Calculo de Pi por el método de Montecarlo

Mauricio Martinez Tovar Diego Alexis Limón Báez Alan Magdiel Villa Herrera
Monica Rangel Guerra Idalia Rivera Del Angel Adrian Isaac Gomez Ocon

9 de septiembre de 2022

Resumen

En el presente trabajo se diseño una simulación para aproximar el valor del número Pi. Utilizamos el Método de Monte Carlo para ejecutar la simulación considerando diferentes tamaños o valores como entrada, utilizando números pseudoaleatorios generados por computadora. Una vez que obtuvimos los resultados se graficaron.

1. Introducción

El objetivo de esta actividad es el de desarrollar un código en lenguaje Python que muestre el Método de Monte Carlo, que como resultado nos nos vaya acercando cada vez más al valor de pi.
Hablando un poco del Método de Monte Carlo, este se basa en crear modelos de posibles resultados mediante la sustitución de un rango de valores (una distribución de probabilidad) para cualquier factor con incertidumbre inherente.Después, calcula los resultados una y otra vez, cada vez usando un grupo diferente de valores aleatorios de las funciones de probabilidad. De esta forma, dependiendo del número de riesgos o incertidumbres y de los rangos especificados, pueden ser necesarios miles de recálculos para completar la simulación[1].

2. Desarrollo

Mediante el método de Monte Carlo para la estimación de Pi, se usan valores aleatorios para aproximar un valor determinístico. El uso del método de Monte Carlo para aproximar el valor de Pi consiste en dibujar un cuadrado, y dentro de ese cuadrado, dibujar un círculo con diámetro de igual medida que uno de los lados del cuadrado.

Luego se dibujan puntos de manera aleatoria sobre la superficie dibujada. Los puntos que están fuera del círculo y los que están dentro, sirven como estimadores de las áreas internas y externas del círculo.

El área total del cuadrado con lado L es: $AT = L^2$ El área total del círculo dentro del cuadrado es:$AC = PI * (L/2)^2$ La relación de áreas entonces es:$AC/AT = pi/4$ A partir de una estimación de esta relación, se multiplica por 4, y se obtiene el estimador de Pi.

Para la simulación, se usan números pseudoaleatorios con distribución uniforme. A partir de esos números se forman las coordenadas de los puntos que se van a dibujar dentro del cuadrado. Una vez dibujados una cantidad suficiente de puntos, se estimará Pi mediante la fórmula:

$$pi_aprox = 4 * \frac{Puntos\,interiores}{Puntos\,totales}$$

3. Experimentación y resultados

Código

Para la implementación del método de Montecarlo se realizo el siguiente programa, en el cual se generan varios puntos aleatorios dentro de un rango determinado, para este programa solo se toma en cuenta un cuadrante del circulo y el cuadrado para calcular pi. Posteriormente se usa el teorema de pitagoras para determinar si un punto se encuentra dentro del circulo y en caso de ser así se incrementa el contador de números dentro del circulo, para finalmente dividir la cantidad de puntos dentro y puntos totales y multiplicar el resultado por 4 para así obtener el valor de pi. Finalmente se calculo el promedio en cada iteración y se graficaron los resultados haciendo uso de la librería matplotlib.

```
import math
from multiprocessing import Pool
from matplotlib import pyplot as plt
from random import uniform
width = 10000
heigth = width
radio = width

npuntos = 0
ndentro = 0
radio2 = radio*radio
replicas = 60
promediopi = []
change_prom = []


def calc_pi(i):
    global npuntos
    global ndentro

    for i in range(1000000):
        x = uniform(0, width)
        y = uniform(0, width)
        npuntos += 1
        if x*x + y*y <= radio2:
            ndentro += 1
    pi = ndentro * 4 / npuntos
    return pi


if __name__ == '__main__':

    with Pool(4) as p:
        results = p.map(calc_pi, [i for i in range(replicas)])
        print(results)
    # Calculo de promedio acumulativo
    avg = [results[0]]
    for i in range(1,len(results)):
        avg.append((avg[-1]*len(avg)+results[i])/(len(avg)+1))

    # Grafica

    plt.plot([i for i in range(len(avg))],avg)
    plt.plot([i for i in range(len(avg))],[math.pi for i in range(len(avg))])
    plt.title("Valor de pi por Método de Montecarlo")
    plt.xlabel("Iteraciones")
```

```
plt.ylabel("Pi")
plt.show()
```

Gráfica de resultados:

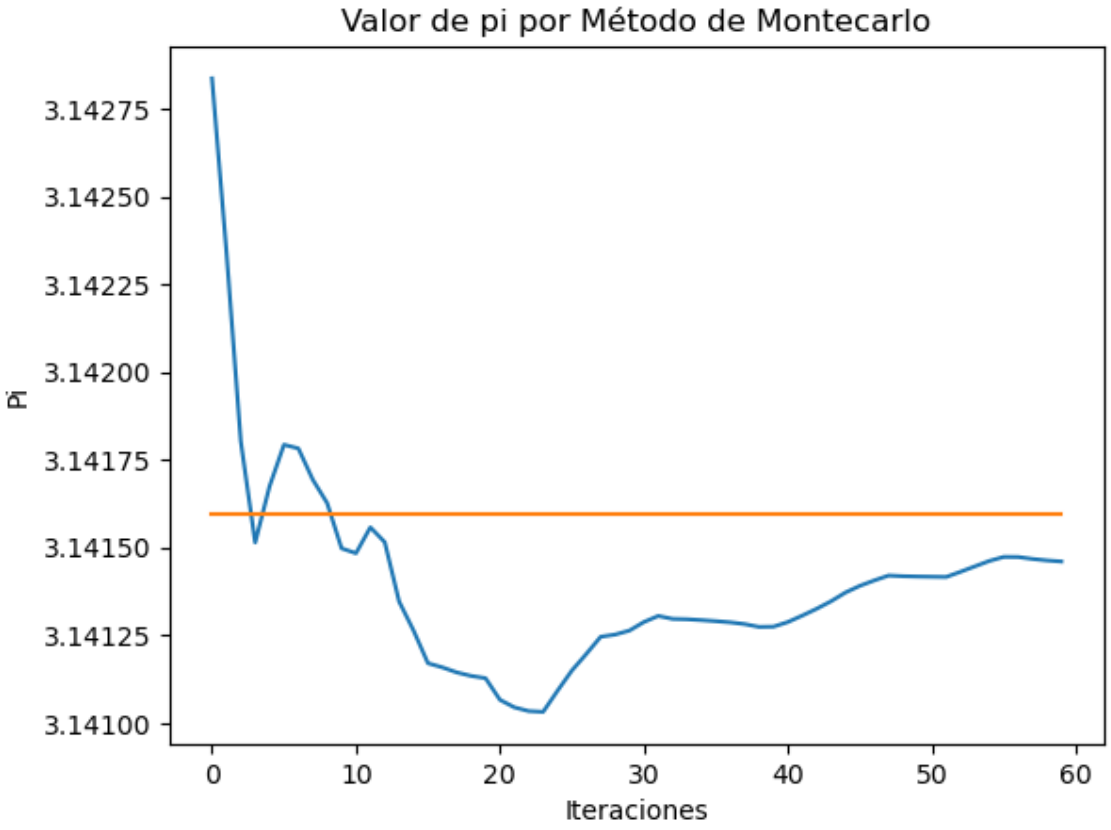

Figura 1: resultados del programa

4. Conclusiones

Este trabajo nos ayudo a entender una de las muchas formas en que se puede calcular pi, y nos ayudo a entender la importancia de las computadoras a la hora de realizar tareas con una gran cantidad de operaciones de por medio. Por otro lado, también aprendimos el lenguaje Python, así como algunas de las librerías que se usan en este lenguaje. Aprendimos a realizar graficas y a como plasmar un algoritmo para usar el método de Montecarlo dentro de este lenguaje. Aprendimos que existen muchas herramientas que no conocíamos para realizar cálculos y gráficas, además de que pudimos ver cómo es que a lo largo de cada iteración recibíamos un valor cada vez más cercano a pi, y como es que este valor podía variar dependiendo de la cantidad de puntos que utilizáramos, así como la cantidad de iteraciones que hiciéramos, sin embargo, para obtener un mejor resultado el tiempo de ejecución también aumentaba considerablemente.

Referencias

[1] EALDE. En qué consiste el método de simulación de monte carlo, Agosto 2020.