TP 2.1 - GENERADORES PSEUDOALEATORIOS

Galarza Karen
Ingeniería en Sistemas de Información
Universidad Tecnológica Nacional
karengalarza94.kg@gmail.com

Gonzalez Aquino Yoana
Ingeniería en Sistemas de Información
Universidad Tecnológica Nacional
yoanakim1@gmail.com

Grancelli Eliseo
Ingeniería en Sistemas de Información
Universidad Tecnológica Nacional
eliseograncelli@gmail.com

Soto Matias Francisco
Ingeniería en Sistemas de Información
Universidad Tecnológica Nacional
matiasfranciscosoto123@gmail.com

Petrelli Juan Franco
Ingeniería en Sistemas de Información
Universidad Tecnológica Nacional
jfpetrelli@gmail.com

Agustina Monti
Ingeniería en Sistemas de Información
Universidad Tecnológica Nacional
agusmonti10@gmail.com

25 de mayo de 2023

ABSTRACT

El siguiente documento tiene por objetivo detallar el trabajo de clase que debe realizarse para introducirnos en uno de los elementos fundamentales para gran parte de las simulaciones, esto son los generadores de números pseudoaleatorios

1. Introducción

Un número pseudoaleatorio es un número generado en un proceso que parece producir números al azar, pero no lo hace realmente. Las secuencias de números pseudoaleatorios no muestran ningún patrón o regularidad aparente desde un punto de vista estadístico, a pesar de haber sido generadas por un algoritmo completamente determinista, en el que las mismas condiciones iniciales producen siempre el mismo resultado.

2. Generadores pseudoaleatorios

En el siguiente trabajo practico estudiaremos distintos tipos de generadores pseudoaleatorios, entre ellos, Generador lineal congruencial(GLC), la media de los cuadrados, otros. Creados en un entorno de trabajo de lenguaje de programación Python.

2.1. Media de los cuadrados

Este método fue propuesto por el matemático John Von Neumann (1903-1957). En el cual la sucesión se obtiene por recurrencia.
• Se inicia con una semilla de 4 dígitos.
• La semilla se eleva al cuadrado, obteniendo un numero de 8 dígitos (si no es así, se le agregan ceros).
• Los 4 números del centro serán el siguiente numero de la secuencia.
Este método tiene dos inconvenientes principales: tiene una fuerte tendencia a degenerar a cero rapidamente (probar por ejemplo con x0 = 1009). los números generados pueden repetirse cíclicamente después de una secuencia corta.

2.2. Generador congruencial lineal

Un generador congruencial lineal (GCL) es un algoritmo que permite obtener una secuencia de números pseudoaleatorios calculados con una función lineal definida a trozos discontinua. Es uno de los métodos más antiguos y conocidos para la generación de números pseudoaleatorios. La teoría que sustenta el proceso es relativamente fácil de entender, el algoritmo en sí es de fácil implementación y su ejecución es rápida.
Los GCL no deberían ser usados en aplicaciones para las que se requiera aleatoriedad de alta calidad. Por ejemplo, esta técnica es inadecuada para el uso en una simulación de Monte Carlo debido a la correlación serial de la secuencia (entre otros motivos). Tampoco deberían usarse para aplicaciones criptográficas; ejemplos de generadores más adecuados para esta función se pueden encontrar en Generador de números pseudoaleatorios criptográficamente seguro. Si un GLC es sembrado con un carácter e iterado una única vez, el resultado es un sencillo cifrado afín clásico, el cual puede ser descifrado con un análisis de frecuencia estándar. Los Generadores Congruenciales Lineales (GCL) comienza con un valor inicial (semilla) y los sucesivos valores se obtiene recursivamente del modo:

$$x_n = (ax_{n-1} + b) mod m$$

Están determinados por los parámetros:
Modulo :

$$m > 0$$

Multiplicador :

$$0 \leq a < m$$

Incremento :

$$c \leq m$$

Semilla :

$$0 \leq X_0 < m$$

3. Test/Pruebas

Someteremos a distintos tipos de pruebas a los GCL, para verificar la calidad de los números pseudoaleatorios. Los items mas significativos en los números son independencia e uniformidad.

3.1. Media de los cuadrados/Prueba de Bondad

La Prueba de Bondad de Ajuste Chi Cuadrado es el test de bondad de ajuste más utilizado. En general un test de bondad de ajuste se utiliza para discriminar si una colección de datos o muestra se ajusta a una distribución teórica de una determinada población. En otras palabras, nos dice si la muestra disponible representa (ajusta) razonablemente los datos que uno espera encontrar en la población. El test de bondad de ajuste chi cuadrado puede ser utilizado para trabajar tanto con distribuciones discretas como, por ejemplo, la Distribución de Poisson o la Distribución Binomial como así también con distribuciones continuas (por ejemplo, Distribución Normal, Distribución Exponencial, etc).
La aplicación de la prueba de bondad de ajuste chi cuadrado requiere que los datos estén agrupados en categorías o clases. Si los datos originalmente no se encuentran agrupados será necesario agruparlos antes de aplicar el test de chi cuadrado para lo cual sería necesario construir una tabla de frecuencia o histograma.

Esta prueba consiste en clasificar números en diferentes intervalos, Creando una tabla de frecuencias en donde ubica a cada numero aleatorio que esta dentro de dichos intervalos. Las frecuencias esperadas las podemos obtener a través de esta ecuación:

$$E_i = \frac{N}{n}$$

Con estos valores podemos aplicar la ecuación de chi-cuadrado para poder obtener nuestro valor de significancia (Z)

$$X^2_{calculada} = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

Gracias a este valor vamos a poder indicar si nuestros números aleatorios se ajustan a una distribución de chi-cuadrada y poder verificar la validez de nuestros números generados.

3.2. Prueba de autocorrelación

En esta prueba se seleccionan todos los números aleatorios generados y se determinan ciertos parámetros para poder hacer dicha prueba, como por ejemplo: El margen de error, el numero por donde se quiere comenzar con la secuencia y la amplitud de la misma.
Para analizar la correlación se utiliza la densidad de probabilidad:.

$$\rho_{im} = \frac{1}{M+1} \sum_{k=0}^{m} r_{(i+km)} r_{|i+(k+1)m)|}$$

(7)

• N representa el tamaño de la muestra.
• i representa el primer numero donde empieza la amplitud de autocorrelación
• m representa la amplitud de la correlación
• M representa el entero mayor de la correlación
• M se obtiene de la siguiente manera, obteniendo un numero con decimales y truncándolo para obtener solamente el numero entero

$$M = Truncar \left\{ \frac{(N-1)}{m} \right\} - 1$$

(8) Luego tenemos que obtener la desviación estándar de nuestra autocorrelación a través de la siguiente ecuación

$$\sigma_{\rho im} = \frac{\sqrt{13M+7}}{12(M+1)}$$

(9) Por último para obtener la significancia (Z), a través de la siguiente ecuación: Z:

$$Z = \frac{\rho_{im} - 0{,}25}{\sigma_{\rho im}}$$

(10)

3.3. Prueba de Poker

Esta prueba examina en forma individual los dígitos del numero aleatorio generado. La forma como esta prueba se realiza es tomando 5 dígitos a la vez y clasificándolos como : Par, dos pares, tercia, póker quintilla full y todos diferentes. Las probabilidades para cada una de las manos del póker diferentes se muestran enseguida:

• Todos diferentes = 0.3024
• Dos pares diferentes
• Un par = 0.504
• Dos pares = 0.108
• Tercia = 0.072
• Full = 0.009
• Quintilla = 0.0001

Al categorizar nuestros números aleatorios de esa manera, se genera una tabla de frecuencia de cada una de esas categorías con relación a los números aleatorios obtenidos y podemos obtener así, las frecuencias observadas, mientras que las frecuencias esperadas son la probabilidad de ocurrencia de cada categoría en particular.

Luego de obtener esos datos podemos aplicarlos a una distribución de chi-cuardado para poder obtener nuestro valor de significancia

$$X^2 = \sum_{i=1}^{7} \frac{(Oi - Ei)^2}{Ei}$$

(11)

Luego, con este valor de Z podemos determinar si los números se ajustan a una distribución de chi-cuadrado o no y determinar la validez de nuestros números aleatorios.

3.4. Prueba de corrida arriba abajo

Este procedimiento consiste en determinar una secuencia de unos y ceros de acuerdo a la comparación de cada número r_i, que cumpla con la condición de ser mayor a 0.5 (en el caso de los unos) o ser menor a 0.5 (en el caso de los ceros).

Luego se determina el número de corridas $\mathbf{C_0}$ y los valores de n_0 y n_1

Valores que se emplean:
$\mathbf{C_0}$= Número de corridas en la secuencia.
$\mathbf{C_0}$= Número de corridas en la secuencia.
$\mathbf{n_0}$= Cantidad de ceros en la secuencia S.
$\mathbf{n_1}$= Cantidad de unos en la secuencia de S.
$\mathbf{n}$ = Cantidad de números
$\mathbf{n}$ = n_0 + n_1

Posteriormente se calcula el valor esperado, la varianza del número de corridas y el estadístico $\mathbf{Z_0}$ con las siguientes ecuaciones:

Valor esperado:
$\mu_{c0} = \frac{2n_0n_1}{n} + \frac{1}{2}$

Varianza del número de corridas:
$\sigma^2_{c0} = \frac{2n_0n_1-n}{n^2(n-1)}$

El estadístico:
$\mathrm{Z}_0 = \frac{C_0-\mu_{c0}}{\sigma_{c0}}$

Para saber si el estadístico $\mathbf{Z_0}$ está fuera del intervalo se emplea la siguiente fórmula:

$$-Z_{\frac{inf}{2}} \leq Z^0 \leq Z_{inf}2$$

Si la condición anterior se cumple, entonces se concluye que los números evaluados son independientes, de lo contrario se rechaza al conjunto.

4. Resultados

Mediante la aplicación de los distintas pruebas a los distintos generadores obtuvimos los siguientes resultados.

4.1. Método de la parte media del cuadrado

Se define la semilla = 1991 y se generan 1500 números aleatorios. En la Figura 1 graficamos los números generados por el generador ordenados en forma ascendente (Naranja) y lo comparamos con el valor esperado (Azul) lo que nos indicaría que la distribución no es uniforme.

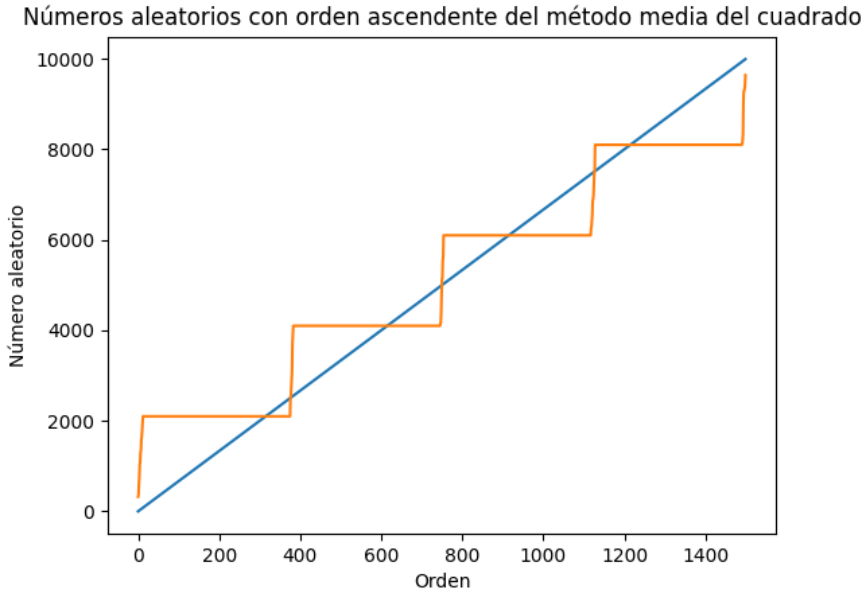

*Figura 1. Numeros pseudoaleatorios

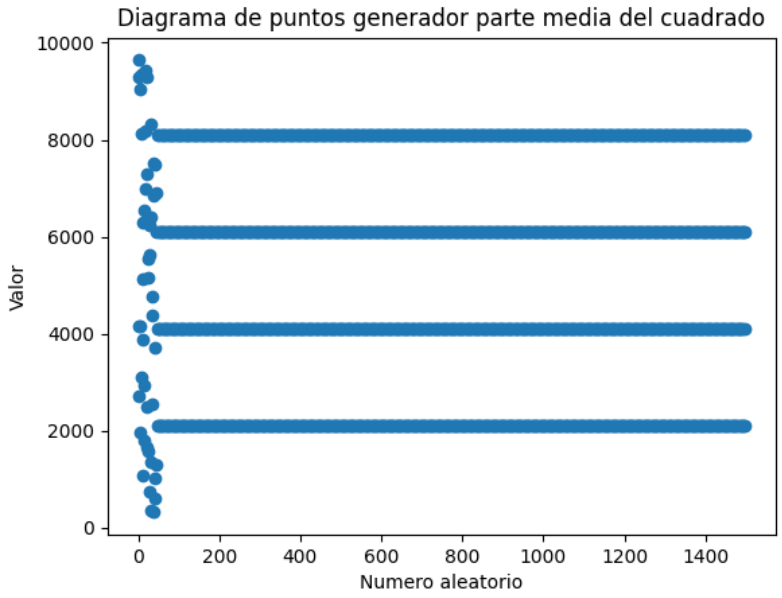

Figura 2. Diagrama de dispersión

En este diagrama podemos detectar que los puntos de dispersion comienzan en todos los valores de la misma manera aleatoria, luego se demuestra una dispersion más homogenea a medida que se avanza hacia el nro 1500.

Resultados de tests generador parte media del cuadrado
Test chicuadrados: Rechazado
Test de corridas: Rechazado
Test prueba de series: Rechazado

4.2. Generador congruencial lineal

Se define la semilla = 1991, módulo = 32768, multiplicador = 26765, incremento = 21001 y se generan 1500 números aleatorios. En la Figura 3 graficamos los números generados por el generador ordenados en forma ascendente (Naranja) y lo comparamos con el valor esperado (Azul) lo que nos indicaría que la distribución podría ser uniforme

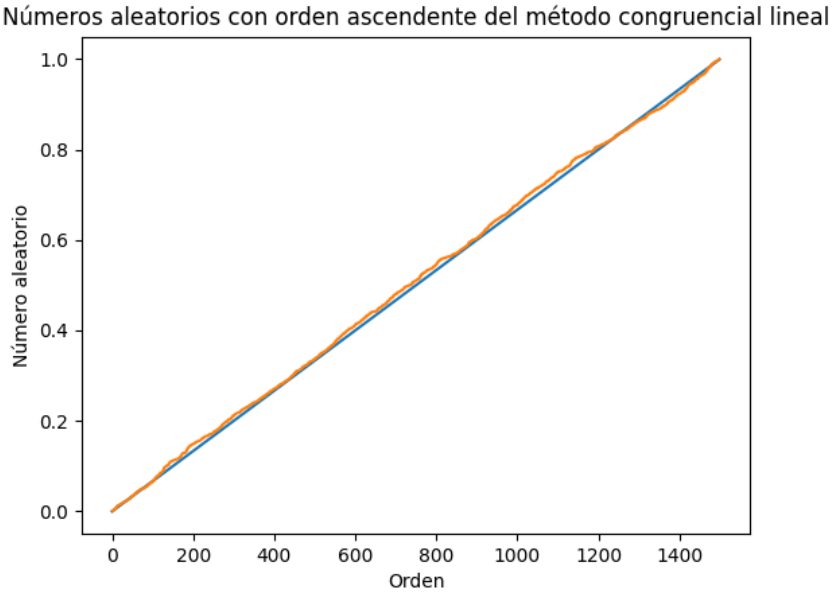

Figura 3. Numeros Pseudoaleatorios

Mediante la figura 4 podemos observar que los números parecen tener una distribución uniforme.

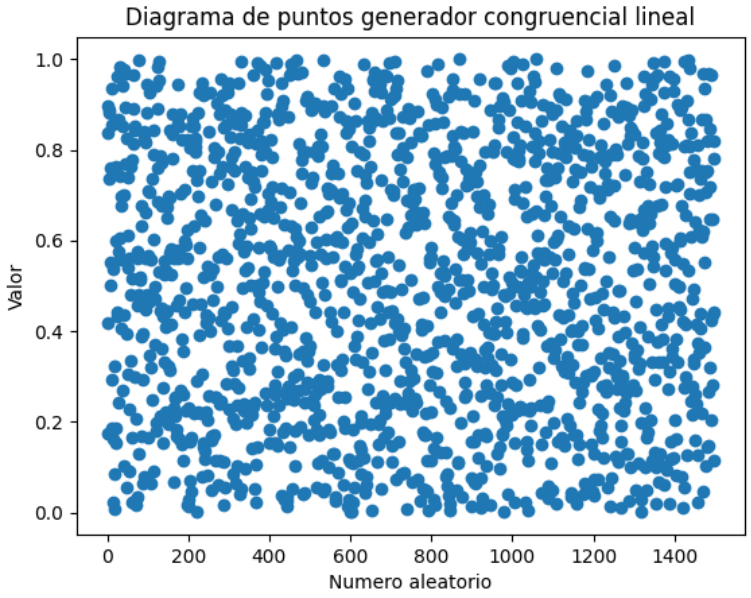

Figura 4. Diagrama de dispersión

En este diagrama se puede apreciar que los puntos entre los valores de numeros aleatorios no representan un patrón definido.

Resultados de tests generador congruencial lineal
Test chicuadrados: Rechazado
Test de corridas: Aceptado
Test prueba de series: Aceptado

4.3. Generador python

En la figura 5 graficamos los números generados por el generador ordenados en forma ascendente (Naranja) y lo comparamos con el valor esperado (Azul) lo que nos indicaría que la distribución podría ser uniforme.

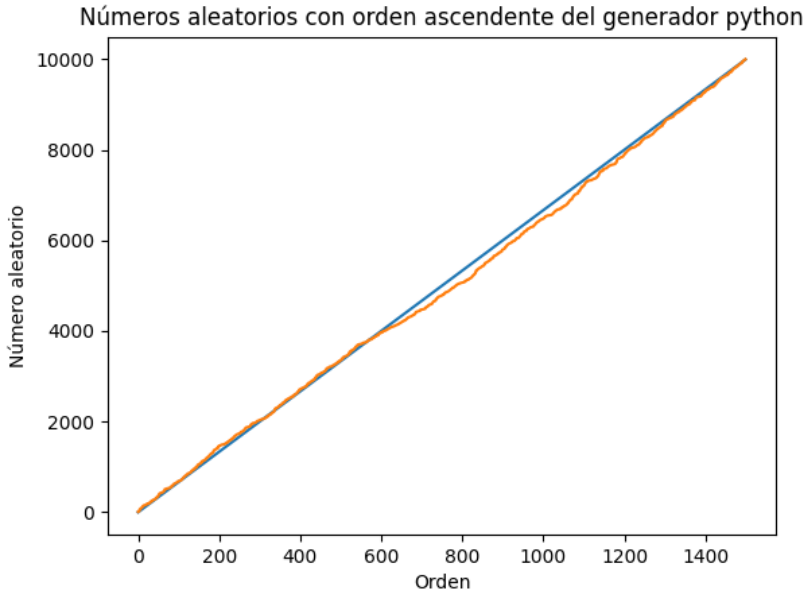

Figura 5. Números Pseudoaleatorios

Mediante la figura 6 podemos observar que los números parecen tener una distribución uniforme.

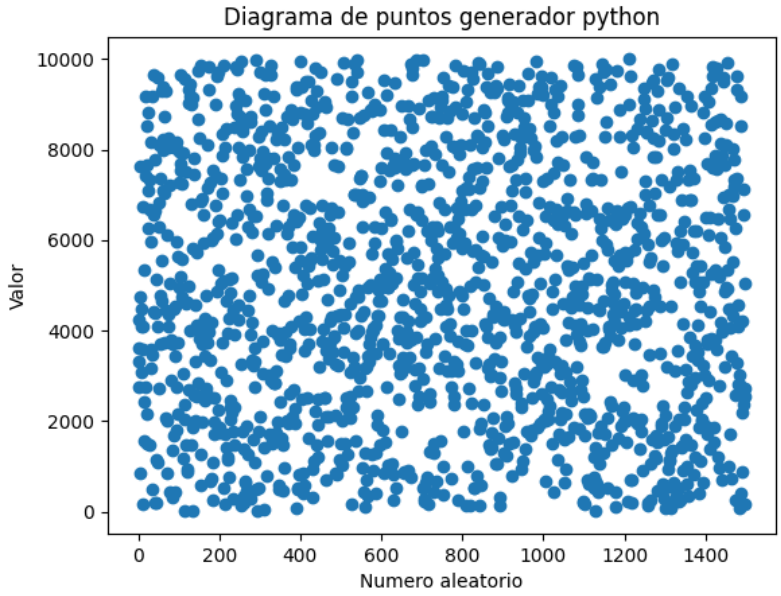

Figura 6. Diagrama de dispersión

En este Diagrama de puntos generador pyhton se puede ver que la representacion de los puntos es muy dispersa y no se puede apreciar un seguimiento de patrones de manera definida.

Resultados de tests generador python

Test chicuadrados: Rechazado
Test de corridas: Aceptado

Test prueba de series: Aceptado

5. Conclusión

Considerando las pruebas realizadas, se puede observar que ninguno de los generadores pseudo-aleatorios han logrado superar todas las pruebas. Esto evidencia que ningún generador produce números aleatorios en realidad, si no que emulan este comportamiento mediante algoritmos informáticos que no se predicen con facilidad y permiten generar estos números. Es realmente imposible modelar con precisión las caracteristicas de la distribución uniforme mediante un buen generador, algunas propiedades no se cumplirán, las cuales pueden no influir mucho en los resultados de determinado estudio, dando lugar a que el criterio de aceptación de un generador dado debe basarse en la aplicación que se le vaya a dar. Es precisamente responsabilidad del analista realizar las pruebas pertinentes.

6. Anexo

6.1. Valores obtenidos

Estos son los resultados en 100 conjuntos con 15.000 elementos.

Generador Congruencial Lineal: aceptado: 82 rechazado: 18

Generador Python: aceptado: 72 rechazado: 28

En el siguiente enlace se puede ver los Test de Pruebas que se usaron.

`https://github.com/jfpetrelli/Simulacion-2023/blob/master/TP-2.1/Resultados.txt`

6.2. Código Python

`https://github.com/jfpetrelli/Simulacion-2023/blob/master/TP-2.1/generadores.py`

Bibliografía

[1] Latex - Documentacion
`https://es.overleaf.com/learn`

[2] Random - Analysis
`https://www.random.org/analysis/`

[3] Numeros Pseudoaleatorios
`https://tereom.github.io/est-computacional-2018/numeros-pseudoaleatorios.html/`

[4] Números aleatorios
`https://www.estadisticaparatodos.es/taller/aleatorios/aleatorios.html`