Método de Monte Carlo para o cálculo de integral e do volume de um toróide parcial

Mariana Bastos dos Santos
Centro Universitário FEI
São Bernardo do Campo, São Paulo
unifmsantos@fei.edu.br

***Resumo*—O artigo apresenta a resolução de uma integral e o cálculo do volume de um toróide parcial pelo método de Monte Carlo.**

***Index Terms*—Integração Numérica, Monte Carlo**

I. Introdução

Uma integração de uma equação pelo método de Monte Carlo consiste em analisar uma amostra de uma distribuição de pontos que probabilisticamente está contido na área abaixo da curva da equação de interesse. Esse tipo de integração permite resolver integrais muito complexas por meio de aproximações que são menos custosas computacionalmente, do que a integração de forma analítica. Sendo assim, o objetivo do presente trabalho é calcular uma integral por meio do método de integração de Monte Carlo e utilizar o mesmo método para calcular o volume de um toróide parcial.

II. Conceitos Fundamentais

A. Integração pelo Método de Monte Carlo

O Método de Monte Carlo estima a integral da função f sobre o volume multidimensional Ω aproximando I por:

$$I = Q_n = V.\langle f \rangle \tag{1}$$

onde $\langle f \rangle$ é a soma das amostras da função:

$$\langle f \rangle = \frac{1}{N} \sum_{i=1}^{n} f(xi) \tag{2}$$

ou seja:

$$\int_{\Omega} f(x)dx = V.\langle f \rangle \tag{3}$$

III. Implementação

Neste trabalho foi implementado o Método de Monte Carlo para o cálculo de integral e do volume de um toróide parcial. Para isso, foi utilizada a linguagem Python.

Para ambos os cálculos (integral e do volume), primeiramente, é feita a definição da função a ser calculada, como pode ser observado na Figura 1. Na sequência, cria-se um looping que irá iterar n vezes. Para cada iteração determina-se o valor de entrada da função a ser calcula por meio da geração aleatório dos números no intervalo entre a e b da integral, e em seguida, calcula-se a Equação 3.

Cálculo do volume de um objeto por Monte Carlo

```
In [35]: def f_toroide(x,y,z):
             if(x<1): return 0.0
             if(y<-3): return 0.0
             if(z**2 + (np.sqrt(x**2 + y**2)-3)**2) > 1: return 0.0

             return (z**2 + (np.sqrt(x**2 + y**2)-3)**2)
```

n = 100

```
In [65]: %%time
         n = 100

         x_a = 1
         x_b = 4
         x = np.zeros(n)

         y_a = -3
         y_b = 4
         y = np.zeros(n)

         z_a = -1
         z_b = 1
         z = np.zeros(n)

         areas = []
         integral = 0.0
         for i in range(n):
                 x = random.uniform(x_a,x_b) #Popula vetor com números aleatórios
                 y = random.uniform(y_a,y_b) #Popula vetor com números aleatórios
                 z = random.uniform(z_a,z_b) #Popula vetor com números aleatórios

                 integral += f_toroide(x,y,z) #Resultado da função tendo x como números aleatórios

         answer = ((x_b-x_a)*(y_b-y_a)*(z_b-z_a))*(integral/float(n)) #Cálculo final de monte carlo
         Wall time: 3.92 ms

In [66]: answer
Out[66]: 12.749736943451282
```

Figura 1. Implementação da integração por Monte Carlo para cálculo do volume de um toróide parcial.

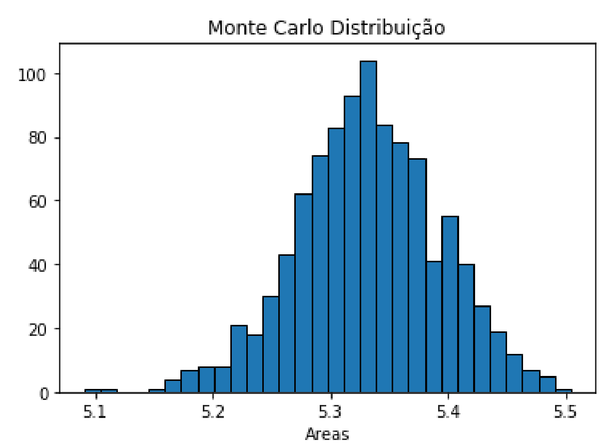

Figura 2. Histograma do cálculo da integral $\int_0^1 \frac{4}{1+x^2}dx$ por Monte Carlo.

IV. Experimentos e Resultados

Foi calculada a integral $\int_0^1 \frac{4}{1+x^2}dx$. Na Figura 2 é possível verificar no histograma a distribuição dos resultados por Monte Carlo. Sendo assim, o valor da integral é mais frequente em cerca de 5.3.

Já para o caso do cálculo do toróide parcial observado na Figura 3, onde $z^2 + (sqrt(x^2 + y^2) - 3)^2 \leq 1$, $x \geq 1$ e $y \geq -3$ os valores obtidos foram: tempo = 3,92 milissegundos e resultado = 12,74 para n = 100, tempo = 13,2 milissegundos e resultado = 10,6 para n = 1000, tempo = 160 milissegundos e resultado = 11,05 para n = 10000. Portanto, a melhor escolha seria n = 1000 por obter o resultado em menor tempo.

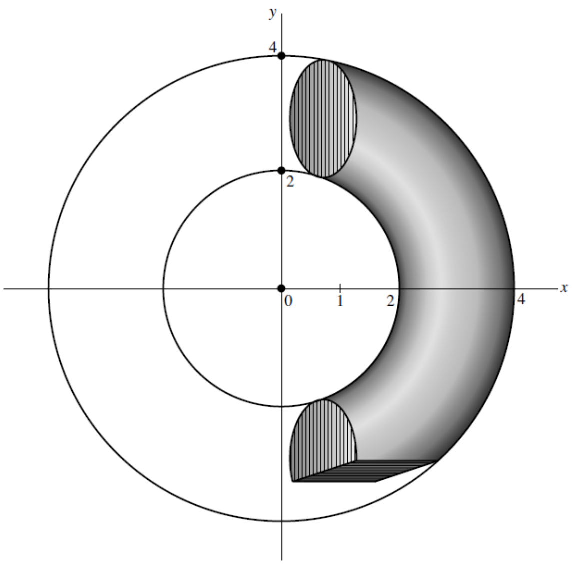

Figura 3. Toróide parcial limitado por: $z^2 + (sqrt(x^2 + y^2) - 3)^2 \leq 1$, $x \geq 1$ e $y \geq -3$.

V. Conclusão

Esse tipo de integração permite resolver integrais muito complexas por meio de aproximações que são menos custosas computacionalmente, do que a integração de forma analítica. Isso pode ser observado nos resultado dos experimentos realizados.

Referências

[1] Press, William H., et al. "Numerical recipes in C++."The art of scientific computing 2 (1992): 1002.