

计算物理第二次作业第 7 题

姓名：姚星宇 学号：PB21000188 班级：2021 级少年班学院四班

2022 年 10 月 18 日

1 题目重述

对一个实验谱数值曲线 $p(x)$ ，自设 $F(x)$ ，分别用直接抽样和舍选法对 $p(x)$ 抽样。比较原曲线和抽样得到的曲线以验证。讨论抽样效率

2 题目分析

本作业使用 Python (Anaconda 3) 进行代码编写。

观察原数据的图像，可得到以下信息：该分布大多集中在 3000eV 附近，若直接舍选抽样，效率会过地，故需要使用阶梯函数舍选抽样。（由于是离散数据，直接抽样可看作以 1 为间距的舍选抽样，故不再赘述）。

和 6 题类似，在抽样过程中，先将抽样区间分为 k 份 (k 大约为五) 并取整，然后穷举搜索到每个区间的最大值，将这个最大值作为该区间的阶梯函数的大小。然后对于阶梯函数积分（即将每个区域的最大值求和），得到与区间编号相对应的、阶梯函数的分布的数组：

$$bmax = [0, \max_{[0,a]}(f(x)), \max_{[0,a]}(f(x)) + \max_{[a,2a]}(f(x)), \dots, \sum_i^k \max_{[ia,ia+a]} f(x)] \quad (1)$$

对该数组进行归一化得到 $b0$ 。生成一个 $[0, 1]$ 之间的随机数 $ran1$ ，在 $b0$ 中进行二分查找，得到区间编号 i 使得 $b0[i] < ran1 < b0[i+1]$ 。然后在将 i 线性转换到这一区间 $i' = \text{int}(\frac{ran1-b[i]}{b[i+1]-b[i]} * (bmax[i+1] - bmax[i]) + bmax[i])$ 。

随后取得 $[0, 1]$ 之间的随机数 $ran2$ ，然后进行比较：如果 $ran2(b[i+1] - b[i]) < f(i')$ ，则为有效点，将 x 加入抽样结果；反之，则舍去 x 。

得到大量的 x 后，将 x 保留两位小数并借助 dictionary 容器进行频率统计，最后画出频数分布直方图并于理论曲线相比较。

对于直接抽样，只需令 $a = 1$ ，并不取第二个随机数而直接将 i' 加入结果即可。

3 代码

```
1  #-*- coding: utf-8 -*-
2  import numpy as np
3  import matplotlib.pyplot as plt
4  import math as m
5
6  def sch_random(N, M = 1, a = 16807, b = 0, m = 2**31 - 1, seed = 1):  
    #N 为生成  
    #个数，M 为生成间隔
```

```

7      q, r = m // a, m % a      #得到 p, r
8      for i in range(N):
9          for j in range(M):
10             seed = a * (seed % q) - r * (seed // q) #进行 schrage 方法
11             if seed < 0:
12                 seed += m
13             yield seed/m#单位化 schrage
14
15
16 def fun_sel_sam(function, f_random, a, b, k=5, N=100000, seed1 = 114514, seed2 =
17     1919810): #抽样函
18     数
19     result = []
20     temp = 0
21     bmax = np.zeros((k+1), dtype = 'double')
22     b0 = np.zeros((k), dtype = 'double')
23     const = (b-a)/k/100
24     for i in range(k):
25         temp = 0
26         for j in range(100):
27             if temp < function((i*100+j+0.5)*const+a):
28                 temp = function((i*100+j+0.5)*const+a)
29             b0[i] = temp
30             bmax[i+1] = temp
31             bmax[i+1] += bmax[i]
32     bmax = bmax/bmax[-1] #产生阶梯分布函数
33
34     brandom = f_random(N, seed = seed1)
35     for i in f_random(N, seed = seed2):
36         temp1, temp2 = 0, k
37         while temp2 - temp1 > 1 :
38             temp = (temp1+temp2)//2
39             if bmax[temp] > i:
40                 temp2 = temp
41             else:
42                 temp1 = temp
43         #二分查找得到位置
44         b_rand = next(brandom)*b0[temp1]

```

```

44         i = ((i-bmax[temp1])/(bmax[temp2]-bmax[temp1])+temp1)*(b-a)/k+a
45         if function(i)>b_rand: #判断是否保留
46             result+=[i]
47     return np.array(result)
48
49
50 def freq(c): #频数统计
51     d = {}
52     for i in c:
53         if round(i,2) in d:
54             d[round(i,2)] += 1
55         else:
56             d[round(i,2)] = 1
57     x , y = np.zeros((len(d)),np.zeros((len(d))))
58     j = 0
59     for i in sorted(d.items(),key = lambda x:x[0]):
60         x[j] = i[0]
61         y[j] = i[1]
62         j += 1
63     return x,y,len(c) #返回长度计算抽样效率
64
65
66 if __name__ == '__main__':
67     fig , ax = plt.subplots(1, 2, figsize = (8,2))
68
69     x,y,eff= freq(fun_sel_sam(lambda x:m.exp(-2*x**2), sch_random, a=-5, b
70                             =5,k=10,N=100000))
71     z = np.exp(-2*x**2) #产生标准曲线
72     y = y/y.max()*1.1
73     ax[0].plot(x, y, color = 'r', linestyle = '-', marker = 'None')
74     ax[0].plot(x, z, color = 'b', linestyle = '-', marker = 'None')
75     print('抽样效率:'+str(eff/100000))
76
77     x,y,eff = freq(fun_sel_sam(lambda x:1/(2*x**4+1), sch_random, a=-5, b=5,
78                             k=10,N=100000))
79     z = 1/(2*x**4+1) #产生标准曲线
80     y = y/y.max()*1.1
81     ax[1].plot(x, y, color = 'r', linestyle = '-', marker = 'None')

```

```
In [4]: runfile('C:/Users/YXY/7.py', wdir='C:/Users/YXY')
抽样效率: 0.34167
```

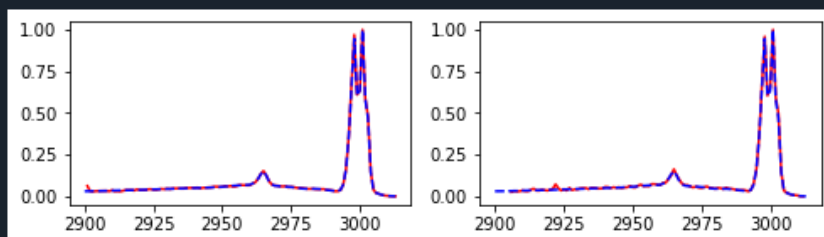


图 1: 程序运行结果截图

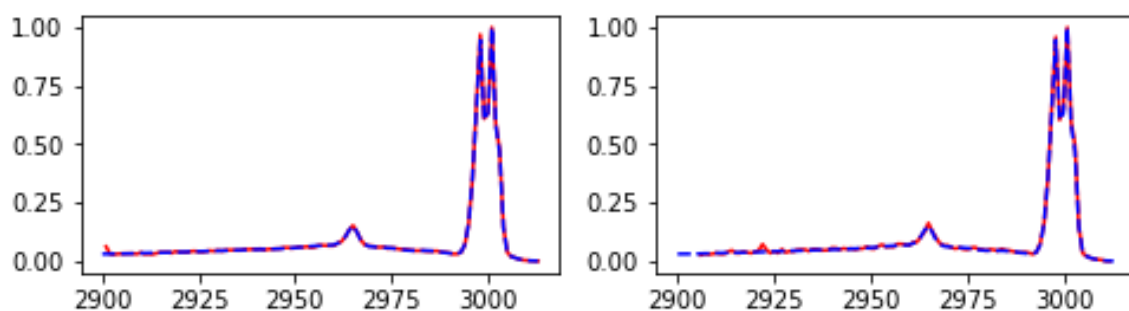


图 2: 得到的抽样图像与原始数据的对比

```
81 ax[1].plot(x, z, color = 'b', linestyle = '-', marker = 'None')
82 print('抽样效率: '+str(eff/100000))
```

4 结果

如图一所示，程序在 Anaconda3 环境下可以正常运行。

如图二所示，两图像基本重合，可认为抽样具有较好的符合度。

舍选法的抽样效率为 34%，较差，在这类数据量较小的情况下效率不如直接抽样。

参考文献

[1] 丁泽军. 计算物理讲义 [M]