

# 蒙特卡洛法估算圆周率实验报告

王松宸

学号：2024201594

## 1 实现步骤

1. 使用 NumPy 生成长度为  $n$  的随机向量  $x, y \sim \mathcal{U}(0, 1)$ ;
2. 判断 `is_in_circle` 的值，从而统计圆内数量  $k$ ;
3. 计算  $\hat{\pi} = 4k/n$  并输出;
4. 变更  $n$  取不同数量，比较估计值的收敛情况;

## 2 源代码

```
1 import numpy as np
2
3 def Monte_Carlo(n):
4     # 生成n个[0,1]区间的随机点
5     x = np.random.rand(n)
6     y = np.random.rand(n)
7
8     # 判断点是否在单位圆内
9     is_in_circle = x**2 + y**2 <= 1
10
11    # 计算圆周率估计值
12    pi_estimate = 4 * np.sum(is_in_circle) / n
13    print(f"估算的π值: {pi_estimate}")
14
15 for n in [1000, 10000, 100000, 1000000, 10000000, 100000000, 500000000]:
16     print('点的数量为', n, '时, ')
17     Monte_Carlo(n)
```

### 3 运行结果

```
点的数量为 1000 时,  
估算的π值: 3.12  
点的数量为 10000 时,  
估算的π值: 3.1444  
点的数量为 100000 时,  
估算的π值: 3.1466  
点的数量为 1000000 时,  
估算的π值: 3.143544  
点的数量为 10000000 时,  
估算的π值: 3.141906  
点的数量为 100000000 时,  
估算的π值: 3.14169092  
点的数量为 500000000 时,  
估算的π值: 3.141609512
```

Figure 1: 不同  $n$  下的  $\pi$  估计输出。

### 4 结论

实验表明，随着采样点数  $n$  的增加，基于蒙特卡洛方法的  $\pi$  估计值逐步收敛到真实值  $3.1415926\dots$ 。然而，收敛速度与方差下降较慢，若需要高精度需显著增大样本规模或采用方差缩减技术。