Generowanie liczb losowych

Seminarium: Algorytmy numeryczne i graficzne

Dawid Żywczak
Opracowano na podstawie [1], [2], [3].

6 kwietnia 2020

Spis treści

1 Wprowadzenie

Z pojęciem losowości jesteśmy dość dobrze zaznajomieni, często przecież mamy z nią do czynienia grając w karty, gry planszowe, czy obserwując miejsca uderzeń pioruna. Zastanówmy się najpierw jak możemy generować liczby losowe w naszym świecie. Można do tego celu użyć tzw. fizycznych generatorów liczb losowych. Najprostszym generatorem tego typu jest moneta. Generuje ona liczby o punktowym rozkładzie gdzie $p = \frac{1}{2}$. Kolejnym przykładem może być maszyna losująca znana z programu Toto Lotek. Jak łatwo sobie wyobrazić, generatory te nie są zbyt przydatne, gdy chcemy badać dużą próbkę danych. Istnieją również sprzętowe generatory liczb losowych, które korzystają z pewnych zjawisk fizycznych oraz ich własności (dla przykładu można tutaj podać rozpad izotopu promieniotwórczego). Obecnie są one wykorzystywane np. w kryptografii, gdyż są "lepszymi" generatorami, niż te programowe. Co oznacza "lepszy" generator liczb losowych dowiemy się w ostatnim rozdziale tego dokumentu. Tematem tej notatki będzie omówienie kilku sposobów na programowe generowanie liczb losowych, których tak często używamy pisząc nasze programy. Dla oszczędności papieru w kolejnych rozdziałach używać będziemy skrótu RNG (ang. random number generator). Gorąco zachęcam do przeglądania na bieżąco dołączonego pliku z implementacjami omawianych zagadnień.

2 Cel generatorów liczb losowych

Głównym celem generatorów losowych jest generowanie ciągów liczb naturalnych postaci $X_0, X_1, ..., X_n$, które następnie przekształcane są na liczby $U_0, U_1, ..., U_n$ z zakresu $[0, 1)$ o rozkładzie jednostajnym. Dlaczego tak? Wynika to faktu, że z rozkładu jednostajnego jesteśmy w stanie uzyskać wiele innych w wyniku nieskomplikowanych przekształceń. Dla przykładu: aby uzyskać rozkład normalny z jednostajnego, wystarczą nam dwie niezależne zmienne losowe X i Y. Definiując odpowiednio $Z_1 := \sqrt{-2 \ln X} \cos(2\pi Y)$ oraz $Z_1 := \sqrt{-2 \ln X} \sin(2\pi Y)$ otrzymamy dwie zmienne Z_0 oraz Z_1 o rozkładzie normalnym N(0,1) (używamy tak zwanej transformacji Boxa-Mullera). Więcej o sposobach transformacji rozkładu jednostajnego na inne dowiemy się w rozdziale 4.

3 Generatory o jednostajnym rozkładzie

3.1 Generatory multiplikatywne/mieszane

Jednym z pierwszych programowych RNG był zaproponowany przez J. von Neumanna w 1949 roku sposób nazywany algorytmem kwadratowym von Neumanna. Jego ideaa jest bardzo prosta i pozwala na generowanie naturalnych losowych liczb N-cyfrowych. Zdefiniujmy funkcję $F(X_n) = X_{n+1}$ gdzie funkcja F oblicza $Z = X_n^2$, jeżeli trzeba uzpełnia liczbę Z wiodącymi zerami, tak aby miała $2 \cdot N$ cyfr, a następnie wycina z liczby Z N środkowych cyfr. Niestety okazało się, że algorytm ten posiada stosunkowo krótki okres oraz jest podatny na zbieżność do 0 (Co zaobserwować możemy w pliku Generatory liczb losowych.ipynb).

Następnym krokiem było uogólnienie idei von Neumanna i zdefiniowanie generatorów liniowych

Definicja 1. *Generator liniowy to generator postaci:*

$$X_{n+1} = (a_1 X_n + a_2 X_{n-1} + ... + a_k X_{n-k+1} + c) \ mod \ m$$

Zdefiniujmy też pojęcie okresu

Definicja 2. *Niech* $P = \min\{i : X_i = X_0, i > 0\}$ *oraz* $v \in \mathbb{N}$ *wtedy jeśli* $\forall i$, $i \geq v$ *zachodzi* $X_i = X_{i+j \cdot P}$, $j = 1, 2, ...$ *to fragment ciągu* $X_0, X_1, ..., X_{v+P-1}$ *nazywamy okresem aperiodyczności ciągu, v parametrem aperiodyczności, a liczbę P okresem ciągu.*

W dalszych rozważaniach skupimy się na generatorach postaci:

$$X_{n+1} = (aX_n + c) \text{ mod m}$$
(Gdy c = 0 to mówimy o generatorze multipikatywnym, wpp. o generatorze mieszanym.)

ponieważ to one są głównie używane we współczesnych implementacjach generatorów w popularnych językach programowania.
Skoro już wiemy jaką postać mają nasze generatory, zastanówmy się nad ich własnościami. Pierwszym pytaniem nasuwającym się jest to, ile liczb losowych możemy wygenerować, zanim się potencjalnie zapętlimy? Szczegółowa odpowiedź na to pytanie nie jest prosta, ale zauważmy, że na pewno nie będzie to więcej niż m. Możemy też zauważyć, że dla m pierwszego okres generatora multiplikatywnego nie będzie większy niż $m - 1$. Naturalną jest chęć dążenia, do stworzenia generatorów posiadających jak największy okres. Tutaj z pomocą przychodzą nam:

Twierdzenie 1. *Generator multiplikatywny osiąga maksymalny okres P=m i tylko wtedy, gdy*
i) c jest wzlędnie pierwsze z m
ii) $b = a - 1$ *jest wielokrotnością p, dla każdego pierwszego p, dzielącego m*
iii) jeśli $m = 4 \cdot x$, *to* $b = 4 \cdot y$ *gdzie* $x, y \in \mathbb{Z}$

Twierdzenie 2 (C. F. Gauss, Disąuisitiones Arithmeticae (1801), §90-92)**.** *Najdłuższy możliwy okres w przypadku, kiedy c = 0, wynosi* $\lambda(m)$, *gdzie* $\lambda(m)$ *jest zdefiniowane wzorem:*
$\lambda(2) = 1,\ \lambda(4) = 2,\ \lambda(2^e) = 2^{e-2}\ ,\ e \geq 3;$
$\lambda(p^e) = p^{e-1}(p-1)\ ,\ p > 2;$
$\lambda(p_1^{e1}...p_n^{en}) = lcm(\lambda(p_1^{e1})...\lambda(p_n^{en}))$.
Okres ten jest osiągany jeśli
i) X_0 *jest względnie pierwsze z m;*
ii) a jest elementem pierwotnym modulo m.

Twierdzenie 3. *Liczba a jest elementem pierwotnym modulo p^e wtedy i tylko wtedy gdy zachodzi jeden z poniższych przypadków:*
i) $p = 2$, $e = 1$ oraz a jest nieparzyste;
ii) $p = 2$, $e = 2$ oraz a mod 4 = 3;
iii) $p = 2$, $e = 3$ oraz a mod 8 = 3, 5 lub 7;
iv) $p = 2$, $e \geq 4$ oraz a mod 8 = 3 lub 5;
v) p jest nieparzyste, $e = 1$, $a \neq 0$ (modulo p) oraz $a^{(p-1)/q} \neq 1$ (modulo p) dla dowolnego dzielnika pierwszego q liczby p — 1;
vi) p jest nieparzyste, $e > 1$, a spełnia warunki z punktu (v) oraz $a^{p-1} \neq 1$ (modulo p2).

Obszerne dowody poszczególnych twierdzeń odnaleźć można w pozycji [1] w rozdziale 3.2.1.2.
Jedynym z wniosków płynących z Twiedzenia 3. jest fakt, że dla $m = 2^e$ $e \geq 4$ oraz $a = 3$ mod 8 lub $a = 5$ mod 8 jest fakt, że maksymalny osiągalny okres to 2^{e-2}. Dlaczego o tym mówimy? Wynika to z faktu, iż często spotykane są generatory korzystające z tej własności. Dla przykładu w języku Java wykorzystywany jest generator o $m = 2^{48}$. Niestety pojawiają się też pewne problemy, mianowicie gdy m nie jest liczbą pierwszą, liczby Z_n uzyskane z liczb X_n, generowanych za pomocą omawianych RNG, przez obcięcie pewnej liczby początkowych bitów posiadają pewną własność okresowości. Okazuje się, że dla $m = 2^l$ i $c = 0$ końcowe bity tworzą ciąg o okresie 1. Np.
Jeśli a=8k+5, $X_0 = 4s+1$ oraz $m = 2^l$ to trzy najmłodsze bity tworzą ciąg o wartościach 001 i 101. (do zaobserwowania w pliku .ipynb). Dlatego generatory te nie są dobrym wyborem, kiedy chcemy końcowe bity traktować jako liczby losowe.

Następną własnością jest pewna struktura przestrzenna omawianych generatorów. Jak możemy zauważyć w dołączonym pliku .ipynb liczby generowane przez nasz generator układają się według pewnych schematów, zatem w ogólności nie możemy traktować ich jako "dobrych" generatorów dla zmiennych losowych wielowymiarowych.

3.2 Generatory oparte na rejestrach przesuwnych

Skoro w komputerze operujemy na bitach, naturalnym wydaje się wykorzystanie pewnych operacji bitowych oraz ich własności do generowania liczb losowych. W ten sposób wymyślone zostały tzw. generatory oparte na rejestrach przesuwnych, które korzystają z własności opreacji XOR. Niech k będzie liczbą naturalną, wtedy niech:

$$b_i = (a_1 b_{i-1} + ... + a_k b_{i-k}) \bmod 2,\ i = k+1, k+2, ...$$

Operację mod 2 na wartościach binarnych możemy zastąpić logicznym XOR, co daje nam dla współczynników $a_{j1} = ... = a_{jl} = 1$ (pozostałe współczynniki a są równe 0) inną postać wzoru przedstawionego wyżej:

$$b_i = (b_{i-j1}\ xor\ ...\ xor\ b_{i-jl})$$

W praktyce najczęściej spotyka się RNG, gdzie kolejne bity zdefiniowane są jako:

$$b_i = (b_{i-p}\ xor\ b_{i-q})\ (1)$$

Mając już wzór na poszczególne bity, możemy generować liczb poprzez ułożenie odpowiedniego ciągu L-bitowego. W ten sposób powstał tzw. generator Tauswortha, który generuje liczby z rozkładem U(0,1) zgodznie z formułą:

$$U_i = \sum_{j=1}^{L} 2^{-j} b_{is+j} \text{ gdzie s jest ustaloną liczbą naturalną}$$

Tak zdefiniowany ciąg korzysta z tych samych bitowych podciągów dla $s < L$, natomiast dla $s \leq L$ z rozłączynch. Warto dodać, że gdy jest względnie pierwsze z liczbą $2^k - 1$ to taki generator osiąga maksymalny ciąg, czyli $2^k - 1$. W załączonym pliku .ipynb zaimplementowanałem generator Tauswortha dla $0 < s \leq p - q$.

Jak to zwykle bywa, powstało uogólnienie omawianych w tym podrozdziale generatorów, nazywane *uogólnionymi generatorami opartymi na rejestrach przesuwnych*. Idea tego uogólnienia sprowadza się do wykorzystania liczb całkowitych $Y_i = b_i b_{i-l2} ... b_{i-lL}$ gdzie liczby l są ustalonymi parametrami przesunięcia. Wtedy dla schematu 1 otrzymujemy $Y_i = Y_{i-p}$ xor Y_{i-q}. Aby takie generatory działały poprawnie wymagają inicjalizacji ciągiem liczb (Y_0,...,Y_p).

3.3 Generatory Fibonacciego

Kolejnym typem generatorów liniowych jaki będziemy rozważać, są tak zwane generatory Fibonacciego. Mają one postać:

$$X_n = X_{n-2} + X_{n-1} \text{ mod m}$$

Korzystamy tutaj z faktu, że reszty dla ciągu takiej postaci, zachowują się w sposób bardzo chaotyczny (ciąg Fibonacciego). RNG tej postaci generują liczby przechodzące testy równomierności rozkładu (takie jak χ^2, który omówimy w dalszej części notatki), ale niestety nie spełniają testów niezależności. Tej wady próbowano pozbyć się zmieniając zależności ciągu na bardziej odległe wyrazy tzn.

$$X_n = X_{n-r} + X_{n-s} \text{ mod m } n \geq r,\ r > s \geq 1$$

co skutkowało stanowczym spowolnieniem procesu generowania liczb. Kolejnym krokiem, było zastąpienie opracji + przez inne. Tak uogólniony generator Fibonacciego oznaczamy przez F(r,s, $\circ$), co oznacza:

$$X_n = X_{n-r} \circ X_{n-s} \text{ mod m } n \geq r,\ r > s \geq 1$$

Generator F(p, q, xor) omawialiśmy w poprzednim rozdziale.

3.4 Generatory nieliniowe o jednostajnym rozkładzie

W poprzednich rozdziałach omawialiśmy RNG korzystające z liniowych przekształceń, ale nic nie stoi na przeszkodzie, żeby korzystać z operacji, który tej liniowości nie zachowują. Rozważmy generator zaproponowany przez Eichenauera i Lehna w 1986 roku. Ma on postać:

$$X_{n+1} = (aX_n^{-1} + b) \text{ mod m gdzie m jest liczbą pierwszą}$$

Założenie, że m jest liczbą pierwszą jest wymagane do szybkiego wyliczania wartości odwrotnej. Wtedy jest to 0 lub (korzystając z Małego Twierdzenia Fermata) X_n^{m-2}. Generatory tej postaci produkują wartości ze zbioru $\{0, 1,..., m-1\}$. Przekształcamy je na rozkład U(0,1) za pomocą wzoru $U_i = X_i/m$. Tak zdefiniowany generator osiąga maksymalny okres równy m, gdy m^2-1 jest najmiejszą liczbą całkowitą taką, że $z^{m^2-1} = 1 \text{ mod } (z^2 - bz - a)$.

Kolejną propozycją jest generator opublikowany w pracy Eichenauera-Hermanna w 1993 roku. Ma on postać:

$$X_{n+1} = (a(n + n_0) + b)^{-1} \text{ mod m}$$

Jak widać, każdy kolejny wyraz powstaje niezależnie od poprzedniego. RNG tej postaci osiąga maksymalny okres równy m dla $a \in \{1, 2,..., m-1\}$.

3.5 Kombinacje generatorów

Skoro znamy już podstawowe typy generatorów, nic nie stoi na przeszkodzie aby je łączyć, w celu uzyskania lepszych własności niż generatory składowe. Załóżmy, że mamy zmienne losowe X i Y określone na zbiorze S=$\{1, 2,..., n\}$ o rozkładach prawdopodobieństwa równych:

$$P(X = i) = p_i,\ P(Y = i) = q_i,\ \text{i=1, 2,..., n}$$

Zdefiniujmy teraz normę wektora t=$(t_1, t_2, ..., t_n)$ dla ustalonego p=1, 2,...,∞ jako:

$$\|t\| = (\textstyle\sum_{i=1}^{n} t_i^p)^{\frac{1}{p}}$$

Teraz dla zdefiniowanej wyżej zmiennej X wprowadźmy miarę podobieństwa do rozkładu jednostajnego na zbiorze S:

$$\delta(X) = \|(p_1, p_2, ..., p_n) - (\tfrac{1}{n}, \tfrac{1}{n}, ..., \tfrac{1}{n})\|$$

Okazuje się, że dla działań $\circ$, których tabelka tworzy kwadrat łaciński, czyli np. $+$, $-$, $*$, xor, rozkład zmiennej X $\circ$ Y jest bardziej podobny do rozkładu jednostajnego na S, niż rozkłady zmiennych składowych tzn.:

$$\delta(X \circ Y) \leq min\{\delta(X), \delta(Y)\}$$

A to oznacza, że ciąg $(X_1 \circ Y_1, X_2 \circ Y_2, ...)$ powinien być bardziej równomiernie rozłożony niż każdy z ciągów składowych. Dodatkowo jeżeli ciąg X ma okres P_1 oraz ciąg Y ma okres P_2 oraz P_1 i P_2 są względnie pierwsze, to okres ciągu wynikowego wynosi $P_1 \cdot P_2$. Dla przykładu weźmy generatory nieliniowe omawiane w poprzednim rozdziale. Przypuśćmy, że mamy $r \geq 5$ takich generatorów $X_n^{(j)}$, j=1,...,r z parametrami m_j które są liczbami pierwszymi. Zdefiniujmy nowy ciąg:

$$U_n^{(j)} = X_n^{(j)}/m_j$$
$$U_n = (U_n^{(1)} + U_n^{(2)} + ... + U_n^{(r)}) \text{ mod 1, n=0, 1, 2,..}$$

Wtedy okres tego ciągu wynosi m=$m_1 \cdot m_2 \cdot ... \cdot m_r$, co pozwala na tworzenie generatorów o bardzo duży okresach.

4 Generatory o dowolnych rozkładach

Omówiliśmy już podstawowe techniki generowana liczb o rozkładzie U(0,1). W tym rozdziale omówimy kilka technik zamiany tego rozkładu na inne. W tym celu rozważymy najpierw metody dla rozkładów ciągłych, następnie dla dyskretnych oraz kilka algorytmów uzskiwania popularnych rozkładów.

4.1 Metody dla rozkładów ciągłych

4.1.1 Metoda odwracania dystrybuanty

Ogólnie metoda odwracania dystrybuanty korzysta z faktu, że dla zmiennej losowej U o rozkładzie jednostanym U(0,1) oraz ściśle rosnącej i ciągłej dystrybuanty F, zmienna X=$F^{-1}(U)$ ma rozkład prawdopodobieństa o dystrybuancie F, ponieważ

$$P(X \leq x) = P(F^{-1}(U) \leq x) = P(U \leq F(x)) = F(x)$$

Zatem aby wygenerować zmienną losową o dystrybuancie F, wystarczy wygenerować ciąg U i zmienić go na ciąg $F^{-1}(U)$. Przypadek można uogólnić na przypadek dowolnych, niekoniecznie ciągłych i ściśle rosnących dystrybuant F. Wtedy

$$F^{-1}(t) = \inf\{x : t \leq F(x)\}$$

i postępować tak jak wyżej.

Twierdzenie 4. *Niech F będzie dystrybuantą pewnego rozkładu prawdopodobieństwa. Jeżeli U jest zmienną losową o rozkładzie U(0,1), to zmienna losowa X=inf{x:U≤F(x)} ma rozkład o dystrybuancie F.*

Dowód. Udowodnimy, że zdarzenie losowe $X \leq t$ zachodzi wtedy i tylko wtedy, gdy zachodzi zdarzenie $U \leq F(t)$. Załóżmy, że zachodzi zdarzenie $X \leq t$. Niech X=t. Z definicji zmiennej losowej X wynika, że dla każdego m=1, 2,... istnieje punkt x_m spełniający warunki $t \leq x_m < t+1/m$ oraz $U \leq F(x_m)$. Gdy $m \to +\infty$, wtedy t≤ $x_m \to t$ (z prawej strony) a stąd, wobec prawostronnej ciągłości dystrybuanty, otrzymujemy nierówność $U \leq F(t)$
Niech $X < t$. W zbiorze $\{x : U \leq F(x)\}$ istnieje zatem punkt $y < t$. Wtedy oczywiście F(y)≤F(t) oraz, z racji przynależności punktu y do zbioru {x:U≤F(x)}, mamy $U \leq F(y)$. Czyli $U \leq F(t)$.
Przypuśćmy, że $U \leq F(t)$, zatem t należy do zbioru $\{x : U \leq F(x)\}$, czyli $t \geq \inf\{x : t \leq F(x)\} = X$. □

Niestety odwracanie dystrybuanty może być dość skomplikowane numerycznie, przez co metody tej nie stosuje się zbyt często. Ale na przykład dla rozkładu wykładniczego o gęstości $f(x) = e^{-x}$, $x \geq 0$ wtedy ciąg $X_n = -\ln U_n$ jest ciągiem liczb losowych o rozkładzie z gęstością $f(x)$.

4.1.2 Metoda eliminacji

Teraz omówimy metodę, której nazwa wzięła się od odrzucania części wyników niespełniających założeń, omówionych w dalszej części rozdziału. Niech f będzie gęstością prawdopodobieństwa rozkładu, który chcemy otrzymać, dodatnią na przedziale (a,b), równą 0 poza przedziałem oraz ograniczoną pewną stałą $d > 0$. Wtedy metoda o podanym niżej schemacie wygeneruje liczby losowe o rozkładzie z gęstością f. Schemat to:
1. Wygeneruj dwie niezależne zmienne losowe U_1 i U_2 o rozkładach jednostajnych U(a,b) i U(0,d)
2. Jeżeli $U_2 \leq f(U_1)$, to $X = U_1$ wpp. odrzucić parę i wygenerować nową tak jak w punkcie 1.
O poprawności powyższej metody świadczą twierdzenia, które zaraz przytoczę, ale najpierw:

Definicja 3. *Niech $C = [a, b] \times [c, d]$ Wtedy przez $l_2(C) = (a - b)(c - d)$ oznaczamy miarę Lebesgue'a na płaszczyźnie zbioru C.*

Teraz niech $(a, b) \times (0, d)$ oznacza prostokąt o wysokości $d > 0$, zbudowany na odcinku (a, b). Wtedy $l_2(a, b) \times (0, d)) = (b - a)d$. Niech $A = \{(x, u) : a \leq x \leq b, 0 \leq u \leq f(x)\}$. Wtedy $l_2(A) = 1$, ponieważ $l_2(A)$ to pole powierzchni pod wykresem funkcji gęstości f. Mówimy, że punkt losowy ω ma rozkład równomierny na zbiorze $D \subset R^2$, jeżeli dla podzbiorów C tego zbioru zachodzi $P(\omega \in C) = l_2(C)/l_2(D)$

Twierdzenie 5. *Niech $(X_1, U_1), (X_2, U_2), ...$ będzie ciągiem punktów losowych o rozkładzie równomiernym na prostokącie $(a, b) \times (0, d)$ i niech (X,U) będzie pierwszym punktem tego ciągu, który wpada do zbioru A. Wtedy punkt losowy (X,U) ma rozkład równomierny na zbiorze A.*

Dowód. Rozważmy pozdbiór B zbioru A i niech $l_2(B)$ oznacza jego pole powierzchni. Chcemy udowodnić, że $P((X, U) \in B) = l_2(B)/l_2(A)$. Zatem

$$P((X,U) \in B) = \sum_{i=1}^{\infty} P((X_1,U_1) \notin A, ..., (X_{i-1},U_{i-1}) \notin A, (X_i,U_i) \in B) = \sum_{i=1}^{\infty}(1 - \frac{l_2(A)}{(b-a)d})^{i-1} \frac{l_2(B)}{(b-a)d} = (\text{z sumy szeregu geometrycznego}) \frac{l_2(B)}{l_2(A)}$$

□

Twierdzenie 6. *a) Jeżeli U ma rozkład jednostajny U(0,1), X ma rozkład o gęstości f(x) oraz X i U są niezależne, to punkt losowy (X, U(f(X)) ma rozkład jednostajny na zbiorze A.*
b) Jeżeli punkt losowy (X, U) ma rozkład jednostajny na zbiorze A, to zmienna losowa X ma rozkład o gęstości f(x).

Dowód. a) Jeżeli U ma rozkład równomierny U(0,1), to dla każdego ustalonego x zmienna losowa $V = Uf(x)$ ma rozkład równomierny $U(0, f(x))$. Dla ustalonego x i dla danego zbioru $B \subset A$ oznaczamy B_x=$\{u : (x, u) \in B\}$. Wtedy

$$P((X, Uf(X)) \in B) = \int (\int_{B_x} \frac{dv}{f(x)}) f(x)dx = \int \int_B dvdx = l_2(B) = \frac{l_2(B)}{l_2(A)}$$

czyli (X, Uf(X)) ma rozkład jednostajny na zbiorze A.
b) Oznaczmy $A_t = \{(x,u) : a \leq x \leq t, 0 \leq u \leq f(x)\}$. Wtedy

$$P(X \leq t) = P((X,U) \in A_t) = \int_a^t \int_0^{f(x)} \frac{dudx}{l_2(A)} = \int_a^t f(x)dx$$

czyli f(x) jest gęstością zmiennej losowej X. □

Omawiana wyżej metoda i twierdzenia ją uzsadniające, działają dla punktów losowych o dwuwymiarowym rozkłazie jednostajnym na prostokącie o podstawie (a,b) i wysokości d. Teraz spróbujemy uogólnić metodę eliminacji do dowolnie wielowymiarowych zmiennych losowych X i dowolnych obszarów zawierających funkcję gęstości o odpowiednich wymiarach.

Twierdzenie 7. *Przypuśćmy, że $\xi_1, \xi_2, ...$ jest ciągiem niezależnych zmiennych losowych o jednakowym rozkładzie w $\mathbb{R}^k$. Niech A będzie zbiorem w $\mathbb{R}^k$, takim że $P(\xi_1 \in A) > 0$ oraz niech η oznacza element losowy równy pierwszemu elementowi ξ_n, przyjmującemy wartość w zbiorze A. Wtedy dla $B \subset \mathbb{R}^k$ zachodzi równość*

$$P(\eta \in B) = \frac{1}{p} P(\xi_1 \in A \cap B)$$

gdzie $p = P(\xi_1 \in A)$. W szczególności, jeżeli $\xi_1, \xi_2, ...$ są niezależnymi punktami losowymi o rozkładzie jednostajnym na pewnym zbiorze $\Omega \supset A$, to η ma rozkład równomierny na zbiorze A.

Twierdzenie 8. *a) Jeżeli X jest punktem losowym o rozkładzie z gęstością g w k-wymiarwoej przestrzeni $\mathbb{R}^k$, U jest zmienną losową o rozkładzie jednostajnym U(0,1) i X oraz U są niezależne, to punkt losowy $(X, cUg(X))$ ma rozkład jednostajny na zbiorze $\Omega = \{(x,u) : x \in \mathbb{R}^k, 0 \leq u \leq cg(x)\} \subset \mathbb{R}^{k+1}$ gdzie $c > 0$ jest dowolną stałą.*
b) Odwrotnie: jeżeli punkt losowy (X, U) ma rozkład równomierny na zbiorze Ω to XX ma rozkład o gęstości g.

Twierdzenia te prowadzą nas do uogólnionego schematu metody eliminacji:
1. Wybierz gęstośc g, żeby generowanie liczb losowych o tej gęstości było łatwe i szybkie oraz wyznacz stałą $c > 0$, taką żeby $f(x) \leq cg(x)$ dla wszystkich x.
Ze względu na ten warunek gęstość g, będziemy nazywać dominującą. Za obszar Ω z twierdzenia 7 przyjąć $\Omega = \{(x,u) : x\mathbb{R}^k, 0 \leq u \leq cg(x)\}$.
2. Wygeneruj punkt losowy X o rozkładzie z gęstością g oraz liczbę losową U U(0,1). Wtedy na mocy pierwszej tezy twierdzenia 8, punkt losowy (X, cUg(X)) ma rozkład jednostajny na zbiorze Ω.
3. Powtarzać generowanie według p. 2, dopóki kolejno wygenerowany punkt nie wpadnie do zbioru $A = \{(x,u) : x \in \mathbb{R}^k, 0 \leq u \leq f(x)\}$, tzn. dopóki nie zostanie spełniony warunek akceptacji

$$U \leq \frac{f(X)}{cg(X)}$$

Teraz na mocy twierdzenia 7 uzyskany punkt ma rozkład jednostajny na A, więc na mocy twierdzenia 8 X ma rozkład o gęstości f(x). Zastanówmy się jak wyznaczyć stałą c, tak aby warunek akceptacji był jak najszybciej osiągany. Możemy to osiągnąć przez

dobranie wartości c takiej, żeby prawdopodobieństwo spełnienia warunku akceptacji było jak największe tzn.

$$P(Ucg(X) \leq f(X)) = \int_{\mathbb{R}^k} g(x)dx \int_0^{f(x)/cg(x)} du = \frac{1}{c}$$

Optymalną wartością powyższego warunku jest $c = \sup_x \frac{f(x)}{g(x)}$. Często gęstość f możemy rozpisać jako iloczyn stałej i innych, prostszych gęstości. Wtedy nasz algorytm, przyjmuje prostszą formę. Dla przykładu:
dla gęstości $f(x) = cp(x)(1-q(x))$ gdzie p(x) to gęstość, a q(x) to dystrybuanta pewnego rozkładu prawdopodobieństwa, to nasz algorytm ma postać
Repeat
    Generuj X o rozkładzie z gęstością p
    Generuj X o rozkładzie z dystrybuantą q
until
    Y > X
Return X
Więcej przykładów takich uproszczeń znaleźć można w [2], str. 52-54.

Często zdarza się, że obliczenie prawej strony warunku akceptacji, może być bardzo czasochłonne, co przy generowaniu ogromnej ilości liczb losowych jest efektem wysoce nieporządanym. Wtedy z pomocą przychodzą nam warunki szybkiej akceptacji i eliminacji. Pomysł ten polega na znalezieniu prostych funckji $\alpha(x)$ i $\beta(x)$ bliskich wartości prawej strony warunku

$$\alpha(x) \leq \frac{f(x)}{cg(x)} \leq \beta(x) \text{ dla wszsytkich x}$$

Zdarzenie $U \leq \alpha(X)$ jest nazywane warunkiem szybkiej akceptacji i skutkuje zaakceptowaniem wygenerowanej liczby, natomiast zdarzenie $U > \beta(X)$ to warunek szybkiej eliminacji i prowadzi do odrzucenia wygenerowanej liczby losowej. Jeżeli żaden z nich nie jest spełniony, musimy obliczyć wartość $\frac{f(x)}{cg(x)}$ ale dla dobrze dobranych funkcji α i β dzieje się to bardzo rzadko. Pojęcie to można uogólnić na ciąg funkcji α i β takich, że

$$\alpha_k(x) \leq ... \leq \alpha_1(x) \leq \frac{f(x)}{cg(x)} \leq \beta_1(x) \leq ... \leq \beta_l(x)$$

4.2 Metody dla rozkładów dyskretnych

Omówiliśmy już podstawowe metody dla przypadków ciągłych. Następnie przedstawione przypadki zmiennych losowych o dyskretnych rozkładach prawdopodobieństwa.

4.2.1 Metoda odwracania dystrybuanty

Metoda odwracania dystrybuanty dla przypadku dyskretnego nie rózni się wiele od wersji ciągłej. Korzystając z obserwacji poczynionych wcześniej możemy zauważyć, że liczby losowe $X_1, X_2, ...$ o rozkładzie dyskretnym $p_k = P(X = k)$, $k = 0, 1, 2, ...$ mogą być generowane przez przekszatłcenie ciągu U za pomocą wzoru

$$X_n = \min\{k : U_n \leq \sum_{i=0}^{k} p_i\},\ n = 1, 2, ...$$

Co prowadzi nas do prostego algorytmu
X=0, S=p_0
Generuj U o rozkładzie jednostajnym U(0,1)
    While $U > S$
        do $X = X + 1$, $S = S + p_x$
Return X
Okazuje się, że warto posortować prawdopodobieństwa p_k w kolejności rosnącej, co pozwala uniknąć możliwej wady tego prostego algorytmu, czyli bardzo długiego czasu wykonania.

4.2.2 Metoda równomiernego rozbicia przedziału (0, 1)

Jak sama nazwa wsakzuje, będziemy rozbijać przedział (0,1) na jednakowo długie podprzedziały, sprawdzać do którego padła wygenerowana liczba U, a następnie identyfikować wartośc generowanej zmiennej X ograniczając się do tego małego przedziału. Rozważmy zmienną losową X przyjmującą skończoną liczbę różnych wartości $P(X = k) = p_k$, $k = 0, 1, ..., K$. Podzielmy przedział (0,1) na K+1 równych podprzedziałów $(\frac{i-1}{K+1}, \frac{i}{K+1})$ i umówmy się, że odcinek $(0, \frac{1}{K+1})$ ma numer 1. Wtedy zmienna U wpada do przedziału o numerze $\lfloor (K+1)U + 1 \rfloor$. Dla podanego rozkładu zdefiniujmy ciąg $q_i = \sum_{j=0}^{i} p_j$, $i = 0, 1, ..., K$ i przyjmijmy $q_{-1} = 0$. Następnie tworzymy pomocniczy ciąg $g_i = max\{j : q_j < \frac{i}{K+1}\}$, $1 \leq i \leq K+1$. Przygotowanie się opłaciło, ponieważ nasz algorytm ma teraz bardzo prostą postać:
Generuj U o rozkładzie jednostajnym U(0,1)
    $X = \lfloor (K+1)U + 1 \rfloor$
    $X = g_X + 1$
    While $q_{X-1} > U$ do $X = X - 1$
Return X
O przydatności tego algorytmu mówi poniższe twierdzenie

Twierdzenie 9. *Oczekiwana liczba porównań $q_{X-1} > U$ jest nie większa od 2.*

Dowód. Zauważmy, że liczba porównań C nie może być większa niż liczba wartości q_i w przedziale X (naszej zwracanej wartości) plus 1. Ale wiemy, że każdy przedział jest równej długości i wybieramy go z równym prawdopodobieństwem, więc

$$C \leq 1 + \frac{1}{K+1} \sum_{i=0}^{K} (\text{liczba wartości } q_j \text{ w i-tym przedziale}) \leq 1 + 1 = 2$$

□

5 Rozkłady wielowymiarowe

Do tej pory generowaliśmy liczby o rozkładach jednowymiarowych, często jednak w naszych obliczeniach potrzebujemy zmiennych losowych o wielu wymiarach. W tym rozdziale omówimy przykład generowania takich zmiennych z rozkładem normalnym. Zanim jednak zaczniemy chciałbym wspomnieć o generowaniu zmiennych wielowymiarowych z

rozkładem jednostajnym U(Ω). Można to osiągnąć przez zdefiniowanie zbioru Ω jako $\prod_{j=1}^{m}[a_j, b_j]$ oraz wprowadzenie miary Lebesgue'a $l_m(\Omega)$ tak jak robiliśmy to wcześniej. Wtedy dla m wymiarowej zmiennej możemy generować liczby losowe U_j z rozkładem $U(a_j, b_j)$ i akceptować $U = (U_1, ..., U_m)$ jako X, gdy $U \in \Omega$. Niestety nie jest to zbyt dobre rozwiązanie, lecz jedyne pozwalające na taką ogólność.

5.1 Wielowymiarowy rozkład normalny

Na początku notatki omówiliśmy po krótce transofrmację Boxa-Mullera, która pozwalała na generowanie dwóch zmiennych losowych o rozkładzie normalnym N(0,1). Co jeżeli chcielibyśmy skorzystać z wielowymiarowego rozkładu normalnego, gdzie poszczególne zmienne losowe są ze sobą skorelowane? Wtedy niech

$$\text{A}= \begin{bmatrix} \sigma_{1,1} & ... & \sigma_{1,n} \\ ... & ... & ... \\ \sigma_{n,1} & ... & \sigma_{n,n} \end{bmatrix}$$

Będzie macierzą kowariancji n-wymiarowej zmiennej losowej $X = (X_1, ..., X_n)$ o rozkładzie normalnym t.że $EX_i = 0$ dla każdego $i = 1, 2, ..., n$. Teraz jesli $Z = (Z_1, ..., Z_n)$ oraz każda składowa Z jest niezależna i ma taki sam rozkład N(0,1), to zmienna losowa CZ, gdzie C jest pewną nieosobliwą macierzą, ma rozkład normalny z macierzą kowariancji CC^T. Zatem aby wygenerować n-wymiarową zmienną losową X z daną macierzą kowariancji A wystarczy skonstruować macierz C taką, że $CC^T = A$, wygenerować n niezależnych zmiennych losowych Z_i o jednakowym rozkłazie N(0,1) i obliczyć $X = CZ$. Nasuwa się pytanie w jaki sposób efektywnie wyznaczyć macierz C? Można to zrobić w następujący sposób

$$c_{i,1} = \frac{\sigma_{i,1}}{\sqrt{\sigma_{1,1}}}$$
$$c_{i,i} = (\sigma_{i,i} - \sum_{r=1}^{i-1} c_{i,r}^2)^{1/2}$$
$$c_{i,j} = \sigma_{j,j}^{-1}(\sigma_{i,j} - \sum_{r=1}^{j-1} c_{i,r}c_{j,r}) \text{ gdy } i > j$$
$$c_{i,j} = 0 \text{ gdy i<j}$$

6 Testowanie poprawności generatorów

Testowanie poprawności naszych generatrów sprowadzać się będzie do testowania poprawności RNG generujących liczby o rozkładzie jednostajnym U(0,1), ponieważ to z nich otrzymujemy inne rozkłady. Metodologia jest dosyć prosta. Będziemy testować hipotezę, że wykorzystywany generator spełnia pewne własności, których od niego oczekujemy. W tym celu będziemy obliczać wartość tzw. statystyki testowej i porównywać ją z wartościami krytycznymi stosowanego testu. Pozytywny wynik testu (czyli fakt, że obliczona wartość nie przekracza wartości krytycznej) przemiawia za tym, że RNG spełnia badane własności. Oczywiście jednorazowe przeprowadznie takiego testu nie jest zbyt dobrym podejściem, dlatego będziemy wykonywać je w seriach. Dostajemy więc ogólny schemat testowania generatora liczb losowych:

1. ustalamy liczbę n i startując z losowo wybranej liczby początkowej, generujemy n kolejnych liczb;
2. obliczamy wartość statystyki testowej - oznaczmy ją przez T;
3. obliczamy wartość F(T), gdzie F jest dystrybuantą rozkładu statystyki T, gdy weryfikowana hipoteza jest prawdziwa; 4. powtarzamy powyższą oprację N razy, jeżeli ciąg $F(T_1), ..., F(T_N)$ jest ciągiem niezleżnych zmiennych losowych o rozkładzie U(0,1), to nasz generator spełnia testowane własności. Oczywiście naprojstszym testem jest np. wyliczenie średniej arytmetycznej ze wszystkich wygenerowanych liczb (o rozkładzie U(0,1)) jej wartość nie powinna znacznie odbiegać od $\frac{1}{2}$. Zastosowanie omawianych niżej testów znaleźć można w dołączonym pliku .ipynb. W tym rozdziale opowiem o kilku metodach testowania, osoby zainteresowane mogą znaleźć ich więcej w [2] w rozdziale 5.

6.1 Test χ^2

Test χ^2 jest jednym z najczęściej stosowanych testów zgodności oraz można go stosować do dowolnych rozkładów prawdopodobieństwa przez co jest bardzo dobrze opisany i dostępny w wielu bibliotekach statystycznych. Hipoteza statystyczna tego testu ma bardzo ogólną postać: zmienna loswa X ma rozkład prawdopodobieństwa o dystrybuancie F. Niech a oraz b będą liczbami takimi, że $F(a) = 0$ oraz $F(b) = 1$. Niech $a = a_0 < a_1 < ... < a_k = b$ będzie rozbiciem zbioru wartości zmiennej losowej X i niech $p_i = P(a_{a-1} < X \leq a_i)$, $i = 1, 2, ...$ Oznaczmy przez n_i liczbę elementów X ciągu $X_1, ..., X_n$, które spełniają warunek $a_{i-1} < X \leq a_i$. Wtedy statystyką naszego testu jest

$$\phi = \sum_{i=1}^{k} \frac{(n_i - np_i)^2}{np_i}$$

Okazuje się, że gdy n jest duże, a rozbicie takie, że liczby np_i nie są zbyt bliskie 0, statystyka ϕ ma w przybliżeniu rozkład chi-kwadrat o (k-1) stopniach swobody. W gwoli wyjaśnienia - rozkład chi-kwadrat o k stopniach swobody to nic innego jak rozkład zmiennej $Y = \sum_{i=1}^{k} (Z_i)^2$, gdzie zmienne losowe Z_i są niezależne i każda ma rozkład normalny N(0,1). W przypadku, gdy testujemy generatory mamy dużą swobodę co do wyboru parametrów, zatem nic nie stoi na przeszkodzie, żeby podzielić zbiór wartości zmiennej losowej X na k przedziałów o równym prawdopodobieństwie $p_i = \frac{1}{k}$. Wtedy nasza statystyka przyjmuje postać

$$\phi = \frac{k}{n} \sum_{i=1}^{k} (n_i^2 - n)$$

Jeżeli obliczona wartość statystyki jest mniejsza lub równa wartości krytycznej dla danej liczby stopni swobody, to generator przechodzi test.

6.2 Test Kołomogorowa

Test Kołomogorowa, podobnie jak test chi-kwadrat, bada czy zmienna losowa X ma rozkład prawdopodobieństwa o dystrybuancie F, z tą różnicą, że stosować go możemy tylko i wyłącznie do rozkładów ciągłych. Idea polega na zbadaniu ódległościempirycznej dystrybuanty, od dystrybuanty hipotetycznej. Dystrybuanta emipiryczna zdefiniowana jest wzorem

$$F_n(x) = \frac{1}{n}\sum_{j=1}^{n} 1_{(-\infty,x]}(X_j)$$
gdzie $1_{(a,b)}$ oznacza funkcję charakterystyczną zbioru (a, b).

Zatem nasza statystyka ma postać

$$D_n = \sup_{-\infty<x<+\infty} |F_n(x) - F(x)|$$

Jeżeli próba $X_1, X_2, ..., X_n$ pochodzi z rozkładu o dystrybuancie F, to $D_n \to 0$ z prawdopodobieństem 1. Duże wartości statystyki przemawiają oczywiście za odrzuceniem hipotezy. Nie będę podawał tutaj wzorów na wyliczanie wartości krytycznych testu, ponieważ są już one obliczone, stablicowane i dostępne w większości pakietów statystycznych.

6.3 Test pokerowy

Test pokerowy jest jednym z testów niezależności próby. Bada on hipotezę, że zmienna losowa $(X_1, X_2, ..., X_n)$ ma rozkład o dystrybuancie będącej iloczynem dystrybuant poszczególnych zmiennych składowych. Schemat testu brzmi następująco:

Niech ciąg $X_1, X_2, ..., X_n$ będzie ciągiem liczb wygenerowanym przez pewien generator z dystrybuantą F. Podzielmy zbióra wartości zmiennych X na k rozłącznych, jednakowo długich przedziałów. Wtedy jeśli zmienne losowe X_j mają rzeczywiście rozkład o dystrybuancie F, to dla każdego przedziału (a_{i-1}, a_i) zachodzi $P(a_{i-1} < X_j \leq a_i) = \frac{1}{k}$. Utwórzmy teraz ciąg zmiennych

$$Y_j = i \text{ jeśli } X_j \in (a_i, a_{i+1})$$

Zauważmy, że przyjmuje on tylko wartości $0, 1, 2, ..., k-1$, każdą z jednakowym prawdopodobieństwem. Podzielmy teraz ciąg Y na pięcioelementowe krotki. Tak zbudowany ciąg zbudowany jest z k^5 różnych krotek. Wyróżniać będziemy następujące typy elementów (podobnie jak w pokerze):
1. abcde (bust)
2. aabcd (para)
3. aabbc (dwie pary)
4. aaabc (trójka)
5. aaabb (full)
6. aaaab (czwórka)
7. aaaaa (piątka)

Teraz korzystając z założenia, że każda z liczb $0, 1, ..., k-1$ występuje w ciągu Y z jednakowym prawdopodobieństwem oraz z faktu, że poszczególne wyrazy są niezależne, prawdopodobieństwo wystąpienia poszczególnych typów piątek wyraża się wzorami:

$$P((abcde)) = \begin{cases} (k-1)(k-2)(k-3)(k-4)/k^4 & \text{dla k} \geq 5 \\ 0 & \text{dla k} < 5 \end{cases}$$

$$P((aabcd)) = \begin{cases} 10(k-1)(k-2)(k-3)/k^4 & \text{dla k} \geq 4 \\ 0 & \text{dla k} < 4 \end{cases}$$

$$P((aabbd)) = \begin{cases} 15(k-1)(k-2)/k^4 & \text{dla k} \geq 3 \\ 0 & \text{dla k} < 3 \end{cases}$$
$$P((aaabc)) = \begin{cases} 10(k-1)(k-2)/k^4 & \text{dla k} \geq 3 \\ 0 & \text{dla k} < 3 \end{cases}$$
$$P((aaabb)) = \begin{cases} 10(k-1)/k^4 & \text{dla k} \geq 2 \\ 0 & \text{dla k} < 2 \end{cases}$$
$$P((aaaab)) = \begin{cases} 5(k-1)/k^4 & \text{dla k} \geq 2 \\ 0 & \text{dla k} < 2 \end{cases}$$
$$P((aaaaa)) = \begin{cases} 1/k^4 & \text{dla k} \geq 2 \\ 0 & \text{dla k} < 2 \end{cases}$$

W praktyce najczęściej stosuje się wartości $k = 2, 8, 10$. Zgodność rozkładu krotek sprawdza się najczęściej za pomocą standardowego testu chi-kwadrat.

Na tym kończymy naszą podróż po świecie programowych generatorów liczb losowych. Gorąco zachęcam do przetestowania omawianych zagadnień w dołączonym pliku .ipynb, a osoby zainteresowane głębiej tematyką RNG, polecam przestudiowanie książek na podstawie, których opracowana została ta notatka.

Literatura

[1] Donald E. Knuth *Sztuka programowania, wydanie trzecie, tom 2: Algorytmy seminumeryczne*. Warszawa, 2002.

[2] Robert Wieczorkowski, Ryszard Zieliński, *Komputerowe generatory liczb losowych*. Warszawa, 1997.

[3] Ryszard Zieliński, *Generatory liczb losowych*. Warszawa, 1972.