机器学习与数据挖掘-HW1

—*Monte Carlo Method*—

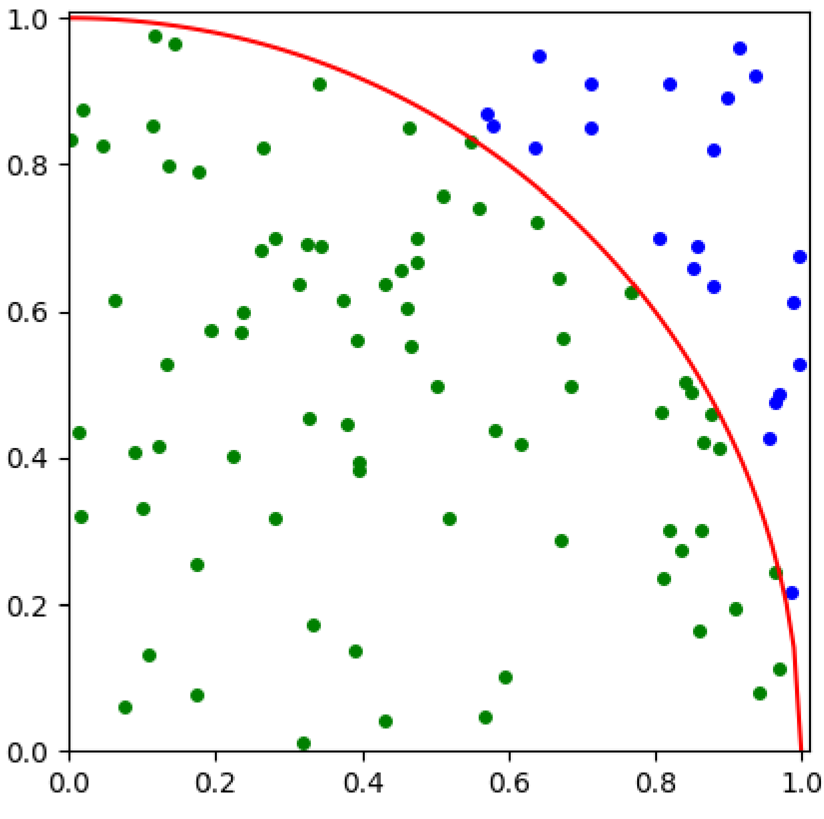
19335253 葉珺明

---

1 计算PI的近似值

- **随机投点法**

- 随机投点法中，以随机投100个点为例，有图像如下：



- 重复一次的实验结果：

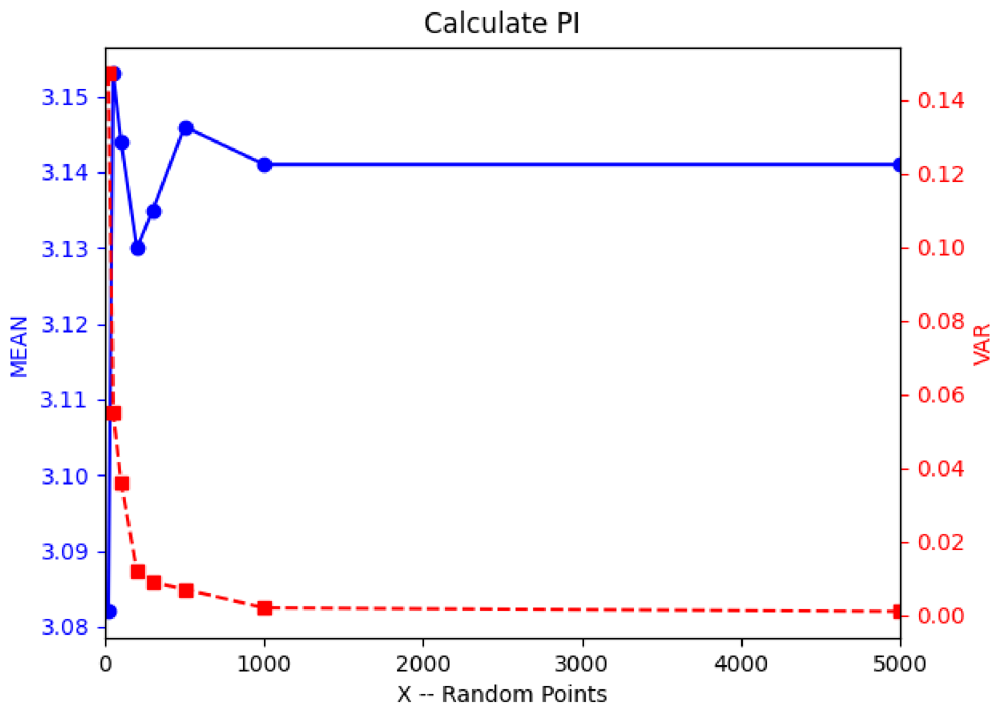
N	20	50	100	200	300	500	1000	5000
PI	3.200	3.280	3.120	3.260	3.187	3.160	3.104	3.162

- 重复100次的实验结果：

N	20	50	100	200	300	500	1000	5000
MEAN	3.082	3.153	3.144	3.130	3.135	3.146	3.141	3.141
VAR	0.147	0.055	0.036	0.012	0.009	0.007	0.002	0.001

- **分析**

  随着N的增大，积分结果的近似值计算趋于平稳，逐渐接近PI的真实值，方差逐渐减小。



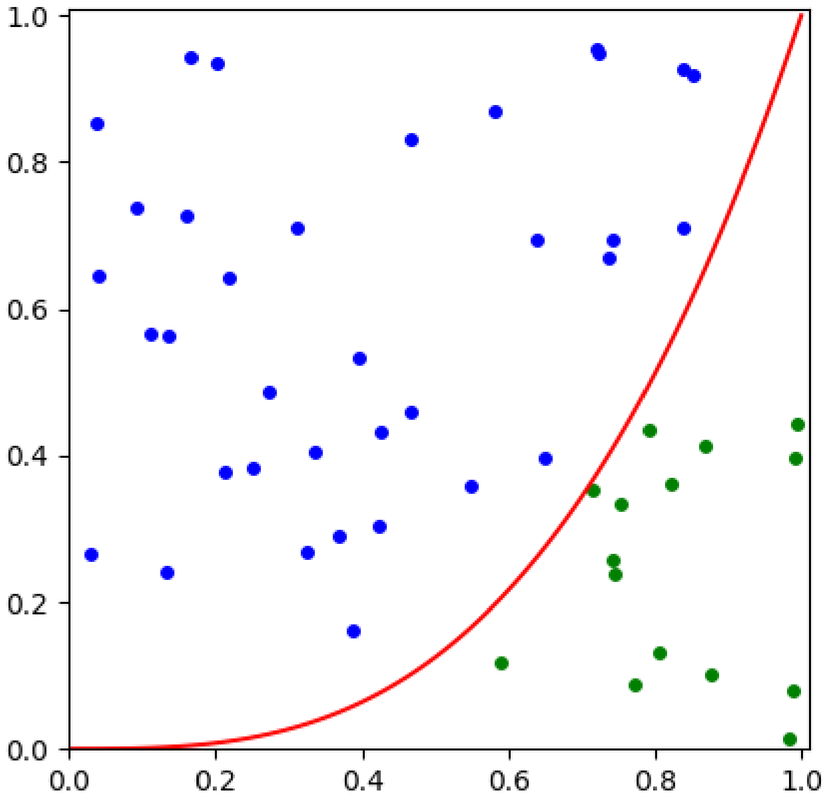
2 求解积分

$$\int_0^1 x^3 dx = \frac{1}{4} = 0.25$$

2.1 随机投点法

> （x, y）在区域中均匀分布

- 随机投点法中，以随机投50个点为例，有图像如下：



- 重复一次的实验结果：

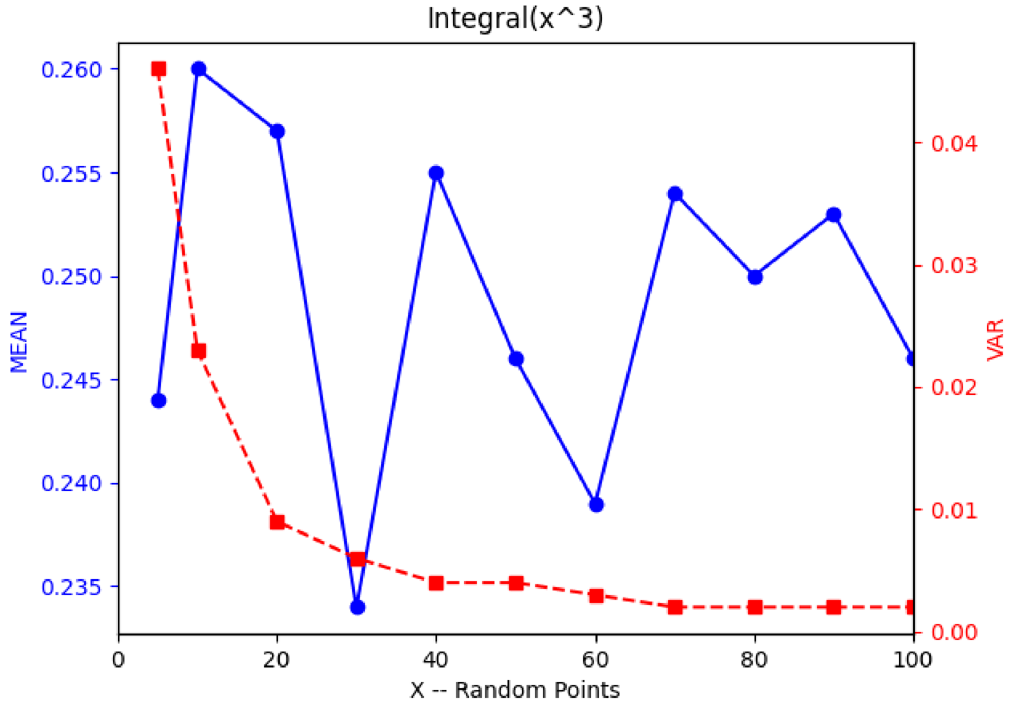
N	5	10	20	30	40	50	60	70	80	90	100
RES	0.200	0.300	0.350	0.367	0.300	0.180	0.250	0.329	0.275	0.222	0.260

- 重复100次的实验结果：

N	5	10	20	30	40	50	60	70	80	90	100
MEAN	0.244	0.260	0.257	0.234	0.255	0.246	0.239	0.254	0.250	0.253	0.246
VAR	0.046	0.023	0.009	0.006	0.004	0.004	0.003	0.002	0.002	0.002	0.002

- **分析**

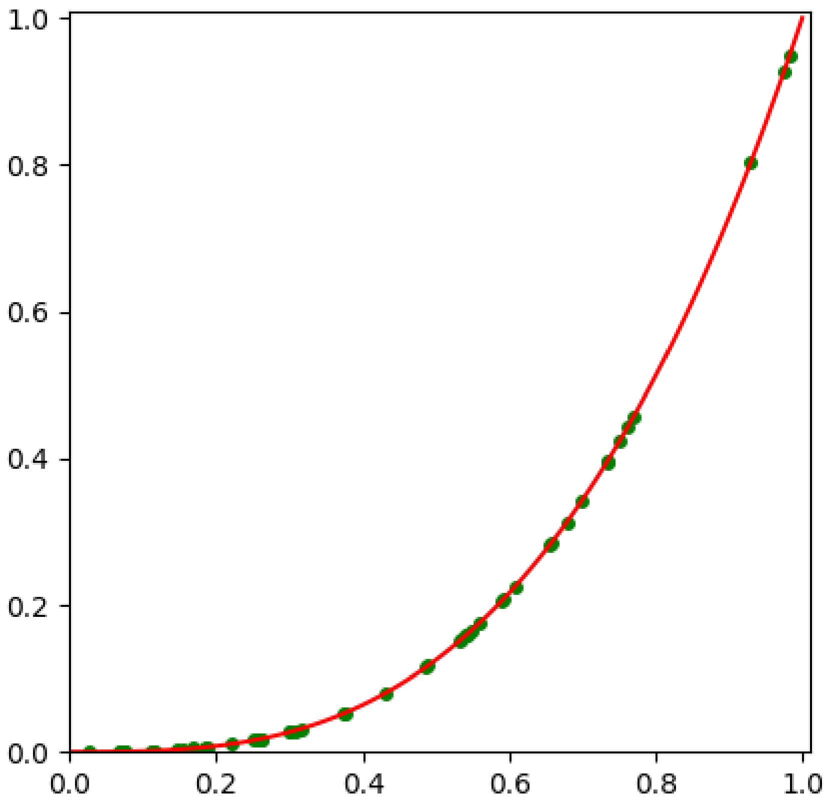
随着N的增大，积分结果的近似值计算趋于平稳，逐渐接近积分的真实值，方差逐渐减小。



2.2 数学期望法

> x通过（0，1）均匀分布得到

- 数学期望法中，在区间随机取50个点为例，有图像如下：



- 重复一次的实验结果：

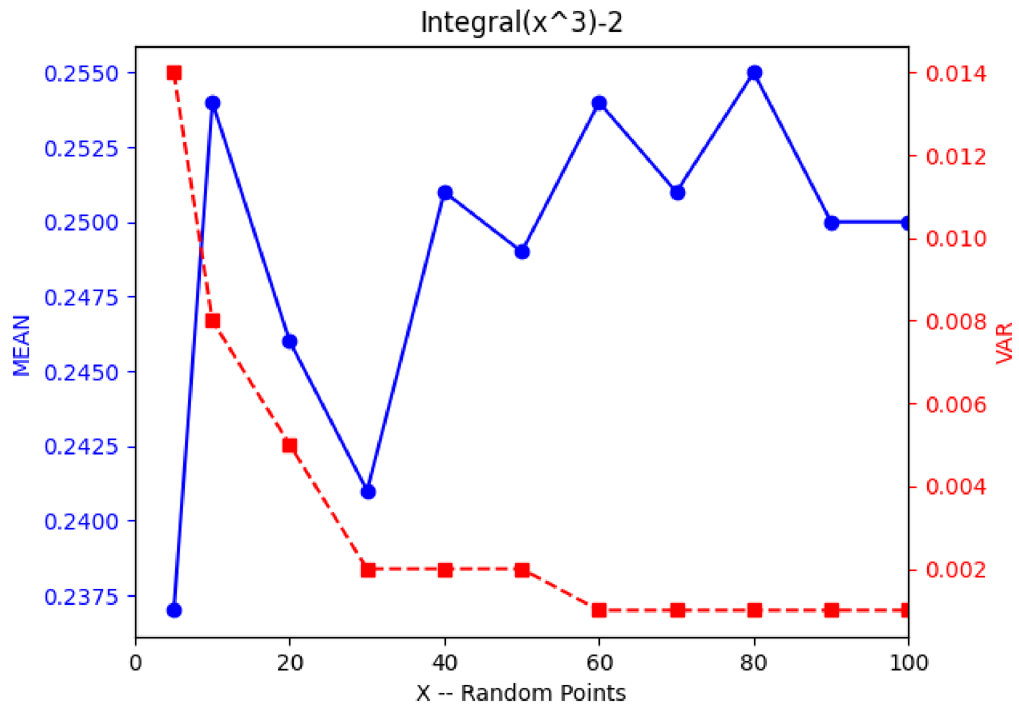
N	5	10	20	30	40	50	60	70	80	90	100
RES	0.370	0.393	0.174	0.243	0.236	0.165	0.222	0.166	0.300	0.307	0.319

- 重复100次的实验结果：

N	5	10	20	30	40	50	60	70	80	90	100
MEAN	0.237	0.254	0.246	0.241	0.251	0.249	0.254	0.251	0.255	0.25	0.25
VAR	0.014	0.008	0.005	0.002	0.002	0.002	0.001	0.001	0.001	0.001	0.001

- **分析**

  随着N的增大，积分结果的近似值计算趋于平稳，逐渐接近积分的真实值，方差逐渐减小。



2.3 方法对比

从上述两种实验方法对比可以看出，两种方法都随着采样点的增多，估计值逐渐趋近积分的真实值；取相同采样点的情况下，数学期望法所得结果向真实值趋近的速度要比随机投点法的快，从方差的比较可以看出数学期望法的计算结果更为稳定。

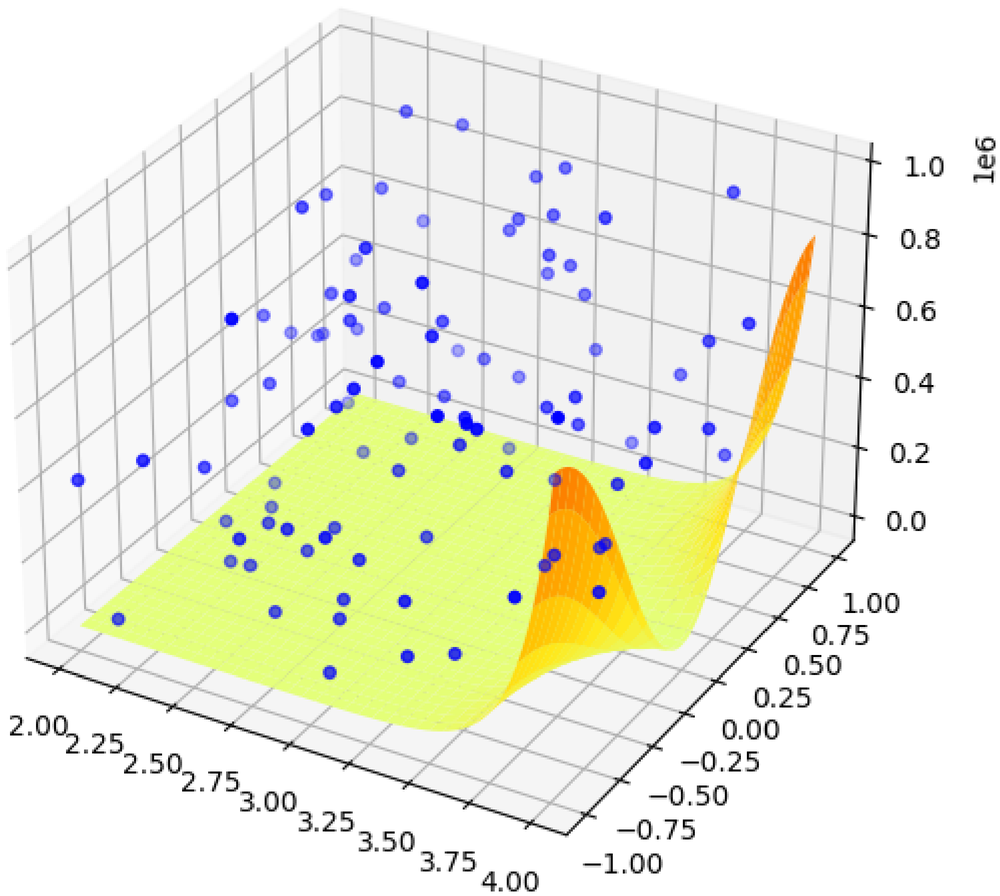
3 计算积分

$$\int_{x=2}^{4}\int_{y=-1}^{1} f(x,y)dxdy = \int_{x=2}^{4}\int_{y=-1}^{1} \frac{y^2 * e^{x^2-y^2}}{x} + x^3 dxdy \approx 1.1296 * 10^5$$

3.1 随机投点法

> （x, y, z）在区域中均匀分布

- 随机投点法中，以随机投100个点为例，有图像如下：（下图没有红点，即没有点投在所求积分区域内）



- 重复一次的实验结果：

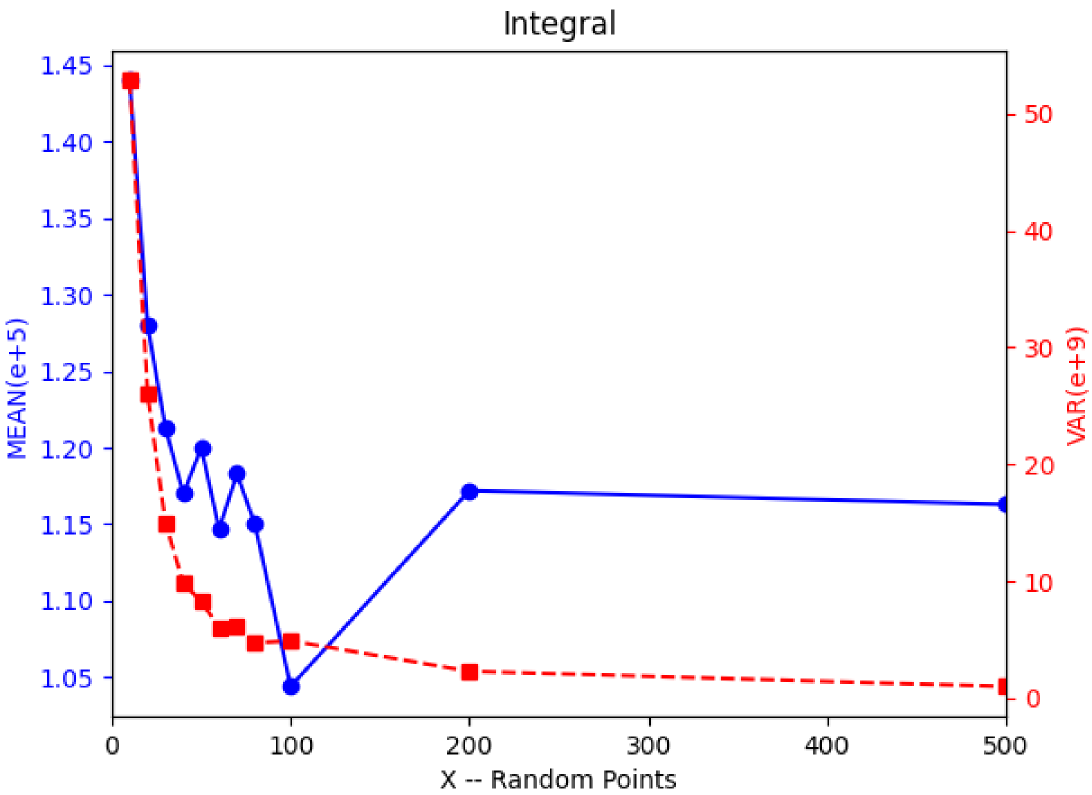
N	10	20	30	40	50	60	70	80	100	200	500
RES(e+5)	0	0	1.333	0	0.800	3.333	1.143	1.000	0	1.600	0.720

- 重复100次的实验结果：

N	10	20	30	40	50	60
MEAN(e+5)	1.440	1.280	1.213	1.170	1.200	1.147
VAR(e+9)	52.86	26.00	15.00	9.811	8.256	6.052
N	70	80	100	200	500	
MEAN(e+5)	1.183	1.150	1.044	1.172	1.163	
VAR(e+9)	6.155	4.725	4.893	2.288	0.9849	

- **分析**

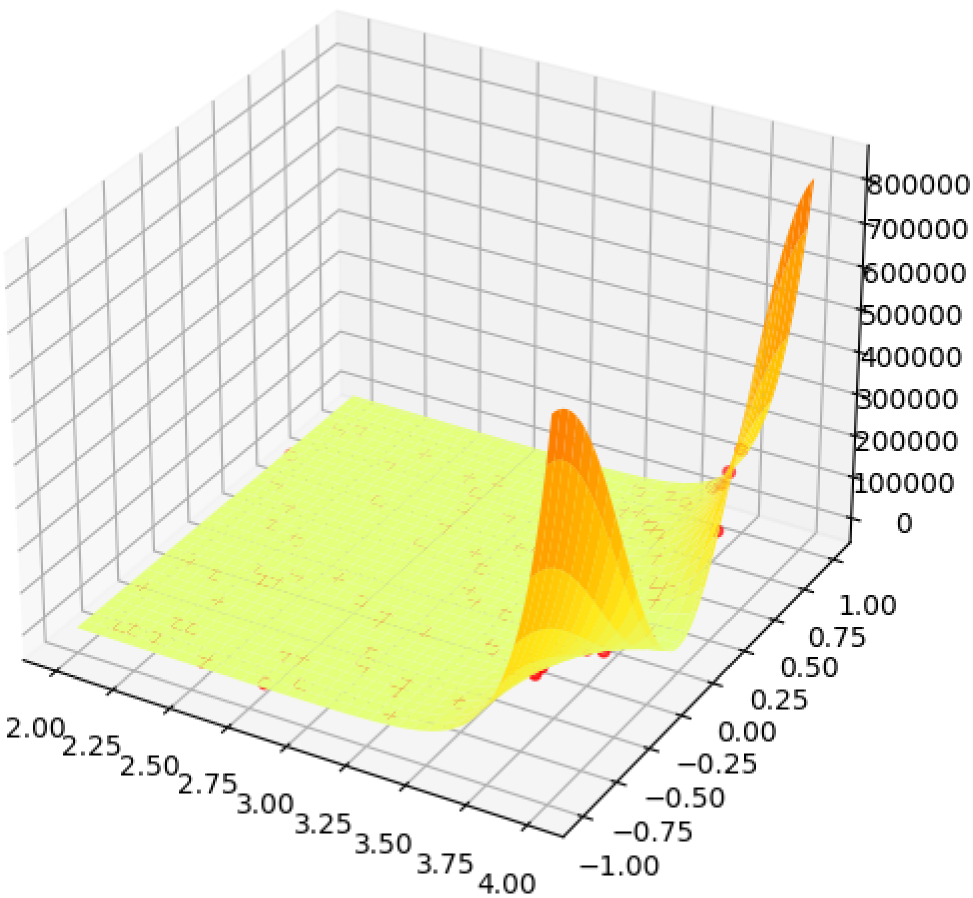
随着N的增大，积分结果的近似值计算趋于平稳，逐渐接近积分的真实值，方差逐渐减小。



3.2 数学期望法

> x通过（2，4）均匀分布得到，y通过（-1，1）均匀分布得到

- 数学期望法中，在区域随机取100个点为例，有图像如下：



- 重复一次的实验结果：

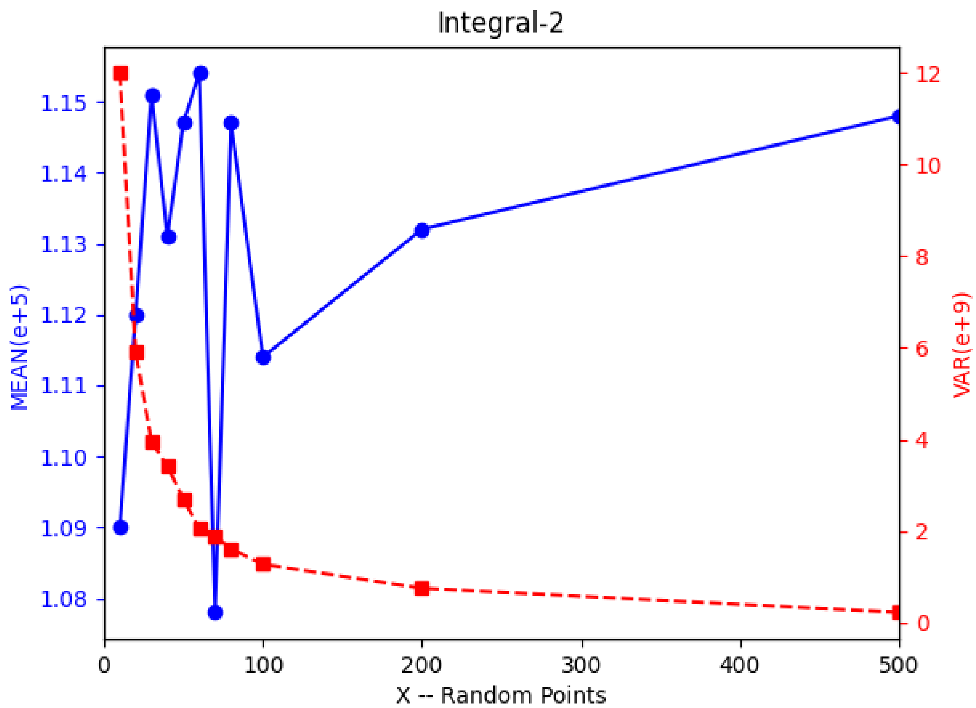
N	10	20	30	40	50	60
RES(e+5)	3.090	1.218	1.384	1.606	0.634	0.853
N	**70**	**80**	**100**	**200**	**500**	
RES(e+5)	0.737	1.232	0.924	1.144	1.265	

- 重复100次的实验结果：

N	10	20	30	40	50	60
MEAN(e+5)	1.090	1.120	1.151	1.131	1.147	1.154
VAR(e+9)	12.00	5.897	3.949	3.406	2.698	2.042
N	**70**	**80**	**100**	**200**	**500**	
MEAN(e+5)	1.078	1.147	1.114	1.132	1.148	
VAR(e+9)	1.886	1.608	1.266	0.746	0.217	

- **分析**

随着N的增大，积分结果的近似值计算趋于平稳，逐渐接近积分的真实值，方差逐渐减小。



3.3 方法对比

所得结论与2.3中的相同，并且在本次求积分的过程中，可以更容易看出，在这种函数分布极其不均匀的情况下，采用随机投点的方法有可能无法将点撒在所求积分区域内，故有结果为0的现象，方差也比较大，相比之下，使用数学期望的方法求得的结果会比较符合预期。

4 探索路

- **思路**
    - 当前状态探索，记录所经路径，对下一时刻的行走方向随机选择
    - 当前状态下，对可行路径进行探索，判断是否越界或重复经过或到达终点，当可选路径为空，即上下左右都不可以走时判定为无解，同时清空所经路径记录；有可行路径时，随机选择前进方向，再对下一时刻状态可行路径进行探索；判断达到终点，则返回记录的路径。
    - 最终当记录的路径不为空时，则判定顺利到达终点；否则，没有成功达到终点。
- **核心代码**

```
def SEEK_PATH(route, visited, x, y):
    route.append((x,y))
    if x==SIZE-1 and y==SIZE-1:
        return route

    path = orientation_exsist(visited, x, y)
    if len(path)==0:
        return []

    x, y = random.choice(path)
    if x==int(SIZE/2) and y==int(SIZE/2):
        visited[x, y] += 1
    else:
        visited[x, y] += 2

    return SEEK_PATH(route, visited, x, y)
```

- **实验结果**

  进行了20000次模拟，最终能够成功跑通的概率为0.25705

5 系统可靠性计算

- **思路**
    - 同时测试系统上路和下路能否通行，即记录至少有一条路是通，最后所得即为该系统的可靠性
- **实验代码**

```
import random

randn = [100, 200, 500, 1000, 2000, 5000, 10000]
```

```python
def processA():
    if random.random() <= 0.85:
        return 1
    else:
        return 0

def processBC():
    if random.random() < 0.95 and random.random() < 0.90:
        return 1
    else:
        return 0

rel = []
for n in range (len(randn)):
    inn = randn[n]
    sum = 0.0
    for i in range (inn):
        if processA() or processBC():
            sum += 1
    rel.append(sum/inn)

print('reliability: ', rel)
```

- **实验结果**

N	100	200	500	1000	2000	5000	10000
REL	0.98	0.975	0.964	0.968	0.9745	0.9774	0.979

随着实验次数的增加，计算所得与逐渐趋近预测值0.97825。