计算物理homework1

李明达 PB18020616[1]*

摘要

这是计算物理第1题作业，作业题目是用Schrage方法编写随机数子程序，用连续两个随机数作为点的坐标值绘出若干点的平面分布图。再用$< x^k >$测试均匀性（取不同量级的N值，讨论偏差与N的关系）、C(l) 测试其2维独立性（总点数$N > 10^7$ ）。

关键词

Schrage方法

[1]中国科学技术大学物理学院　　***作者**: dslmd@mail.ustc.edu.cn

1. 算法及主要公式

1.1 随机数产生器的实现

我们采用线性同余法

$$I_{n+1} = aI_n + b$$

$$x_n = I_n/m$$

并且取参数 m 为 int 型最大正整数$2^{31} - 1$, 同时取 b = 0,a = 16807 来产生随机数（即生成一个 16807 产生器），同时使用 Schrage 算法进行取模运算。此次作业用 C 语言完成，具体操作如下：

1. 定义全局变量seed起种子作用，并且定义函数Seed供程序员改变种子的值。同时，Schrage_int在程序中起到 In 的作用，存储上一次产生的随机数，为下一次递归运算作准备

2. 函数 Schrage_int，通过输入的自变量, 用 schrage 算法自动求出 In 的结果，并返回该结果。在 schrage 算法中，我们采用 C自带的求余号完成运算 z mod q, 并且利用 C中整数除法自动向下取整的特性完成运算 [z/q] 之后根据条件返回运算结果.

3. 最后通过函数 Write_RandN(int seed, int N)来产生N个随机数,并写入文件之中。结果存储在data.txt中。

1.2 随机数产生器的检验

这里我们一共采用三种办法： 1.平面分布图：用连续两个随机数作为点的坐标值绘出若干点的平面分布图，从而来看是否随机。

2. 定义函数 Aver_RandN(int seed, int N, int k) 产生N个随机数,并计算x^k平均值，将其与$\frac{1}{k+1}$比较得出误差，用于分析

3.以及定义CL(int l, int N)产生N个随机数，并且用于计算C(L)，从而可以得出数据与数据之间的独立性。

$$c(L) = \frac{< x_n x_{x+l} > - < x_n >^2}{< x_n^2 > - < x_n >^2}$$

理论上如果完全独立，则C(L)为零：

$$C(L) = 0$$

所以从这个我们也可以得出随机数的效果。

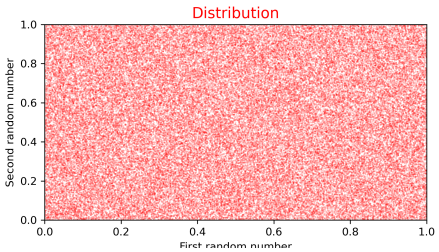

图 1. 连续两个随机数作为点的坐标值绘出若干点的平面分布图

2. 计算结果的分析与讨论

2.1 均匀性讨论

首先我们用第一种办法：平面分布图。图一所示。用连续两个随机数作为点的坐标值绘出若干点的平面分布图，从而来看是否随机。在这里我取了80000个点，通过data.txt里生成的数据来画出该图。画图代码如图二所示。

其次我们计算x^k平均值，将其与$\frac{1}{k+1}$比较得出误差，平均值如图三所示：

通过计算，我们可以得出图四表：

如果画成图，如图五图六所示：

```python
# -*- coding: utf-8 -*-
import numpy as np
import math
import matplotlib.pyplot as plt

plt.figure(num=2,dpi=300)

f = open('data.txt')     #用test.txt先试试
s = f.read()
p = list(float(i) for i in s.split())
print(p)
l=len(p)

plt.scatter(p[0:80000:1], p[1:80001:1], s=0.01, c='r')

ax=plt.gca()
plt.xlim(0,1)
plt.ylim(0,1)
plt.xlabel('First random number')
plt.ylabel('Second random number')
ax.set_title('Distribution',fontsize=14,color='r')
plt.show()
```

图 2. 画图代码

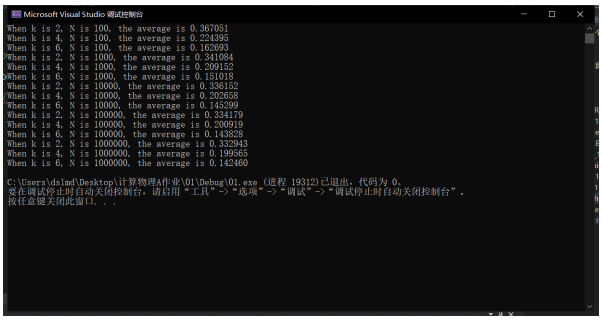

图 3. x^k平均值

k \ N	100	1000	10000	100000	1000000
2	0.03	0.01	0.003	0.0008	-0.0004
4	0.02	0.01	0.003	0.0009	-0.0004
6	0.02	0.01	0.002	0.001	-0.0004

图 4. x^k与$\frac{1}{1+k}$的偏差

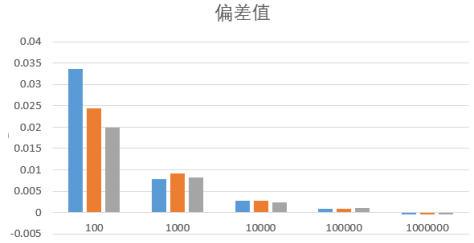

图 5. x^k与$\frac{1}{1+k}$的偏差

可以看出，随着N的增加，x^k平均值与$\frac{1}{k+1}$的偏差值会越来越小，这说明，N越高，伪随机数生成器的均匀性就越好！而这个均匀性，除了在样本很小的情况下，与k没有特别大的关系。所以我们最终的结论是，**N越高，伪随机数生成器的均匀性就越好！且近似于$\frac{1}{\sqrt{N}}$成正比**

2.2 独立性讨论

二维平面图可以看出伪随机数产生器的独立性：

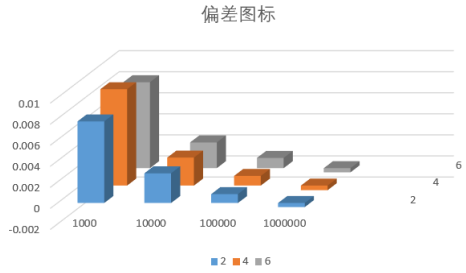

图 6. x^k与$\frac{1}{1+k}$的偏差

由二维平面图可以看出，没有明显的条纹，且基本上在平面内密铺，初步证明，我们随机数产生器的独立性有较好的保证。

接着，我们进一步用 C(2) 、 C(3) 以及更高的 C(L) 测试我们随机数产生器的独立性。如下图所示在N=1e7的量级，可以看出C(L)基本上在0.03左右，

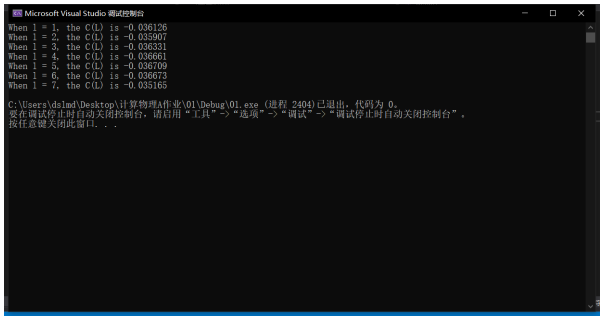

图 7. C(L)的值

这比理想值要高出不少，说明这个随机数发生器并不是真正的“随机”，还是有非独立性的成分在里面。

3. 结论

我们用 Schrage 算法产生了一个 16807 产生器，并对这个产生器进行了均匀性和独立性检验。

均匀性检验中 x^k 的结果符合 $|<x^k-\frac{1}{k+1}>|$ 按照$\frac{1}{\sqrt{N}}$ 衰减的预期，且检测效果较好，证明我们随机数产生器的均匀性比较可靠。

没有在二维平面图上发现明显的图案条纹也说明均匀性和独立性状态良好。在C(l) 检验独立性中，我们发现其距离真正的“随机数”还差一些水平，这个随机数发生器并不是真正的“随机”，还是有非独立性的成分在里面。但总体来说效果良好。

综上所述，我们随机数产生器的实验比较成功，在这个实验中学到了很多知识。