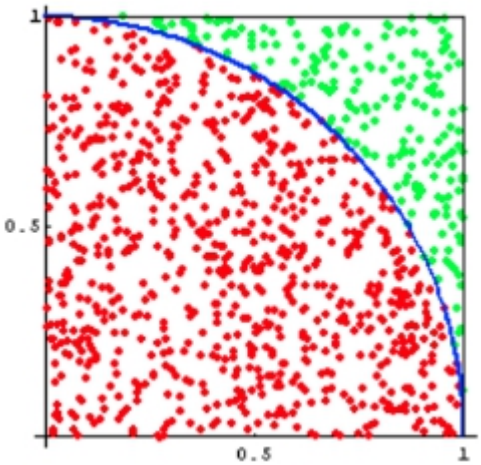
DATA-mining：Exercises for Monte Carlo Methods

Exercise1

蒙特卡洛方法可以用于产生接近pi的近似值。图1显示了一个带有1/4内切圆在内的边长为1的正方形。正方形的面积是1，该1/4圆的面积为pi/4。通过编程实现在这个正方形中产生均匀分布的点。落在圈内（红点）的点和总的投在正方形（红和绿点）上的点的比率给出了pi/4的近似值。这一过程称为使用蒙特卡洛方法来仿真逼近pi实际值。令N表示总的投在正方形的点。当投点个数分别是20, 50, 100, 200, 300, 500, 1000, 5000时，pi值分别是多少？对于每个N，每次实验算出pi值，重复这个过程20次，并在表中记下均值和方差。



环境配置

本题目使用MATLAB实现

算法设计与实现

根据题目要求，首先需要在x=0~1，y=0~1范围内使用均匀分布取随机点，判断随机点到圆心（0,0）的距离是否小于1，进而将随机点分为圆内和圆外两个部分，计数处于圆内点的个数，然后将圆内点的个数比上生成的随机点的总个数，得到的结果乘以4就得出pi的近似值。使用的方法为投值法，核心代码如下：

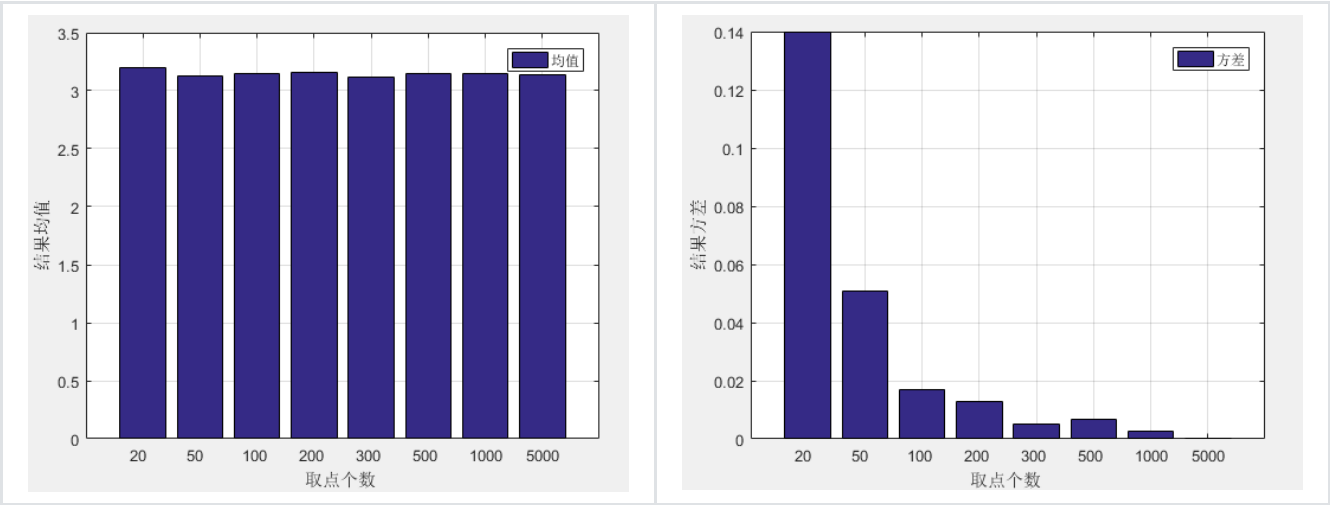
```
r = 1;
centerX = 0;
centerY = 0;
s = rng;
rng(s);
points = rand(2, n);
inCircle = 0;
for i=1:n
dist = sqrt((points(1, i)-0)^2+(points(2, i)-0)^2);
if dist < 1
inCircle = inCircle + 1;
end
end
myPi = inCircle/n*4;
```

多次选取点的个数不同，每次重复20次得出的均值和方差为：

1	2	3	4	5	6	7	8
20	50	100	200	300	500	1000	5000
3.2000	3.1240	3.1440	3.1590	3.1160	3.1452	3.1432	3.1390
0.1400	0.0509	0.0170	0.0128	0.0049	0.0067	0.0025	3.4717e-04

其中，第一行数据为取点个数，第二场为每次取点重复试验20次后结果的均值，第三行为重复20次结果的方差。

画出柱状图如下：



结果分析：

可以看出随着取点个数的增加，每次取点后重复20次试验结果的均值间没有很大差距，即不同的取点个数得出的均值相似，没有很大偏差。但是随着去点个数的增加，方差会逐渐缩小，即20次试验每次试验得出的结果差距会逐渐缩小。这是因为随着去点个数的增加，得出的结果会越来越接近正确值，这个时候每次的结果就不会有很大的偏差。

Exercise2

我们现在尝试通过蒙特卡洛的方法求解如下的积分：

$$\int_0^1 x^3$$

该积分的求解我们可以直接求解，即有

$$\int_{x=0}^{1} x^3 = 1/4$$

。如果你用蒙特卡洛的方法求解该积分，你认为x可以通过什么分布采样获得？如果采样次数是分别是N = 5, 10, 20, 30, 40, 50, 60, 70, 80, 100，积分结果有多好？对于每个采样次数N，重复蒙特卡洛过程100次，求出均值和方差，然后在表格中记录对应的均值和方差。

环境配置

python3

算法设计与实现

首先是取点，这里我选择对x进行均匀分布随机采样。然后我使用的是平均值法，原理为：随机变量 XX 服从 [0,1]上的均匀分布，则 Y=f(X) 的数学期望为

$$E(f(X))=\int_0^1 f(x)dx = J$$

所以估计 J的值就是估计 $f(X)$的数学期望值。由辛钦大数定律，可以用 $f(X)$的观察值的均值取估计 f(X)f(X) 的数学期望。具体做法：

先用计算机产生 n个服从 [0,1] 上均匀分布的随机数：x_i,i=1,2,...,n。

对每一个 x_i ，计算 $f(x_i)$。

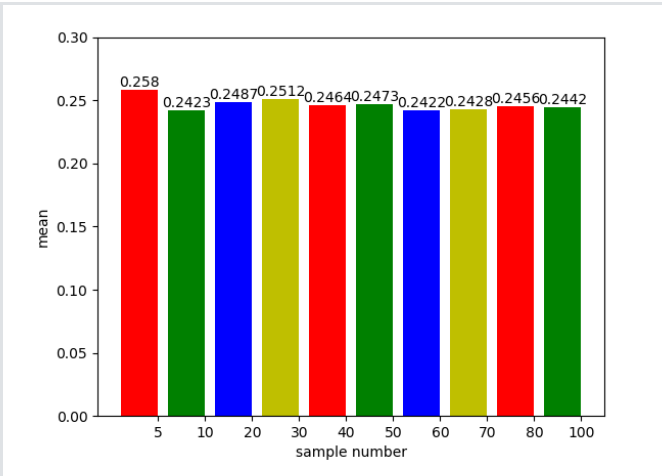
计算 $J = \frac{1}{n}\sum_{i=1}^{n} f(x_i)$

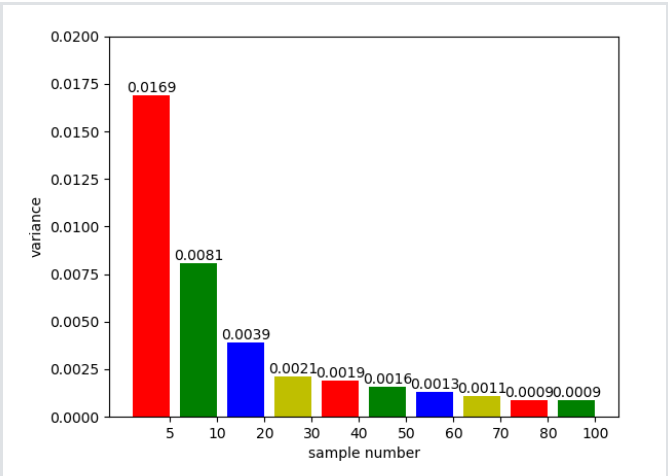
```
def f(x):
    return x**3

def mentoCalo(n):
    x=np.random.uniform(0,1,n)
    c=f(x)
    s=0
    for i in c:
        s=s+i
    integral = 1.00*s / n
    return integral

def loopMC(n):
    result = []
    for i in range(100):
        temp_result = mentoCalo(n)
        result.append(temp_result)

    the_mean = round(np.mean(result), 4)
    the_variance = round(np.var(result), 4)
    return the_mean, the_variance
```

多次选点，每次重复100次得出的均值和方差为：

number of	mean	variance
5	0.258	0.0169
10	0.2423	0.0081
20	0.2487	0.0039
30	0.2512	0.0021
40	0.2464	0.0019
50	0.2473	0.0016
60	0.2422	0.0013
70	0.2428	0.0011
80	0.2456	0.0009
100	0.2442	0.0009

画出柱状图如下：





结果分析

在这次实验结果中，可以看出随着采样点的个数不同，重复100次后得出的结果求均值虽然不同但十分相近，总体偏差不是很大，而且随着采样点的个数的增多，均值会越来越接近准确值。随着采样点的个数的增加，在重复100次运算后，每次运算结果的偏差逐渐缩小，这是因为随着采样点的个数增加每次运算得出的结果会逐渐接近准确值，这就使得不同次试验的结果彼此之间差距会逐渐缩小。

Exercise3

我们现在尝试通过蒙特卡洛的方法求解如下的更复杂的积分：

$$\int_{x=2}^{4}\int_{y=-1}^{1} f(x,y) = \frac{y^2 * e^{-y^2} + x^4 * e^{-x^2}}{x * e^{-x^2}}$$

你能够通过公式直接求解上述的积分吗？如果你用蒙特卡洛的方法求解该积分，你认为(x, y)可以通过什么分布采样获得？如果点（x, y）的采样次数是分别是N = 10, 20, 30, 40, 50, 60, 70, 80, 100, 200, 500, 积分结果有多好？对于每个采样次数N，重复蒙特卡洛过程100次，求出均值和方差，然后在表格中记录对应的均值和方差。

环境配置

python3

算法设计与实现

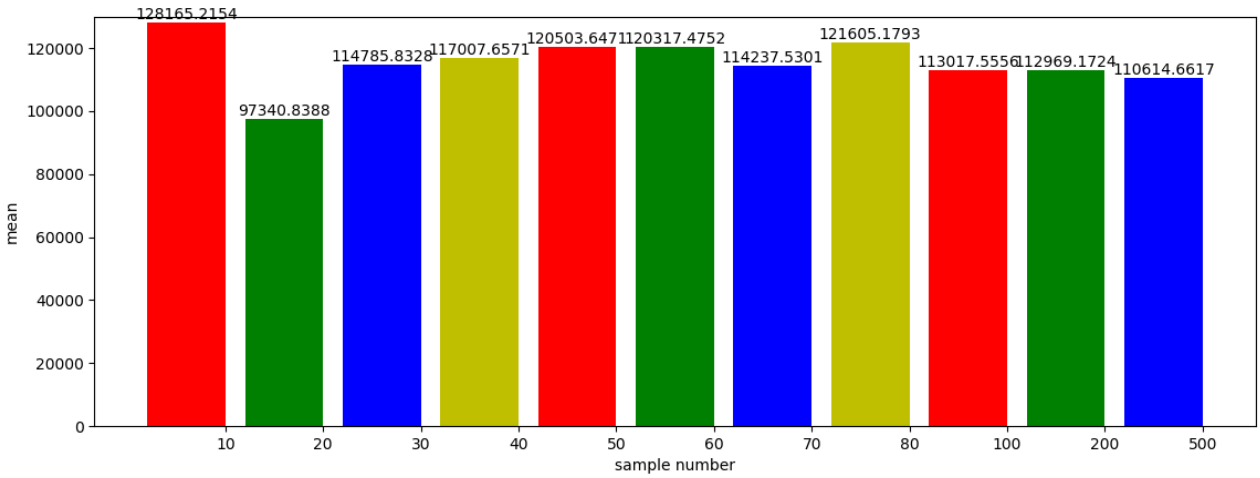
首先，（x，y）使用均匀分布进行随机采样。这一问我同样使用平均值法，只是这次积分区间发生了变化，因此要分别乘上x的积分区间和y的积分区间。具体实现如下：

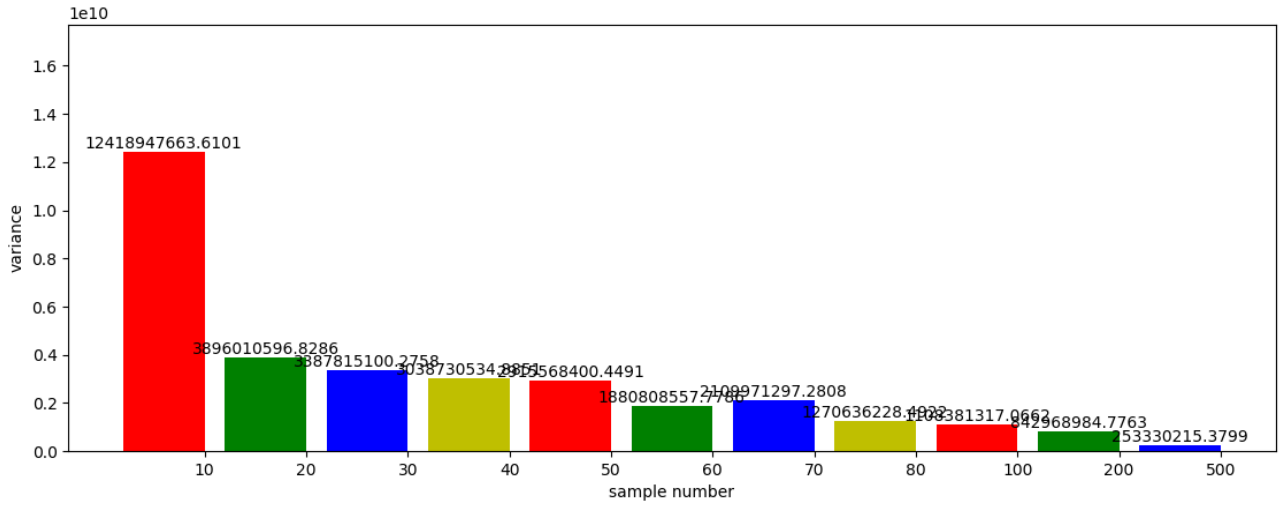
```
def ff(x, y):
    return (pow(y, 2) * math.exp(-pow(y, 2)) + pow(x, 4) * math.exp(-pow(x, 2))) / (x *
math.exp(-pow(x, 2)))

def monteCarlo(n):
    sum = 0
    x = np.random.uniform(0,1,n)*2+2
    y = np.random.uniform(0,1,n)*2-1
    z = ff(x, y)
    for i in z:
        sum = sum + i

    return sum/n

def loopMC(n):
    result = []
    for i in range(100):
        temp_result = monteCarlo(n)
        result.append(temp_result)

    the_mean = round(np.mean(result), 4)
    the_variance = round(np.var(result), 4)
    return the_mean, the_variance
```

多次选点，每次重复100次得出的均值和方差为：

points	mean	variance
10	128165.2	1.24E+10
20	97340.84	3.9E+09
30	114785.8	3.39E+09
40	117007.7	3.04E+09
50	120503.6	2.92E+09
60	120317.5	1.88E+09
70	114237.5	2.11E+09
80	121605.2	1.27E+09
100	113017.6	1.11E+09
200	112969.2	8.43E+08
500	110614.7	2.53E+08

画出柱状图如下：





结果分析

从实验结果可以看出，随着采样点的个数的增加，每次试验得出的结果偏差会逐渐减小，进而导致方差逐渐减小。而对于不同采样点个数的均值，他们之间的偏差不会很大且都与准确值相接近。