Laboratorio 2 - Simulación de Sistemas

INTRODUCCIÓN

HISTORIA DEL NÚMERO PI

Pi (insertar símbolo) es el resultado de dividir la circunferencia de un círculo entre su diámetro, es decir, en tomar el largo del perímetro de un círculo de un radio cualquiera y dividirlo por el diámetro de ese mismo círculo. El resultado es un número irracional que tiene infinitos resultados. sus primeras cifras son 3,1415926535.

Es utilizado en el campo de la topografía, la geodesia y la navegación, en las distribuciones estadística y en numerosas ecuaciones fundamentales de la física moderna, como el principio de incertidumbre de Heisenberg o las ecuaciones del campo de Einstein.

Se estima que ya en el año 2.000 a.C. los babilonios tuvieron un acercamiento al averiguar que la circunferencia de un círculo suele ser poco más de tres veces el equivalente a su diámetro. Sin embargo, en el siglo III a.C., Arquímedes establecería su valor en 3,14 tras utilizar polígonos para afinar su cálculo. A partir de ahí, el número fue adquiriendo mayores aproximaciones. Ptolomeo lo cifró en 3,14166 y a finales del siglo V, el matemático y astrónomo chino Zu Chongzhi añadió dos dígitos más, adquiriendo el valor 3.1415927, un resultado que no fue superado hasta el siglo XVI, cuando el Ludolph Van Ceulen obtuvo un total de 35 decimales[1].

El número fue bautizado π, la letra griega inicial para las palabras “periferia’ y ‘perímetro’. El término fue empleado por primera vez por el matemático William Oughtred, aunque lo popularizó el físico y matemático suizo Leonhard Paul Euler en su obra Introducción al cálculo infinitesimal de 1748.

DESCRIPCIÓN DEL PROBLEMA

El problema consiste en calcular Pi con ayuda a dos métodos como pueden ser el método de Monte-Carlo y el método de Buffon. Para esto se explicara como funciona cada método de una manera sencilla y se simularán en un entorno

controlado a través de algoritmos para llegar a aproximar el valor lo más posible a Pi.

ALGORITMOS DE SIMULACIÓN

Método de Monte-Carlos
A continuación se presenta el algoritmo de simulación del número pi a través del método de Monte-Carlo usando lenguaje Python.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import random
import math

def pi(n): #función que devuelve aproximación del número pi

      #Contador que cuenta cuantas veces el
      #punto seleccionado queda dentro del círculo
      aciertos = 0

      #Iteramos en el rango n (número de iteraciones que se harán)
      for i in range(n):
      #generamos los números aleatorios
      x = random.random()
      y = random.random()

      #calcular distancia media
      deltaX = math.pow((x-0.5), 2)
      deltaY = math.pow((y-0.5), 2)

      #Distribución
      distE = math.sqrt(deltaX+deltaY)

      #Comprobación si el punto está dentro del círculo
      if distE < 0.5:
            aciertos = aciertos + 1

      return (4 * float(aciertos) / n) #calculo del numero pi

#Mandamos a llamar la función dardos
print (pi(1000000)) #Número aleato
```

Método de Aguja de Buffon
A continuación se presenta el algoritmo de simulación del número pi a través del método de Aguja de Buffon usando lenguaje Python.

```
# -*- coding: utf-8 -*-
from random import random
from math import cos,pi

def corta(l=1,d=1): #comprueba si corta o no la línea
    #válido únicamente si l <= d
    x = d*random()
    #Un poco anticlimático tener que usar pi
    y = l*cos(pi/2*random())
    if y>x:
        return True
    else:
        return False

intentos = 1000000
N = 0 #Número de Tiros
C = 0 #Número de tiros que Cortan
L=0.25 #Longitud de la aguja
D=0.75 #Distancia entre líneas de corte, D<1

while N < intentos:

    N+=1
    if corta(L,D): #Si corta se suma otro acierto
        C+=1


print 'Pi = ',(2.0*L*N)/(D*C)
```

MÉTODO USADO PARA GENERAR LOS NÚMEROS ALEATORIOS

En Python el módulo random nos proporciona un rápido generador de números pseudoaleatorios basado en el algoritmo Mersenne Twister; el cual genera números con una distribución casi uniforme y un período grande, haciéndolo adecuado para una amplia gama de aplicaciones.

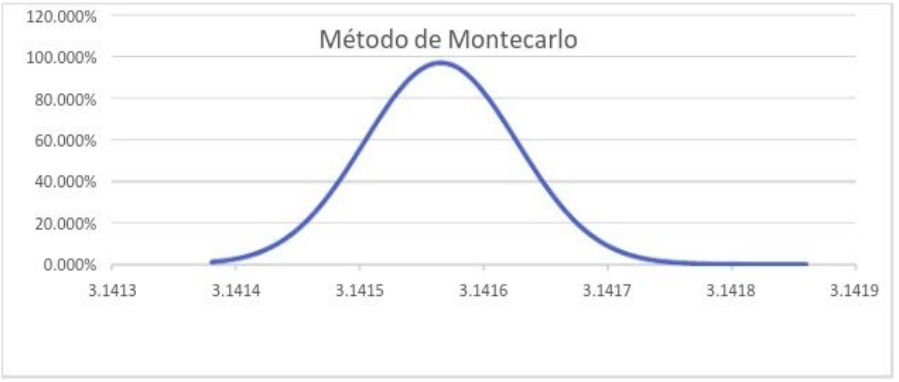
RESULTADOS: TABLAS Y GRÁFICOS DE SALIDA

Método de Monte-Carlos

Intento	Valores de pi
1	3.14152663
2	3.14158089
3	3.14156052
4	3.14153631
5	3.14157849
6	3.14148043
7	3.1416172
8	3.14160201
9	3.14165453
10	3.14154453
11	3.14153606
12	3.14150892
13	3.14168012
14	3.14165588
15	3.14148496
16	3.14156629
17	3.14148962
18	3.14165803
19	3.14152712
20	3.14151907
Media Aritmética:	3.141565379
Desviación estándar:	6.16498E-05

De acuerdo a los intentos realizados que fueron 20 (con n=1000000000) ejecutando el algoritmo propuesto se obtiene la tabla que está en el lado izquierdo, comprobando que los valores de pi se acercan bastante por lo menos hasta el 4 o 5 decimal.

El siguiente gráfico muestra cómo se reparten los resultados en un gráfico, viendo que la tendencia está cerca del 3.1416, un valor cercano a Pi



Método de Aguja de Buffon

Intento	Valores de pi
1	3.141397325
2	3.141411901
3	3.141439119
4	3.141492525
5	3.1415077
6	3.141517792
7	3.141538349
8	3.141550478
9	3.141568968
10	3.14162325
11	3.14164029
12	3.141650513
13	3.141655343
14	3.141659245
15	3.141760593
16	3.141797654
17	3.141867827
18	3.141907237
19	3.141977835
20	3.142177816
Media aritmética:	**3.141657088**
Desviación estándar:	**0.000203923**

De acuerdo a los intentos realizados que fueron 20 (con n=1000000000) ejecutando el algoritmo propuesto se obtiene la tabla que está en el lado izquierdo, comprobando que los valores de pi se acercan bastante por lo menos hasta el 3 decimal, algunos obtienen acercarse hasta el 4 decimal.

El siguiente gráfico muestra cómo se reparten los resultados en un gráfico, viendo que la tendencia está cerca del 3.1416, similar al otro método.

CONCLUSIONES

Tanto el método de Monte Carlo o el Método de Aguja de Buffon, son técnica que ayudan a calcular acercándose lo más posible al valor real de Pi. Aunque hay métodos más eficientes, estos métodos son muy gráficos y fáciles de entender, por lo cual crear un algoritmo que pueda simular estos métodos a traves de codigo de programacion y así obtener un valor aproximado es facil.

Los códigos escritos en Python son simples de entender, para los dos son ocupado una librería llamada Random, que da numero aleatorios para poder ser usados, estos números son los ocupados para simular dos puntos en un plano cartesiano y así poder calcular lo correspondiente a cada método.

BIBLIOGRAFÍA

[1] – ¿Qué tiene de especial este curioso número? - National Geographic España (2019)