Całkowanie metodą Monte Carlo
RAPORT

Maciej Borowiec, Jacek Pabis, Jagoda Wójcik

25 stycznia 2017

1 Wstęp

Celem naszego projektu było zaprezentowanie metody całkowania numerycznego Monte Carlo. W prezentacji przedstawione zostały inne popularne metody całkowania numerycznego, czyli metoda trapezów, metoda Simpsona (parabol) oraz kwadratury Gaussa.

Metoda Monte Carlo jest metodą numeryczną, która korzysta z definicji całki oznaczonej Riemanna, mówiącej, że wartość całki równa jest polu obszaru pod wykresem funkcji w zadanym przedziale całkowania. Ważnym założeniem dla tej metody jest przyjęcie, że wartości funkcji w obszarze całkowania mieszczą się w przedziale $[y_0, y_{max}]$. Pole prostokąta wyznaczonego przez zakres wartości funkcji w tym przedziale oraz przez przedział całkowania wynosi $rectArea = |x_{max}| - |x_0| * |y_{max}| - |y_0|$.
Metoda Monte Carlo polega na wygenerowaniu N punktów znajdujących się w obrębie tego prostokąta i na ich podstawie obliczenia stosunku pola powierzchni pod krzywą do pola wyznaczonego prostokąta.

2 Opis metody Monte Carlo na przykładzie "rzucania kamykami"

Całkowanie za pomocą metody Monte Carlo można opisać na przykładzie hodowcy ryb, który wybrał się do swoich najdalszych włości w celu dodania glonojadnych ryb do zainfekowanego przez glony stawu.

Hodowca ten, aby dodać odpowiednią ilość ryb do stawu, zmuszony jest poznać poznać powierzchnię swojego stawu.

Dla swojego celu nie potrzebuje on znać dokładnej powierzchni stawu, a jedynie oszacowaną wartość.

Problemem hipotetycznego hodowcy jest więc **zmierzenie powierzchni nieregularnie ukształtowanego stawu używając wyłącznie tego, co "pod ręką"**.

3 Implementacja metody Monte Carlo w celu wyliczenia Π

1. Załóżmy, że rozważany staw jest okrągły i wpisany w kwadrat o boku 2 ($r = 1$).
2. Wiemy, że analityczna powierzchnia koła $\oint dA = \Pi$.
3. Od $i = 1$ do $i = N$ generowana jest para losowych liczb x_i, y_i z przedziału $[0, 1]$.
4. Jeżeli ${x_i}^2 + {y_i}^2 < 1$, to liczba trafień += 1.
5. Liczba trafień dzielona jest przez ilość iteracji $\implies$ oszacowane Π.
6. Im większe N tym większa dokładność oszacowania.

4 Opis programu i kod

W naszym programie przedstawione zostały dwie implementacje metody Monte Carlo - pierwsza (Opcja 1) służy do obliczenia liczby Π i nie uwzględnia granic całkowania.

Druga implemetacja (Opcja 2) może posłużyć do obliczenia całki dowolnej funkcji dla podanych granic całkowania.

Kod programu:

```
# -*- coding: utf-8 -*-

from __future__ import division
import random
import math
menu = {}
menu['1']="Circle"
menu['2']="Sin(x)"
menu['3']="Exit"
while True:
  options=menu.keys()
  options.sort()
  for entry in options:
    print entry, menu[entry]
  choose=raw_input("Please Select:")
  if choose =='1':
    def f1(x,y):
        return x**2 + y**2 < 1
    def est_pi(samples):
        counts = 0
        for _ in range(samples):
            if f1(random.random(), random.random()):
                counts += 1
        return (counts / samples) * 4
    print "Monte Carlo pi estimation = " + str((est_pi(10000000)))
  elif choose == '2':
    def f2(x):
        return math.sin(x)
    x0 = 0
```

```
    xmax = 2
    steps = 100000
    y0 = f2(x0)
    ymax=f2(y0)
    for i in range(steps):
        x = x0 + (xmax - x0) * float(i) / steps
        y = f2(x)
        if y < y0: y0 = y
        if y > ymax: ymax = y
    rectArea = (xmax - x0) * (ymax - y0)
    iterac = 100000
    sign = 0
    for _ in range(iterac):
        x = x0+(xmax-x0) * random.random()
        y = y0+(ymax-y0) * random.random()
        if math.fabs(y) <= math.fabs(f2(x)):
            if f2(x) > 0 and y > 0 and y <= f2(x):
                sign += 1
                integral = rectArea * float(sign)
            if f2(x) < 0 and y < 0 and y >= f2(x):
                sign -= 1
                integral = rectArea * float(sign)

    integral = integral / iterac
    print "Monte Carlo integral = " + str(integral)

elif choose == '3':
    break
else:
    print "Unknown Option Selected!"
```

5 Wyniki i wnioski

Dokładność oszacowania liczby Π zależy bezpośrednio od ilości iteracji N. Poniżej przesdstawione zostały uzyskanie oszacowania liczy Π wraz z wyliczonym błędem, w zalezności od liczby iteracji N.

KOLUMNA	WARTOŚĆ
N=1000	3,216 3,128 3,012 3,196 3,132
ŚREDNIA	3,1368
BŁĄD	0,48 %
N=10000	3,1468 3,1388 3,1404 3,1188 3,1588
ŚREDNIA	3,14072
BŁAD	0,0873 %
N=100000	3,14392 3,14012 3,14332 3,1482 3,1384
ŚREDNIA	3,142792
BŁĄD	0,12 %
N=1000000	3,14186 3,142772 3,1404 3,14154 3,1415
ŚREDNIA	3,1416144
BŁĄD	0,00217 %
N=10000000	3,141718 3,141574 3,1416832 3,1411576 3,1412084
ŚREDNIA	3,14146824
BŁĄD	0,0124 %
N=100000000	3,14150228 3,14163196 3,14159432 3,1416826 3,14160884
ŚREDNIA	3,141604
BŁĄD	0,00113 %

Z powyższych wyników widać, że ogólna zasada "im więcej iteracji tym dokładniejszy wynik" sprawdza się w większości przypadków.
Odstępstwa od tej reguły wynikają z zasady działania metody Monte Carlo - czyli z przypadkowości.
Liczba Π została najdokładniej oszacowana w ostatnim przypadku, gdy liczba iteracji była największa, a najgorzej w pierwszym, przy zaledwie tysiącu iteracji.
Można jednak założyć, że hipotetyczny hodowca ryb nie będzie potrzebował dokładności rzędu setnych części procenta w celu oszacowania ilości ryb potrzebnych do zwalczenia infekcji glonów.