计算物理 A 第六题

杨旭鹏 PB17000234

2019 年秋季

1 题目描述

对两个函数线型（Gauss 分布和类 Lorentz 型分布），设其一为 $p(x)$，另一为 $F(x)$，用舍选法对 $p(x)$ 抽样。将计算得到的归一化频数分布直方图与理论曲线 $p(x)$ 进行比较，讨论差异。讨论抽样效率。

$$Gaussian :\sim e^{-\frac{x^2}{2}} \qquad Lorentzian\ like :\sim \frac{1}{1+x^4} \tag{1}$$

2 算法

2.1 舍取法

所要实现的抽样是 $p(x) = \frac{1}{1+x^4}$，比较函数为 $F(x) = 1.1e^{-\frac{x^2}{2}}$。由于 $p(x)$ 为峰状结构函数，故我们采取变换抽样与舍取抽样相结合的方式来提高抽样效率。我们有：

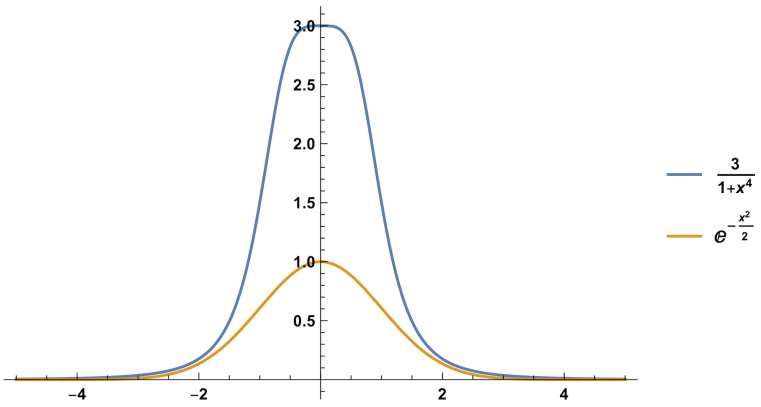

图 1: $p(x)$ 与 $F(x)$

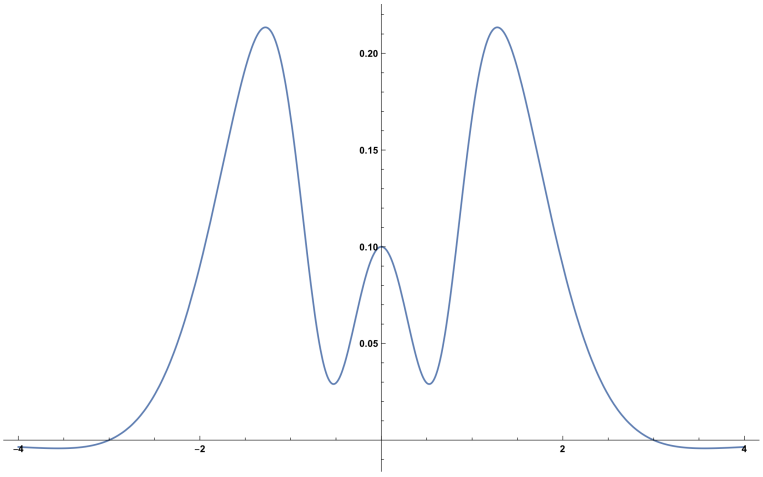

图 2: $F(x) - p(x)$

对 $F(x)$，其为高斯分布，我们可选定对其 3σ 区间即 $x \in [-3, 3]$ 抽样。在此区间满足 $F(x) > p(x)(F(3) - p(3) \doteq 0.0000247742 > 0)$，于是可采取变换抽样与舍取抽样相结合的方式对 $p(x)$ 进行抽样。对于其中的高斯函数部分，我们可以利用 Box-Muller 法进行抽样。而 Box-Muller 法的实现方法为：

1 随机抽样一对均匀分布的随机数，$u \in [0, 1]$ $v \in [0, 2\pi]$ ；

2 令 $x = \sqrt{-2lnu}cosv$

则得到概率密度函数 $c\ e^{-\frac{x^2}{2}}$ 的抽样。(其中 c 为一正常数，由于概率密度函数的绝对大小没有意义，只有相对大小有意义，故同样的抽样方法可对应相差为正常数倍的概率密度函数。) 再在 $[0, 1]$ 均匀抽取抽取点 η，则判断 $F(x)\eta$ 是否小于等于 $p(x)$，若小于，则取此 x 值，若不是，则舍。

2.2 16807 产生器

16807 产生器属于线性同余法产生器的特例。而线性同余法方法为：

$$\begin{aligned} I_{n+1} &= (aI_n + b)\ mod\ m \\ x_n &= I_n/m \end{aligned} \tag{2}$$

其中整数 $I_i \in [0, m-1]$，a, b, m 为算法中的可调参数，其选取直接影

响产生器的质量。选取参数：

$$\begin{cases} a = 7^5 = 16807 \\ b = 0 \\ m = 2^{31} - 1 = 2147483647 \end{cases} \tag{3}$$

即为所谓的 16807 产生器。由于直接利用2编写程序时计算 $(aI_n \ mod \ m)$ 时很容易造成数据溢出，故采取 Schrage 方法进行具体编程的实现：

$$aI_n \ mod \ m = \begin{cases} a(I_n \ mod \ q) - r[I_n/q], & if \geq 0 \\ a(I_n \ mod \ q) - r[I_n/q] + m, & otherwise \end{cases} \tag{4}$$

其中 $m = aq + r$，即 $q = m/a = 127773$，$r = m \ mod \ a = 2836$。即可利用此方法产生伪随机数序列。

3 程序使用方法

在运行程序后，会看到请求输入所需总随机点数的提示，按照提示在后面输入所需要的总随机点数，摁回车继续。然后经过计算在屏幕上给出舍选法的抽样效率并统计在 $[-3, 3]$ 上剖分的 o 个小区间上抽样点出现的概率到数据文件。程序输出完这些后会自动退出。

```
请输入您所需要的总点数：100000000
您输入的参数已接受，正在计算请稍等片刻~
抽样效率为：0.796780
Program ended with exit code: 0
```

图 3: 一个典型程序的运行示例

4 程序结果与讨论

当输入一些不同的点数时，得到如下结果[1]：

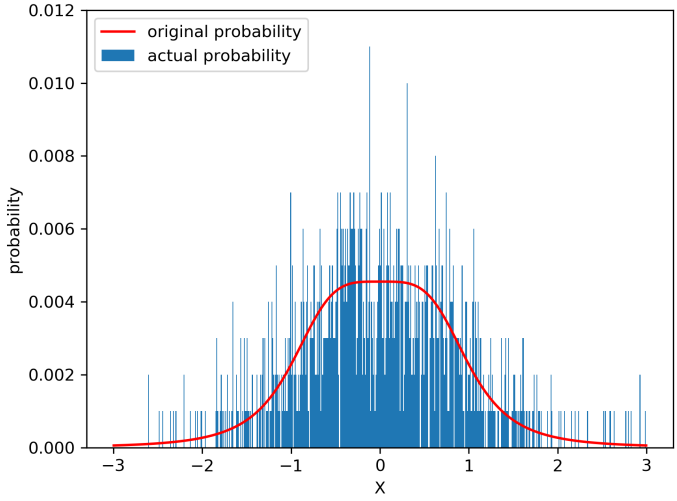

图 4: 产生 10^3 个点时的结果

[1]以下的结果是将 $[-3,3]$ 剖分为 600 个小区间得到的 (即每个小区间的长度为 0.01) 其中概率为在 $[-3,3]$ 之间归一化后的结果，例如对于在 $[x_i, x_{i+1}]$ 中出现的概率，对于抽样点来说即为：$p_i = \frac{n_i}{N}$(n_i 为在此区间内抽样得到的点数，N 为抽样总点数)；对于理论频率来说，即为：$p_i = \frac{\int_{x_i}^{x_{i+1}} p(x)dx}{\int_{-3}^{3} p(x)dx} \doteq \frac{p(\frac{x_2+x_{i+1}}{2})\Delta x}{\int_{-3}^{3} p(x)dx}$(其中 Δx 为区间长度)

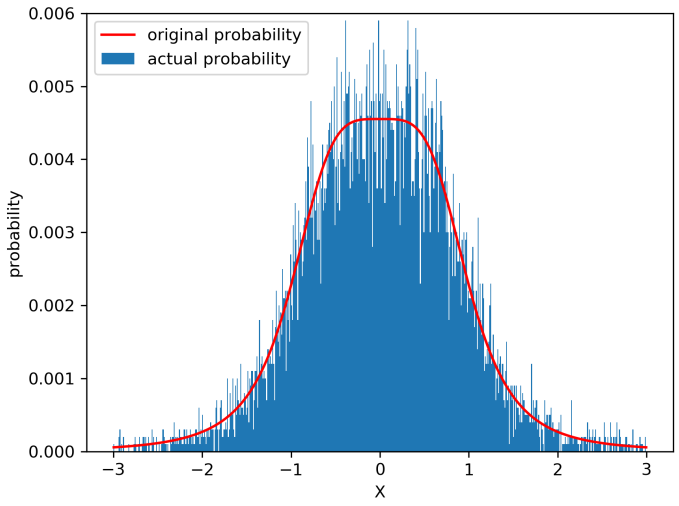

图 5: 产生 10^4 个点时的结果

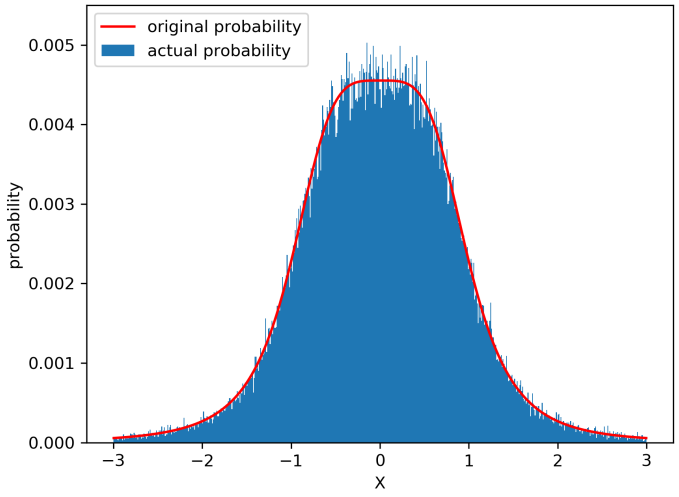

图 6: 产生 10^5 个点时的结果

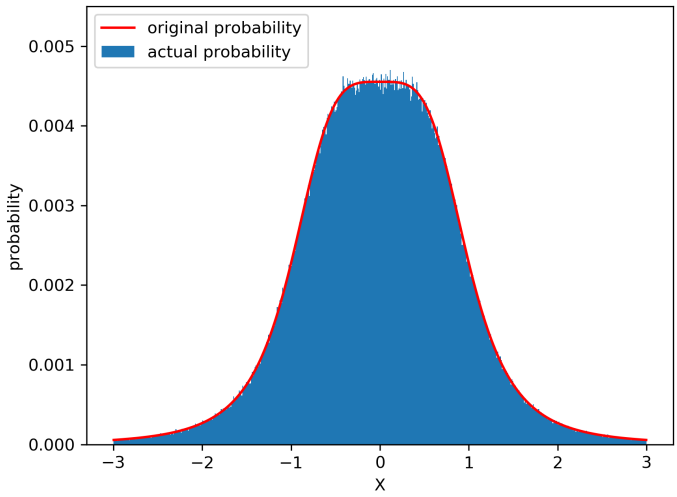

图 7: 产生 10^6 个点时的结果

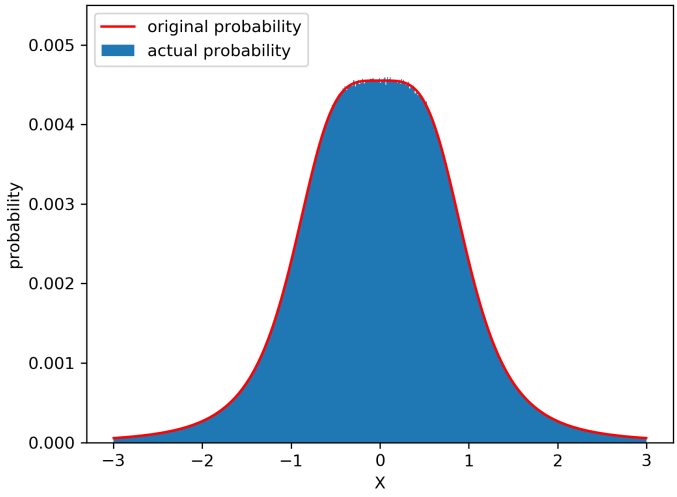

图 8: 产生 10^7 个点时的结果

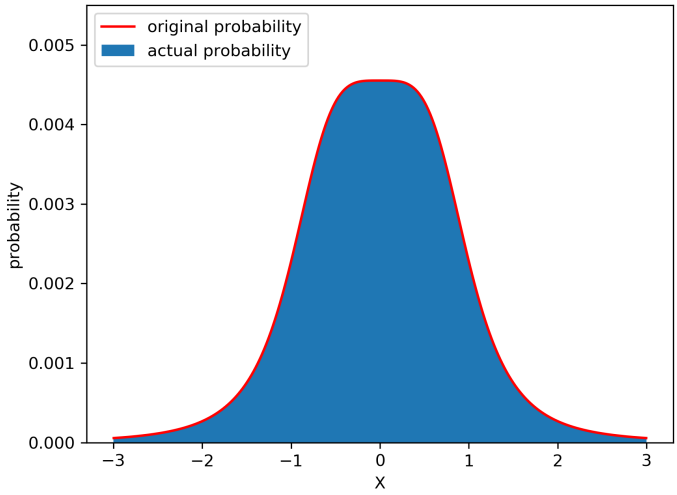

图 9: 产生 10^8 个点时的结果

由上述结果可以看出当总抽样点数 $N = 10^8$ 时，其概率分布已基本和理论概率分布重合，说明此抽样方法是成功的。而当 N 比较小时，实际抽样得到的概率分布会有“毛刺”现象，此为统计涨落造成的，特别对于 $N = 10^3$ 时，能明显看出有的剖分区间内甚至没有出现抽样点，而有些区间内抽样点出现的概率甚至是理论概率的 2 倍左右。

而对不同抽样点数的抽样效率如下：

	$N = 10^3$	$N = 10^4$	$N = 10^5$	$N = 10^6$	$N = 10^7$	$N = 10^8$
抽样效率	0.781861	0.791703	0.797486	0.796541	0.796881	0.796780

表 1: 抽样效率一览表

可以看出此方法的抽样效率还是蛮高的，而且对于不同的抽样点数基本稳定。

5 心得与体会

通过此次作业，了解了对于高斯分布，一般只需对其 3σ 区间抽样的约定。也更加熟悉了一般连续情形的舍选抽样方法。

通过编程作业，也更加熟悉了一些 C 语言和 LaTeX。

6 附录

A C 语言源程序

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
#define a 16807
#define b 0
#define m 2147483647
#define r (m%a)
#define q (m/a)
#define Pi 3.1415926
#define e 2.718281828
#define o 600 //统计概率所剖分区域的总数


int my_filewriter_double(char str[],double num[],int n){
    FILE * fp;
    fp = fopen(str,"w+");

    for(int i=0;i<n;i++)
    {
        if (i == (n-1)){
            fprintf(fp,"%lf",num[i]);
            break;
        }
        fprintf(fp,"%lf,",num[i]);

    }
    fclose(fp);
    return 0;
}

int my_filewriter_int(char str[],int num[],int n){
```

```
33      FILE * fp;
34      fp = fopen(str,"w+");
35
36      for(int i=0;i<n;i++)
37      {
38          if (i == (n-1)){
39              fprintf(fp,"%d",num[i]);
40              break;
41          }
42          fprintf(fp,"%d,",num[i]);
43
44      }
45      fclose(fp);
46      return 0;
47  }
48
49
50
51  // 舍选法抽样
52  int my_choose(int seed[], double ran[], int n){
53      double x;
54      double y;
55      double u,v;
56      int flag = 0; //记录总抽样次数
57      for (int j = 0; j <= n; ) {
58          flag++;
59          if (seed[0] >= 0) { //产生在[0,2,4199]间均匀抽取的\xi
60              u = (seed[0] / (double)m) ;
61          }
62          else{
63              u = ((seed[0] + m) / (double)m) ;
64          }
65
66          if (seed[1] >= 0) { //产生在[0,1]间均匀抽取的\eta
67              v = 2*Pi*(seed[1] / (double)m);
68          }
69          else{
70              v =  2*Pi*((seed[1] + m) / (double)m);
```

```
71          }
72          if (seed[2] >= 0) { //产生在[0,1]间均匀抽取的\eta
73              y = (seed[2] / (double)m);
74          }
75          else{
76              y =  ((seed[2] + m) / (double)m);
77          }
78
79
80          x = sqrt(-2*log(u))*cos(v);
81
82          if( x <= 3 && x >=-3 && y*1.1*pow(e,-x*x/2) <=
                (1/(1+pow(x,4))) ){ //舍选条件判断
83              ran[j] = x;
84              j++;
85          }//满足舍选法条件后写入数据
86
87          for(int k = 0;k<3;k++){
88              if(seed[k] == m-1){
89                  if(a >= b){   //由于Schrage方法只对z in
                        (0,m-1)成立，故这里要讨论z == m-1的情况
90                      seed[k] = m + (b-a) % m;
91                  }
92                  else  seed[k] = (b-a) % m;
93              }
94              seed[k] = ((a * (seed[k] % q) - r * (seed[k] / q)) + b %
                    m ) % m;
95          }
96      }
97
98      return flag;
99  }
100
101
102
103
104 int main() {
105     int N;
```

```
106     int flag; //记录舍选法抽样总次数
107     double p[o]; //记录在每(6/o)长度的区间内点出现的概率
108 
109     char str[50];
110     printf("请输入您所需要的总点数: ");
111     while (!scanf("%d",&N)) {
            //简单的输入检查，此时N变为需要产生的随机点的总数
112         gets(str);
113         printf("\nInput error,please try again\n");
114         printf("请输入您所需要的总点数: ");
115     }
116     if(N >1000000)
            printf("您输入的参数已接受，正在计算请稍等片刻~\n");
117     for(int i = 0;i<o;i++){ //数组初始化
118         p[i] = 0;
119     }
120 
121     int seed[3] = {809576131,-1025892587,558213681};
            //设置随机数产生器的初始种子值
122     double *ran = malloc(sizeof(double) * N);
            //用来存放舍选法产生的随机数
123 
124     flag = my_choose(seed, ran, N); //舍选法抽样
125 
126     printf("抽样效率为: %lf\n",(double)N/flag);
127     int j;
128     for(int i = 0;i<N;i++){ //对抽样点在每一剖分区间内出现概率进行计算
129         j = (int)floor((ran[i]+3)/6*o);
130         p[j] += (double)1/N;
131     }
132     my_filewriter_double("p.txt", p, o);
            //输出抽样点在每一剖分区间内出现概率至文件
133 
134     return 0;
135 }
```

B 可视化绘图 python 程序源码

```

import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
#from IPython.core.pylabtools import figsize # import figsize
#figsize(12.5, 4) # 设置 figsize
plt.rcParams['savefig.dpi'] = 300 #图片像素
plt.rcParams['figure.dpi'] = 300 #分辨率
# 默认的像素: [6.0,4.0], 分辨率为100, 图片尺寸为 600&400
fig = plt.figure()
ax1 = fig.add_subplot(111)
X = []
Y = []

with open('problem 6/p_108.txt', 'r') as f:
    while True:
        lines = f.readline() # 整行读取数据
        if not lines:
            break
        Y = [float(i) for i in lines.split(',')] # 将整行数据分割处理
    Y = np.array(Y) # 将数据从list类型转换为array类型。


X = np.arange(-2.995, 3.005, 0.01)

plt.bar(x=X, height=Y, width=0.01, label='actual probability')
ax1.legend(loc=1)
ax1.set_ylabel('probability')


X = np.arange(-2.995, 3.005, 0.01)
oY = 0.01/(1+X**4)/2.196879736
ax1.plot(X, oY, 'r', label='original probability')
ax1.legend(loc=2)
plt.ylim([0,0.0055])
```

```
plt.xlabel('X')


plt.savefig("2.png")
```