**浅谈伪随机数**

计算机科学与技术系 181860007 陈盛恺

1. **前言**

一想到这个东拼西凑的五一小长假，就想到了有伪随机数，为啥这么说呢，因为小长假这个定义就很奇怪，哪有“小”的“长假“啊，就像有个段子要么是“小短腿”要么是“大长腿”，哪有“小长腿”这种奇怪的设定。其实随机数也是这样一个东西，利用计算机算法生成的伪随机数可以通过算法变得杂乱无章，但是又有一些规律，而生活中的真随机数看上去完全随机却又服从着本福特定律这样的神奇规律。

1. **伪随机与真随机**

伪随机数：通过公式或者算法生成的一个数值序列，虽然会遵循某种规律，在数学意义上并不是随机的，但是在统计意义上具有随机数的一些特性。

真随机数：产生的数值序列不可预计，几乎不可能产生两个完全相同的真随机数序列。真随机数是通过一些随机的物理过程来产生的，例如放射性衰变，电子设备噪声等。

1. **如何评价一个伪随机数序列**
   1. **相同序列的概率低**
   2. **符合统计意义上的平均性**
   3. **不应该通过一段序列预测下一段序列**
   4. **不应该通过随机数发生器的状态猜测出随机数发生器之前的状态**

ab为常用的随机数生成器应满足的标准，而cd为加密应用上需要满足的标准。由于数学水平有限，所以只给出相关定义不做数学验证。

1. **如何生成均匀伪随机数**
   1. **线性同余法**

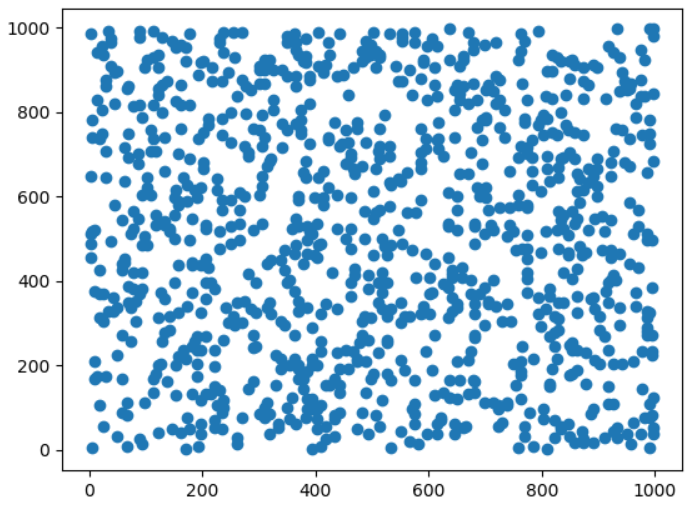
线性同余法是目前最广泛应用的一种伪随机数生成算法，其算法思想是通过前一个数的线性运算获得下一个数，其递推公式如下：

ai+1=( ai \* b + c ) mod m

通过该算法确定一个随机数序列需要确定b，c，m。该序列的最大周期是m，也就是在序列长度超过m之后将不符合第一条验证，为保证最大周期，bcm应满足一下条件：

* + 1. c和m互质
    2. m的所有质因子的积能整除b-1
    3. 若m是4的倍数，则b-1也是
    4. b，c，a(0)（初值，一般即种子）都比m小
    5. b，c是正整数

c++中的rand()函数用的就是这种随机数生成算法。这种算法有一定的局限性，不能用于加密和蒙特卡洛算法，在高维空间中分布均匀度并不理想。该方法所生成的分布图如图所示：

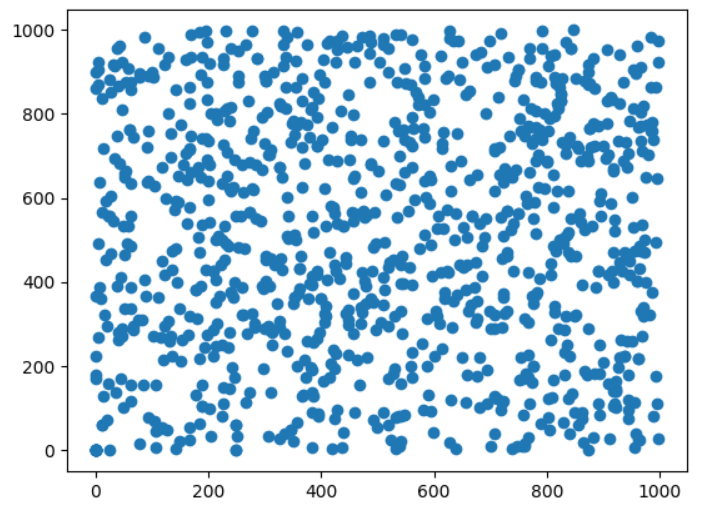


* 1. **平方取中法**

平方取中法是由冯诺依曼1946年提出的一个随机数生成算法其算法思想是将数列中的第ai项（假设其有m位）平方，取得到的2m位数（若不足2m位，在最高位前补0）中间部分的m位数字，作为ai的下一项ai+1，由此产生一个伪随机数数列：

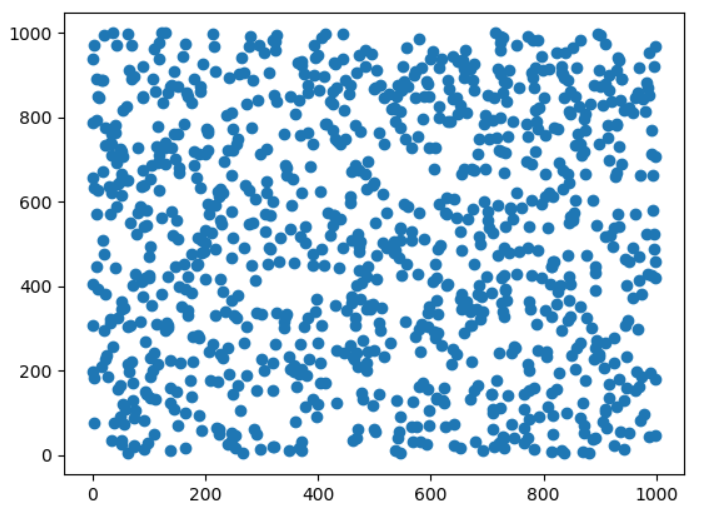
ai+1= (10^(-m/2)\*ai^2)mod(10^m)

但在实际情况中容易出现周期性，a退化为0等状况。分布如图所示：

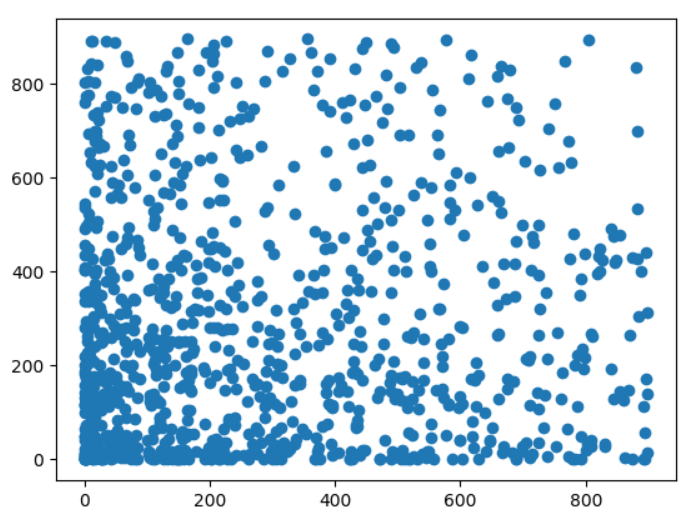


1. **梅森旋转法**

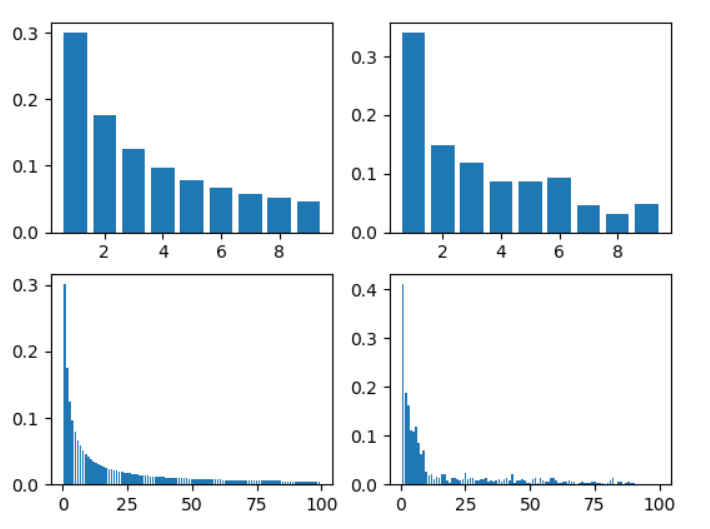
梅森旋转算法是一个伪随机数发生算法.由松本真和西村拓士开发，是一个基于有限二进制字段上的矩阵线性递归的算法.可以快速产生高质量的伪随机数，修正了朴素随机数生成的缺陷。其随机分布如下：



1. **具有特殊分布规律的随机数——以本福特定律为例**

因为本福特定律对于不同位数的数有不同的分布规律，所以该算法讨论如何生成n位下符合本福特定律的随机数，比较朴素的想法是随机生成一个[0,1]的概率然后再根据该位数出现的概率进行映射，图像如图所示：

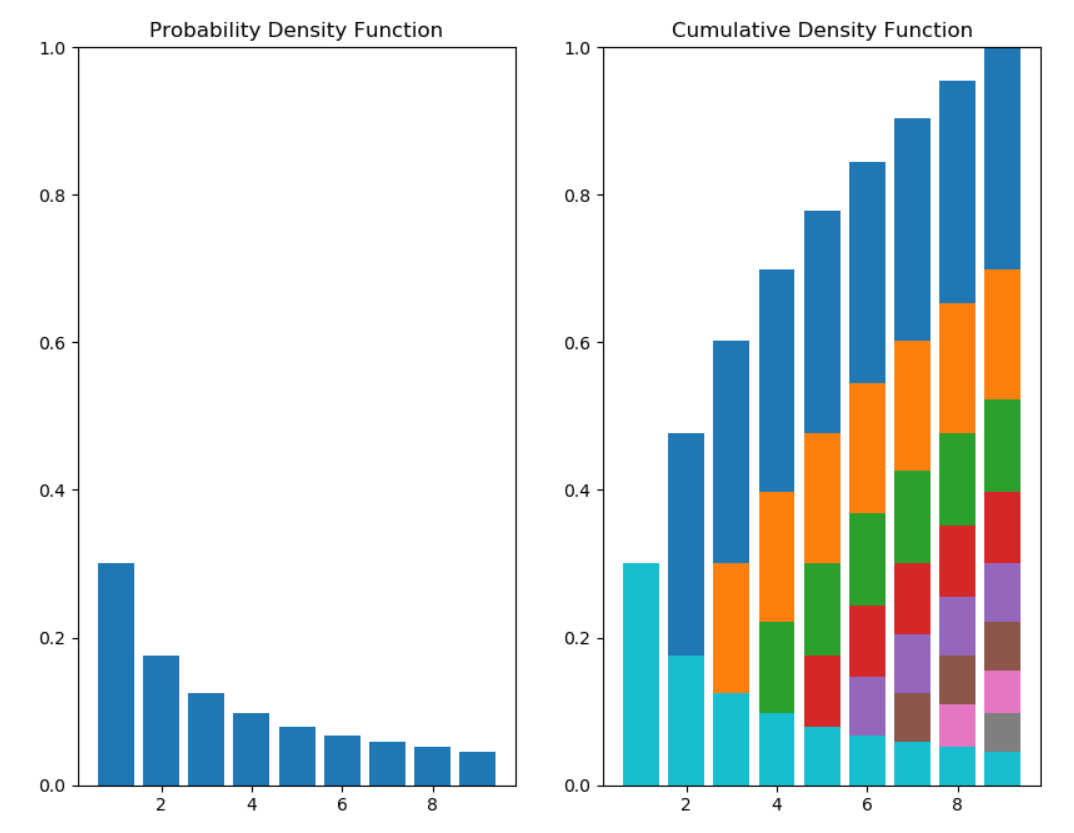
利用之前写的本福特定律的验证器验证结果：



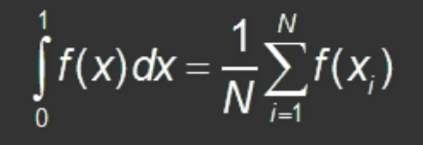
虽然不能完美的逼近理论值，但是基本符合了本福特定律。具有较高的相似度。

（5.16update）

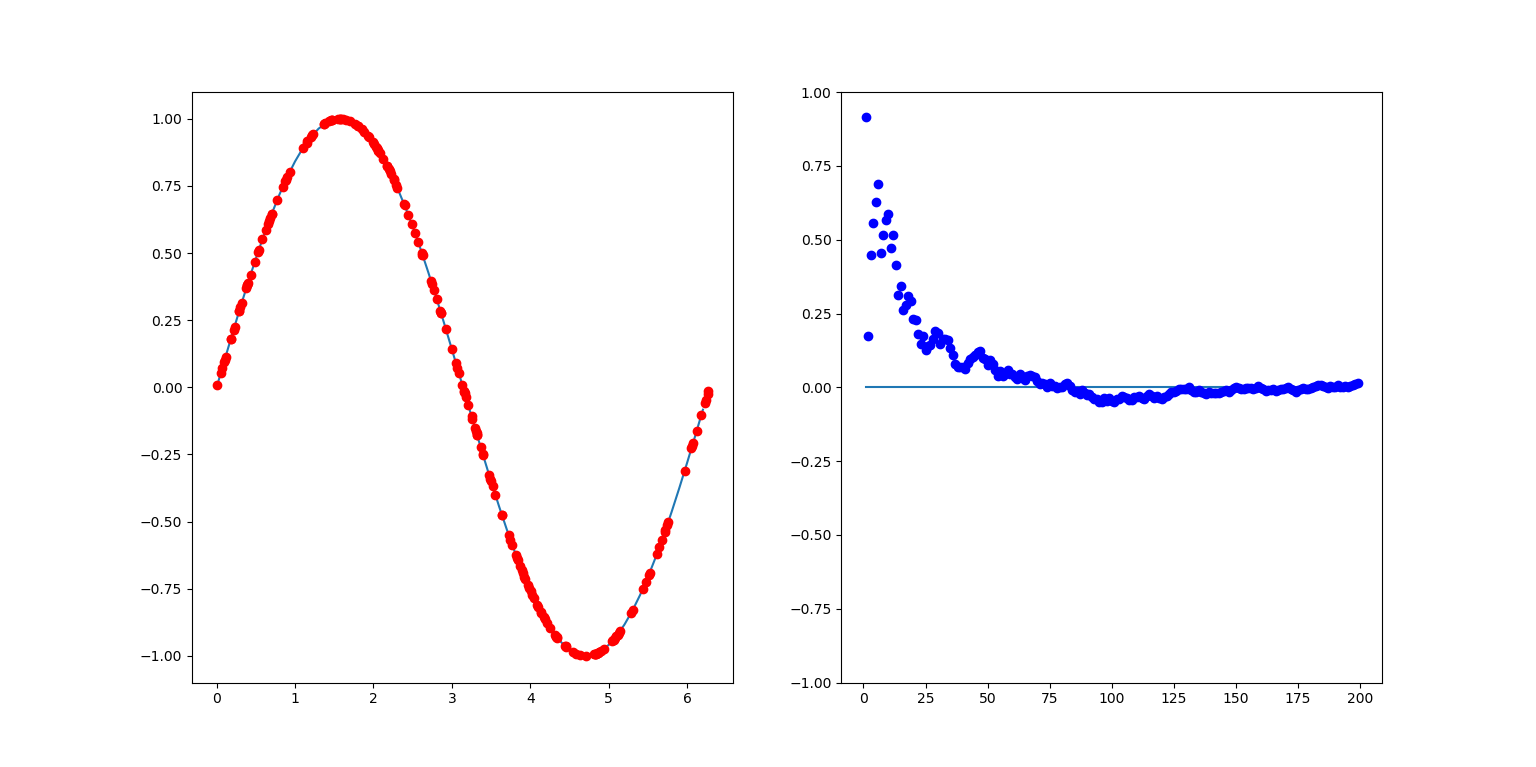
具体的算法就是将概率密度函数（PDF）积分（求前缀和）变为累计概率函数（CDF）然后在随机一个[0,1]的随机数，对CDF求反函数，映射回原来的数值。



1. **蒙特卡洛随机算法**

蒙特卡洛是著名的赌城，所以不少的随机算法都以蒙特卡洛命名，如蒙特卡洛搜索树，蒙特卡洛数值积分也是其中的一种，其思想简单粗暴，对于一个未知解析式的函数，随机投点计算函数的值，求平均数即可：

该方法收敛速度较一些有针对性的算法来说较慢，但是具有很高的普适性，可以应对各种不连续，斜率较大的函数。以下为蒙特卡洛算法计算sin(x)积分的收敛情况：

****

1. **参考资料**

<https://www.cnblogs.com/forget406/p/5294143.html>

<http://johnhany.net/2013/11/random-algorithm-and-performance/>

<https://www.cnblogs.com/lzxwalex/p/6880748.html>

<https://github.com/xlxwalex/lab102/tree/master/W7>

1. **附录**

**本福特定律生成**

import matplotlib.pyplot as plt

import numpy as np

import random

def update(s:list):

    x = [x + 1 for x in range(0, 9)]

    global arr

    plt.subplot(1, 4, 3)

    ran = random.random()

    res = 0

    while (ran >= s[res]):

        res += 1

    arr[res] += 1

    su = sum(arr)

    plt.plot(9, ran, "o", color="w")

    plt.plot(res+1,ran,"o",color ="w")

    plt.subplot(1, 4, 4)

    plt.cla()

    plt.ylim(0,1)

    plt.bar(x,[arr[i]/su for i in range(0,9)])

    plt.pause(0.01)

x = [x+1 for x in range(0,9)]

f = [0 for x in range(0,9)]

s = [0 for x in range(0,9)]

global arr

arr = [0 for x in range(0, 9)]

for i in range(1, 10):

    f[i-1] = np.log10(i + 1) - np.log10(i)

    s[i-1] = s[i - 2] + f[i-1]

al = [[s[j]-(0 if i==0 else s[i-1]) if j>=i else 0 for j in range(0,9)] for i in range(0,9)]

plt.subplot(1, 4, 1)

plt.title("Probability Density Function")

plt.ylim(0,1)

plt.bar(x,f)

plt.subplot(1, 4, 2)

plt.title("Cumulative Density Function")

plt.ylim(0, 1)

for i in range(0, 9):

    plt.bar(x, al[i])

plt.bar(x, f)

plt.subplot(1, 4, 3)

plt.ylim(0, 1)

for i in range(0, 9):

    plt.bar(x, [s[8 - i] if j>=8-i else 0 for j in range(0, 9)])

for i in range(0, 200):

    update(s)

plt.show()

**蒙特卡洛数值积分**

import numpy as np

import matplotlib.pyplot as plt

from matplotlib import animation

import random

def animation\_update(num):

    global point

    global res

    tmp = np.pi\*2\*random.random()

    res += np.sin(tmp)

    point.append(tmp)

    plt.subplot(1, 2, 1)

    plt.plot(tmp,np.sin(tmp),"o",color = "r")

    plt.subplot(1, 2, 2)

    plt.plot(num, res / len(point), "o", color="b")

    plt.pause(0.01)

if \_\_name\_\_ == "\_\_main\_\_":

    global point

    global res

    point = []

    res = 0

    sin\_x = np.arange(0, np.pi\*2,0.1)

    sin\_y = np.sin(sin\_x)

    sta\_x = np.arange(1, 200, 1)

    sta\_y = sta\_x \* 0

    fig = plt.subplots(2,2)

    plt.subplot(1,2,1)

    l1, = plt.plot(sin\_x, sin\_y)

    plt.subplot(1, 2, 2)

    plt.ylim(-1, 1)

    l2, = plt.plot(sta\_x, sta\_y)

    for i in range(1, 200):

        animation\_update(i)

    plt.show()

**扫描二维码下载资源：**

****