Projekty z Metod Monte Carlo

Maciej Romaniuk*

29 lipca 2014

1 Wprowadzenie i przykłady zastosowań

Projekt 1.1. Wyjaśnij do czego służą funkcje `srand ()` i `rand ()` w C++. Korzystając z `srand ()` i `rand ()`, napisz algorytm w C++ do obliczania estymatora całki

$$\int_0^1 \sqrt{1-x^2}\,dx = \frac{\pi}{4} \tag{1}$$

dla dowolnej ustalonej przez użytkownika liczby symulacji n.

Projekt 1.2. Załóżmy, że dysponujemy generatorem, który losuje liczby z rozkładu jednostajnego na przedziale $[0;1]$ poprzez wywołanie funkcji `GenerujU`.

```
for i=1 to 12 do
 X(i)=GenerujU
return X(1)+X(2)+...+X(12)-6
```

Napisz program realizujący powyższy algorytm dla dowolnej ustalonej przez użytkownika wielkości próby n (tzn. n razy powtarzający cały algorytm) i zapisujący wyniki w pliku danych. Następnie wygeneruj dużą (tzn. przynajmniej kilkusetelementową) próbę. Korzystając z pakietu statystycznego, stwórz dla otrzymanych danych histogram i inne wykresy statystyczne. Z jakiego rozkładu (w przybliżeniu) pochodzą otrzymane próbki? Wyjaśnij dlaczego. Następnie przeprowadź testy zgodności ze wskazanym rozkładem.

2 Generatory liniowe

Projekt 2.1. Zaimplementuj generator liniowy postaci

$$X_{n+1} = (a_1 X_n + a_2 X_{n-1} + c) \mod m \tag{2}$$

dla ustalanych przez użytkownika parametrów a_1, a_2, c, m, n (gdzie n to wielkość generowanej próby) oraz wartości inicjalizujących. Program wyniki powinien zapisywać do pliku. Przetestuj okres generowanej próby dla różnych zestawów wartości parametrów. Napisz też implementację generatora ALFG postaci

$$X_n = (X_{n-s} + X_{n-r}) \mod m \tag{3}$$

dla podanych przez użytkownika wartości parametrów $n, s < r, m$ oraz wartości inicjalizujących. Program wyniki powinien zapisywać do pliku.

* *e-mail: mroman@ibspan.waw.pl*

Projekt 2.2. Uogólnij program napisany w projekcie 2.1 do postaci

$$X_n = (X_{n-s} \diamond X_{n-r}) \mod m \ , \tag{4}$$

gdzie $\diamond$ jest dowolnym operatorem (dodawaniem, mnożeniem, odejmowaniem, itp.). Zaimplementuj kilka przykładowych funkcji, które użytkownik powinien móc wybrać w trakcie działania programu. Przetestuj wygenerowane próbki za pomocą odpowiedniego pakietu statystycznego.

3 Generatory nieliniowe

Projekt 3.1. Napisz w C++ implementację generatora odwrotności modulo postaci

$$X_n = (a\check{X}_{n-1}^{-1} + b) \mod m \tag{5}$$

dla dowolnych podanych przez użytkownika parametrów a, b, m, n, gdzie m jest liczbą pierwszą. Program wyniki powinien zapisywać do pliku. *Podpowiedź:* Skorzystaj z uogólnionego algorytmu Euklidesa.

4 Kombinowanie algorytmów

Projekt 4.1. Korzystając z procedur przygotowanych we wcześniejszych projektach zaimplementuj kombinację algorytmów ALFG i Lehmera za pomocą formuły

$$Z_n = (X_n + Y_n) \mod m \ . \tag{6}$$

Przekształć otrzymane rezultaty za pomocą funkcji

$$U_i = \frac{Z_i}{m} \tag{7}$$

do próbek z rozkładu na przedziale jednostkowym. Przeanalizuj otrzymane wyniki wykorzystując pakiet statystyczny. Zwróć szczególną uwagę na zgodność z rozkładem jednostajnym.

5 Algorytm Mersenne Twister

Projekt 5.1. Omów działania algorytmu Mersenne Twister. Zapoznaj się z implementacją tego algorytmu w C lub C++ (możesz skorzystać np. ze strony `http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/emt.html`) i sposobami jego wykorzystania w generowaniu liczb (pseudo)losowych.

Projekt 5.2. Przedstaw i omów funkcje, za pomocą których można korzystać z algorytmu Mersenne Twister w różnych pakietach i językach (np. R, Mathematica, GSL).

6 Metoda ROU

Projekt 6.1. Napisz program w C++ generujący liczby (pseudo)losowe dla rozkładu Couchy'ego wykorzystując metodę ROU. Za pomocą pakietu statystycznego przeanalizuj otrzymane wyniki.

7 Biblioteki i programy symulacyjne

Projekt 7.1. Zapoznaj się z możliwościami generowania liczb (pseudo)losowych za pomocą biblioteki GSL. Skorzystaj z dokumentacji dostępnej na stronie `http://www.gnu.org/software/gsl/`. Wyjaśnij, do czego w bibliotece GSL służą funkcje:

- `gsl_rng * gsl_rng_alloc (const gsl_rng_type * T)`
- `gsl_rng_set (const gsl_rng * r, unsigned long int s)`
- `void gsl_rng_free (gsl_rng * r)`

W jaki sposób korzystając z biblioteki GSL, można wygenerować próbę z rozkładu:

1. normalnego?
2. wykładniczego?
3. gamma?

Projekt 7.2. Stwórz program w C++ wykorzystujący bibliotekę GSL. Użytkownik powinien mieć możliwość wybrania dowolnego z rozkładów: normalnego, wykładniczego lub gamma.

Projekt 7.3. Wyjaśnij, w jaki sposób można generować liczby (pseudo)losowe w wybranym pakiecie lub języku (np. R, Mathematica) dla kilku różnych rozkładów. Następnie za pomocą tego pakietu oblicz miary statystyczne, wykonaj wykresy i testy sprawdzające zgodność próbki z zadeklarowanym wcześniej rozkładem prawdopodobieństwa.

8 Metody wielowymiarowe

Projekt 8.1. Zaimplementuj „naiwny" algorytm losowania jednorodnego z kuli p-wymiarowej. Sprawdź doświadczalnie zależność liczby odrzuconych punktów i czas trwania symulacji od wymiaru kuli. Następnie porównaj otrzymane wyniki z algorytmem wykorzystującym wielowymiarowy rozkład normalny.

9 Metody Monte Carlo

Projekt 9.1. Napisz programy obliczające całki

$$\int_{-1}^{6} \frac{\sin x}{1+x^2} dx \ , \ \int_{2}^{\infty} \frac{1}{x^2} \exp\left(-\frac{x^2-4x+6}{2}\right) dx \tag{8}$$

dla podanej przez użytkownika liczby symulacji n. Przeanalizuj doświadczalnie zbieżność generowanych wartości i czas trwania symulacji w zależności od n.

Projekt 9.2. Heurystycznie porównaj zbieżność estymatorów prostego i próbkowania ważonego w metodzie MC dla całki postaci

$$\int_{1}^{2} \left(x^{10}-1\right) dx \ . \tag{9}$$

10 Łańcuchy Markowa

Projekt 10.1. Napisz program symulujący zachowanie się cząsteczek w urnie Ehrenfestów w zależności od chwili t dla dowolnej liczby cząsteczek n. Zaobserwuj zachowanie się liczby cząsteczek w wybranej połowie pojemnika dla różnych stanów ich początkowego rozłożenia w pojemniku (np. wszystkie cząsteczki w jednej połowie, jedna czwarta w jednej połowie i pozostałe w drugiej, itd.).