基于自然图像的随机数发生器

——物理科学与技术学院 193 班 10 号高旭

一．背景

目前，随机数已被广泛应用在不同的场景中，大体可以分为两种类型，一种是强调安全的场景，包括密钥生成、椭圆曲线签名中的随机数 k 的产生，销售终端（PoS） 算法中节点的选取；另一种是强调公平的场景，例如随机抽样、抽奖、抽签等。而且在统计学中我们也常常需要使用到随机数，比如进行蒙特卡罗模拟等。

虽然产生随机数有多种不同的方法。但是真正的随机数是利用物理现象产生的：比如掷钱币、骰子、转轮、使用电子元件的噪声等等，这样的随机数发生器叫做物理性随机数发生器，但是它们的缺点也很明显，就是对技术、时间和实验环境的要求比较高。

在实际应用中使用的大多都是伪随机数。它们是通过一个固定的、可以重复的计算方法产生的，具有类似于随机数的统计特征。但是不可否认的是在真正关键性的应用领域中需要使用真随机数。而本次实验则是向真随机数逼近了一步，利用一幅图像，可以简单、快速、安全地生成一个随机数的密钥。

二．原理

一幅 RGB 图像就是彩色像素的一个 M×N×3 数组，其中每一个彩色像素点都是在特定空间位置的彩色图像相对应的红、绿、蓝三个分量。图片在 MATLAB 中以无符号数组（数组中值范围为 0-255）的形式存在，且每一副自然图像的数组中的数值都是互不相同的，数组中的一组数就相当于一组随机数的密钥/随机数序列，于是我们可以通过对图像的数组中的数值进行提取，从而得到随机数。

三．基于自然图像的随机数发生器的设计

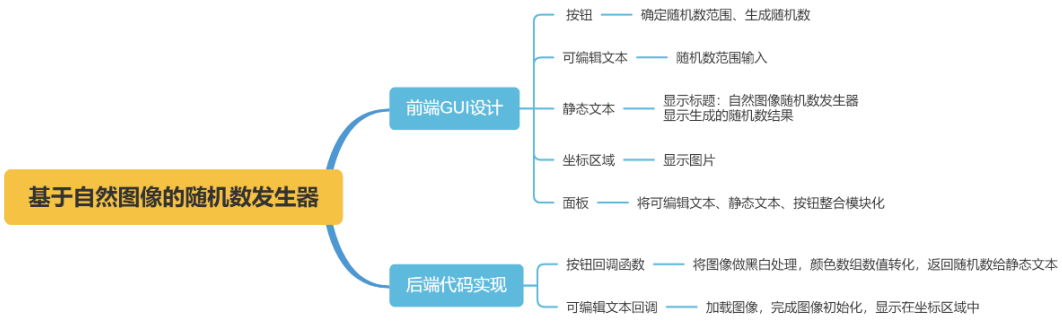

图 1-1 设计思路

(1) 前端 GUI 设计:

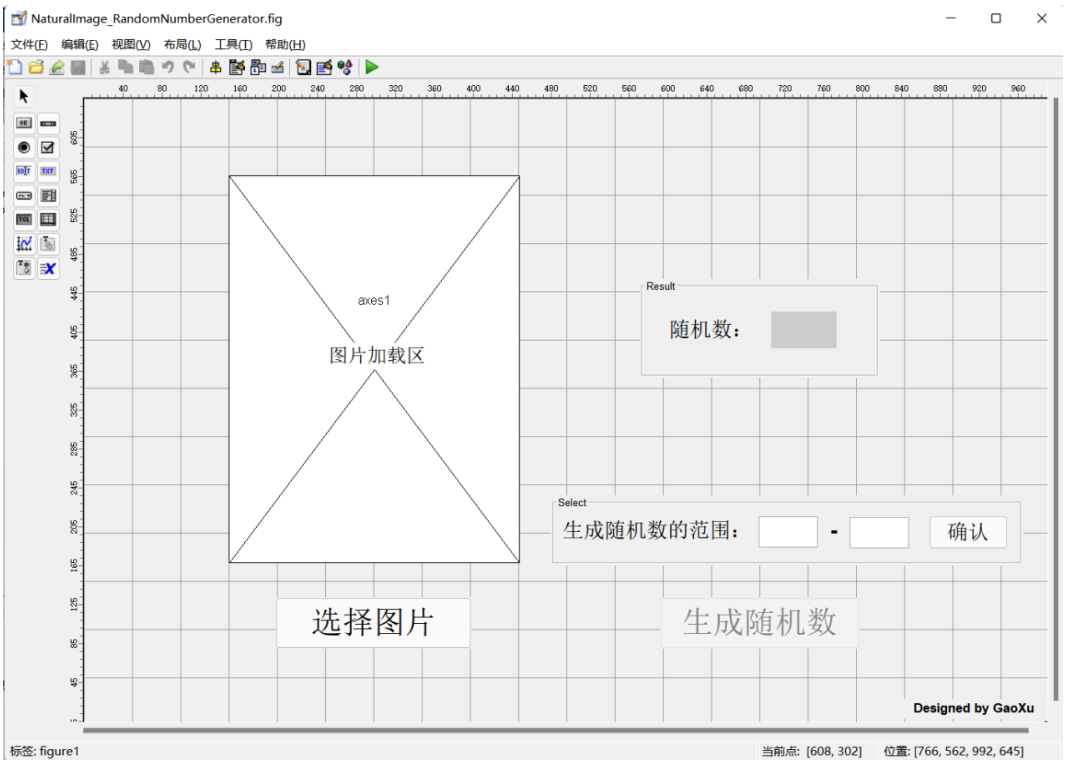

图 1-2 前端 GUI 设计

图 1-3 GUI 主界面

在初期的后续的优化中，将进度条显示时长调节为根据文件字节大小来进行显示，给用户更好的交互体验。

图 1-4 图片加载

图 1-5 随机数范围选取

图 1-6 随机数生成

为了提高交互效果，设计了在图片未成功添加和输入随机数生成范围有误时，会有警告窗口提示信息。

图 1-7（a） 图片未成功添加提示弹窗

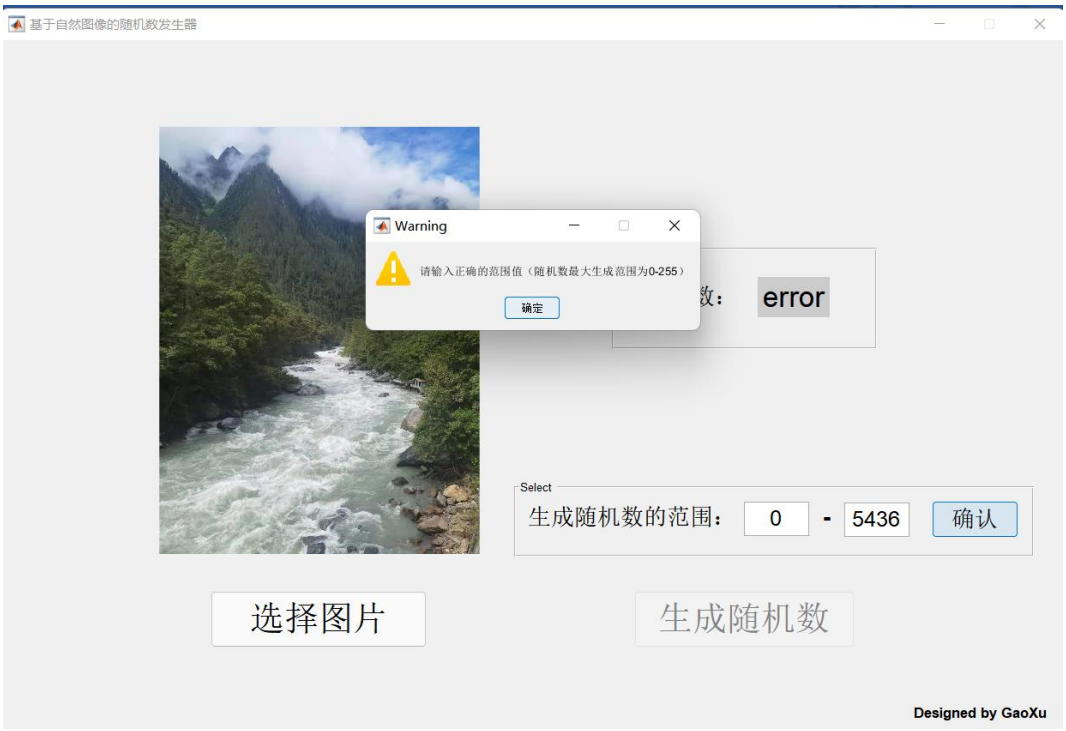

图 1-7（b）输入随机数生成范围有误提示弹窗

（2）后端代码实现：

```
function varargout = NaturalImage_RandomNumberGenerator(varargin)
% Last Modified by GUIDE v2.5 15-Nov-2021 16:10:54
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',          mfilename, ...
                   'gui_Singleton',   gui_Singleton, ...
                   'gui_OpeningFcn',
@NaturalImage_RandomNumberGenerator_OpeningFcn, ...
                   'gui_OutputFcn',
@NaturalImage_RandomNumberGenerator_OutputFcn, ...
                   'gui_LayoutFcn',    [] , ...
                   'gui_Callback',     []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function NaturalImage_RandomNumberGenerator_OpeningFcn(hObject,
```

```
eventdata, handles, varargin)
handles.output = hObject;
guidata(hObject, handles);
global pic_flag;
pic_flag=0;

function varargout =
NaturalImage_RandomNumberGenerator_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function pushbutton_createRandom_Callback(hObject, eventdata, handles)
global pic_flag;
global imgfile;
global leftboundary;
global rightboundary;
img_matrix = imread(imgfile);
y=img_matrix();
if pic_flag == 0
    set(handles.text_showResult,'String','error');
    warndlg('请添加生成随机数的图像源','Warning','modal');
else
    for i=1 : length(y)
        randomnum =unifrnd (1,length(y));
        temp = y(uint16(randomnum));
        if temp >= leftboundary && temp <= rightboundary
            result=num2str(temp);
            set(handles.text_showResult,'String',result);
        end
        if temp > rightboundary
            result = rem(temp,(rightboundary-leftboundary+1))+leftboundary;
            set(handles.text_showResult,'String',result);
        end
    end
end

function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function pushbutton_selectPic_Callback(hObject, eventdata, handles)
global imgfile;
global pic_flag;
[filename,pathname] =uigetfile({'*.jpg';'*.bmp';'*.jpeg';'*.*'},'载入图片');
if isequal(filename,0) || isequal(pathname,0)
```

```
    errordlg('没有选中文件','Warning');
else
    imgfile=[pathname,filename];
    progress_bar = waitbar(0,'图片加载中，请稍后...');
    imgdata = dir(imgfile);
    end_time = imgdata.bytes/1000;
    for i=1 : end_time
        waitbar(i/end_time);
    end
    close(progress_bar);
    set(handles.text_tip,'Visible','off');
    pic_flag = 1;
    img=imread(imgfile);
    axes(handles.axes1);
    axis off
    imshow(img);
end

function edit_selectRightBoundary_Callback(hObject, eventdata, handles)
global rightboundary;
input=str2num(get(hObject,'String'));
rightboundary=input;

function edit_selectRightBoundary_CreateFcn(hObject, eventdata, handles)

if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit_selectLeftBoundary_Callback(hObject, eventdata, handles)
global leftboundary;
input=str2num(get(hObject,'String'));
leftboundary=input;

function edit_selectLeftBoundary_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit9_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit10_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function pushbutton_confirmRange_Callback(hObject, eventdata, handles)
global leftboundary;
global rightboundary;

if (isempty(leftboundary)) || (isempty(rightboundary)) || (isempty(leftboundary))
&& (isempty(rightboundary))
    set(handles.pushbutton_createRandom,'enable','off');
    warndlg('请给出随机数范围的左、右边界值','Warning','modal');
else
    if leftboundary < 0 || rightboundary > 255 || leftboundary >= rightboundary
        set(handles.text_showResult,'String','error');
        set(handles.pushbutton_createRandom,'enable','off');
        warndlg('请输入正确的范围值（随机数最大生成范围为0-255）
','Warning','modal');
    else
        set(handles.pushbutton_createRandom,'enable','on');
    end
end
```

四．随机数的验证

为快速高效的验证，选择使用 Python 语言编写程序进行验证。

（1）随机数随机性验证

蒙特卡洛方法验证随机数：利用随机数生成器生成多组随机数，将其存放在 mydata.txt 文件中，执行如下代码加以验证。

程序代码如下：

```
import pylab as pl                                  # 导入 matplotlib 中的 pylab
fi=open("mydata.txt")
str=fi.read()
fi.close()
n=len(str)
print("n",n)
d=5                                                 # 间隔 d 个取一个数
max=1
for i in range(d):
```

```
        max*=10
max-=1
print(max)                                          # d 位数的最大值 99999（d=5）
length=int(n/d)
data=[1]*length
i,j=0,0
while j<length:
        data[j]=int(str[i:i+d])/max
            #  随机数与数的取法有联系，分开取数可以，连续取不可以
        i+=d
        j+=1
counter,num=0,length-1
for i   in range(num):
         x,y=data[i],data[i+1]
         if x*x+y*y<1:
                 pl.scatter(x,y,s=.1,c="r")
                 counter+=1
         pl.scatter(x,y,s=.1,c="b")
pl.text(0,0,"pi=%f"%(4*counter/num))
pl.show()
```

运行代码得到结果：

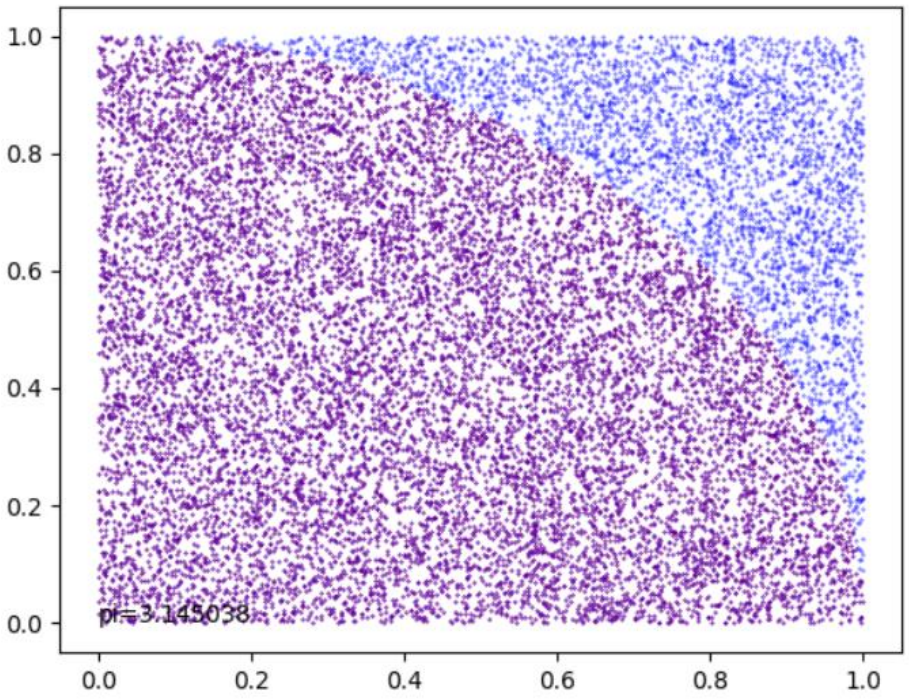

图 1-8 程序运行结果

在使用蒙特卡洛方法计算π值的 Python 程序生成的图像中，随机数对应的点均匀分

布，计算结果的π值为：3.145038，说明图像生成的随机数具有较好的随机性。

（2）随机数生成方法普遍性验证

重复 100 次蒙特卡洛验证实验，得到 100Π值，将其存放在 pi.csv 文件中。

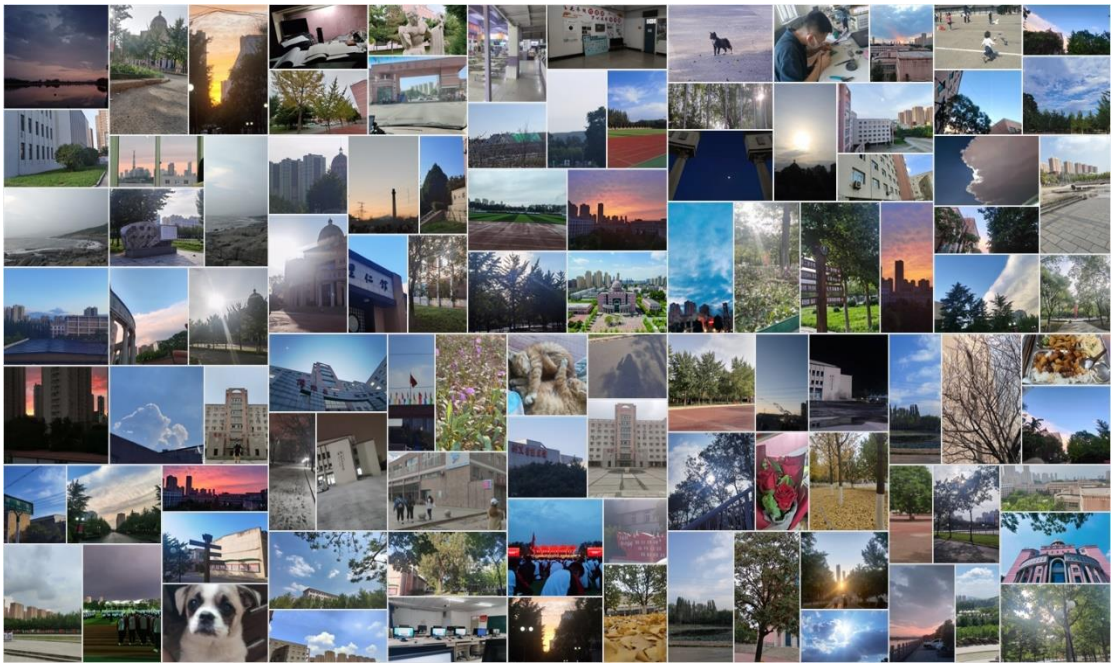

图 1-9 100 次实验的实验图像源

3.13992	3.14664	3.14344	3.1342	3.14768	3.12704	3.14296	3.14028	3.13936	3.13088
3.13156	3.12788	3.13828	3.14572	3.15224	3.12788	3.14448	3.13924	3.14284	3.14148
3.14652	3.1474	3.14396	3.14768	3.1498	3.14585	3.13542	3.13201	3.14198	3.15249
3.14243	3.13982	3.14273	3.13857	3.14356	3.12989	3.14115	3.13723	3.14179	3.14671
3.14028	3.14524	3.13144	3.13928	3.14464	3.14152	3.1398	3.1492	3.13904	3.14024
3.13764	3.14432	3.1452	3.13012	3.14158	3.13392	3.12776	3.13696	3.13892	3.13248
3.14884	3.13612	3.14604	3.14148	3.1512	3.14596	3.13392	3.13948	3.14573	3.14618
3.13597	3.14196	3.14044	3.14088	3.14312	3.14515	3.13816	3.14021	3.14336	3.14284
3.14528	3.14124	3.13796	3.1386	3.14228	3.13916	3.13392	3.14364	3.13392	3.1426
3.14428	3.14396	3.13984	3.13696	3.1446	3.13936	3.14552	3.14688	3.14352	3.13532

图 1-10 利用蒙特卡洛方法进行 100 次实验验证计算π值结果（pi.csv 数据）

代码如下：

```
import numpy as np
# 导入 numpy 包
import pandas as pd
# 导入 pandas 包
from scipy import stats
# 导入 scipy 包
import pylab as pl
# 导入 matplotlib 中的 pylab
df=pd.read_csv("pi.csv")
data=df["pi"]
```

```
bin=10
t=np.linspace(data.min(),data.max(),bin)
y=stats.norm(data.mean(),data.std())
# 产生与目标数据相吻合正态分布数据对象
pl.subplot(131)
pl.hist(data,10,edgecolor="y",density=True,label="reality")
# 绘制目标数据直方图
pl.title("histogram")
y2=y.rvs(100)
# 生成 100 个正态分布数据
p,t2=np.histogram(y2,bins=bin,density=True)
t2=(t2[:-1]+t2[1:])/2
pl.subplot(132)
pl.plot(t2,p,label="reality",color="b")
pl.plot(t,y.pdf(t),label="experience",ls="-.",color="r")
# 绘制概率密度图
pl.title("probability density")
pl.legend()
pl.subplot(133)
pl.plot(t,y.cdf(t),label="experience", ls="-.",color="r")
# 绘制累计概率密度图
pl.plot(t2,np.cumsum(p)*(t2[1]-t2[0]),label="realtiy",color="b")
pl.title("accumulate")
pl.legend()
pl.show()
```

运行代码得到结果：

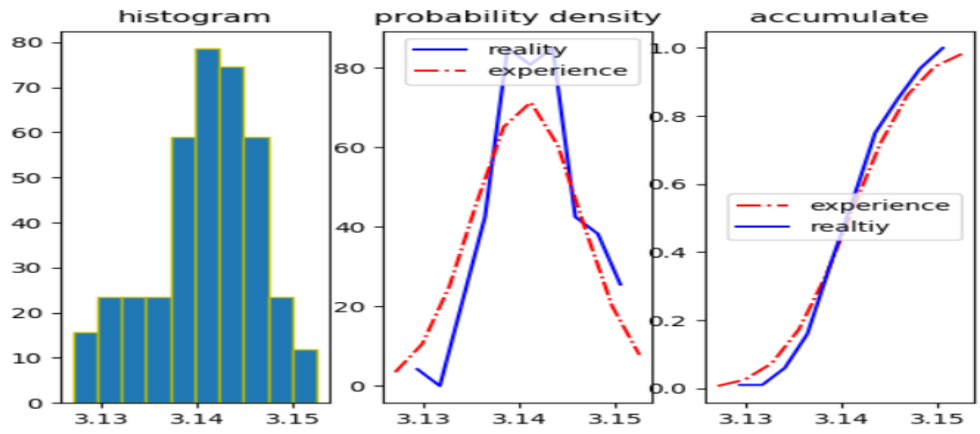

图 1-11 绘制 100 次实验结果与正态分布数据比较分析图

经过大量重复实验，可以看出：直方图中数据在 3.14 中心接近正态分布；实际概率密度曲线与理想概率密度曲线大体吻合；累计概率密度曲线基本重合，且接近到 1。说明该随机数生成方法具有普遍性。

五．基于自然图像的随机数发生器特点

1、取材方便，一幅自然图片甚至是一张人脸照片就可以作为生成随机数的材料；
2、生成速度快，利用 GUI 界面进行鼠标点击可直接操作处理，没有模数转换过程，不受外界变量控制；
3、成本低，几乎没有实验器材，只需要图片和计算机，得到的是优于数学算法的随机数；
4、个性化特点明显，调整不同的照片，就可以提取不同的序列号,确保随机数的唯一性。

六．参考文献

[1] Aj‘s Blog.科普:什么是随机数
[EB/OL].https://www.6zou.net/docs/what_is_random.Html.2021-9-10．
[2]百度百科.蒙特卡洛 [EB/OL].http://baike.baidu.com.2021-9-10．
[3]知乎.如何评价一个伪随机数生成算法的优劣
[EB/OL].https://www.zhihu.com/question/20222653.2021-9-10.
[4]郭弘,刘钰,党安红,等.物理真随机数发生器[J].科学通报,2009(23):3651-3657．
[5]于全福.基于量子效应的随机数产生研究[D].北京:北京大学.2013．
[6]钟志强.基于 Python 语言的大学物理实验数据处理方法研究[J].鞍山师范学院学报 2016，18(2):77-81．