TP 2.1 - GENERADORES PSEUDOALEATORIOS

Abud Santiago Elias
Legajo 47015
sabudvicco@gmail.com

Buchhamer Ariel
Legajo 46217
arielbuchhamer1@outlook.com

Castellano Marcelo
Legajo 39028
marce.geek22@gmail.com

Dolan Guillermo Patricio
Legajo 46101
guillermo230899@gmail.com

Navarro Franco
Legajo 46387
franconavarro1889@gmail.com

8 de mayo de 2022

ABSTRACT

Los números pseudoaleatorios se generan de manera secuencial con un algoritmo determinístico. Construir un buen algoritmo de números pseudoaleatorios es complicado, por eso hemos hecho un estudio sobre cómo funcionan y de la manera que se comportan.

1. Introducción

Los números aleatorios son muy útiles en distintos ámbitos de aplicación. En nuestro caso, resulta especialmente importante su aplicación al campo de la simulación estocástica.[2]

Hace muchísimos años, las personas que necesitaban números aleatorios en sus trabajos científicos tenían que recurrir a fuentes "convencionales" de aleatoriedad (bolilleros, dados, cartas, entre otras). Más tarde se publicaron tablas, como la publicada en 1927 por H. L. C. Tippett, de 40000 dígitos. Posteriormente surgieron dispositivos que generaban números aleatorios de manera mecánica. La computadora Ferranti Mark I, instalada por primera vez en 1951, tenía una instrucción que obtenía 20 bits aleatorios usando el ruido de una resistencia eléctrica. En 1955 la RAND Corporation publicó una ampliamente usada tabla de un millón de dígitos aleatorios obtenidos con la ayuda de otro dispositivo especial. Una famosa máquina de números aleatorios llamada ERNIE (el primer modelo, construido en 1956[1]) fue utilizada por muchos años (desde 1957 hasta 1973[1]) para seleccionar los números ganadores de los bonos de lotería *Premium Savings Bonds* británicos. Esta última fue luego reemplazada por sucesivas versiones, siendo la última y actual la quinta, de 2019, que funciona con tecnología cuántica, empleando luz en lugar de ruido térmico como las anteriores[1].[2]. Hoy en día otro generador de números aleatorios reales y de acceso al público en general es el servicio de RANDOM.ORG, que se basa el procesamiento de ruido atmosférico captado mediante un arreglo de receptores de radio[3].

Con la introducción de las computadoras, se comenzó a buscar maneras eficientes de generar números aleatorios dentro de los programas. Las tablas no eran prácticas porque tenían un tamaño finito, requerían memoria y tiempo para cargarlas, además de que había que mantenerlas[1]. Las máquinas como ERNIE podían servir, pero dificultaban probar el software, debido a la no repetibilidad de sus resultados; por otro lado, esa clase de máquinas tendían a sufrir de malfunciones muy difíciles de detectar.[2]

Fue esta insuficiencia de los métodos mecánicos en los primeros días lo que llevó a un interés en producir números aleatorios empleando las operaciones aritméticas ordinarias de una computadora. John von Neumann sugirió por primera vez este enfoque alrededor de 1946, ideando el método de los cuadrados medios, que luego analizaremos en mayor detalle.[2]

[1] Los avances tecnológicos desde la década de 1990 hasta la actualidad las han hecho nuevamente útiles, debido a la posibilidad de almacenar, transportar y transmitir grandes volúmenes de datos con facilidad[2]

Estos métodos, que parecen generar números aleatorios, pero no lo hacen realmente, se denominan *pseudoaleatorios* o *cuasialeatorios*[2]. Existen diferentes tipos y familias de generadores números pseudoaleatorios, algunos de ellos son el método de los cuadrados medios y los generadores congruenciales lineales (GCL) (y los generadores congruenciales en general). En nuestro estudio nos enfocaremos en el método de los cuadrados medios y los GCL.

2. Descripción del trabajo

Utilizando en Python 3.7, reproducimos y comparamos algoritmos generadores de números pseudoaleatorios, poniendo a prueba la aleatoriedad de cada método por medio de diferentes pruebas.

Los algoritmos generadores de números pseudoaleatorios utilizados fueron:

1. Método de los cuadrados medios.
2. Generador lineal congruencial (GLC), en sus formas:
   a) Ansi
   b) Randu

Y las pruebas realizadas, fueron:

1. Test de Chi Cuadrado
2. Test de póker

3. Marco teórico

Existen diferentes tipos y familias de generadores números pseudoaleatorios, algunos de ellos son el método de los cuadrados medios y los generadores congruenciales lineales (y los generadores congruenciales en general).

A grandes rasgos, un generador de números pseudoaleatorios se basa en aplicar una serie de operaciones aritméticas a un estado inicial (la *semilla*) para obtener el siguiente, y así sucesivamente.

Período de un generador Debido a que el estado de un generador es finito, finalmente el mismo se repetirá en algún punto y habrá un ciclo que se repite infinitamente. La cantidad de iteraciones necesaria para que esto ocurra es lo que se denomina el *período del generador*. Según el tipo de generador, sus parámetros y tal vez la semilla empleada puede ser que el mismo sea el máximo soportado por el tamaño de su estado o que sea menor. Normalmente es deseable que este sea suficientemente grande, idealmente el máximo posible. Sin embargo, vale la pena notar que un período grande no determina que un generador es bueno, debemos verificar que los números que se generan se comportan como si fueran aleatorios.

Recordemos que lo que nos interesa para trabajar con un buen generador de números aleatorios es que la distribución de los números obtenidos tiene que ser uniforme, no deben de haber correlaciones entre los términos de la secuencia, el período debe ser lo más largo posible, y el algoritmo debe ser de ejecución rápida.

El problema es saber qué generador de números es mejor, ya que la razón es que si su generador de números aleatorios es bueno, es igualmente probable que aparezca cada posible secuencia de valores. Esto significa que un buen generador de números aleatorios también producirá secuencias que parecen no aleatorias para el ojo humano y que también fallan en cualquier prueba estadística a la que podamos exponerlo.

Es imposible probar definitivamente la aleatoriedad. Una forma de aproximar esto es tomar muchas secuencias de números aleatorios de un generador dado y someterlos a una batería de pruebas estadísticas. A medida que las secuencias pasan más pruebas, aumenta la confianza en la aleatoriedad de los números y también la confianza en el generador. Sin embargo, debido a que esperamos que algunas secuencias no parezcan aleatorias, debemos esperar que algunas de las secuencias fallen al menos en algunas de las pruebas. Sin embargo, si muchas secuencias fallan en las pruebas, deberíamos sospechar.

Hay varias formas de examinar un generador de números aleatorios, las distintas pruebas estadisticas son: analisis visual simple, análisis estadístico de Charmaine Kenny y análisis estadístico de Louise Foley.

3.1. Tipos de generadores

3.1.1. Método de los cuadrados medios

Como se mencióno anteriormente, este método fue desarrollado por Jonh von Neumann, quien sugirió usar las operaciones aritméticas de una computadora para generar secuencias de números pseudoaleatorios. Con este procedimiento se generan números pseudoaleatorios de 4 dígitos de la siguiente forma:

1. Se inicia con una semilla de 4 dígitos.
2. La semilla se eleva al cuadrado, produciendo un número de 8 dígitos (si el resultado tiene menos de 8 dígitos se añaden ceros al inicio).
3. Los 4 números del centro serán el siguiente número en la secuencia, y se devuelven como resultado.

Este generador cae rápidamente en ciclos cortos, por ejemplo, si aparece un cero se propagará por siempre.

A inicios de 1950s se exploró el método y se propusieron mejoras, por ejemplo para evitar caer en cero. Metrópolis logró obtener una secuencia de 750,000 números distintos al usar semillas de 38 bits (usaba el sistema binario), además la secuencia de Metrópolis mostraba propiedades deseables. No obstante, el método del valor medio no es considerado un método bueno por lo común de los ciclos cortos.

3.1.2. Generadores congruenciales lineales

Los generadores congruenciales lineales (GCL) fueron introducidos en 1949 por D.H. Lehmer, son muy populares y utilizados incluso hoy en día, en casos donde no se requieran mejores generadores (por ejemplo, no se pueden usar en aplicaciones criptográficas). Tienen la forma:

$$X_{n+1} = (aX_n + c) \mod m \tag{1}$$

Donde a es el multiplicador, m el módulo, c el incremento y X la semilla.

Hay distintas variantes de este generador según sus parámetros... Lehmer Parker-Miller, minstd, rand, randu, etc.

Los GCLs continúan siendo utilizados en muchas aplicaciones porque con una elección cuidadosa de los parámetros (la elección de los parámetros determina la calidad del generador) pueden pasar muchas pruebas de aleatoriedad, son rápidos y requieren poca memoria[2].

3.2. Pruebas estadísticas

3.2.1. Prueba χ^2 de Pearson

La prueba χ^2 de Pearson se considera una prueba no paramétrica que mide la discrepancia entre una distribución observada y otra teórica (bondad de ajuste), indicando en qué medida las diferencias existentes entre ambas, de haberlas, se deben al azar en el contraste de hipótesis.[7]

Pone a prueba una hipótesis nula que establece que la distribución de frecuencia de ciertos eventos obervados en una muestra es consistente con una distribución teórica en particular. Los eventos considerados deben ser mutuamente excluyentes y tener una probabilidad total igual a 1. Un caso común para esto es donde cada uno de los eventos cubren un resultado de una variable categórica.[6]

La prueba χ^2 de Pearson se usa para realizar distintos tipos de contrastes, como bondad de ajuste, homogeneidad, e independencia.[6] Para nuestro estudio, como los valores generados son números enteros y deberían estar distribuidos de manera uniforme, llevamos a cabo pruebas de bondad de ajuste a la distribución uniforme discreta.

Una prueba de bondad de ajuste determina si una distribución de frecuencia observada difiere de una distribución teórica dada.

Planteamos el test de hipótesis siguiente, la hipótesis nula (H_0) de que los datos siguen la distribución esperada y la alternativa (H_1), de que los datos no siguen la misma:

H_0 : no hay diferencia entre las distribuciones

H_1 : hay diferencia entre las distribuciones

El procedimiento de cálculo de la prueba de χ^2 para bondad de ajuste incluye los pasos siguientes[6]:

[2] Quizás esto no sea tan relevante hoy en día, pero sí que lo es especialmente en dispositivos muy limitados (como pueden ser los microcontroladores)

1. Calcular el estádistico χ^2, que se asemeja a una suma normalizada de los desvíos al cuadrado entre las frecuencias observadas y las teóricas (ecuación 2).
2. Determinar los grados de libertad, **gl**, del estadístico: para nuestro caso, de bondad de ajuste, gl = $n - m$, donde n es el número de valores distintos de la distribución, y m es el número de parámetros ajustados para hacer que la distribución se ajuste mejor a las observaciones: el número de valores reducidos por el número de parámetros ajustados en la distribución.
3. Seleccionar el nivel deseado de confianza (nivel de significancia, valor p[3] o el nivel alfa correspondiente) para el resultado de la prueba. Usualmente y para nuestro caso seleccionamos un alfa de 0.05, lo que corresponde a un 95 % de confianza.
4. Comparar χ^2 con una distribución χ^2 con gl grados de libertad para obtener el valor p correspondiente y emplear y el nivel de confianza seleccionado (de un solo lado, puesto que la prueba es solamente en una dirección, esto es, ¿es el valor del estadístico de prueba mayor que valor el crítico? o ¿es el valor p menor o igual al alfa?), lo que en muchos casos da una buena aproximación de la distribución χ^2.
5. Rechazar o mantener la hipótesis nula de que la distribución de frecuencias observadas es la misma que la teórica empleando el valor p correspondiente. Si el valor p es menor o igual que el nivel alfa seleccionado, se rechaza la hipótesis nula (H_0) y se acepta la alternativa (H_1), con el nivel de confianza seleccionado. Si en cambio el valor p supera dicho umbral, no se puede llegar a una conclusión clara, y se mantiene la hipótesis nula (no la podemos rechazar), lo que no significa necesariamente que la misma sea aceptada.

Prueba de bondad de ajuste - distribución discreta uniforme En este caso se dividen N observaciones entre n valores[6]. Una aplicación simple es probar la hipótesis de que, en la población general, los distintos valores se producirán con la misma frecuencia. La frecuencia teórica absoluta para cualquier valor (bajo la hipótesis nula de una distribucíón discreta uniforme) se calcula como

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

y la reducción en los grados de libertad es $p = 1$, dado que las frecuencias observadas O_i deben cumplir con la restricción de sumar N. Esto es, los grados de libertad resultan ser gl = $n - 1$

El valor del estadístico de prueba es

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} = N \sum_{i=1}^{n} \frac{(O_i/N - p_i)^2}{p_i} \tag{2}$$

donde

- χ^2: estadístico acumulativo de prueba de Pearson, que se aproxima asintóticamente a una distribución χ^2
- O_i: número de observaciones de tipo i
- $E_i = Np_i$: la cantidad esperada (teórica) de observaciones de tipo i, afirmada por la hipótesis nula de que la fracción de observaciones de tipo i en la población es p_i

Finalmente, el estadístico χ^2 puede entonces emplearse para calcular un valor p comparando el valor del estadístico con una distribución χ^2.[6] Esto es

$$\text{valor p} = 1 - P(X \leq \chi^2) \tag{3}$$

Como ya se ha mencionado, si el valor p resulta menor o igual que el nivel de significancia $\alpha = 0{,}05$, rechazamos la hipótesis nula y concluimos con un 95 % de confianza que los valores generados difieren de manera estadísticamente significativa de una distribución discreta uniforme y por lo tanto no son aleatorios, en caso contrario, concluimos que no se puede rechazar la hipótesis nula y por lo tanto los valores pueden ser aleatorios.

Implementación En nuestro trabajo empleamos la función `scipy.stats.chisquare`, del paquete SciPy[4], que se encarga tanto de calcular el estadístico como el valor p correspondiente y ya tiene en cuenta por defecto la reducción en los grados de libertad $p = 1$.

[3] Probabilidad de obtener resultados de prueba al menos tan extremos como los observados, bajo la suposición de que la hipótesis nula es correcta[5]

4. Análisis de resultados

5. Conclusiones

Referencias

[1] Virtual ERNIE. *E.R.N.I.E - the Electronic Random Number Indicator Equipment*. [Internet; consultado 8-mayo-2022]. URL: https://www.virtualcolossus.co.uk/ERNIE/ernie.html.

[2] Donald E Knuth. *The Art of Computer Programming, volume 2: Seminumerical Algorithms (3rd Edition)*. Addison-Wesley Professional, 1997.

[3] RANDOM.ORG. *RANDOM.ORG - Frequently Asked Questions (FAQ)*. [Internet; consultado 8-mayo-2022]. URL: https://www.random.org/faq/.

[4] SciPy. *SciPy v1.8.0 Manual - scipy.stats.chisquare*. [Internet; consultado 5-mayo-2022]. 2022. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.chisquare.html.

[5] Wikipedia. *P-value — Wikipedia, The Free Encyclopedia*. [Internet; consultado 5-mayo-2022]. 2022. URL: https://en.wikipedia.org/w/index.php?title=P-value&oldid=1086336141.

[6] Wikipedia. *Pearson's chi-squared test — Wikipedia, The Free Encyclopedia*. [Internet; consultado 5-mayo-2022]. 2022. URL: https://en.wikipedia.org/w/index.php?title=Pearson%27s_chi-squared_test&oldid=1082334148.

[7] Wikipedia. *Prueba χ^2 de Pearson — Wikipedia, La enciclopedia libre*. [Internet; consultado 5-mayo-2022]. 2022. URL: https://es.wikipedia.org/w/index.php?title=Prueba_%CF%87%C2%B2_de_Pearson&oldid=142386482.