Índice general

Capítulo 1

Números aleatorios

En el año 1945, en la Universidad de Pensilvania, fue inventado el primer ordenador electrónico, conocido como *ENIAC*. Al mismo tiempo, John von Newman, un investigador en Los Alamos, estaba muy interesado en las investigaciones termonucleares que estaban siendo llevadas a cabo. Cuando Newman se enteró de la existencia del ENIAC, propuso un modelo relativamente simple para estudiar la viabilidad de un arma termonuclear.

Los resultados obtenidos del ENIAC fueron revisados por varios científicos en 1946, en Los Alamos. Entre ellos se encontraba Stan Ulam, un investigador que tenía gran interés en el estudio de los procesos aleatorios presentes en los juegos de azar como el poker. Él sabía que las técnicas de muestreo estadísticas estaban en desuso, pero al ver los resultados obtenidos pensó que deberían volver a usarse. Ulam discutió la idea con Newman, creando así en 1947, una aproximación para resolver el problema de la difusión de neutrones en material fisionable usando métodos estadísticos.

Debido a las características del método y al hecho de que un tío de Ulam solía pedir prestado dinero a su familia para ir al casino de Monte Carlo, Nicholas Metropolis sugerió darle ese mismo nombre, surgiendo así el *Método de Monte Carlo*.

Este método necesita números aleatorios, ya que los comportamientos de los neutrones siguen ciertas distribuciones de probabilidad, que pueden ser obtenidas a partir de una distribución uniforme. Por esa razón, en este capítulo vamos a ver qué son los números aleatorios, cómo generarlos y cómo comprobar cómo de aleatorios son los números generados.

1.1. Noción de secuencia aleatoria

Un número aleatorio es un número generado por un proceso cuyo resultado es impredecible y que no se puede reproducir posteriormente de forma fiable. A primera vista, esta definición no parece práctica, ya que necesitamos algún mecanismo que nos dé números aleatorios. Existen infinidad de

procesos a partir de los cuales podemos obtener números aleatorios: como los juegos de azar perfectos o la medición de ruidos electrónicos. El principal problema de estos procesos es que no son reproducibles, luego no podemos contrastar los resultados que obtenemos.

Por esa razón, es preferible usar una definición matemática, que tenga en cuenta ciertas características de los números aleatorios. Como es esperable, esta definición no es única. En este trabajo se ha escogido la que da Antonio Salmerón [1]:

Definición 1.1. *Una secuencia de números aleatorios es una sucesión de variables aleatorias independientes* $\{X_1, \dots X_n\}$ *donde* $X_i \rightsquigarrow \mathcal{U}[0,1)$ *para todo* $i = 1, \dots, n$.

Hay que notar que el dominio de cada X_i no tiene por qué ser el intervalo $[0,1)$, sino que puede ser cualquier intervalo de $\mathbb{R}$. Sin embargo, es conveniente añadir esa restricción ya que facilita algunos cálculos, porque como se verá en una sección posterior, a partir de una distribución uniforme se pueden generar otras muchas distribuciones aleatorias. Otro aspecto importante a notar es que él no define un *número aleatorio*, sino una *secuencia aleatoria*. Esto es debido a que no tiene mucho sentido verificar la aleatoriedad de un único número.

Por último, es importante tener en cuenta que, en principio, es imposible generar números realmente aleatorios usando un ordenador, ya que es una máquina totalmente determinística. Por esa razón, a los generados a partir de un proceso determinista se llaman números *pseudoaleatorios*.

Definición 1.2. *Los números de una secuencia creada a través de un proceso determinista (una rutina, un programa, etc) reciben el nombre de números pseudoaleatorios.*

1.2. Métodos de generación

El objetivo de los científicos, como John von Newman, era generar números aleatorios [6], para posteriormente obtener otras distribuciones probabilísticas. Como ya se ha comentado, es posible obtener estos números a partir de juegos de azar perfectos, pero el proceso es largo y tedioso, luego el interés de estos métodos para fines estadísticos es escaso.

1.2.1. Métodos manuales

Estos procedimientos, al ser tan engorrosos, promovieron la creación de tablas de números aleatorios, como la publicada por L.H.C. Tippett, con 40000 entradas diferentes, obtenida a partir de registros del censo [11]. Posteriormente, en 1955, simulando una ruleta en un dispositivo hardware conectado a un ordenador, la RAND Corporation creó una tabla con un millón

de entradas diferentes [12]. Estos métodos eran denominados *métodos manuales* y sí creaban números realmente aleatorios. Esta tabla fue tan exitosa que incluso se crearon *punched cards* de IBM para que fuera posible usarlas desde un ordenador.

Los principales incovenientes del uso de tablas eran que requería una gran cantidad de espacio y las secuencias eran siempre las mismas. Estos problemas fueron solventados por el avance de los ordenadores, la creación de dispositivos como el ERNIE (usado para la lotería inglesa) y el desarrollo de los *métodos digitales*.

1.2.2. Métodos digitales

Los métodos digitales son algoritmos numéricos que, a partir de una *semilla*, generan secuencias de números de la forma

$$X_{n+1} = f(X_0, \ldots, X_n)$$

donde f es una función de cualquier tipo. La dependencia funcional de f hace evidente que las secuencias generadas por estos métodos no son realmente aleatorias, sino pseudoaleatorias, ya que responden a una formulación matemática totalmente determínistica. Sin embargo, esta pérdida se suple, en cierta medida, por las propiedades descritas en la definición 1.1. Además, estos métodos o generadores tienen que evitar todos los problemas que hemos descrito anteriormente, es decir, deben caracterizarse por su:

- Reproductividad: toda secuencia de números que generen debe poderse reproducir para que los experimentos que se realicen a partir de ella se pueden contrastar.
- Eficiencia: el método debe ser relativamente rápido de ejecutar y debe consumir poca memoria.

A continuación, vamos a describir el primero de estos métodos y luego vamos a estudiar el caso más simple de los métodos congruenciales.

Middle Square method

El primero método digital desarrollado es denominado *método del centro de los cuadrados*, propuesto por John Von Neumann en 1946 [13]. El método es bastante simple:

1. Fijar un número de dígitos $N \in \mathbb{N}$ y una semilla inicial $n \in \mathbb{N}$.
2. Calcular n^2 y tomar las $N/2$ cifras del medio.
3. Repetir (2) con el número obtenido. En caso de que n^2 no tenga N dígitos, se añadiran tantos ceros a la izquierda como sea necesario.

Ejemplo 1.1. *Tomando $N = 4$ y $n = 24$, se obtiene la siguiente tabla:*

n	n^2	X_i
24	0|57|6	*0.52*
57	3|24|9	*0.24*
24	0|57|6	*0.52*
57	3|24|9	*0.24*
...	...	...

Como se aprecia en el ejemplo, el método va a repetir los números 24 y 57 sucesivamente. Este ejemplo da lugar a la siguiente definición:

Definición 1.3. *A la longitud de la secuencia máxima que se pueda obtener a partir de un generador de números pseudoaleatorios se le denomina longitud de periodo o simplemente periodo.*

En el ejemplo 1.1, el periodo es igual a 2. Es evidente que un periodo tan bajo no tiene ningún interés, luego en los generadores siempre nos va a intersar obtener el máximo periodo posible. Este hecho nos hace preguntarnos qué factores afectan al periodo. En principio, lo más lógico es que dependa de la elección del número inicial, X_0, al que llamaremos *semilla* y de los propios parámetros del método. Si en el ejemplo anterior, se escogiera como semilla el número 1234, se obtendría un periodo igual a 55. Sin embargo, a partir de ese término, lo único que obtendríamos es una sucesión de ceros, con lo cual, hemos perdido la uniformidad, incumpliendo la definición 1.1.

Debido a estos problemas, de los que el propio Newman era consciente [6], generadores más eficaces fueron diseñados, teniendo en cuenta las tres características anteriores: reproductividad, eficiencia y periodo máximo.

Generadores congruenciales

A continuación vamos a describir una familia de generadores que han sido muy usados debido a que tienen una teoría muy bien desarrollada. Además, son muy fáciles de implementar y requierenn muy poca memoria para funcionar. En concreto, vamos a estudiar el generador congruencial simple, que como su nombre indica, es el caso más simple de esta familia pero podemos determinar de forma exacta su periodo máximo. Al final veremos otro caso particular de esta familia, el registro de retroalimentación lineal con desplazamiento.

El generador congruencial simple se basa en calcular congruencias módulo un número natural m, al resultado de evaluar una determinada función f:

$$X_{n+1} = f(X_0, \ldots, X_n) \ mod \ m \tag{1.1}$$

Dentro de este tipo de métodos, vamos a centrarnos en el más básico, en el cual f es una función lineal, es decir:

$$X_{n+1} = (aX_n + c) \; mod \; m \tag{1.2}$$

donde

- a, es el multiplicador $(0 \leq a < m)$
- m, el módulo $(m > 0)$
- c, el incremento $(0 < c < m)$
- X_0, la semilla $(0 < m)$

A este método se le conoce como Generador Congruencial Lineal (GCL) y fue propuesto por D.H. Lehmer en 1949 [4]. Es fácil deducir que el periodo siempre será igual o menor que m (debido a la operación módulo), luego cuanto más grande sea m, más largo podrá ser el periodo. Esto nos hace preguntarnos cómo influye el resto de los parámetros en el valor del periodo. Para ello tenemos el siguiente teorema [3], que nos da condiciones necesrias y suficientes para obtener un GCL de longitud máxima m.

Teorema 1.1 (Hull-Dobell)**.** *La secuencia definida por la equación* (1.2)*, tiene periodo máximo si se cumple:*

(i) c *es primo relativo con* m*.*

(ii) $a \equiv 1 \; (mod \; p)$*, si* p *es un factor primo de* m*.*

(iii) $a \equiv 1 \; (mod \; 4)$*, si* 4 *es un factor de* m*.*

Nota 1.1. *Si* $c = 0$*, el teorema anterior no se podría aplicar. En ese caso, tendríamos otro tipo de generador llamado Generador Congruencial Multiplicativo (GCM), que precisa otro estudio aparte.*

Demostración. El caso $a = 1$ es fácil, ya que si $mcd(c, m) = 1$, el periodo es evidentemente m, por lo explicado anteriormente. Por lo tanto, solo tenemos que demostrar el caso $a \neq 1$. Partiendo de (1.2), se puede sustituir recursivamente la expresión de X_n, se obtiene:

$$x_n \equiv a^n x_0 + \frac{(a^n - 1)c}{a - 1} \; (mod \; m)$$

queremos encontrar la longitud de ese generador, es decir, el número $n \in \mathbb{N}$ tal que $X_n = X_0$, ya que eso implica que se va a producir exactamente la misma secuencia después de X_n. Operando en la ecuación anterior:

$$\frac{(a^n - 1)(x_0(a - 1) + c)}{a - 1} \equiv 0 \; (mod \; m)$$

Para reducir esa expresión, usamos que $x_0(a-1)+c$ es primo relativo con m. Ese hecho se razona por reducción al absurdo usando las condiciones del teorema. Llamemos $\lambda = x_0(a-1)+c$. Supongamos p un número primo tal que $p|m$ y $p|\lambda$. Por la condición ii, se tiene que $a \equiv 1\ (mod\ p)$, es decir:

$$a = 1 + kp,\ k \in \mathbb{N} \Rightarrow x_0(1+kp-1)+c\ (\ mod\ p) = c \Rightarrow \lambda \equiv c\ (mod\ p)$$

Pero hemos asumido que $p|\lambda$, luego $\lambda \equiv 0\ (mod\ p)$, pero $p \nmid c$ (por i), así que tenemos una contradicción. Luego podemos obviar esa parte de la ecuación y resolver, en su lugar:

$$\frac{a^n-1}{a-1} \equiv 0\ (mod\ m) \tag{1.3}$$

Queremos ver que si a satisface las condiciones del teorema, entonces n es igual a m. Vamos a demostrar primero el resultado si m es una potencia de un primo mayor que 2, es decir, $m = p^\alpha$, donde $\alpha \in \mathbb{N}$ y $\alpha \geq 2$ (si $\alpha = 1$ es el caso del principio).

Como a satisface la condición (ii), a se expreserá como:

$$a = 1 + kp^\beta \tag{1.4}$$

donde $mcd(k,p) = 1$ y $k \neq 0$ porque $a \neq 1$, y $\beta \in \mathbb{N}$. Para comprobar que $n = p^\alpha = m$ satisface (1.3), sustituimos el valor de n y a, obteniendo:

$$\begin{aligned}\frac{a^n-1}{a-1} = \frac{(1+kp^\beta)^{p^\alpha}-1}{kp^\beta} = \frac{1+\sum_{j=1}^{p^\alpha}\binom{p^\alpha}{j}(kp^\beta)^j-1}{kp^\beta} = \\ = \frac{p^\alpha kp^\beta + \frac{p^\alpha(p^\alpha-1)}{2!}(kp^\beta)^2+\cdots+(kp^\beta)^{p\alpha}}{kp^\beta} = \\ = p^\alpha + \frac{p^\alpha(p^\alpha-1)}{2!}kp^\beta+\cdots+(kp^\beta)^{p^\alpha-1}\end{aligned} \tag{1.5}$$

Ahora solo tenemos que ver que esa expresión es divisible por p^α, o lo que es lo mismo, que cada término es divisble por p^α. Para ello, podemos reescribir el término j-ésimo como sigue:

$$\frac{p^\alpha}{j}\binom{p^\alpha}{j-1}k^{j-1}p^{(j-1)\beta},\quad (j>1)$$

Se tiene que $\binom{p^\alpha}{j-1}$ y $k^{j-1}p^{(j-1)\beta}$ son enteros, luego ninguno de ellos *necesitará* parte de p^α. Por tanto, el único elemento que puede *tomar* algo de p^α es j. Sin embargo, como j toma valores en $\{2,\ldots,p^\alpha\}$, el número de veces que el factor p puede aparecer en j es menor que (usando la fórmula de Legendre):

$$v_p(j!) = \sum_{i=1}^{\infty}\left\lfloor\frac{n}{p^i}\right\rfloor \leq \frac{j}{p}+\frac{j}{p^2}+\frac{j}{p^3}+\cdots = \frac{j}{p-1} \tag{1.6}$$

y por lo tanto, es menor o igual que $j-1$. Pero el factor p aparece al menos $j-1$ veces in $p^{(j-1)\beta}$, ya que $\beta \geq 1$. Por consiguiente, el factor p^α no es necesario para que j sea dividido por un elemento del numerador. Como todo elemento de (1.5) es divisible por p^α, la ecuación (1.3) se cumple para $n = p^\alpha$.

Todavía queda demostrar que, efectivamente, $n = p^\alpha$ es el menor valor que satisface (1.3). Tenemos que ver que ningún valor menor a p^α satisface (1.3). Como (1.3) es equivalente a $a^n \equiv 1 \ (mod\, m)$, tenemos que a pertenece al exponente de $n \, (mod\, m)$. Por tanto, se puede usar el teorema descrito en [7], que nos dice que si hay otro número N que cumple (1.3), entonces n divide a N. Como $n = p^\alpha$, viendo que no se cumple para $p^{\alpha-1}$ es suficiente para asegurar que n es el mínimo.

Repitiendo los mismos pasos que en (1.5), pero con $n = p^{\alpha-1}$, se obtiene la siguiente expresión para el término j-ésimo:

$$\frac{a^n - 1}{a-1} = p^{\alpha-1} + \frac{p^{\alpha-1}(p^{\alpha-1}-1)}{2!}kp^\beta + \cdots + (kp^\beta)^{p^{\alpha-1}-1} \tag{1.7}$$

donde el coeficiente j-ésimo es igual a

$$\frac{p^{\alpha-1}}{j}\binom{p^{\alpha-1}}{j-1}k^{j-1}p^{(j-1)\beta}, \quad (j > 1)$$

Al contrario que antes, ahora tenemos que ver que ese número no es divisible por p^α. Evindetemente, el primer término, $p^{\alpha-1}$, no es divisible por p^α, por consiguiente, con demostrar que los otros términos sí dividen a p^α, es suficiente. El argumento es análogo al anterior, pero esta vez nos falta otro término p. Ese término se obtiene en $p^{(j-1)\beta}$, ya que como p es impar, (1.6) es menor o igual que $j-2$.

Con esto queda concluida la prueba para $m = p^\alpha$ si p es un primo impar. Para $p = 2$, simplemente hay que considerar la fórmula alternativa de Legendre:

$$v_p(n!) = \frac{n - s_p(n)}{p-1} \Rightarrow v_2(j!) = j - s_2(j) \leq j - 1$$

Ahora faltaría el caso en el que m es producto de potencias de primos, es decir, cualquier número natural, pero tomando

$$m = p_1^{\alpha_1} \cdots p_s^{\alpha_s}, \quad a = 1 + kp_1^{\alpha_1} \cdots p_s^{\alpha_s}$$

donde $\{p_1, \ldots, p_s\}$ son primos, $\{\alpha_1, \ldots, \alpha_s\}$ son enteros positivos y $k \neq 0$ con $mcd(k, m) = 1$, la demostración es prácticamente igual al caso que hemos desarrollado. □

Este teorema tiene gran importancia ya que nos asegura que si escogemos los parámetros a y c cumpliendo ciertas condiciones, podemos obtener

un generador congruencial lineal de longitud arbitraria m. Esa característica combinada con lo simple que es de calcular X_{n+1}, hace a este tipo de G.C.L una buena elección de generador de números pseudoaleaorios. En [2], puede verse una lista de los valores (a, c, m) escogidos por diferentes implementaciones de este proceso.

Nota 1.2. *Es importante notar, que aunque el teorema es válido para cualquier producto de primos, casi siempre se toma m como una potencia de 2, normalmente* 2^{32} *o* 2^{64}*. Esto es debido a razones de eficiencia por la forma en la que se almacenan los datos en la memoria de un ordenador.*

Por último, vamos a ver una generalizacion del generador congruencial multiplicativo.

Linear-Feedback Shift Register Generators

Un registro de retroalimentación lineal con desplazamiento, o LFSR, es un registro de desplazamiento (circular) cuyo bit de entrada es el resultado de evaluar una función lineal usando su estado previo. Este tipo de generadores fue propuesto por Tausworthe en 1965. El bit resultante se puede expresar mediante la siguiente recurrencia:

$$b_i \equiv (a_p b_{i-p} + a_{p-1} b_{i-p+1} + \ldots + a_1 b_{i-1}) \ (mod\ 2) \tag{1.8}$$

Como el módulo del generador, $m = 2$, es un número primo, podemos expresar la parte derecha de la congruencia como el siguiente polinomio:

$$f(z) = z^p - (a_1 z^{p-1} + \ldots + a_{p-1} z + a)$$

sobre el cuerpo finito de Galois $\mathcal{G}(2)$, definido sobre los enteros $\mathbb{Z}_2$ con las operaciones suma y producto usuales, seguidas de una reducción módulo 2. Como consecuencia de la teoría de Galois, se tiene que mientras haya al menos un $b_i \neq 0$, el periodo de la recurrencia será $2^p - 1$ si y solo si f es irreducible en $\mathcal{G}(2)$.

Si el grado es igual a 2, solo hay un binomio irreducible, $x + 1$. Sin embargo, hay una gran variedad de trinomios módulo 2 (ver [14]). Tales trinomios suelen denotarse por $R(p, r) = x^p + x^r + 1$. Si tomamos uno de esos polinomios como f, obtenemos la siguiente recurrencia:

$$b_i \equiv b_{i-p} + b_{i-r} \ (mod\ 2)$$

donde $q = p - r$. La suma en binario se realiza mediante la operación lógica or-exclusivo (xor), denotada por $\oplus$, quedando:

$$b_i = b_{i-p} \oplus b_{i-r}$$

Una vez hayamos evaluado esta recurrencia cierto número de veces l, con $l \leq p$, podemos interpretar la secuencia de bits como tuplas de longitud

l contiguas (si llegamos al final, se empieza por el principio, ya que es un registro con desplazamiento). Es evidente ver que si l es primo relativo con $2^p - 1$, entonces el periodo de las l-tuplas también será $2^p - 1$.

Ejemplo 1.2. *Primero tomamos uno de los polinomios de [14], como por ejemplo,* $x^3 + x^5 + 1$. *Como este polinomio es irreducible, la teoría de Galois nos dice que el periodo será* $2^{\text{máx}(3,5)} - 1 = 31$, *es decir, podremos generar 31 polinomios diferentes de grado 5 reducción módulo 2. Antes de empezar la secuencia, necesitamos definir los primeros 5 bits. Se puede tomar, por ejemplo, 11111 como secuencia inicial. Fácilmente se puede generar el resto de la secuencia, donde cada bit es el coeficiente de uno de los términos de tales polinomios.*

En nuestro caso, las primeras 32 interacciones del método nos dan como resultado

$$11111000110111010100001001011001$$

Una vez tenemos la secuencia, necesitamos escoger un número l *que sea primo relativo con 32, como* $l = 17$. *Ahora simplemente tenemos que coger los 17 primeros bits e interpretárlos en decimal, obteniendo 30941. Para el siguiente número, hay que coger los 17 siguientes, teniendo en cuenta que esos 32 bits son circulares, es decir, si alcanzamos el final, tenemos que continuar desde el principio. Si repetimos el proceso descrito 32 veces, obtenemos:*

30941, 33970, 58229, 4811, 36308, 19247, 14160, 11452, 56642, 45809, 29961, 52166, 54309, 12059, 20630, 48238, 16985, 61882, 2405, 50922, 9623, 7080, 38494, 28321, 22904, 47748, 26083, 59922, 38797, 43083, 24119, 41260

Es evidente que estos números no pertenecen al intervalo $[0, 1]$, *pero es fácil transformarlos a ese intervalo dividiéndolos por el número máximo que se puede alcanzar, es decir,* $2^{17} - 1$.

1.3. Calidad de los PRNG

En la sección anterior hemos visto algunos ejemplos de PRNG. En cada ejemplo se ha hecho un razonamiento sobre cuánto vale el periodo, es decir, de cuántos números *distintos* pueden ser generados. El problema es que *distinto* no es lo mismo que *aleatorio*. Por ejemplo, supón dos PRNG de periodo 5 y dos secuencias generadas por cada uno de ellos: $\{3, 2, 4, 1, 5\}$ y $\{1, 2, 3, 4, 5\}$. La primera secuencia parece más aleatoria que la segunda. Por esa razón, es necesario definir una serie de herramientas que nos permitan medir la calidad de los números aleatorios generados, para posteriormente poder comparar generadores.

1.3.1. Test χ^2

El test *chi-cuadrado*, denotado por χ^2, es un test de contraste de hipótesis que puede ser usado para contrastar a partir de una muestra aleatoria simple de una variable X, si X sigue una distribución determinada. En este caso, como queremos comprobar si los números generados son efectivamente aleatorios, lo que debemos comprobar es si los datos siguen una distribución uniforme en el intervalo $[0, 1]$. Por lo tanto, las hipótesis del test serán:

- H_0: los datos proceden de una distribución $\mathcal{U}[0, 1]$.
- H_1: los datos no proceden de una distribución $\mathcal{U}[0, 1]$.

Como el test χ^2 es cualitativo, necesitamos clasificar los números generados en $d \in \mathbb{N}$ subconjuntos, denominados *clases*. A la cantidad de números que encontramos dentro de cada clase, se le llama *frecuencia observada de la clase* i, y se denota por N_i. Calcular estas frecuencias es fácil, ya que podemos realizar una homotecia del intervalo $[0, 1]$ al intervalo $[0, d]$, usando la función parte entera para obtener un índice válido: $C_i = \lfloor dx_i \rfloor$. Una vez tenemos la frecuencia observada, nos falta calcular la *frecuencia esperada*, E_i, que se puede calcular como $E_i = np_i$. Como queremos constrastar que los datos siguen una distribución uniforme, se tiene que $p_i = \frac{1}{d}$, luego $E_i = \frac{n}{d}$.

Una vez tenemos todos los datos necesarios, podemos usar el estadístico del test χ^2, que es sabido que sigue una distribución $\chi^2(d-1)$:

$$\chi^2 = \sum_{i=1}^{d} \frac{(N_i - E_i)^2}{E_i} \longrightarrow \chi^2(d-1)$$

Como las frecuencias esperadas son constantes, la expresión anterior puede simplificarse en términos de operaciones a realizar:

$$\chi^2 = \sum_{i=1}^{d} \frac{(N_i - E_i)^2}{E_i} = \frac{d}{n} \sum_{i=1}^{d} (N_i - E_i)^2 = \\ = \frac{d}{n} \left(\sum_{i=1}^{d} N_i^2 + \sum_{i=1}^{d} \frac{n^2}{d^2} - 2\frac{n}{d} \sum_{i=1}^{d} N_i \right) = \frac{d}{n} \sum_{i=1}^{d} N_i^2 - n \quad (1.9)$$

Para poder aplicar este test, es obligratorio que las frecuencias esperadas en cada clase sea mayor que 5, es decir, $E_i > 5$, luego elegiremos el número de clases d como el mínimo número de clases necesario para cumplir esa condición.

Por último, es necesario fijar un nivel de significación α (normalmente 0.05), y comprobar si el valor resultanto del estadístico pertenece al siguiente conjunto:

$$\{\chi^2 \leq \chi^2_{d-1,\frac{\alpha}{2}}\} \cup \{\chi^2 \geq \chi^2_{d-1,1-\frac{\alpha}{2}}\}$$

Ejemplo 1.3. *Vamos a usar el test χ^2 para analizar la aleatoriedad de la secuencia generada en el ejemplo 1.2, usando un generador LFSR. Supongamos que previamente los datos han sido transformados al intervalo* $[0, 1]$.

Lo primero es dividir el intervalo $[0, 1]$ *en d partes de forma que la frecuencia esperada en cada clase sea mayor a 5. Como la distribucion que queremos contrastar es la uniforme, la probabilidad de que un elemento caiga en una clase es la misma para todas,* $\frac{1}{d}$. *Por lo tanto podemos tomar* $d = 6$, *ya que* $n/d = 32/6 > 5$. *Obteniendo:*

i	*0*	*1*	*2*	*3*	*4*	*5*
N_i	*10*	*12*	*10*	*0*	*0*	*0*

Calculamos el valor del estadístico:

$$\chi^2 = \frac{d}{n}\sum_{i=1}^{d} E_i^2 - n = 32.5$$

La región crítica es

$$\{\chi^2 \leq \chi^2_{d-1,\frac{\alpha}{2}}\} \cup \{\chi^2 \geq \chi^2_{d-1,1-\frac{\alpha}{2}}\} = \{\chi^2 \leq 0.8312\} \cup \{\chi^2 \geq 12.8325\} \ni 32.5$$

Luego tenemos que rechazar la hipótesis nula, ya que el valor del estadístico se encuentra en la región crítica. Esto quiere decir que nuestro generador LFSR no ha generado números suficientemente aleatorios. Esto es debido a que hemos elegido un trinomio con grados muy pequeños.

Los resultados obtenidos tienen sentido ya que la mitad de las clases están vacías cuando deberían estar uniformemente distribuidas.

Una vez conocemos el principal test de contraste usado, podemos describir brevemente algunos de los métodos usados en *Diehard*.

1.3.2. Test de rachas

Esta prueba se realiza considerando a los números generados como dígitos. La prueba consiste en contar el número de dígitos que aparecen entre ocurrencias sucesivas de un mismo dígito. Por ejemplo, el número 834938 presenta un hueco de longitud cuatro entre los dos ochos.

La probabilidad de que aparezca cada uno de los tamaños de longitud i se obtiene con la siguiente expresión:

$$p_i = 0.1(0.9)^n \text{ para } i = 0, 1, 2, \ldots$$

Sin embargo, como teóricamente el valor del tamaño del hueco puede ser infinito, es conveniente agrupar las probabilidades para valores de i mayores o iguales a un determinado natural k, fijado arbitrariamente por el realizador del test.

Contabilizamos las fecuencias observadas N_i, con $i = 0, \ldots, k$ y las frecuencias esperadas $E_i = np_i$, con $i = 0, \ldots, k$. Una vez obtenidas ambas frecuencias, podemos calcular el valor del estadístico χ^2:

$$\chi^2 = \frac{d}{n} \sum_{i=1}^{d} E_i^2 - n \longrightarrow \chi^2(k)$$

Y construir un contraste para las hipótesis

- H_0: los datos proceden de una distribución $\mathcal{U}[0,1]$ y son independientes.
- H_1: los datos no proceden de una distribución $\mathcal{U}[0,1]$ y no son independientes.

1.3.3. Diehard Tests

George Marsaglia, publicó en 1996 [**marsaglia1996**] un conjunto de 16 test de aleatoriedad diferentes llamados *Diehard*. Cada uno de estos test mide una propiedad que deberían cumplir números aleatorios reales. Algunos hacen operaciones con bits, otros cuentan las permutaciones de longitud 5, el rango de ciertas matrices, etc. 11 de ellos usan el test χ^2 para constrastar la hipótesis. A continuación se describen algunos de los métodos:

Contar los 1

Primero se elige un número de bytes determinado, se cuentan los bits con valor 1 en cada uno de los bytes elegidos y se le asocia a cada posible valor del recuento una letra. Finalmente, se cuentan todas las apariciones de palabras de cinco letras.

Esferas aleatorias

Se escogen 4000 puntos aleatorios dentro de un cubo de lado 1000 y se centra una esfera en cada punto con radio la distancia al punto más cercano. El volumen de la esfera más pequeña debería seguir una ditribución exponencial con una media determinada.

Mínima distancia

Se escogen de forma aleatoria 8000 puntos en un cuadrado de lado 10000 y se calcula la distancia mínima entre las parejas. El cuadrado de esa distancia debería estar distribuida exponencialmente con una cierta media.

Bibliografía

[1] María Morales Giraldo Antonio Salmerón Cerdán. *Estadística Computacional*. 2001.

[2] James E Gentle. *Random number generation and Monte Carlo methods*. Springer Science & Business Media, 2006.

[3] Thomas E Hull y Alan R Dobell. "Random number generators". En: *SIAM review* 4.3 (1962), págs. 230-254.

[4] Derrick H Lehmer. "Mathematical methods in large-scale computing units". En: *Annu. Comput. Lab. Harvard Univ.* 26 (1951), págs. 141-146.

[5] George Marsaglia. "DIEHARD: a battery of tests of randomness". En: *http://stat. fsu. edu/geo* (1996).

[6] N Metropolis. *Beginning of the Monte Carlo Method. 1987*.

[7] Oystein Ore. *Number theory and its history*. Courier Corporation, 1988, pág. 280.

[8] Peter Shirley. *Ray Tracing in One Weekend*. 2020. URL: `https://raytracing.github.io/books/RayTracingInOneWeekend.html`.

[9] Peter Shirley. *Ray Tracing: The Next Week*. 2020. URL: `https://raytracing.github.io/books/RayTracingTheNextWeek.html`.

[10] Peter Shirley. *Ray Tracing: The Rest of Your Life*. 2020. URL: `https://raytracing.github.io/books/RayTracingTheRestOfYourLife.html`.

[11] L.H.C Tippett. *Random sampling numbers*. 1927.

[12] John W Tukey. *A Million Random Digits with 100,000 Normal Deviates*. 1955.

[13] John Von Neumann. "13. various techniques used in connection with random digits". En: *Appl. Math Ser* 12.36-38 (1951), pág. 5.

[14] Neal Zierler y John Brillhart. "On primitive trinomials (mod 2), II". En: *Information and Control* 14.6 (1969), págs. 566-569.