Sprawozdanie 1
Modelowanie i Identyfikacja

Maciej Kajdak
Nr indeksu: 226256

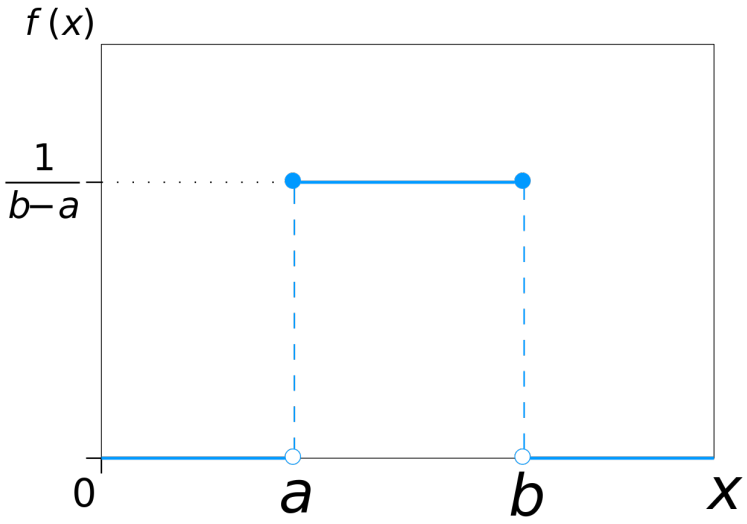
25 marca 2019

1 Wstęp

Przy generowaniu liczb losowych w najprostszym przypadku można posłużyć się tzw. generatorami fizycznymi. Do nich można zaliczyć chociażby rzut monetą (losowanie liczb ze zbiory $0, 1$) lub pobieranie co pewien kwant czasu próbki z przebiegu pewnego szumu elementu elektronicznego. Takie rozwiązania są jednak pod wieloma względami niepraktyczne. Postęp, któremu z biegiem lat ulega nauka i technologia pozwala na coraz coraz wydajniejsze programowe generowanie liczb losowych. Do celu generowania liczb można wykorzystać np. generatory liczb losowych o rozkładzie równomiernym. Wśród nich najczęściej spotykane są generatory liniowe, nieliniowe, Fibonacciego lub generatory oparte na rejestrach przesuwnych. Chcąc wygenerować liczby losowe o dowolnym rozkładzie prawdopodobieństwa również można posłużyć się algorytmami powszechnie znanymi, takimi jak np. metoda odwracania dystrybuanty albo metoda eliminacji. Ważnym aspektem generacji liczb losowych jest także testowanie generatorów. Dzięki temu możliwe jest ocenianie jakości generatora pod konkretnym kątem. W niniejszym sprawozdaniu opisane zostały wyniki badań, którym zostały poddane różne algorytmy generowania liczb losowych. Generatory implementowano w środowisku Python w wersji 3.7.1 [3] wraz z pakietami numpy w wersji 1.15.4 [2] oraz matplotlib w wersji 3.0.2 [1]. Wszystkie stworzone w ramach laboratoriów skrypty można znaleźć w repozytorium twórcy w serwisie Github

2 Generator liczb losowych o rozkładzie jednostajnym

Jednostajny (równomierny) rozkład prawdopodobieństwa na przedziale $[a, b]$ charakteryzuje się tym, że jego gęstośc na tym przedziale jest stała i różna od zera, a poza wspomnianym przedziałem przyjmuje wartości 0 jak zostało to przedstawione na rysunku 1.

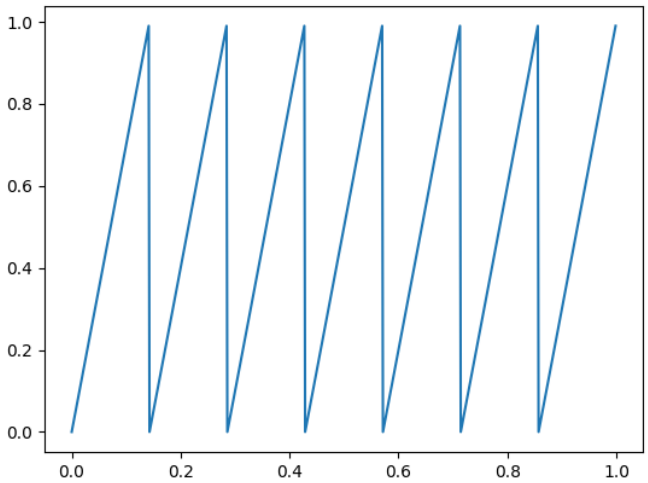


Rysunek 1: Gęstość rozkładu jednostajnego.

Zaimplementowano i przetestowano 2 rodzaje generatorów: oparty na przebiegu piłokształtnym oraz generator liniowy. Wyniki badań zostały przedstawione w dalszej części sprawozdania.

2.1 Generator oparty na przekształceniu piłokształtnym

Sygnał piłokształtny zawdzięcza swoją nazwę graficznej reprezentacji, która przypomina zęby piły (rysunek 2). Odpowiednie próbkowanie takiego sygnału pozwala na generowanie liczb losowych o rozkładzie jednostajnym.



Rysunek 2: Przebieg piłokształtny o 7 „zębach” wygenerowany na przedziale $[0, 1)$ w środowisku Python

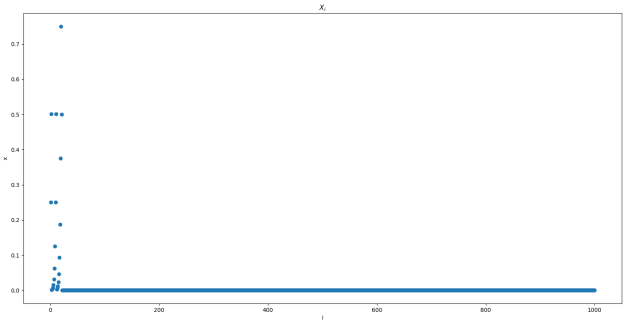
Generacja liczb losowych z takiego przebiegu opiera się o algorytm, którego istotą jest równanie 1.

$$X_{n+1} = X_n \cdot Z - \lfloor X_n \cdot Z \rfloor \tag{1}$$

Rysunek 3: Histogramy generowanych próbek z przebiegu piłokształtnego

Jak można od razu zauważyć dla liczby zębów piły, czyli $Z = 1$ histogram przedstawia 1 słupek. Dzieje się tak, gdyż równanie 1 przyjmuje w takim wypadku postać 2 co oznacza, że każdy kolejny element jest powtórzeniem poprzedniego. Dodatkowo dla parzystej liczby zębów również występuje problem skupienia zdecydowanej większości liczb w jednym przedziale. Sprawdzając poszczególne liczby wygenerowane dla takiej ilości „zębów" (rysunek 4) można zauważyć, że liczby generowane są na całym przedziale, lecz po pewnym czasie wszystkie liczby są zerami. Dzieje się tak dlatego, że podczas mnożenia przez 2 w pewnym momencie „zerują się" dalsze miejsca po przecinku co prowadzi do sytuacji, gdzie każda kolejna liczba będzie zerem.

$$X_{n+1} = X_n - \lfloor X_n \rfloor \tag{2}$$



Rysunek 4: Wykres kolejnych generowanych liczb dla liczby „zębów" piły równej 2

2.2 Generator liniowy

Badając o właściwości generatorów liniowych ma się na myśli generatory zapisywane w postaci przedstawionej jako równanie 3

$$X_{n+1} = (a_1 \cdot X_n + a_2 \cdot X_{n-1} + a_3 \cdot X_{n-2} + ... + a_k \cdot X_{n-k+1} + c) \ mod \ m \tag{3}$$

, gdzie $a_1 - a_k,\ c,\ m$ są pewnymi stałymi parametrami generatora, a operacja $mod\ m$ oznacza wyznaczanie reszty z dzieleia przez m.

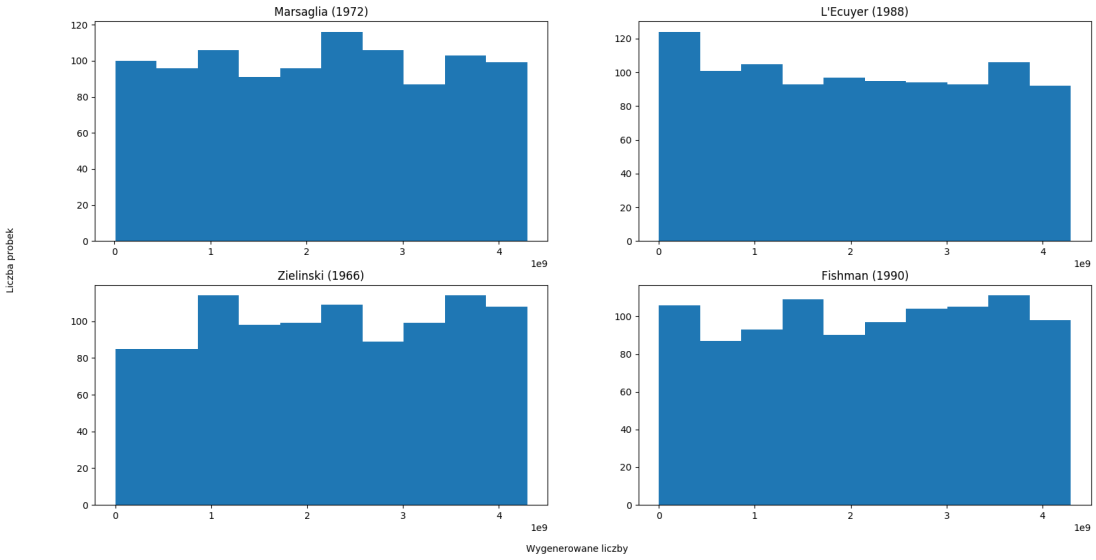
W szczególnym przypadku, gdy $k = 0$ to opis generatora przyjmuje postać równania 4. Co więcej jeżeli $c = 0$ to taki generator nazywany jest generatorem multiplikatywnym, z kolei, gdy $c \neq 0$ – generatorem mieszanym.

$$X_{n+1} = (a \cdot X_n + c) \ mod \ m \tag{4}$$

Do celów badań zaimplementowano 3 generatory multiplikatywne oraz generator mieszany, a ich parametry przedstawiono w tabeli 1. Zestawy parametrów zaczerpnięto z książki Wieczorkowskiego i Zielińskiego [5]. Wygenerowano przy jego pomocy po 1000 liczb dla każdego zestawu parametrów. Wyniki przedstawiono w postaci histogramu dzielącego przedział $[0, m)$ na 10 podprzedziałów. Histogram został przedstawiony na rysunku 5. Jak widać histogramy przypominają (z niewielkimi odchyleniami) wykres gęstości rozkładu jednostajnego.

a	c	m	Źródło
69069	1	2^{32}	Marsaglia (1972)
40692	0	$2^{31} - 249$	L'Ecuyer (1988)
$2^2 \cdot 23^7 + 1$	0	2^{35}	Zielinski (1966)
1099087573	0	2^{32}	Fishman (1990)

Tablica 1: Tabela parametrów generatorów liniowych.



Rysunek 5: Histogramy wyników generacji liczb losowych generatora liniowego dla poszczególnych zestawów parametrów.

3 Metoda odwrotnej dystrybuanty

Chcąc wygenerować liczby losowe o dowolnym rozkładzie można wykorzystać algorytm nazywany metodą odwrotnej dystrybuanty. Jak sama nazwa wskazuje, bazuje ona na wykorzystaniu dystrybuanty rozkładu do generowania liczb. Podstawowym jednak założeniem jest, że „jeżeli istnieje gęstość g rozkładu μ na $\mathbf{R}$, to $F_\mu(t) = \int_{-\infty}^{t} g(x)dx$" [4]. Dodatkowo niech U będzie zmienną losową o rozkładzie jednostajnym na $[0, 1)$. Dzięki takim założeniom można wyznaczyć nową zmienną losową $X = F^{-1}(U)$, wtedy zmienna losowa X ma rozkład prawdopodobieństwa o dystrybuancie F. Zatem generując ciąg liczb $X_i = F^{-1}(U_i)$ otrzymuje się ciąg liczb losowych o rozkładzie z dystrybuantą F.
W taki sposób zaimplementowano 4 generatory liczb losowych o rozkładach:

- równomiernym,
- trójkątnym,
- wykładniczym,
- Laplace'a.

3.1 Rozkład równomierny

Gęstość rozkładu prawdopodobieństwa dla implementowanego generatora wynosi:

$$f(x) = \begin{cases} 2x, & \text{dla } x \in [0,1] \\ 0, & \text{dla } x \in (-\infty, 0) \bigcup (1, \infty) \end{cases} \tag{5}$$

3.2 Rozkład trójkątny

Gęstość rozkładu prawdopodobieństwa dla implementowanego generatora wynosi:

$$f(x) = \begin{cases} x+1, & \text{dla } x \in (-1,0) \\ x-1, & \text{dla } x \in [0,1) \\ 0, & \text{dla } x \notin (-1,1) \end{cases} \tag{6}$$

3.3 Rozkład wykładniczy

Gęstość rozkładu prawdopodobieństwa dla implementowanego generatora wynosi:

$$f(x) = e^{-x},\ x \geq 0 \tag{7}$$

3.4 Rozkład Laplace'a

4 Metoda odrzucania

4.1 Rozkład równomierny

4.2 Rozkład wykładniczy

4.3 Rozkład Laplace'a

5 Wnioski

Literatura

[1] Dokumentacja biblioteki matplotlib. `https://matplotlib.org/contents.html`, 03 2019.

[2] Dokumentacja biblioteki numpy. `https://docs.scipy.org/doc/`, 03 2019.

[3] Dokumentacja python 3.7.1. `https://docs.python.org/release/3.7.1/`, 03 2019.

[4] R. S. Jacek Jakubowski. *Wstęp do teorii prawdopodobieństwa.* Jacek Jakubowski, Rafał Sztencel, 2001.

[5] R. Z. Robert Wieczorkowski. *Komputerowe generatory liczb losowych.* Wydawnictwa Naukowo-Techniczne, 1997.