Lab 3 Report

蒙特卡洛模拟求解定积分

王世炟

PB20151796

March 23, 2023

Part 1: 实验要求

本次实验要求使用 python 进行蒙特卡罗模拟，求定积分的近似值。使用“向不规则图形扔大量的针”的模拟方式，统计落在长方形中不规则图形内的针的数量，利用公式计算出函数 $f(x) = \sin(x)$ 在 $0 \leq x \leq \pi$ 范围内的定积分近似值。实验需要计算 $i = 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000$（i 为针的数量）时定积分的近似值以及近似值与真实值之差的绝对值（保留 5 位有效数字）。

Part 2: 实验环境

本次实验的实验环境为：

- Windows10 64 位
- IDE：VS Code
- python==3.9.5
- jupyter==1.0.0
- matplotlib==3.5.1
- numpy==1.23.0

Part 3: 实验内容

3.1 实验设计

蒙特卡罗模拟是一种基于随机数的数值计算方法，其基本思想是通过随机抽样的方式，模拟某个过程的概率分布，从而得到该过程的统计特性。在本实验中，我们使用“向不规则图形扔大量的针”的模拟方式，统计落在长方形中不规则图形内的针的数量，利用公式计算出函数 $f(x) = \sin(x)$ 在 $0 \leq x \leq \pi$ 范围内的定积分近似值。

3.2 算法流程

具体实现步骤如下：

1. 在 $0 \leq x \leq \pi$ 范围内随机生成 x 和 y 坐标，其中 x 坐标的生成方式为在 [0,] 范围内均匀分布，y 坐标的生成方式为在 [0,1] 范围内均匀分布。

2. 判断该点是否在不规则图形内，判断方法为将该点的 y 坐标乘以 sin(x)，若小于等于该点的 x 坐标，则该点在不规则图形内。

3. 统计在长方形中不规则图形内的点的数量，计算出该数量与总点数的比值，乘以长方形面积，即可得到函数 $f(x) = \sin(x)$ 在 $0 \leq x \leq \pi$ 范围内的定积分近似值

3.3 核心代码解释

生成随机点以及投针过程：

```
    def monte_carlo_integrate(self):
        count = 0
        # 生成随机点
        for i in range(self.n):
            x = random.uniform(0, np.pi)
            y = random.uniform(0, 1)
            if y <= self.f(x):                  # 如果随机点落在函数下方
                count += 1                      # 计数器加一
                self.x_hits.append(x)           # 将随机点的x坐标添加到x_hits列
                    表中
                self.y_hits.append(y)           # 将随机点的y坐标添加到y_hits列
                    表中
            else:                               # 如果随机点落在函数上方
                self.x_miss.append(x)           # 将随机点的x坐标添加到x_miss列
                    表中
                self.y_miss.append(y)           # 将随机点的y坐标添加到y_miss列
                    表中
        self.area = np.pi * count / self.n # 计算面积
```

计算误差：

```
    print('n='+str(n)+' 时 sin(x) 的近似积分值为 '+str(np.around(model.area
        ,5))+' ,近似值与真实值之差的绝对值为 '+ str(np.around(np.abs(model.
        area-actual_value),5)))
```

绘图：

```
    def Show(self,):
        # 绘图
        plt.figure(dpi = 150)
        plt.plot(np.linspace(self.x_min, self.x_max, 100), self.f(np.
            linspace(self.x_min, self.x_max, 100)))
        plt.plot(self.x_hits, self.y_hits, 'o', markersize=1, color='green'
            , label='Hit')
        plt.plot(self.x_miss, self.y_miss, 'o', markersize=1, color='red',
            label='Miss')
        ax = plt.gca()
        ax.set_xlim([self.x_min, self.x_max])
        ax.set_ylim([self.y_min, self.y_max])
        ax.set_title('Monte Carlo Integration(n='+str(self.n)+'),result='+
            str(np.around(self.area,5)))
        ax.set_xlabel('x')
        ax.set_ylabel('y')

        # 显示图形
        plt.show()
```

Part 4: 实验结果

运行上述代码可得到以下结果：

表 1: n 取不同值时 $f(x) = \sin(x)$ 在 $0 \leq x \leq \pi$ 范围内的定积分近似值结果 (保留五位小数)

n	1000	2000	4000	8000	16000	32000	64000	128000
积分近似值	2.04832	2.0169	2.01219	1.96978	2.00453	2.00178	1.99349	1.99948
误差值	0.04832	0.0169	0.01219	0.03022	0.00453	0.00178	0.00651	0.00052

将上表中的数据在同一张图中画出，得到以下结果：

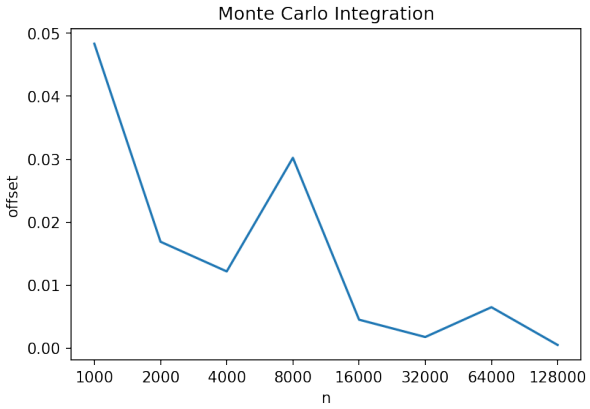
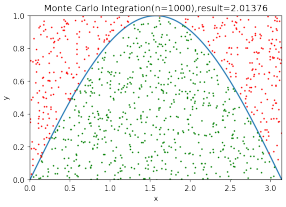
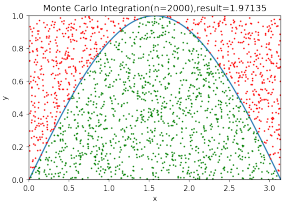


图 1: n-offset picture

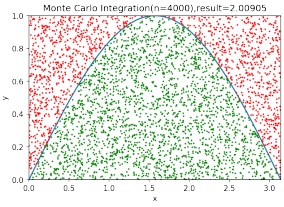
从图中可以看出，随着投针次数的增加，误差呈减小趋势，这也符合我们的预期。下面给出对于不同的 n 值投针法的过程图像：



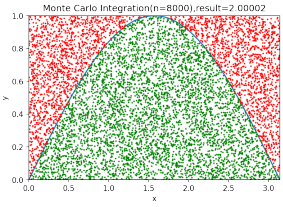
(a) n=1000



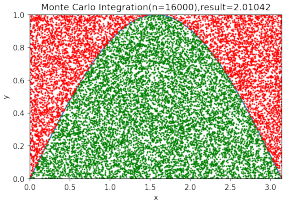
(b) n=2000



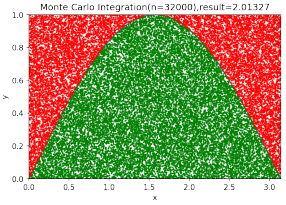
(c) n=4000



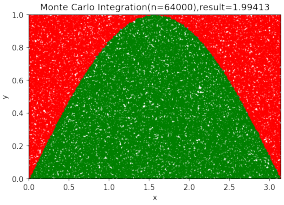
(d) n=8000



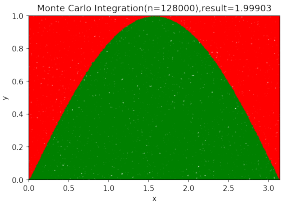
(e) n=16000



(f) n=32000



(g) n=64000



(h) n=128000

Part 5: 实验总结

本实验使用 python 进行蒙特卡罗模拟，求解定积分的近似值。通过不断增加针的数量，得到了越来越接近真实值的近似值，验证了蒙特卡罗模拟方法的有效性。在实际应用中，蒙特卡罗模拟方法可以用于求解各种复杂的数学问题，具有广泛的应用前景。