Wykład 1

SYMULACJE MONTE CARLO – GENERATORY LICZB LOSOWYCH

Wyznaczanie liczby π

Prosty przykład MC

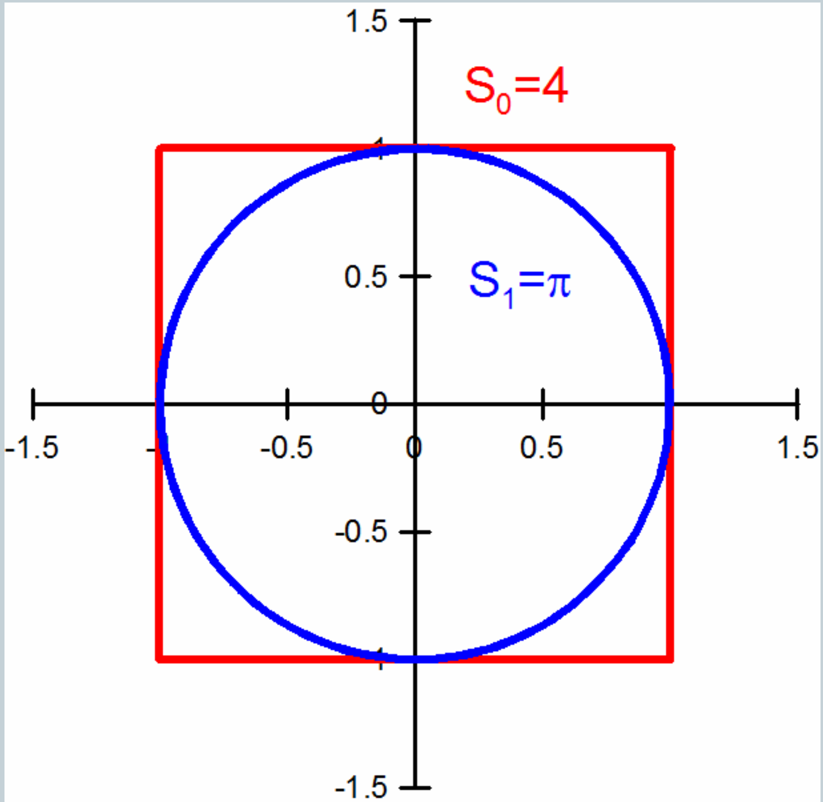
Liczbę π możemy oszacować

•Oddając N_0 „strzałów” do kwadratu

•Zliczając liczbę trafień w koło, N_1

$$\pi \approx 4\frac{N_1}{N_0}$$

Model



Wyznaczanie liczby π

Losowe punkty w kwadracie

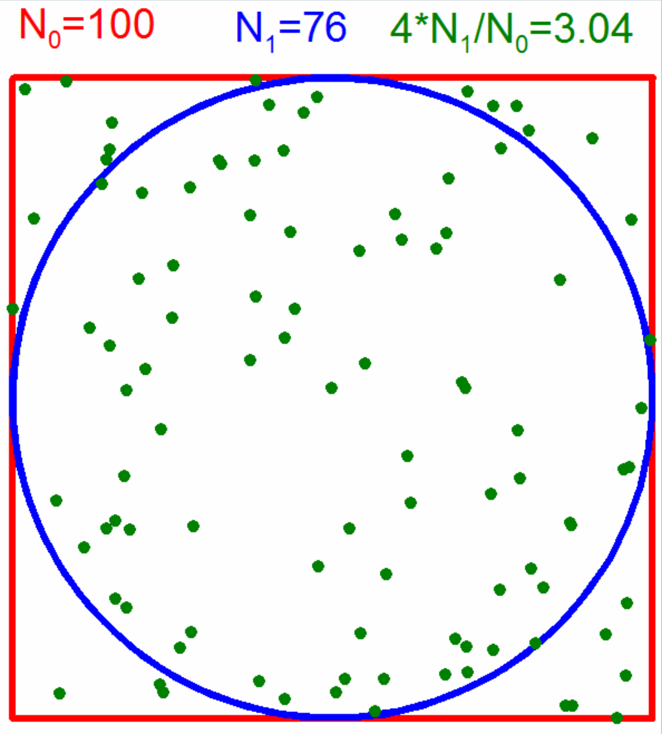
Jak wygenerować losowe punkty w kwadracie?

Wystarczy generator liczb losowych z przedziału [0,1)

$$x = -1 + 2 * u_1 \qquad u_1 \in [0,1)$$

$$y = -1 + 2 * u_2 \qquad u_2 \in [0,1)$$

Losowa próba



Wyznaczanie liczby π

Błąd w metodzie MC

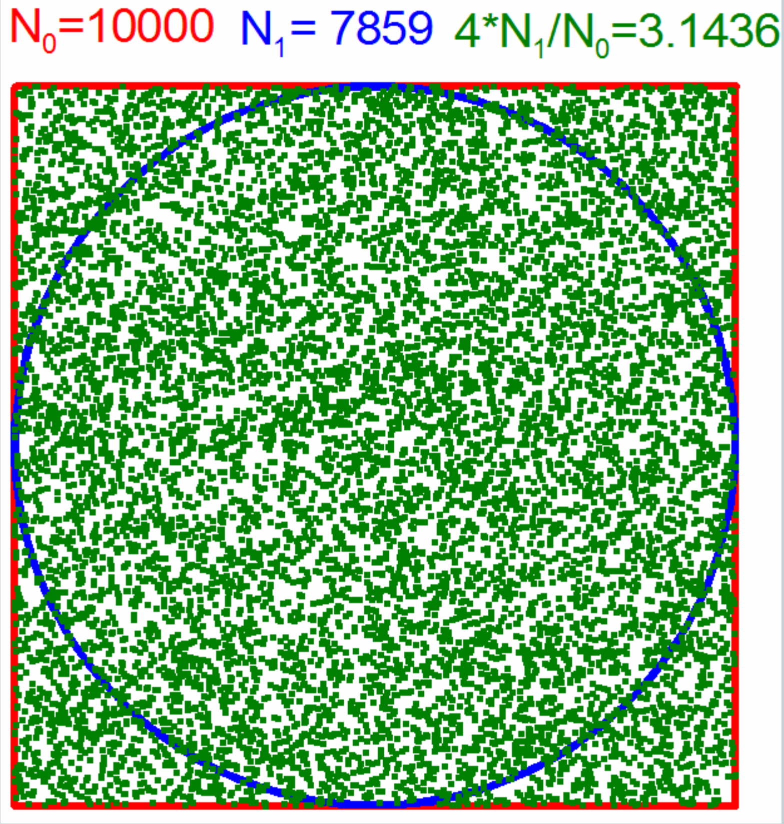
Jaka jest dokładność takiego oszacowania liczby π ?

$$\Delta \propto \frac{\sigma}{\sqrt{N_0}}, \qquad \sigma^2 = \frac{\pi}{4}\left(1 - \frac{\pi}{4}\right)$$

Poprawienie wyniku o 1 cyfrę wymaga 100 razy więcej strzałów

Dokładność w metodzie MC jest ograniczona, stąd główne jej zastosowanie w wielowymiarowych problemach

Losowa próba



Wyznaczanie liczby π

Podsumowując zastosowanie MC do obliczenia π

- Zastąpiliśmy podejście deterministyczne przez probabilistyczne
- Wygenerowaliśmy losową próbę punktów
- Oszacowaliśmy liczbę trafień w koło , a stąd wartość π
- Zbadaliśmy wpływ liczby punktów na dokładność wyniku

Całkowanie Monte Carlo

Zaczniemy od obliczenia prostej całki, żeby pokazać jak metoda stochastyczna zastępuje podejście deterministyczne

$$I = \int_0^1 g(x)dx$$

Do tego celu wykorzystamy zmienną losową X o wartościach z (0,1) z gęstością prawdopodobieństwa f

f(x)=1 dla 0<x<1

Wykorzystując definicję wartości oczekiwanej

$$E[X] = \int_0^1 xf(x)dx$$

Całkowanie Monte Carlo

Sprowadzamy ten problem do obliczenia wartości oczekiwanej

$$I = \int_0^1 g(x)dx = \int_0^1 g(x) f(x)dx = E[g(X)]$$

Wartość oczekiwaną E[g(X)] można oszacować za pomocą średniej arytmetycznej

$$I_{MC} = \frac{1}{M}\sum_{i=1}^{M} g(x_i)$$

obliczonej na losowej próbie $x_1, x_2, \cdots, x_M$

Całkowanie Monte Carlo

Przykład

$$I = \int_0^1 \exp(-x)dx$$

Wyniki oraz błędy dla $M=10^n$

n	I	Δ
2	0.**62**661496	0.00550560
4	0.**63**148474	0.00063582
6	0.**632**01419	0.00010637
8	0.**6321**3715	0.00001659
	0.63212056	

Próbkowanie ważone

Wybierając rozkład f(x) inny niż jednostajny, możemy całkę

$$I = \int_a^b g(x)dx$$

zastąpić obliczeniem wartości oczekiwanej $$I = E\left[\frac{g(X)}{f(X)}\right]$$

W tym przypadku generowanie próby losowej $x_1, x_2, \cdots, x_M$ może być bardziej skomplikowane – zależy od rozkładu f.

Również oszacowanie wartości oczekiwanej $$I_{MC} = \frac{1}{M}\sum_{i=1}^{M}\frac{g(x_i)}{f(x_i)}$$

zależy od wybranego rozkładu f(x), optymalny f~|g|

Generatory liczb losowych

Niezbędnym składnikiem metody Monte Carlo są liczby losowe .

Zwykle korzystamy z generatorów liczb losowych dostępnych w bibliotekach numerycznych, np. GSL .

Generator liczb losowych to deterministyczny algorytm „produkujący" ciąg liczb pseudolosowych $x_1, x_2, x_3, \ldots$.

np.
$$x_{n+1} = Ax_n \bmod m$$

W obliczeniach MC należy używać rekomendowanych generatorów (jak Mersenne Twister), gdyż kiepskiej jakości generatory mogą prowadzić do artefaktów.

Generatory liczb losowych

Wymagania jakie powinien spełniać dobry generator

-
- Dobra losowość (przechodzi przez wszystkie znane testy teoretyczne i empiryczne).
- Długi okres (większy od 10^{20}, obecnie jedną liczbę losową można wygenerować w czasie 10^{-14} sekundy).
-
- Powtarzalność (możliwość wygenerowania tego samego podciągu np. w celach testowych).
- Przenośność (przenośność kodu i produkcja tych samych liczb na różnych komputerach i systemach).

Generatory liczb losowych

Przykład kodu z generatorem mt19937 z gsl

```
#include<gsl/gsl_rng.h>
int main()
{unsigned long int zarodek;
gsl_rng  *r= gsl_rng_alloc(gsl_rng_mt19937);
cout<<"seed >0     =   "; cin>>zarodek;
gsl_rng_set(r, zarodek);

 for(int n=1;n<=20;n++)
   {double u=gsl_rng_uniform(r);
    cout<<n<<"    "<<u<<endl; }
return 0;
}
```

Generatory liczb losowych

Użycie generatora liczb losowych – **random()**

1. Akceptacja z prawdopodobieństwem **p**

jeśli random()<=p to akceptuj

2. Wybór jednej z **n** wartości **p =1/n**

j= [n * random()]

Generatory liczb losowych

3. Generowanie **x** z przedziału **[a, b]** z rozkładem jednostajnym

x=a+(b-a)*random()

4. Generowanie **x** z rozkładu normalnego N(0,1)

u_1= random(), u_2= random(),

$$x_1 = \sqrt{-2\ln u_1}\cos(2\pi u_2), \quad x_2 = \sqrt{-2\ln u_1}\sin(2\pi u_2)$$