大连理工大学

本科实验报告一

课程名称：随机信号分析实验

学院（系）：电子信息与电气工程学部

专　　业：电子信息工程

班　　级：电信 1806 班

学　　号：201871080

学生姓名：刘祎铭

2020 年 10 月 23 日

大连理工大学实验预习报告

学院（系）：电子信息与电器工程学部 专业：电子信息工程 班级：电信 1806

姓　　名：刘祎铭 学号：201871080 组：

实验时间：2020.10.28 实验室： 实验台：

指导教师：

实验 I：随机信号的产生、相关分析及其应用实验

一、 实验目的和要求

通过实验理解掌握随机信号样本生成的原理和方法、掌握随机过程相关函数的计算原理和方法。训练 MATLAB 程序代码编写能力，要求完成以下工作，并将实验结果与理论分析对照。

1.基于均匀分布伪随机数，掌握均匀分布白噪声典型生成方法。

2.基于均匀分布伪随机数，掌握高斯分布白噪声典型生成方法。

3.掌握随机信号相关函数计算、相关分析及实现方法。

二、实验原理和内容

1 实验原理

(1)均匀分布随机数

较简单的伪随机序列产生方法是采用数论中基于数环理论的线性同余法（乘同余法、混 合同余法），其迭代公式的一般形式为 f(x) = (r • x + b) Mod M，其离散形式为 s(n + 1) = [r • s(n) + b] Mod M。其中，s(n)为 n 时刻的随机数种子，r 为扩展因子，b 为固定扰动项，M 为循环模，Mod M 表示对 M 取模。为保证 s(n)的周期为 M，r 的取值应满足 r = 4k + 1， $M = 2^p$ ，k 与 p 的选取应满足：r < M，r(M-1) + 1< 2^{31} -1。通常公式中参数常用取值为 s(0) =12357，r = 2045， b = 1，M =1048576。

1.1 混合同余法：选定参数 a、c ，M， 再选定一个初值 y(0)，按下式产生随机数：

y (n+1) = a*y(n)+c (mod M) (n=1,2…)，x(n+1)=y(n+1)/M，其中 M 为足够大的整数. 在 C 语言中，$M = 2^{16}$，a、c 和 y(0)均为 0 至 M 中的常数。按上面算法公式，则 y(1),…,y(N)为（0，M）之间的均匀随机数，x(1),…,x(N)为（0,1）之间的均匀随机数。

1.2 乘同余法：在混合同余法算法中， 令 c=0，则为乘同余法。

(2)高斯随机数

a.变换抽样法，如果 X1、X2 是两个互相独立的均匀分布随机数，那么如下 Y1、Y2 是期 望为 m，方差为 σ^2 的高斯分布函数，且互相独立：$\begin{cases} Y_1 = \sigma\sqrt{-2\ln X_1}\cos(2\pi X \\ Y_2 = \sigma\sqrt{-2\ln X_1}\sin(2\pi X \end{cases}$为简单起见，可以令期望 m=0，方差为 σ^2=1。

b.较简单的高斯白噪声产生方法是基于概率论中的中心极限定理。即无穷多个同分布随机变量之和构成随机变量服从高斯分布。方便起见，可用 N 个（通常 N=12）均匀分布随机变量

之和 X_i 近似高斯分布随机变量。若 X_i，i = 0, 1, …, 11 在[0, 1)上服从均匀分布，则 $Y=\sum_{i=0}^{11}X_i-6$ 近似服从均值为 0, 方差为 1 的高斯分布。

(3)相关函数估计

离散随机序列自相关函数定义为 Rx(m)=E[x(n)x(n+m)]。对于各态历经随机过程，统计平均可用时间平均代替，即$R_x(m)=\lim_{N\to\infty}\frac{1}{2N+1}\sum_{n=-N}^{N}x(n)x(n+m)$。工程实践中，无法获得无限长数据，只能用有限平均来近似，即$R_x(m)=\frac{1}{(N-m)}\sum_{n=0}^{N-1}x(n)x(n+m)$，m=0, 1, …, K-1；K<N。为保证估计质量，通常要求 K<<N, 此时 Rx(m)也可以简化为$R_x(m)=\frac{1}{N}\sum_{n=0}^{N-1}x(n)x(n+m)$。

同理, 也类似地计算互相关函数。

（4）声音延迟及噪声分布律估计

对于噪声延迟信号，通过求解两信号的互相关函数峰值点得到延迟点数，再通过延迟点数除以采样频率即可得到信号的延迟时间。之后利用 matlab 的自带函数通过假设检验判断噪声信号的分布类型，提出检验假设，选择检验水准 α，通常为 0.05，即检验假设为真，但被错误地拒绝的概率。然后通过统计方法确定假设成立的可能性 P 的大小并判断，若 P>α，不拒绝 H_0，即认为差别很可能是由于抽样误差造成的；如果 P≤α，则拒绝 H_0，接受 H_1。

2 实验内容

(1) 编程实现产生 10000 个在(0, 1)区间均匀分布随机数。计算生成随机数的 1~4 阶矩，最大值，最小值，频度直方图。

(2) 编程实现产生 10000 个 N(3, 4) 高斯随机数。计算生成随机数的 1~4 阶矩，最大值，最小值，频度直方图。

(3) 编程实现产生 10000 个 N(1, 2) 高斯随机数和 10000 个 N(3, 4) 高斯随机数。计算其自相关函数，计算两个高斯随机信号的互相关函数。

(4)探究式实验内容

声音延迟及噪声分布律估计

研究随机信号相关分析方法在实际中的应用。针对指定的两路语音信号：已知其中一路是另外一路的延迟，延迟的时间未知，填充延迟时间的数据是标准的随机分布噪声数据，两路信号可能存在一定的干扰。要求找出两路信号之间的时间差，并且估计出延迟时间中随机数分布律及其参数。

声音延迟及噪声分布律估计

【研究内容】

研究随机信号相关分析方法在实际中的应用。

针对指定的两路语音信号：已知其中一路是另外一路的延迟，延迟的时间未知，填充延迟时间的数据是标准的随机分布噪声数据，两路信号可能存在一定的干扰。要求找出两路信号之间的时间差，并且估计出延迟时间中随机数分布律及其参数。

【实验布置】

两路信号文件：学号-采样 1.wav 和学号-采样 2.wav；

提示：可以用 audioread、wavread 等函数取得采样率和数据，或者用 kmplayer 等其他工具取得数据信息。

提示：可以用相关分析的方法，需要查询假设检验的一般方法。

提示：数据具有特异性，不能抄袭答案。

三、主要仪器设备

DELL 笔记本电脑，Windows10 专业版操作系统，Matlab R2016a

四、实验步骤

实验内容 1：

(1) 实现用同余运算产生 10000 个在(0, 1)区间均匀分布随机数。

```
M=1048576;
b=1;
r=2045;
first=12357;
num=10000;%定义初始值
s=zeros(1,num);
s(1)=first;
for i=2:num
    s(i)=mod(s(i-1)*r+b,M);%得到[0，M]之间的均匀分布随机数
end
s=s/M; %得到[0，1]之间的均匀分布随机数
```

(2) 计算生成随机数的 1~4 阶矩，最大值，最小值，用 hist()函数画频度直方图。

```
n=zeros(1,4);%生成[0 0 0 0]
for i=1:10000
    n(1)=n(1)+s(i);%均值
    n(2)=n(2)+s(i)^2;%二阶矩
    n(3)=n(3)+s(i)^3;%三阶矩
    n(4)=n(4)+s(i)^4;%四阶矩
end
n=n/10000;
disp(['自编随机数的各阶矩',num2str(n),'。']);
Max=max(s);disp(['自编最大值',num2str(Max),'。']);
Min=min(s);disp(['自编最小值',num2str(Min),'。']);
```

(3) 利用 ksdensity 函数估计概率密度并与预设好的[0.1]均匀分布进行比较。

```
% ksdensity 函数
[f,xi]=ksdensity(s);
figure,subplot(2,1,1),plot(xi,f);
```

```
title('利用 MATLAB 函数 ksdensity 估计的概率密度')
%预设的[0 ,1]均匀分布概率图
t=-0.2:1/10000000:1.2;
ff=ones(size(t)).*(t<=1&t>=0)+0*(t<0&t>=-0.2)+0*(t<=1.2&t>1); %利用分段函数画出预设的[0 ,1]均匀分布概率图
subplot(2,1,2),plot(t,ff);
title('预设的[0 ,1]均匀分布理论概率图')
```

实验内容 2：

(1) 用变换法和中心极限法编程实现产生 10000 个 N(3, 4)高斯随机数。

```
%变换法高斯
x=randuniform(M,b,r,first1,num);
y=randuniform(M,b,r,first2,num);
y1=sqrt((-2)*log(x)).*cos(2*pi*y);
s=a*y1+m;
figure
hist(s,100);
title('变换法高斯');
%中心极限定理高斯
x=randuniform(M,b,r,first,num*12);
% x = rand(1,num*12);
y = reshape(x,12,num);
y1 = sum(y)-6;
% 线性变换，加上均值和标准差，返回两路高斯。
s = a*y1+m;
figure
hist(s,100);
title('中心极限定理高斯');
%% 产生高斯随机变量
m=3;
a=2;
num=10000;
s=rnd1(m,a,num);
y=rnd2(m,a,num);
```

(2) 计算生成随机数的 1~4 阶矩，最大值，最小值，并与预设值进行比对验证。

```
%比对验证
meanValue = mean(y);
stdValue  = std(y);
disp('----------')
```

```
disp(['预设参数，均值为：',num2str(m),',标准差为：',num2str(a)]);
disp(['计算参数，均值为：',num2str(meanValue),',标准差为：',num2str(stdValue)]);
meanErr = (meanValue - m)/(m)*100;
stdErr  = (stdValue - a)/(a)*100
disp(['相对误差分别为：',num2str(meanErr),' %, 和：',num2str(stdErr),' %'])
disp('两者相近。从直方图和低阶矩上看，基本符合要求。')
```

(3) 画出数据的点，线，用 hist()函数画频度直方图，并利用 ksdensity 函数分别画出概率分布密度图并与预设好的 N(3,4)高斯分布进行比较。

```
%预设的 N（m，a）高斯分布理论概率图
t=-50:1/100:50;
k=1/(sqrt(2*pi)*a)*exp(-(t-m).^2/(2*a*a));
subplot(3,1,3),plot(t,k);
title('预设的 N（m，a）高斯分布理论概率图')
xlim([-6,12]);
```

实验内容 3：

(1) 编程实现产生 10000 个 N(1, 2)高斯随机数和 10000 个 N(3, 4)高斯随机数。

(2) 画出两个高斯信号的自相关函数，并计算出其峰值。

```
[zxg1,n1]=ycorr(y1,y1,'coeff');
[mx,indx]=max(zxg1);
title(['最大值点位于',num2str(abs(indx-num))]);
hold on
subplot(2,1,2);plot(n2,zxg2);
```

(3) 画出两信号的互相关函数，并计算出峰值。

实验内容 4：探究性实验

(1) audioread()等函数取得采样率和数据。

```
%% 读取数据
[filename1,filepath1]=uigetfile('.wav');
audeofile1= strcat(filepath1,filename1);
[y1,Fs1] = audioread(audeofile1);
[filename2,filepath2]=uigetfile('.wav');
audeofile2= strcat(filepath2,filename2);
%取得采样率和数据
[y2,Fs2] = audioread(audeofile2);
%获取信号长度
num=length(y1);
```

(2) 用求互相关函数最大值的方法找到延迟点数，并计算延迟时间。

```
[x,lags]=xcorr(y1,y2,'coeff');
figure;plot(lags,x)
title('两路同源信号的互相关，峰值处最相关');
[mx,indx] = max(x);
TLag = abs(num - indx);% 延迟点数
```

```
disp(['经计算相关函数，估计延迟点数为：',num2str(TLag),'。'])
TsLag=TLag/Fs1*1000; %延迟时间
disp(['估计延迟时间为：',num2str(TsLag),'ms。'])
```

(3) 截取出噪声信号。

```
noise=y2(1:TLag);
```

(4)计算噪声信号的数字特征，画出其直方图，概率密度分布图，进行初步判断。

(5)猜想填充噪声为高斯分布，用假设检验证明猜想成立，不成立再依次检验其他分布的可能性。

```
p_judge(noise,0.05);%调用函数，定义 0.95 的置信水平

function p_judge(A,alpha)
[mu,sigma]=normfit(A); %求解噪声信号的均值和方差
p1=normcdf(A,mu,sigma); %求噪声信号的概率分布
[H1,s1]=kstest(A,[A,p1],alpha); %利用 kstest 函数比对噪声信号频率分布与理论分布，来检验是否满足正态分布
n=length(A);
if H1==0
disp('该数据源服从正态分布。')
else
disp('该数据源不服从正态分布。')
end
if H1==1
[mu,sigma]=unifit(A);
p1=unifcdf(A,mu,sigma);
[H6,s6]=kstest(A,[A,p1],alpha);%进行了均匀分布的检验
n=length(A);
if H6==0
disp('该数据源服从均匀分布。')
else
disp('该数据源不服从均匀分布。')
end
if H6==1
phat=gamfit(A,alpha);
p2=gamcdf(A,phat(1),phat(2));
[H2,s2]=kstest(A,[A,p2],alpha);
if H2==0
disp('该数据源服从γ分布。')%进行了γ分布的检验
else
disp('该数据源不服从γ分布。')
end
if H2==1
lamda=poissfit(A,alpha);
p3=poisscdf(A,lamda);
[H3,s3]=kstest(A,[A,p3],alpha);
if H3==0
```

```
disp('该数据源服从泊松分布。') %进行了泊松分布的检验
else
disp('该数据源不服从泊松分布。')
end
if H3==1
mu=expfit(A,alpha);
p4=expcdf(A,mu);
[H4,s4]=kstest(A,[A,p4],alpha)
if H4==0
disp('该数据源服从指数分布。') %进行了指数分布的检验
else
disp('该数据源不服从指数分布。')
end
if H4==1
[phat, pci] = raylfit(A, alpha);
p5=raylcdf(A,phat);
[H5,s5]=kstest(A,[A,p5],alpha);
if H5==0
disp('该数据源服从 rayleigh 分布。') %进行了 rayleigh 分布的检验
else
disp('该数据源不服从 rayleigh 分布。')
end
end
end
end
end
end
```

五、实验数据记录和处理

1.均匀随机数

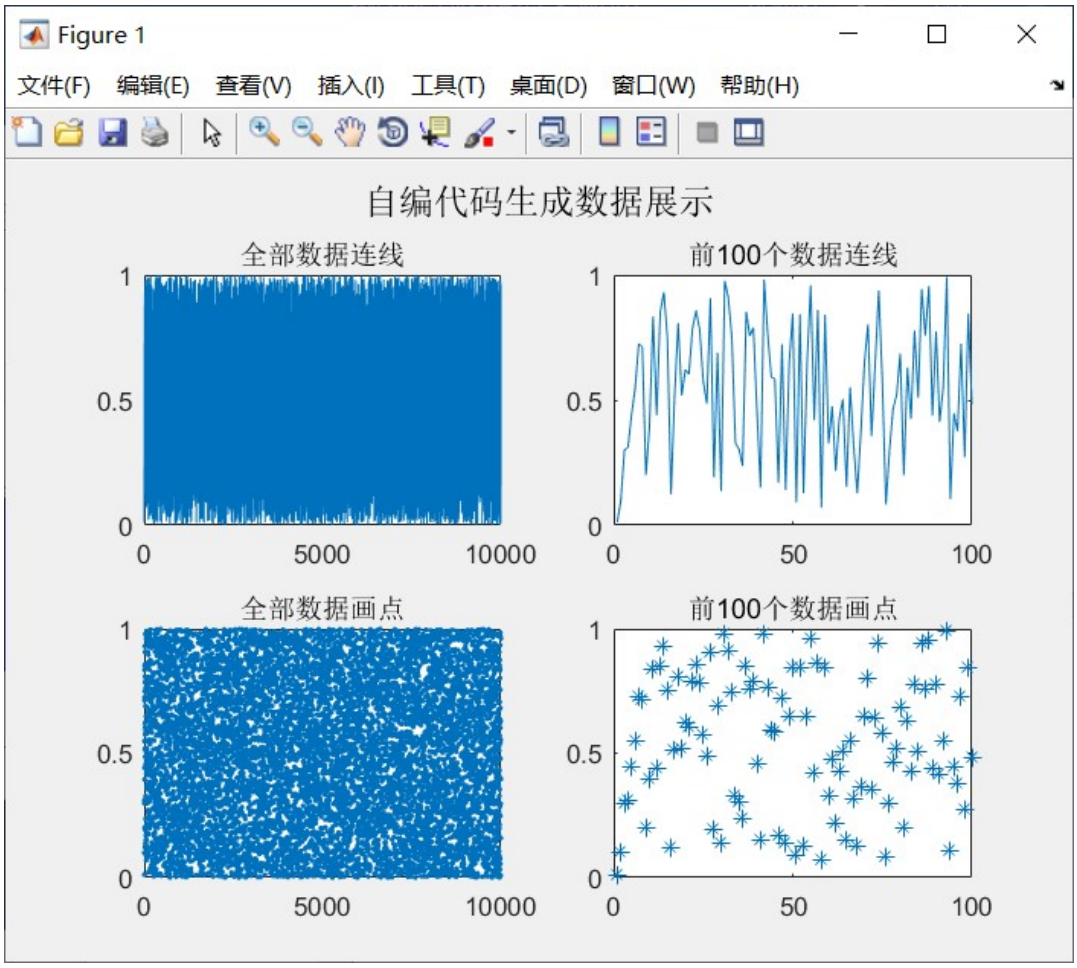
自编随机数的各阶矩

一阶矩	二阶矩	三阶矩	四阶矩	最大值	最小值
0.49724	0.33062	0.24762	0.19795	0.99986	7.6294e-06

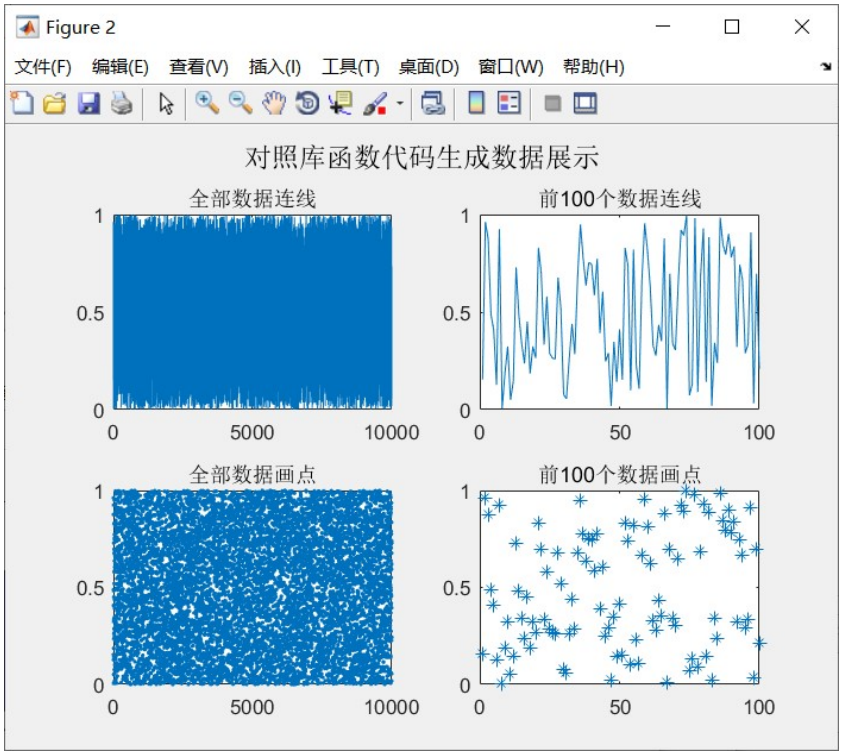
对照库函数随机数的各阶矩

一阶矩	二阶矩	三阶矩	四阶矩	最大值	最小值
0.49628	0.33012	0.24748	0.198	0.99997	7.2029e-05

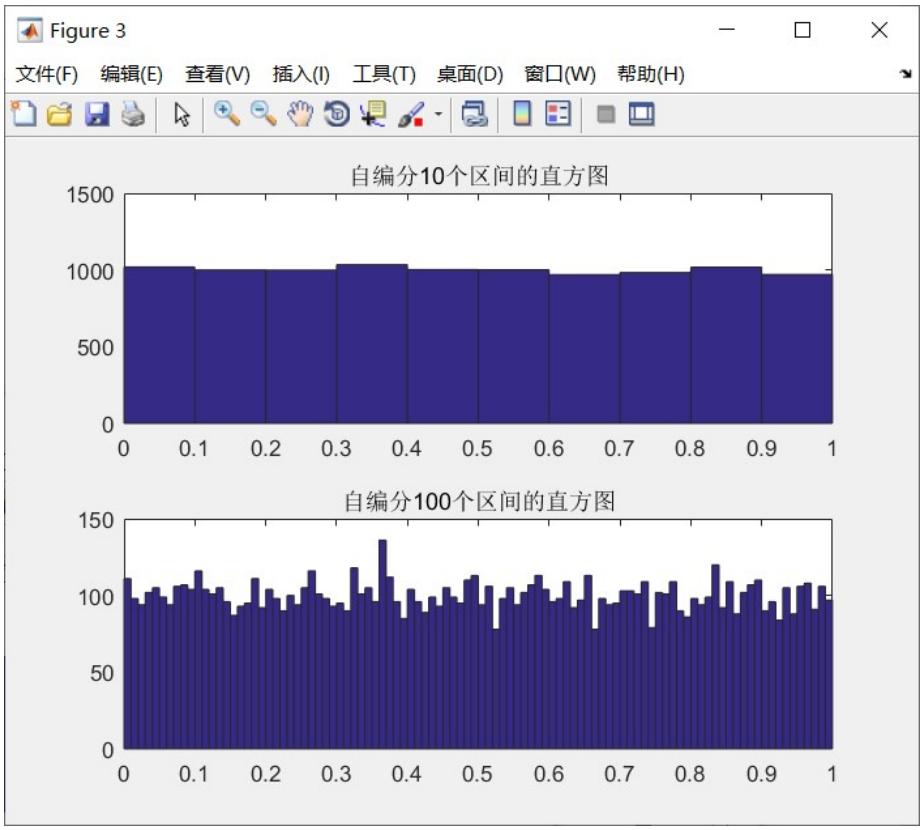
（1）混合同余法产生的 10000 个在[0,1]区间均匀分布随机数的点和连线图



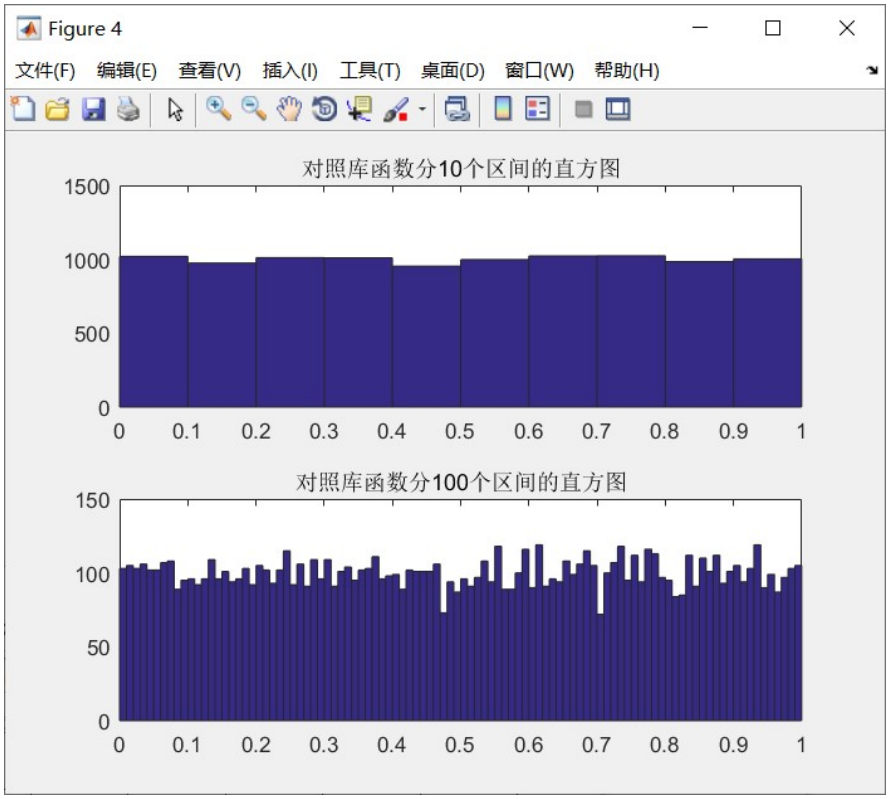
对照库函数产生的 10000 个在[0,1]区间均匀分布随机数的点和连线图



(2)混合同余法产生的 10000 个在[0,1]区间均匀分布随机数的直方图

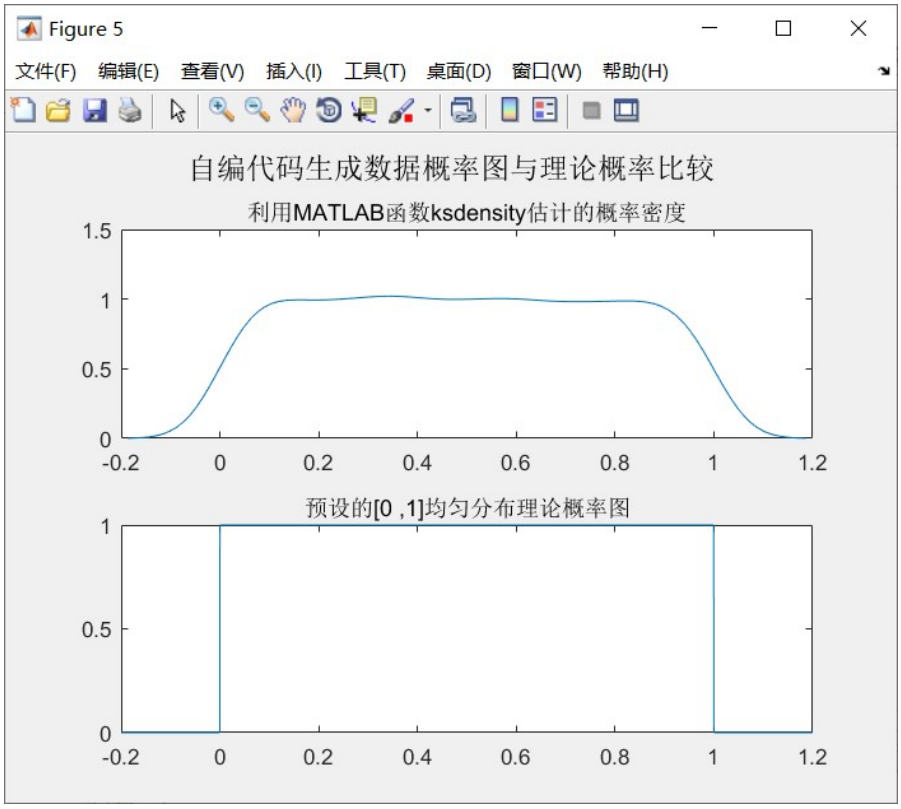


对照库函数产生的 10000 个在[0,1]区间均匀分布随机数的直方图

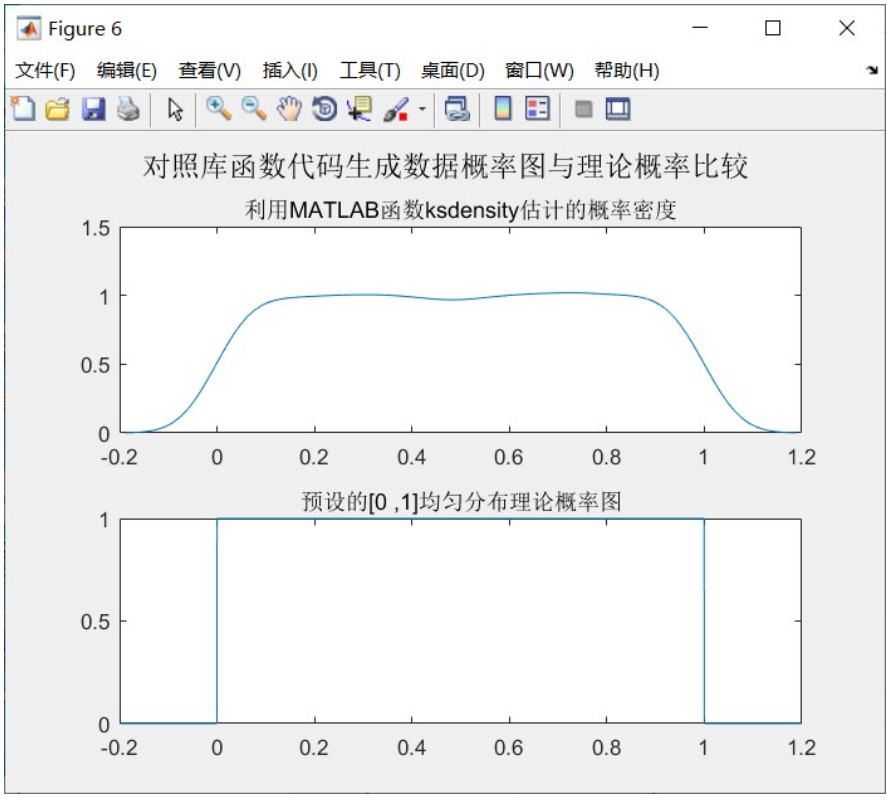


可以观测到点在 0,1 间分布均匀，且直方图显示在[0,1]各个区间分布密度几乎相同。

(3)利用 ksdensity 函数估自编函数产生的随机信号的概率密度并与预设理论图对比



（5）利用 ksdensity 函数估计的对照库函数产生的均匀随机数概率密度与预设理论图对比



通过与理论概率分布进行比较可知，在实际上（0,1）区间上的均匀分布应该只在（0,1）区间上密度为 1,而在其余的区间应均为 0。用 ksdensity 分布却是在（-0.2,1.2）上且在（0，1）上概率并不完全为 1，这是由于 ksdensity 函数是平滑密度估计，是无法应对在-1 和 1 处的突变情况的，它在拟合曲线时有一个从 0 至 1 的渐变的过程，所以才出现了（-0.2,0）和（1,1.2）的缓冲带。Ksdensity 本质上是对缺少的点进行适当补足，是对一小段区间的平均补充，因为在 1 附近大于 1 概率密度是 0，所以小于 1 处的概率要减小，而大于 1 时正好相反。实际的数据中并没有小于 0 以及大于 1 的数据存在，满足均匀分布条件。

2.高斯随机数

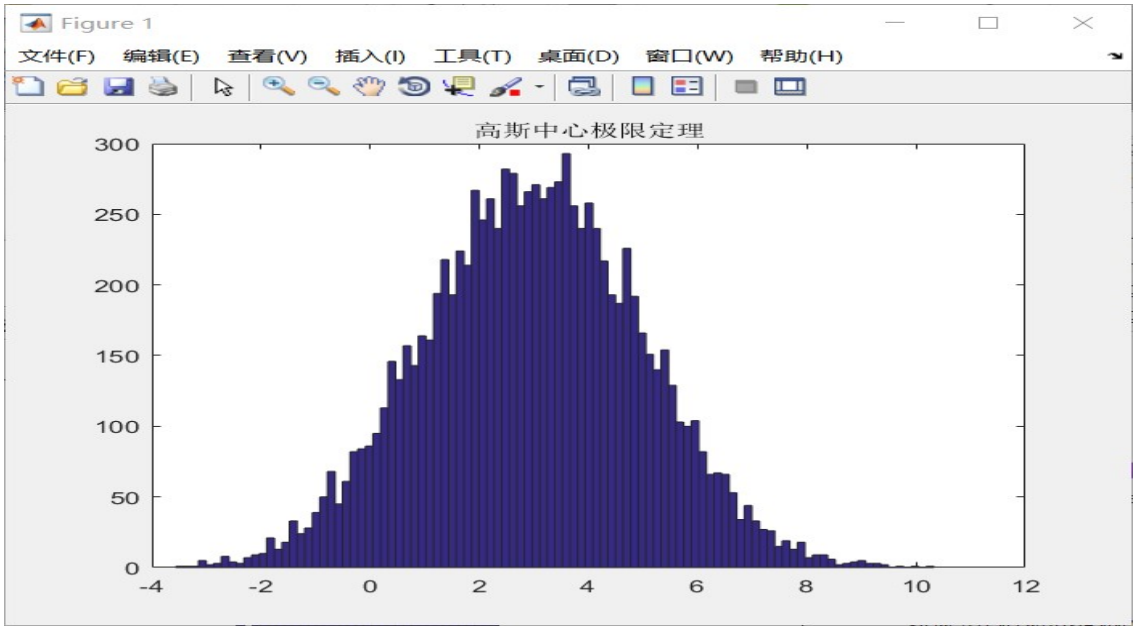
中心极限累加法

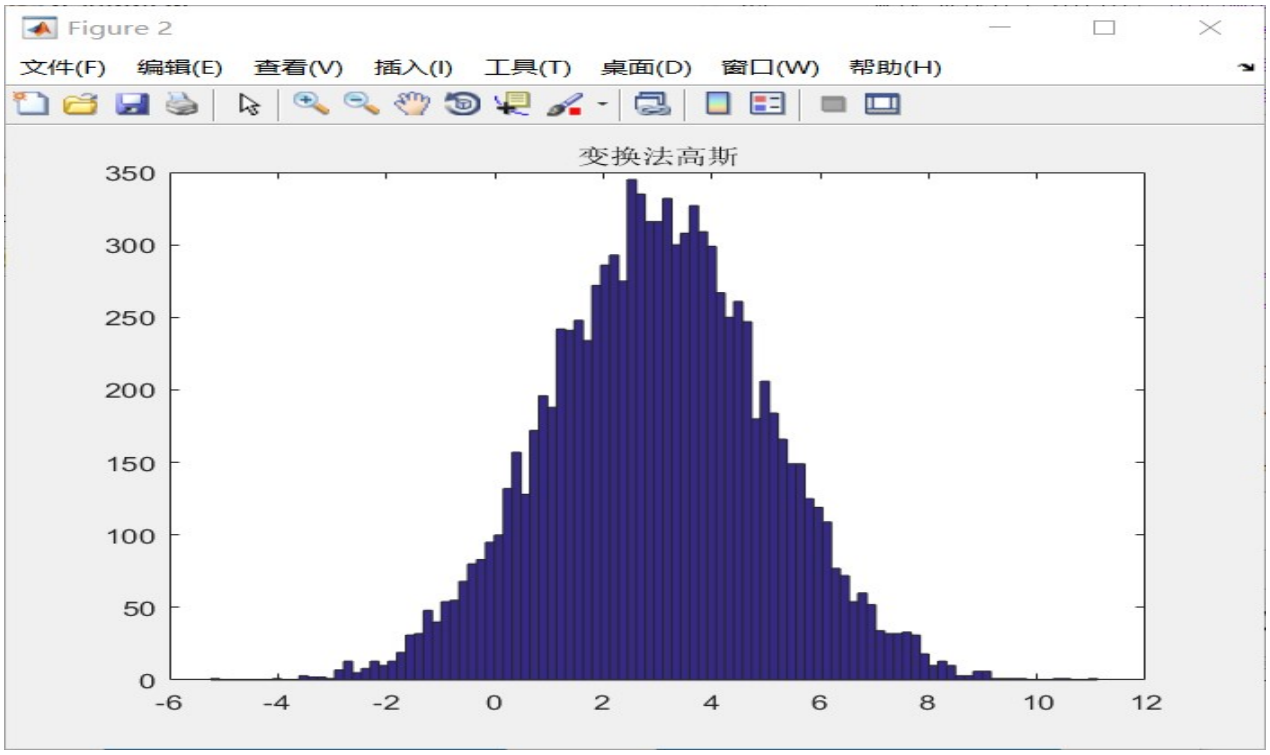
一阶矩	二阶矩	三阶矩	四阶矩	最大值	最小值
2.9808	12.82	61.7863	334.6656	10.3253	-3.562

变换法

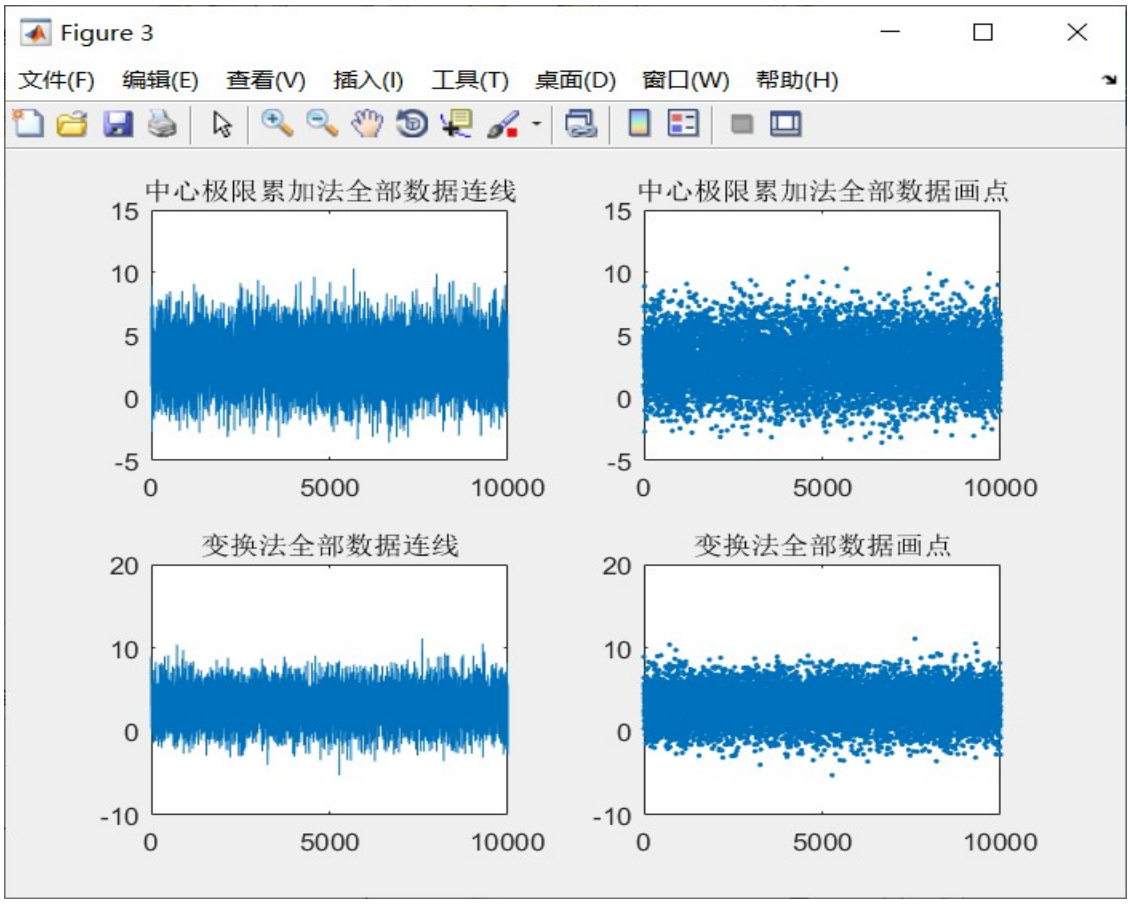
一阶矩	二阶矩	三阶矩	四阶矩	最大值	最小值
3.013	13.1628	64.2087	353.7376	11.1127	-5.239

（1） 直方图

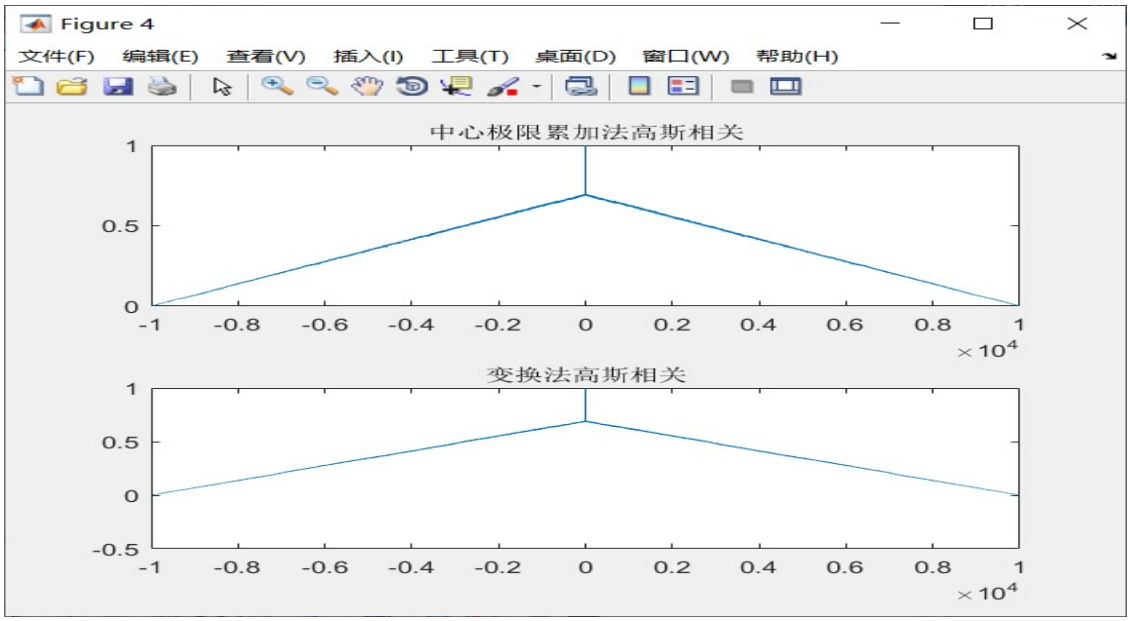


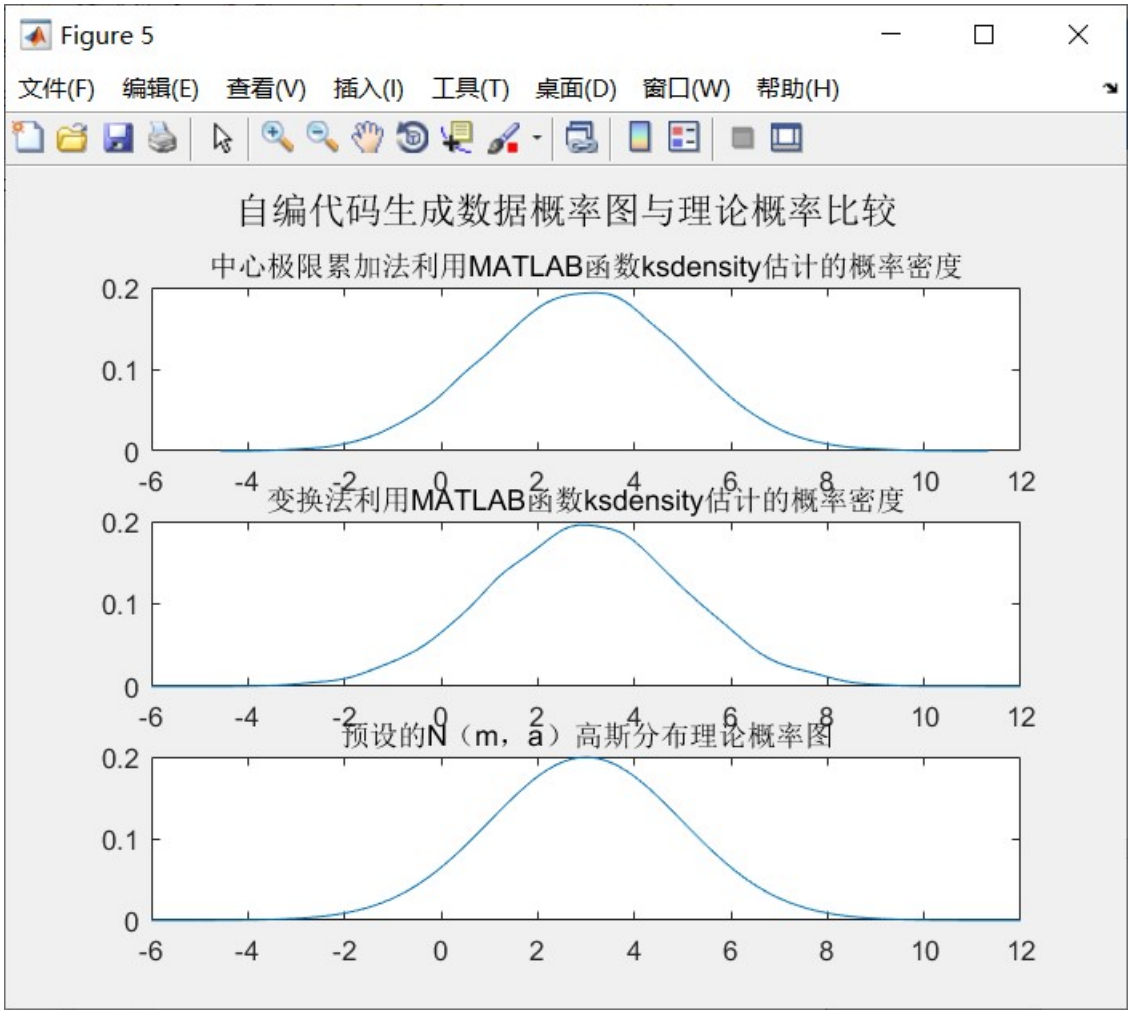


上述两种方法产生的高斯随机数点分布均以 3 左右为中心集中，并向两边逐渐稀疏，分布直方图与高斯图像一致。



两种方法产生图像保持一致。





可见，由两种方法生成的高斯随机数概率分布图像和预设的理论 N(3,4)高斯分布图像一

致，均值点为 3，概率密度峰值均为 0.2。

中心极限累加法

预设参数，均值为：3,标准差为：2

计算参数，均值为：2.9808,标准差为：1.9838

相对误差分别为：-0.64052 %, 和：-0.81214 %

两者相近。从直方图和低阶矩上看，基本符合要求。

变换法

预设参数，均值为：3,标准差为：2

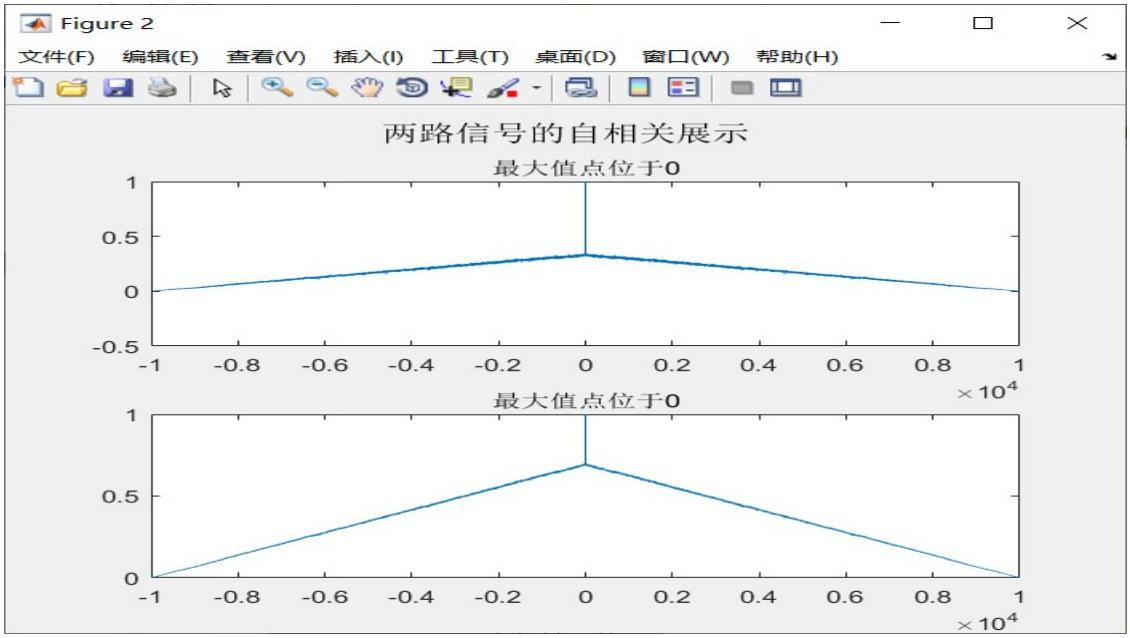
计算参数，均值为：3.013,标准差为：2.0212

相对误差分别为：0.43337 %, 和：1.0578 %

两者相近。从直方图和低阶矩上看，基本符合要求。

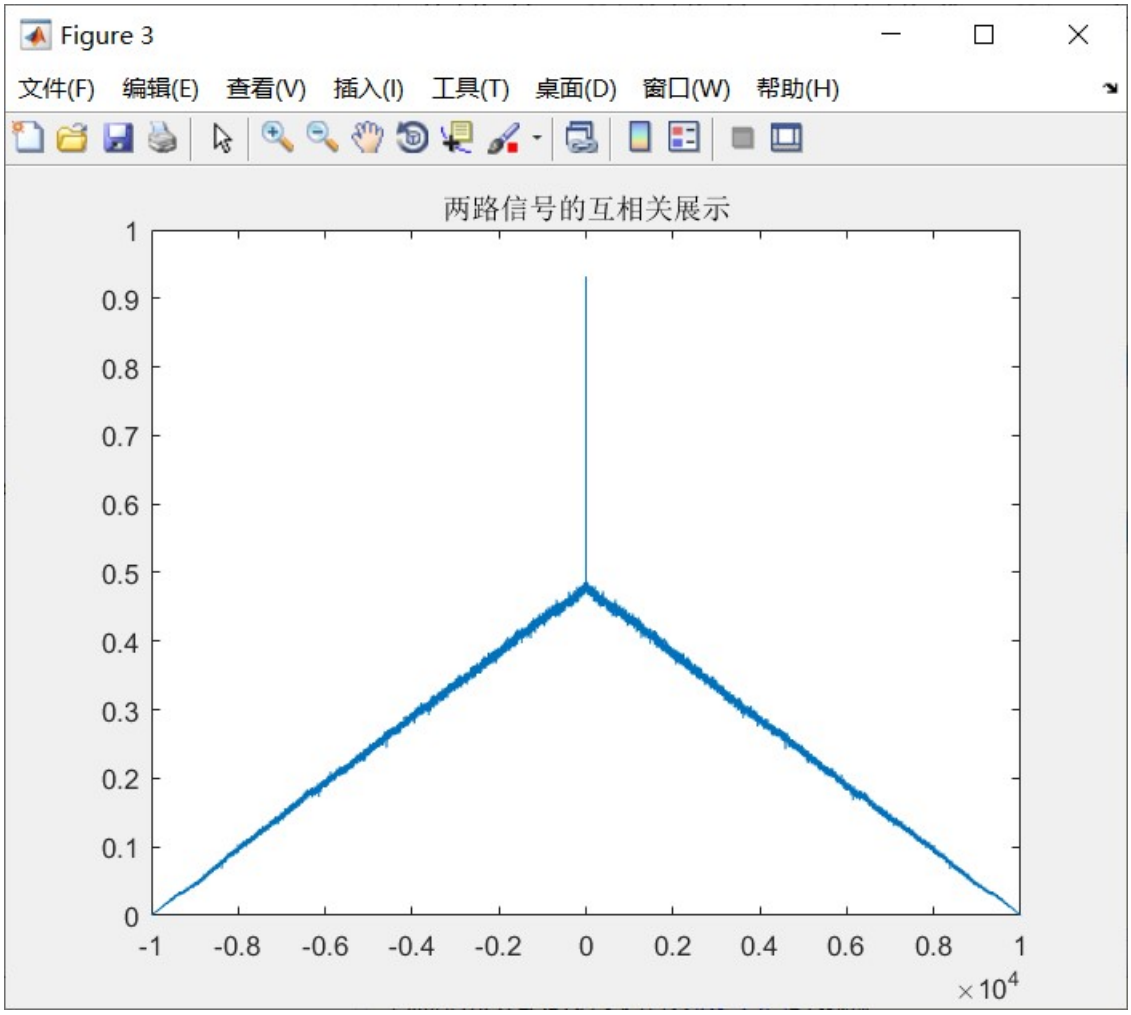
3. 高斯函数的自相关和互相关

（1） 10000 个 N(1,2)高斯随机数和 10000 个 N(3,4)高斯随机数的自相关函数图



平稳随机信自相关函数为偶函数，且在 0 点处取得最大值，并且 N（3,4）的均方值大于 N（1,2），所以自相关函数斜率也比他大，图像保持一致。

（2）两高斯信号的互相关函数图

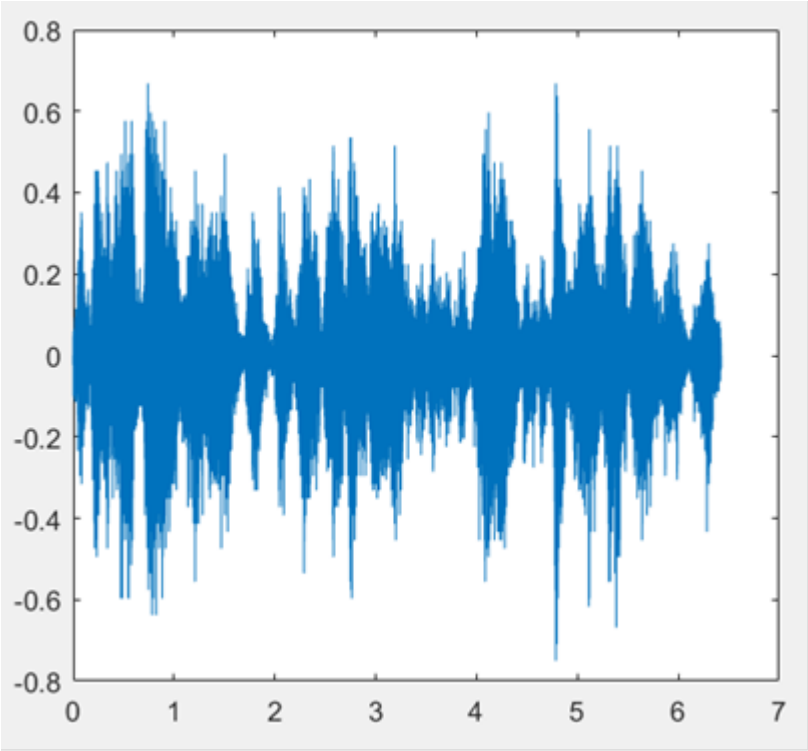


---

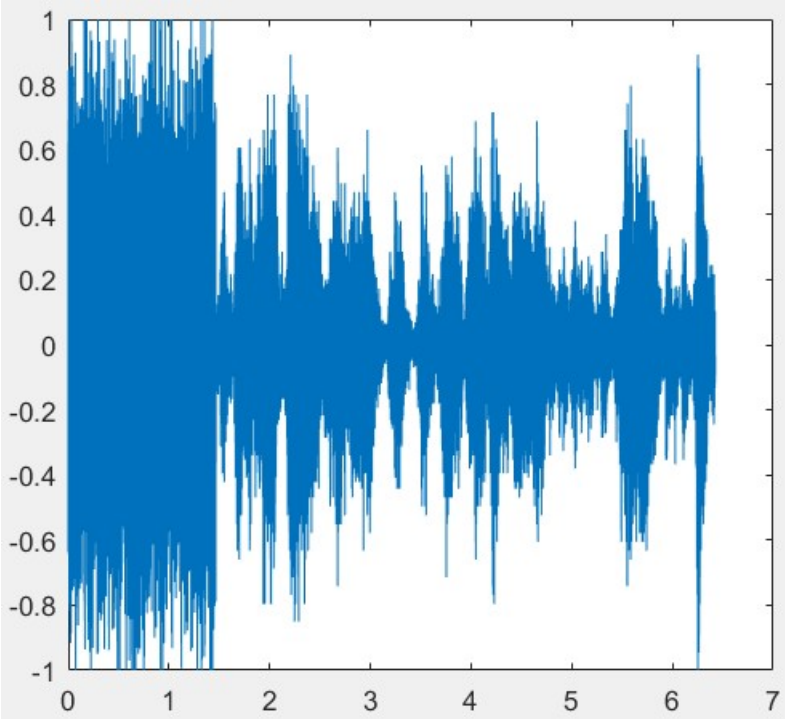
4. 探究性实验

（1）两路音频信号的时域波形图

采样 1 时域波形图：

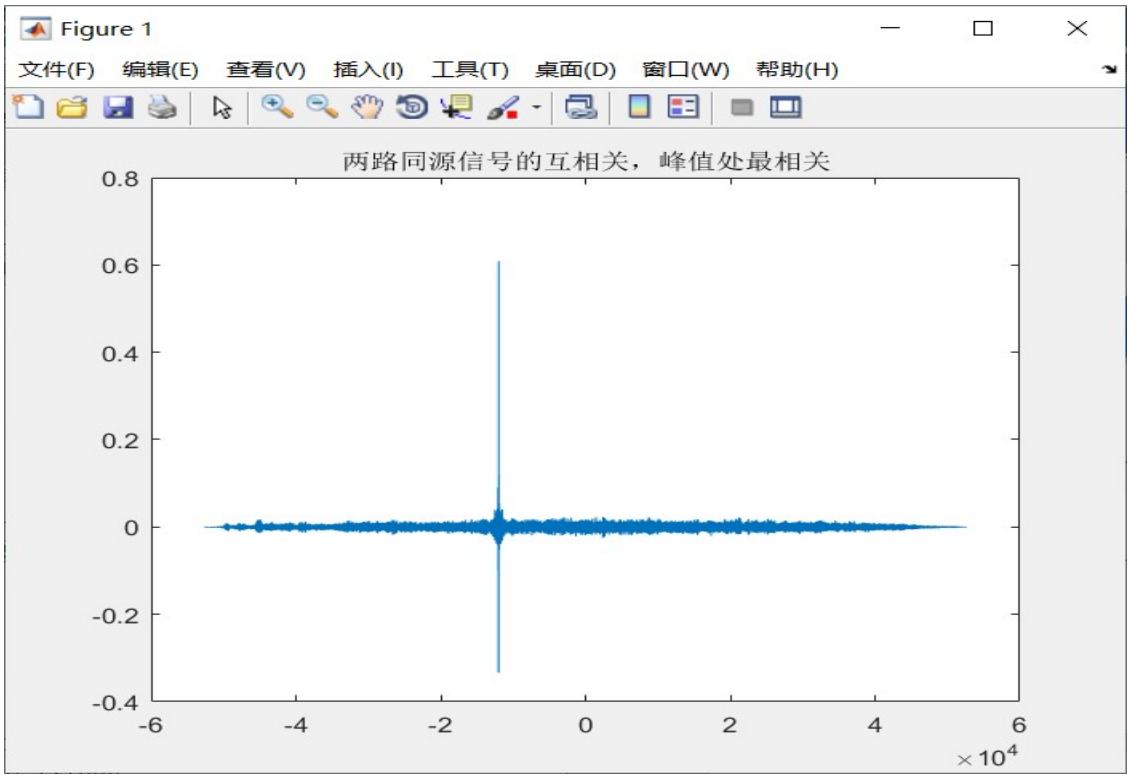


采样 2 时域波形图：



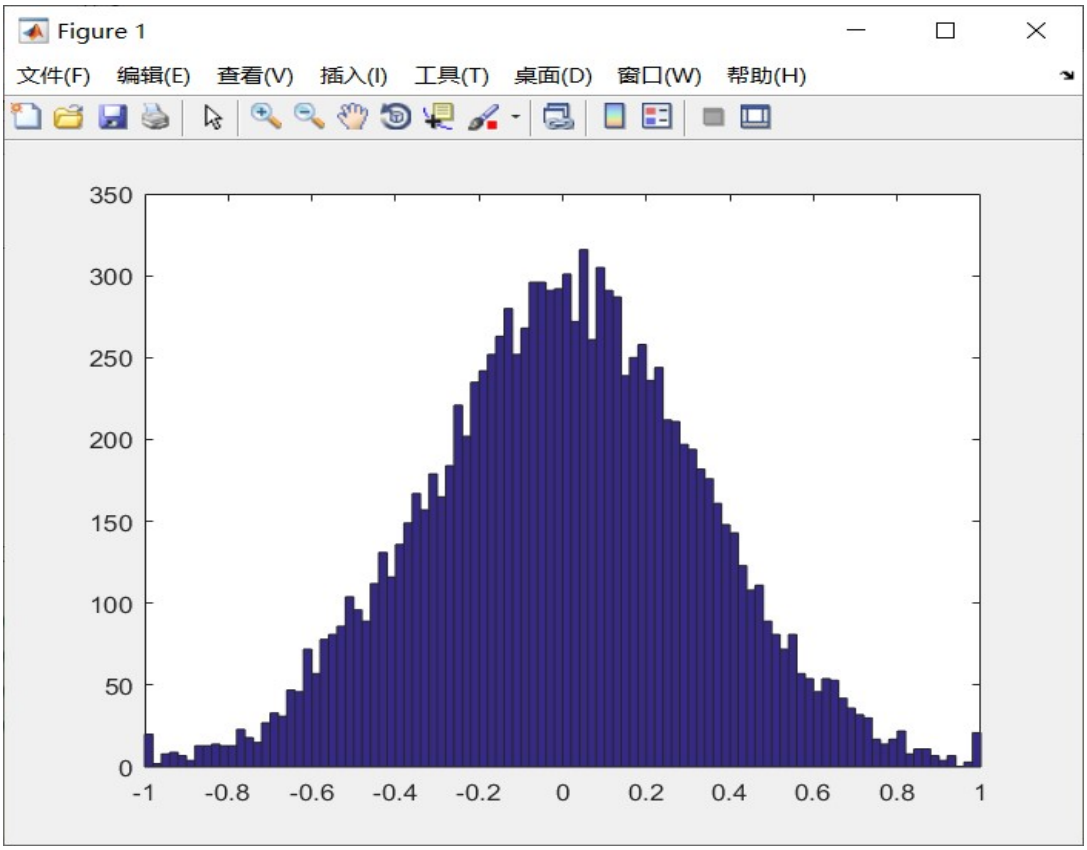
图像可知，延迟信号前有一段噪声信号，需要确定其延迟时间和分布类型。

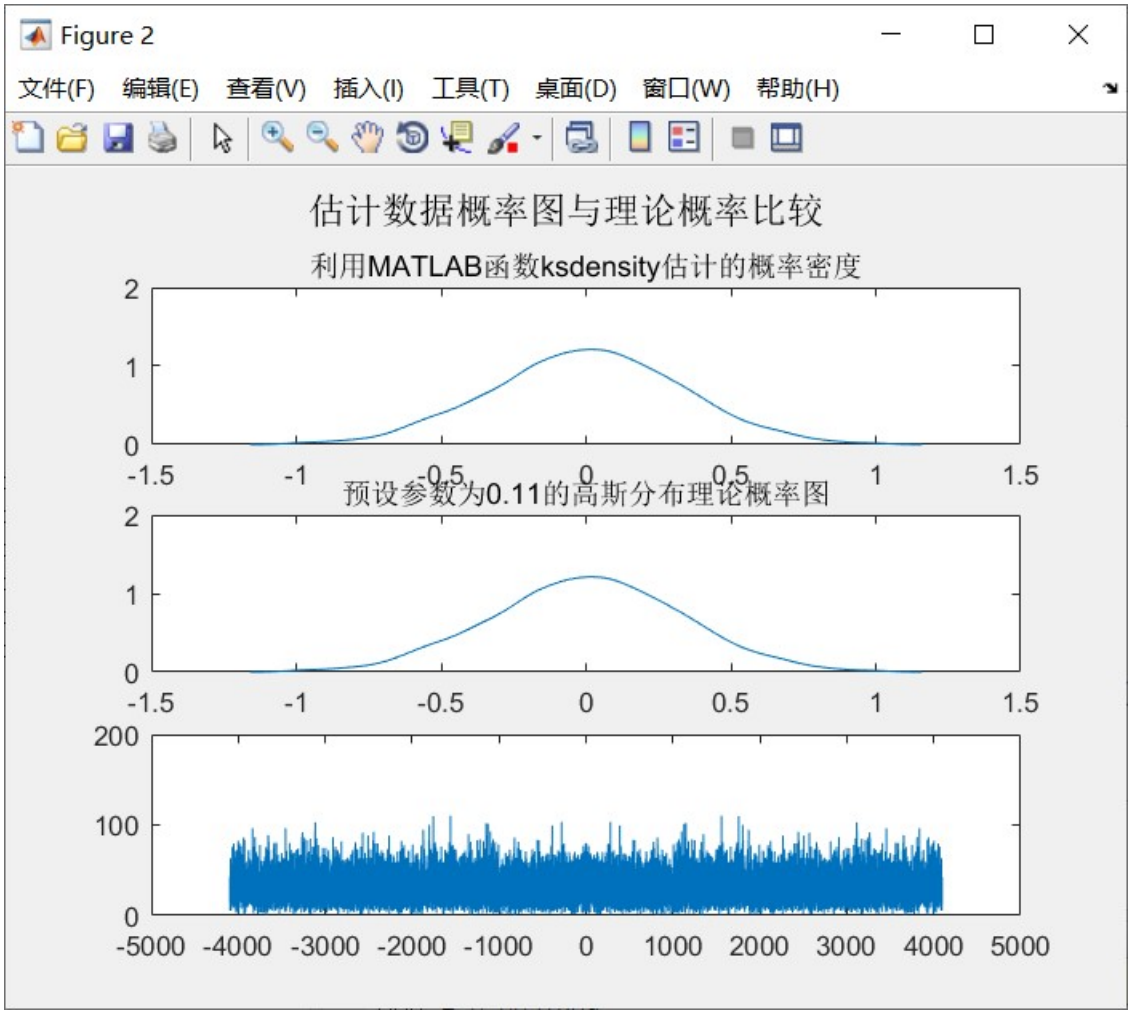
（2）两路音频信号的互相关函数图像



互相关函数的峰值点就是当两个音频相同部分重合时所求的互相关值，因为两段信号等长，所以峰值所在的点数就是噪声信号的点数。

（2） 噪音信号直方图，概率密度分布图像





根据直方图和概率密度分布可以初步认定噪音信号为高斯分布，均值在 0 点附近。

探究式实验 数据处理记录表

预设参数，均值为：0,方差为：0.11

时间差	延迟点数
1464.8438ms	12000
均值	方差
0.0032869	0.1093
均值绝对误差为	方差相对误差
0.0032869	-0.63765 %
声音信号频率	分布类型

8192Hz	高斯分布
三阶原点矩	四阶原点矩
-0.00026162	0.035162

通过 kstest 函数进行假设性检验确定噪声信号确实为高斯分布，否定了均匀分布和指数分布加强确认。并且求得均值为 0，方差为 0.11，是 N（0,0.11）的高斯分布噪声信号。

六、实验结果与分析

（1）均匀随机数

通过混合同余法得到的 10000 个在[0,1]区间均匀分布随机数的描点，连线，直方图正常，在各个区间均匀分布且和由系统函数 rand 所生成的随机数图像保持一致。均值 0.49724，约为 0.5，最大值 0.99986，最小值 7.6294e-06，与理论值非常接近，满足数字特征。而理论概率密度分布图不像预设的那样，只在（0,1）区间上有且为 1,在其余的区间均为 0，而是在（-0.2,1.2）。这是由于 ksdensity 函数在拟合曲线时会用一些数据平滑的方式（矩形拟合，三角拟合等）对样本点进行平滑，有一个从 0 至 1 渐变的过程，无法产生突变，所以出现了缓冲带。可以通过设置 ksdensity 的参数来优化概率密度曲线，比如将取值区间限制在[0,1]，以及改变样本的平滑方式，但只能做到优化，而不能完全解决。

至于问题为什么减 6，是因为期望和为 12*0.5=6，选择 12 为一组，是因为 sqrt（x/12），选择 12 方便计算。

（2）高斯随机数

使用自编的函数（中心极限法）产生的高斯随机数。预设参数，均值为：3,标准差为 2。实际均值为：2.9808,标准差为：1.9838。相对误差分别为：-0.64052 %, 和：-0.81214 %

。误差很小，可以认为服从高斯分布。并且由画出的频度分布直方图可以进一步证明正确性。

变换抽样法则预设参数，均值为：3,标准差为：2，计算参数，均值为：3.013,标准差为：2.0212，

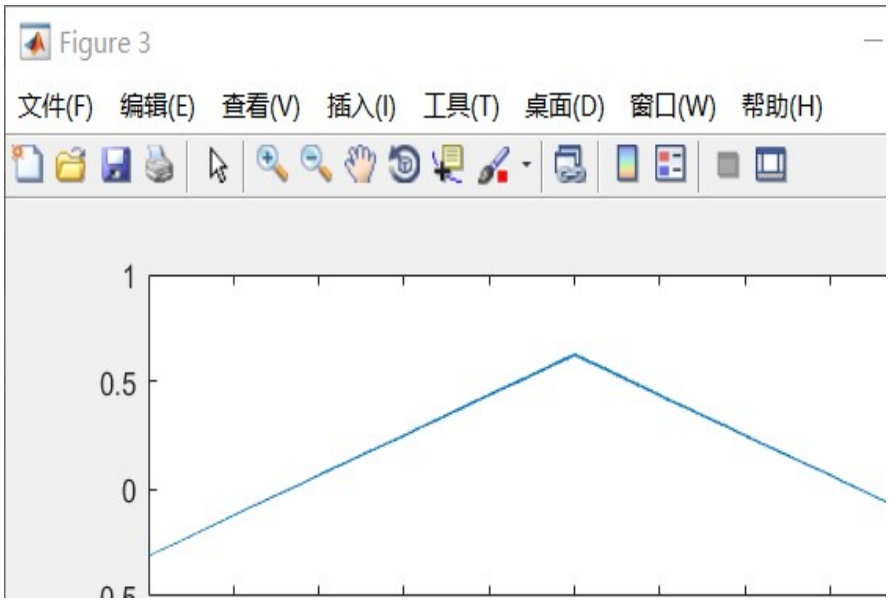
相对误差分别为：0.43337 %, 和：1.0578 %。误差很小，可以认为服从高斯分布。

概率密度分布图正常，分布均以 3 为中心集中，向两边逐渐稀疏减少，与预设理想 N（3,4）概率密度分布保持一致。

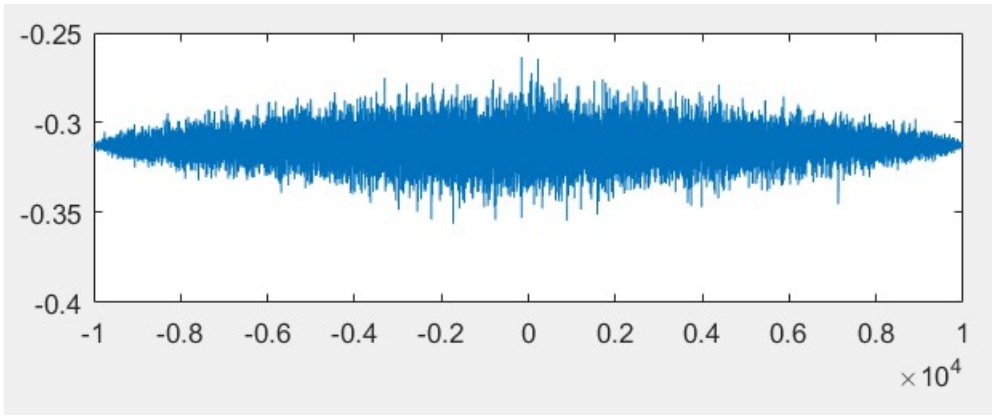
对于由交换法生成的两组高斯随机数的关系，理论上应该是相互独立但不正交。但事实上不正交且相关。说明 matlab 只能模拟，与实际有区别。两组高斯随机数的协方差为 0，说明是相互独立的（高斯随机数线性无关和统计独立等效）。通过计算其量组随机数的互相关函数，发现其不为 0，说明两组高斯随机数不正交。

```
num=10000;
```

```
s=rand(1,num);
y=rand(1,num);
[zxg1,n1]=xcorr(s,y,'coeff');
figure,subplot(2,1,1),plot(n1,zxg1-0.25);
```




```
s=random('norm',0,1,1,num);
y=random('norm',0,1,1,num);
[zxg1,n1]=xcorr(s,y,'coeff');
figure,subplot(2,1,1),plot(n1,zxg1-0.25);
```



自带高斯函数是互相独立的，说明我们的算法需要改进，并进行进一步思考。

（3）高斯分布的自相关以及互相关函数

其中自相关函数的图像能够比较好的用理论知识进行解释，为偶函数并且在 0 处取得最大值。比较难解释的是互相关函数为什么呈偶对称，从理论上看，互相关函数不一定是偶对称的，但是我测了多组高斯分布的互相关函数，无一例外的图像都近似是偶对称，而数据却不是偶对称的。按我的理解，两个独立的高斯分布，是由高斯变换法中的 ax+b 生成的，x 是生成的标准正态函数，所以他们俩的互相关函数实质上是 x 互相关函数和它们期望的线性和，而由于 x 具有宽平稳的性质，为常数，所以它们的互相关有着 x 的互相关函数的性质。

（4）探究性实验

计算两个函数互相关函数，得到延迟点数为 12000 点，根据采样率算出延迟时间为 1464.8438ms。接着画出概率密度图像，发现与高斯分布的图像相似。最后使用 matlab 自带的 kstest 函数进行置信度为 95%的假设检验，得出结果为符合均值为 0,方差为 0.11 的高斯分布，猜想正确。

kstest 函数不需要知道数据原分布是什么类型，是一种非参数检验方法。通过 normfit 函数得到噪声信号在置信区间下的高斯均值和方差的估计值，然后由 normcdf 函数根据估计值得到噪声信号的高斯概率密度分布，再由 kstest 函数进行判断其是否在置信区间内满足估计概率密度分布与实际的一致。当实际观测值 D>D(n, α)，在置信水平上不满足假设，则拒绝 H0，认为两者分布不同，否则认为 H0 成立。还可以通过 unifit 和 unifcdf 函数检验其是否满足均匀分布，答案是不满足，还可以通过改变检验条件检验很多类型的分布，验证结束后通过画噪声信号的直方图和概率分布图以及对误差的计算进一步验证了其为高斯分布，且近似标准高斯分布。

下面是 ks 检验的解释，感觉这个解释最通俗易懂：

Kolmogorov-Smirnov 是比较一个频率分布 f(x)与理论分布 g(x)或者两个观测值分布的检验方法。其原假设 H0:两个数据分布一致或者数据符合理论分布。D=max| f(x)- g(x)|, 当实际观测值 D>D(n, α)则拒绝 H0，否则则接受 H0 假设。

KS 检验与 t-检验之类的其他方法不同是 KS 检验不需要知道数据的分布情况，可以算是一种非参数检验方法。当然这样方便的代价就是当检验的数据分布符合特定的分布事，KS 检验的灵敏度没有相应的检验来的高。在样本量比较小的时候，KS 检验最为非参数检验在分析两组数据之间是否不同时相当常用。

PS：t-检验的假设是检验的数据满足正态分布，否则对于小样本不满足正态分布的数据用 t-检验就会造成较大的偏差，虽然对于大样本不满足正态分布的数据而言 t-检验还是相当精确有效的手段。

KS 检验是如何工作的?

1. 首先观察下分析数据

对于以下两组数据：

controlB={1.26, 0.34, 0.70, 1.75, 50.57, 1.55, 0.08, 0.42, 0.50, 3.20, 0.15, 0.49, 0.95,

0.24, 1.37, 0.17, 6.98, 0.10, 0.94, 0.38}

treatmentB= {2.37, 2.16, 14.82, 1.73, 41.04, 0.23, 1.32, 2.91, 39.41, 0.11, 27.44, 4.51, 0.51, 4.50, 0.18, 14.68, 4.66, 1.30, 2.06, 1.19}

对于 controlB，这些数据的统计描述如下：

Mean = 3.61

Median = 0.60

High = 50.6 Low = 0.08

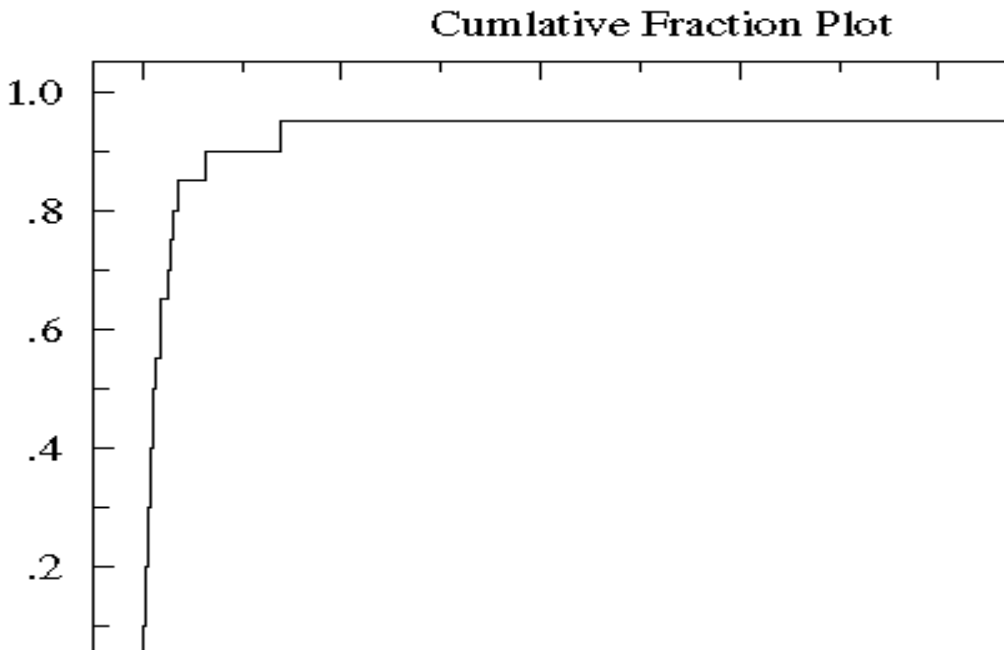
Standard Deviation = 11.2

可以发现这组数据并不符合正态分布， 否则大约有 15%的数据会小于均值-标准差（3.61-11.2），而数据中显然没有小于 0 的数。

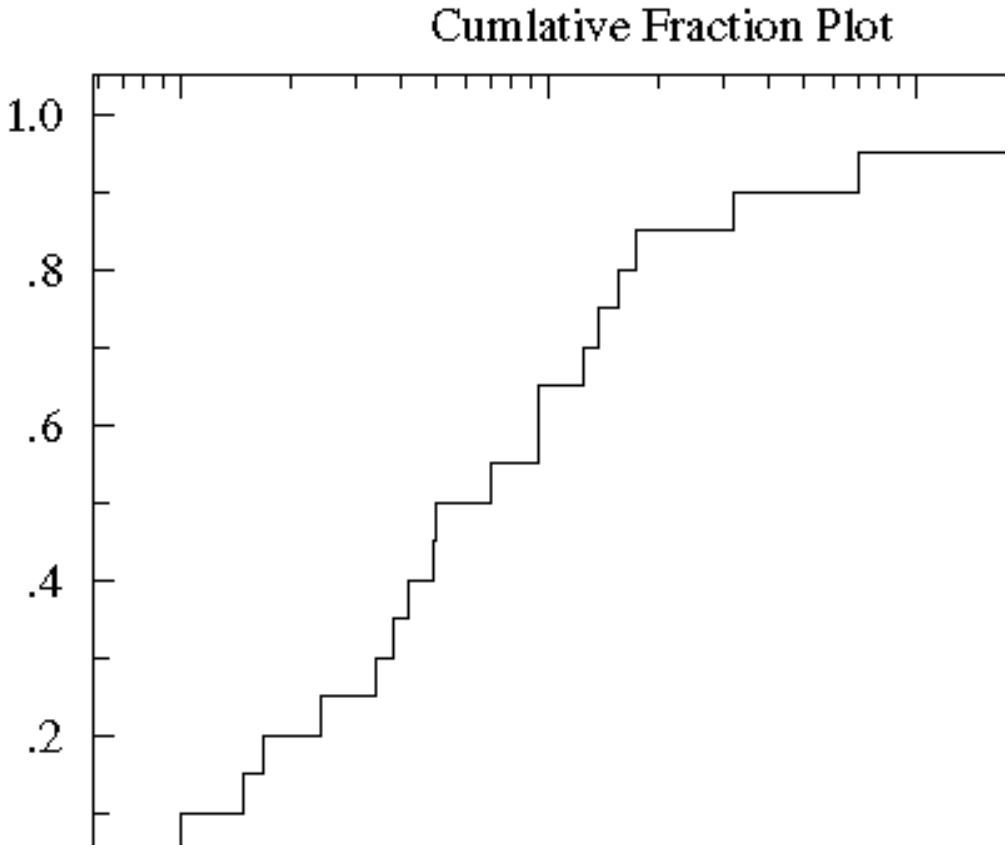
2. 观察数据的累计分段函数（Cumulative Fraction Function）

对 controlB 数据从小到大进行排序：

sorted controlB={0.08, 0.10, 0.15, 0.17, 0.24, 0.34, 0.38, 0.42, 0.49, 0.50, 0.70, 0.94, 0.95, 1.26, 1.37, 1.55, 1.75, 3.20, 6.98, 50.57}。10%的数据(2/20)小于 0.15，85%(17/20)的数据小于 3。所以，对任何数 x 来说，其累计分段就是所有比 x 小的数在数据集中所占的比例。下图就是 controlB 数据集的累计分段图

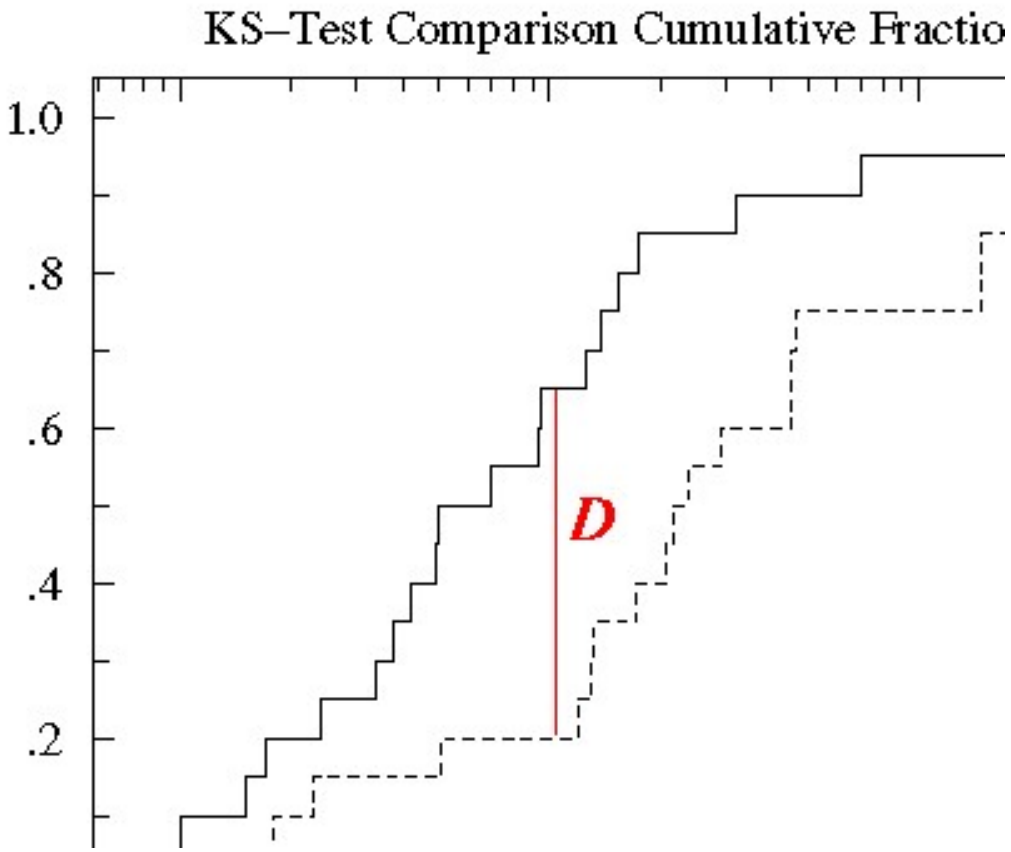


可以看到大多数数据都几种在图片左侧（数据值比较小），这就是非正态分布的标志。为了更好的观测数据在 x 轴上的分布，可以对 x 轴的坐标进行非等分的划分。**在数据都为正的时候有一个很好的方法就是对 x 轴进行 log 转换。**下图就是上图做 log 转换以后的图：



将 treatmentB 的数据也做相同的图（如下），可以发现 treatmentB 和 controlB 的数据

分布范围大致相同（0.1 - 50）。但是对于大部分 x 值，在 controlB 数据集中比 x 小的数据所占的比例比在 treatmentB 中要高，也就是说达到相同累计比例的值在 treatment 组中比 control 中要高。KS 检验使用的是两条累计分布曲线之间的最大垂直差作为 D 值（statistic D）作为描述两组数据之间的差异。在此图中这个 D 值出现在 x=1 附近，而 D 值为 0.45（0.65-0.25）。



值得注意的是虽然累计分布曲线的性状会随着对数据做转换处理而改变（如 log 转换），但是 D 值的大小是不会变的。

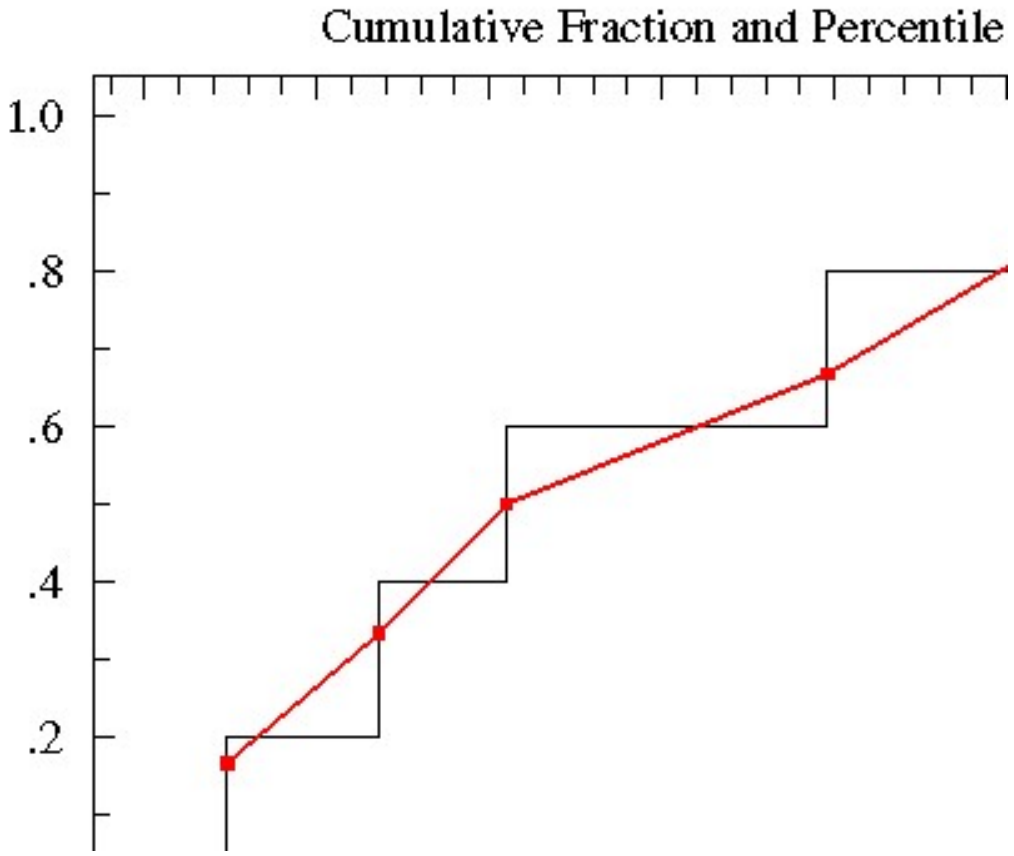
3. 百分比图（percentile plot）

估算分布函数肩形图（Estimated Distribution Function Ogive）是一种累计分段图的替代方式。其优势在于可以让你使用概率图纸作图（坐标轴经过特殊分段处理，y 轴上的数值间隔符合正态分布），从而根据概率在 y 轴上的分布可以直观的判断数据到底有多符合正态分

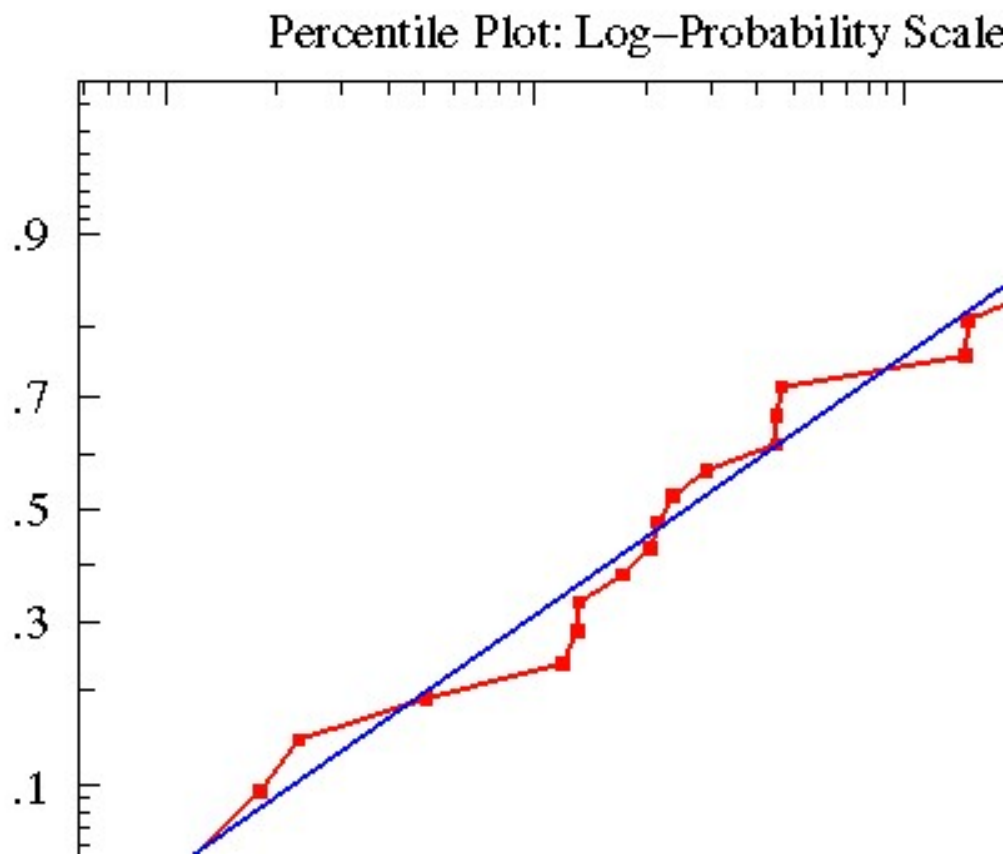
布，因为正态分布的数据在这种坐标上是呈一条直线。

那么这种图是如何画的呢?

假设我们有这 5 个数{-0.45, 1.11, 0.48, -0.82, -1.26}，从小到大对它们进行排序，{ -1.26, -0.82, -0.45, 0.48, 1.11 }。0.45 是中位数，百分比为 0.5，而 0.45 的累计分布函数中占了 0.4 到 0.6 的区间。根据数据 x 在数据集（N）中排位 r 可以计算 x 的百分数（percentile）为 r/(N+1)。将上述数据与他们的百分数配对，得到{ (-1.26,.167), (-0.82,.333), (-0.45,.5), (0.48,.667), (1.11,.833) }。然后将各点之间用直线连接就是百分比图了。如下图中红线所示（另一条线为累计分段曲线）。



treatmentB 的数据近似对数正态分布，其几何均值为 2.563，标准差为 6.795。该数据的百分图（红）与其近似的对数正态分布曲线（蓝）如下。



由于数据近似正态分布，所以对其采用 t-检验是最佳的检验方法。

七、讨论、建议、质疑

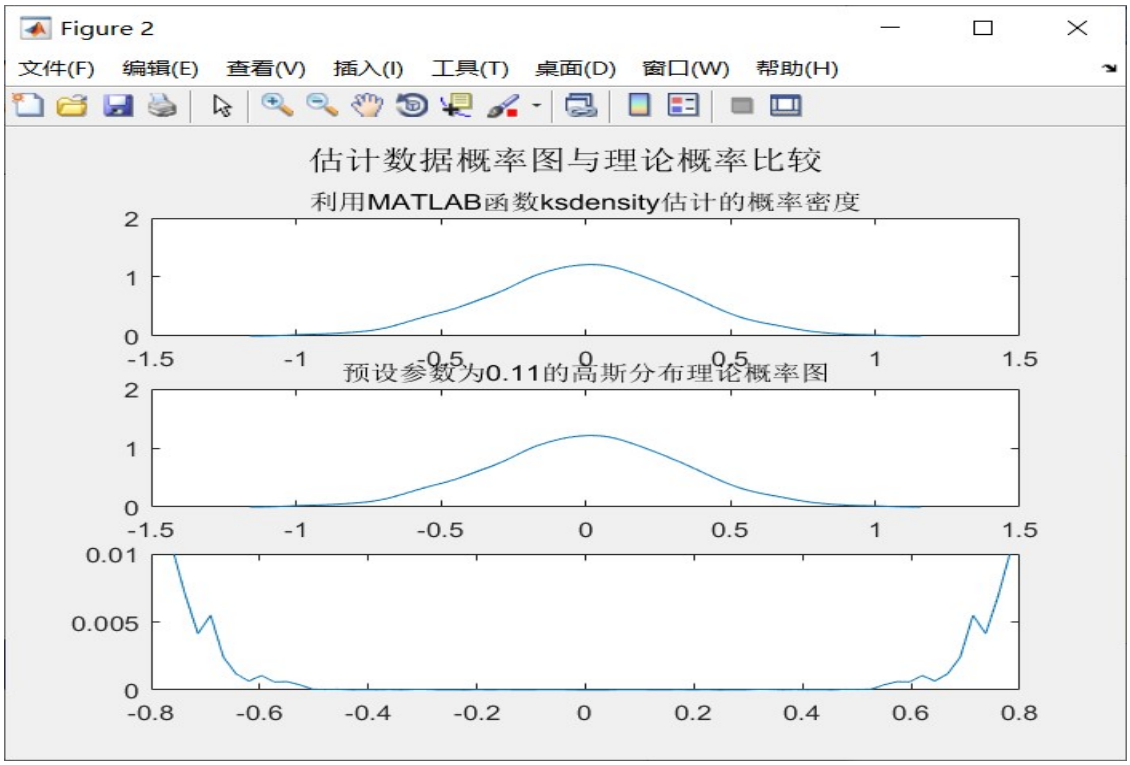
1.之前使用 matlab 时基本都是直接应用它提供的库函数，但是通过这个实验，通过自编函数和库函数对比，发现库函数不是特别完美的，也不能涵盖所有情况，遇到很多问题是无法给出有效解释的。很多时候一些理论上正确的东西但用计算机模拟却出现较大偏差，这个时候就必须深刻探讨函数的算法和思想，才能对我们的问题提供有效解释。

2.通过这次试验，学习了不通过 MATLAB 内置函数生成均匀分布随机数，高斯分布随机数的方法，求自相关，互相关函数以及截取延迟噪声，判断其延迟时间与分布类型等知识，并且加深的 MATLAB 软件的使用熟练度以及认识。

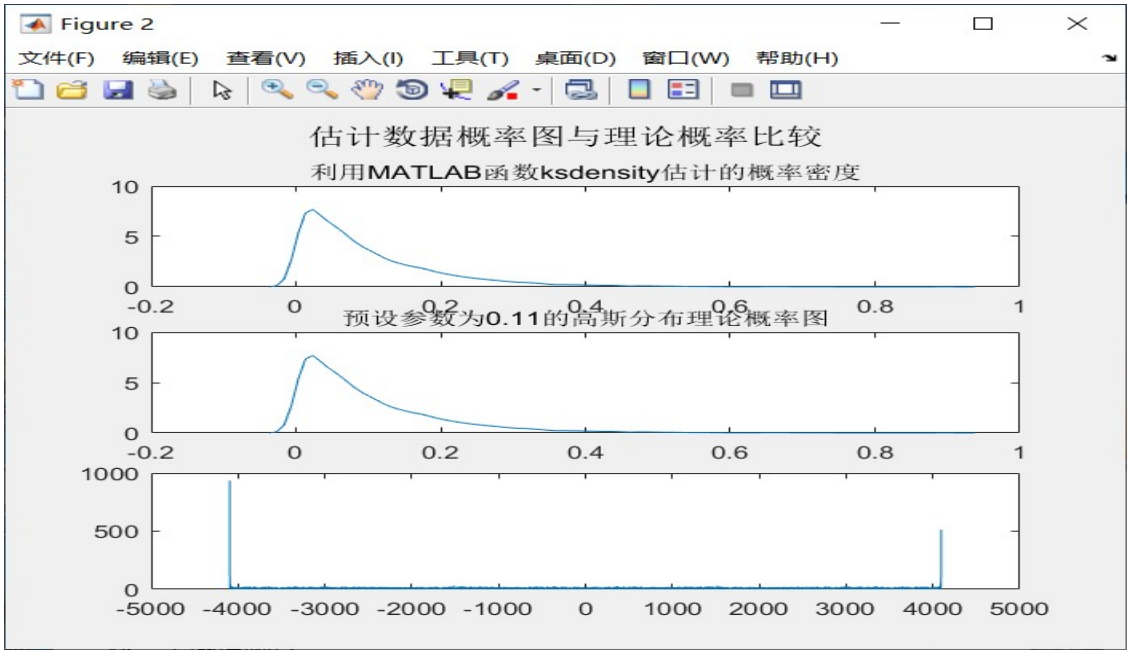
3.本次实验我尽量使用自编的代码，相关函数那部分使用了老师给的自编函数，自己也试着编了编函数，但求互相关时还是调用了库函数，因为自编函数还是存在一些问题，没有库函数考虑的情况充分。

4.老师在课堂上提出的问题给了我们很多思考，尤其是互相关的对称性确实使我对随机信号的计算过程有了更加深刻的理解，尤其是在证明函数分布规律时，开始想采用 fft 变换直接做，认为变换后的频域应该是一条直线，但是在老师的问题问过后，深深意识到高斯分布的是直方图而非是时域图，

使用 FFT 变换后生成的是杂乱无章的图形，后来对 ksdenity 进行变换，生成了两侧有频率变化，中间无变化的波形，肯定也是不对的。FFT 的波形本质是时域波形在频率图中的开方，在高斯检测方面，这条路不太行的通。所以最后使用了 kstest，进行了检验，这个函数普适性很高。不过在指数检测方面，似乎 FFT 法有些用途。



改成了 ksdenity 的波形



针对指数

代码附录：

函数 1　randuniform：

自编产生随机数函数：

```
function s=randuniform(M,b,r,first,num);

s=zeros(1,num);
s(1)=first;
for i=2:num
    s(i)=mod(s(i-1)*r+b,M);
end

s=s/M;
```

任务1：

使用自编函数生成随机数并估计数字特征：

```
%% task1_1
%% 配置环境
clear all;
close all;
clc;
%% 产生随机数
% 参数
M=1048576;
b=1;
r=2045;
first=12357;
num=10000;
s=randuniform(M,b,r,first,num);%自编
y=rand(1,num);%对照
%% 画线和画点
% 画线
figure,subplot(2,2,1),plot(s)  %全部
title('全部数据连线');
subplot(2,2,2),plot(s(1:100)) %前100个数据
title('前100个数据连线');
% 画点
```

```
subplot(2,2,3),plot(s,'.')  %全部
title('全部数据画点');
subplot(2,2,4),plot(s(1:100),'*') %前100个数据
title('前100个数据画点');
suptitle('自编代码生成数据展示')
% 画线
figure,subplot(2,2,1),plot(y)  %全部
title('全部数据连线');
subplot(2,2,2),plot(y(1:100)) %前100个数据
title('前100个数据连线');
% 画点
subplot(2,2,3),plot(y,'.')  %全部
title('全部数据画点');
subplot(2,2,4),plot(y(1:100),'*') %前100个数据
title('前100个数据画点');
suptitle('对照库函数代码生成数据展示')
%% 统计量
%求各阶矩
n=zeros(1,4);
for i=1:10000
    n(1)=n(1)+s(i);
    n(2)=n(2)+s(i)^2;
    n(3)=n(3)+s(i)^3;
    n(4)=n(4)+s(i)^4;
end
n=n/10000;
disp(['自编随机数的各阶矩',num2str(n),'。']);
%% 对比求各阶矩
m=zeros(1,4);
for i=1:10000
    m(1)=m(1)+y(i);
    m(2)=m(2)+y(i)^2;
    m(3)=m(3)+y(i)^3;
    m(4)=m(4)+y(i)^4;
end
m=m/10000;
disp(['对照库函数随机数的各阶矩',num2str(m),'。']);
%直方图
figure,subplot(2,1,1),hist(s);title('自编分10个区间的直方图');
subplot(2,1,2),hist(s,100);
title('自编分100个区间的直方图');
figure,subplot(2,1,1),hist(y);title('对照库函数分10个区间的直方图');
subplot(2,1,2),hist(y,100);
title('对照库函数分100个区间的直方图');
```

```
Max=max(s);disp(['自编最大值',num2str(Max),'。']);
Min=min(s);disp(['自编最小值',num2str(Min),'。']);
Max1=max(y);disp(['对照库函数最大值',num2str(Max1),'。']);
Min1=min(y);disp(['对照库函数最小值',num2str(Min1),'。']);

[f,xi]=ksdensity(s);
figure,subplot(2,1,1),plot(xi,f);
title('利用MATLAB函数ksdensity估计的概率密度')

% 预设的[0 ,1]均匀分布概率图
t=-0.2:1/10000000:1.2;
ff=ones(size(t)).*(t<=1&t>=0)+0*(t<0&t>=-0.2)+0*(t<=1.2&t>1);
subplot(2,1,2),plot(t,ff);
title('预设的[0 ,1]均匀分布理论概率图')

suptitle('自编代码生成数据概率图与理论概率比较')
[f,xi]=ksdensity(y);
figure,subplot(2,1,1),plot(xi,f);
title('利用MATLAB函数ksdensity估计的概率密度')

% 预设的[0 ,1]均匀分布概率图
t=-0.2:1/10000000:1.2;
ff=ones(size(t)).*(t<=1&t>=0)+0*(t<0&t>=-0.2)+0*(t<=1.2&t>1);
subplot(2,1,2),plot(t,ff);

title('预设的[0 ,1]均匀分布理论概率图')

suptitle('对照库函数代码生成数据概率图与理论概率比较')
```

函数2　rnd2

自编变换法生成高斯分布：

```
function [s]= rnd2(m,a,num);

%%

% x=rand(1,num);

% y=rand(1,num);

M=1048576;

b=1;

r=2045;
```

```
first1=12357;
first2=12359;
x=randuniform(M,b,r,first1,num);
y=randuniform(M,b,r,first2,num);
y1=sqrt((-2)*log(x)).*cos(2*pi*y);
s=a*y1+m;
figure
hist(s,100);
title('变换法高斯');
```

函数 3　rnd1:

自编中心极限法生成高斯分布：

```
function [s]= rnd1(m,a,num)
%%
% 用中心极限定理方法生成高斯随机数
% m：均值，a：标准差，num：数据点数
% 生成一路独立的均匀分布随机数

M=1048576;
b=1;
r=2045;
first=12357;
x=randuniform(M,b,r,first,num*12);
% x = rand(1,num*12);
y = reshape(x,12,num);
y1 = sum(y)-6;
% 线性变换，加上均值和标准差，返回两路高斯。
```

```
s = a*y1+m;
%%
% s=zeros(1,num);
% x=randuniform(1048576,2045,1,12357,num*12);
% for i=1:num
% y=sum(x(1+12*(i-1):1+12*(i-1)+11)-1/2)/sqrt(12/12);
% s(i)=a*y+m;
% end

%%
figure
hist(s,100);
title('高斯中心极限定理');
```

任务 2：

（3）使用中心极限法和变换法产生高斯分布，并估计数字特征：

```
%% task1_2
%% 配置环境
clear all;
close all;
clc;
%% 产生高斯随机变量
m=3;
a=2;
num=10000;
s=rnd1(m,a,num);
y=rnd2(m,a,num);
%% 画线和画点
% 画线
```

```
figure,subplot(2,2,1),plot(s);
title('中心极限累加法全部数据连线');
% 画点
subplot(2,2,2),plot(s,'.');
title('中心极限累加法全部数据画点');
% 画线
subplot(2,2,3),plot(y);
title('变换法全部数据连线');
% 画点
subplot(2,2,4),plot(y,'.');
title('变换法全部数据画点');

%% 求随机数的均值、均方值、一维三阶原点矩、一维四阶原点矩
n=zeros(1,4);
for i=1:10000
    n(1)=n(1)+s(i);
    n(2)=n(2)+s(i)^2;
    n(3)=n(3)+s(i)^3;
    n(4)=n(4)+s(i)^4;
end
n=n/10000;
disp(['中心极限累加法生成数据的数字特征'])
disp(['均值 = ',num2str(n(1))] );
disp(['均方值 = ',num2str(n(2))] );
disp(['三阶原点矩 = ',num2str(n(3))] );
disp(['四阶原点矩 = ',num2str(n(4))] );
m1=zeros(1,4);
for i=1:10000
    m1(1)=m1(1)+y(i);
    m1(2)=m1(2)+y(i)^2;
```

```
    m1(3)=m1(3)+y(i)^3;
    m1(4)=m1(4)+y(i)^4;
end
m1=m1/10000;
disp(['变换法生成数据的数字特征'])
disp(['均值 = ',num2str(m1(1))] );
disp(['均方值 = ',num2str(m1(2))] );
disp(['三阶原点矩 = ',num2str(m1(3))] );
disp(['四阶原点矩 = ',num2str(m1(4))] );
%% 最大最小值
Max=max(s);disp(['中心极限累加法最大值',num2str(Max),'。']);
Min=min(s);disp(['中心极限累加法最小值',num2str(Min),'。']);
Max1=max(y);disp(['对照库函数最大值',num2str(Max1),'。']);
Min1=min(y);disp(['对照库函数最小值',num2str(Min1),'。']);
%%
disp('中心极限累加法')
meanValue = mean(s);
stdValue   = std(s);
disp('----------')
disp(['预设参数，均值为：',num2str(m),',标准差为：',num2str(a)]);
disp(['计算参数，均值为：',num2str(meanValue),',标准差为：',num2str(stdValue)]);

meanErr = (meanValue - m)/(m)*100;
stdErr   = (stdValue - a)/(a)*100;

disp(['相对误差分别为：',num2str(meanErr),' %, 和：',num2str(stdErr),' %'])
disp('两者相近。从直方图和低阶矩上看，基本符合要求。')
%% 验证均值和方差
disp('变换法')
meanValue = mean(y);
```

```
stdValue    = std(y);
disp('----------')
disp(['预设参数，均值为：',num2str(m),',标准差为：',num2str(a)]);
disp(['计算参数，均值为：',num2str(meanValue),',标准差为：',num2str(stdValue)]);

meanErr = (meanValue - m)/(m)*100;
stdErr    = (stdValue - a)/(a)*100;

disp(['相对误差分别为：',num2str(meanErr),' %, 和：',num2str(stdErr),' %'])
disp('两者相近。从直方图和低阶矩上看，基本符合要求。')
%%
[zxg1,n1]=xcorr(s,'coeff');
[zxg2,n2]=xcorr(y,'coeff');
figure,subplot(2,1,1),plot(n1,zxg1);
title('中心极限累加法高斯相关');
subplot(2,1,2),plot(n2,zxg2);
title('变换法高斯相关');
[f,xi]=ksdensity(s);
figure,subplot(3,1,1),plot(xi,f);
title('中心极限累加法利用 MATLAB 函数 ksdensity 估计的概率密度')
[f,xi]=ksdensity(y);
subplot(3,1,2),plot(xi,f);
title('变换法利用 MATLAB 函数 ksdensity 估计的概率密度')
xlim([-6,12]);
t=-50:1/100:50;
k=1/(sqrt(2*pi)*a)*exp(-(t-m).^2/(2*a*a));
subplot(3,1,3),plot(t,k);
title('预设的 N（m，a）高斯分布理论概率图')
xlim([-6,12]);
suptitle('自编代码生成数据概率图与理论概率比较')
```

任务 3：

用中心极限法产生高斯分布并估计自，互相关函数：

%% task1_3

```
%% 配置环境
clear all;
close all;
clc;
num =10000;
% %% 产生 N(1,2),N(3,4)
y1=rnd1(1,sqrt(2),10000);
y2=rnd1(3,2,10000);
% %% 统计量


% [x1,lags]=xcorr(y1,'coeff');
% [x2,lags]=xcorr(y2,'coeff');
[zxg1,n1]=ycorr(y1,y1,'coeff');
[zxg2,n2]=ycorr(y2,y2,'coeff');
[x12,lags]=xcorr(y1,y2,'coeff');
subplot(2,1,1);plot(n1,zxg1);
%ylim([0,1]);
[mx,indx]=max(zxg1);
title(['最大值点位于',num2str(abs(indx-num))]);
hold on
subplot(2,1,2);plot(n2,zxg2);
[mx,indx]=max(zxg2);
title(['最大值点位于',num2str(abs(indx-num))]);
suptitle('两路信号的自相关展示')
figure;plot(lags,x12)
```

```
title(['两路信号的互相关展示'])
% subplot(3,1,1);plot(n1,zxg1)
% title('N(1,2)自相关函数');
% subplot(3,1,2);plot(n2,zxg2)
% title('N(3,4)自相关函数');
% subplot(3,1,3);plot(n3,hxg)
% title('互相关函数');
```

任务 4

读取语音并估计分布：

```
%% task1_4
%%  准备环境
clear all;
close all;
clc;
%%  读取数据
[filename1,filepath1]=uigetfile('.wav');
audeofile1= strcat(filepath1,filename1);
[y1,Fs1] = audioread(audeofile1);
[filename2,filepath2]=uigetfile('.wav');
audeofile2= strcat(filepath2,filename2);
[y2,Fs2] = audioread(audeofile2);
num=length(y1);
disp(['声音信号频率',num2str(Fs1),'Hz。']);
%%  判断延迟
[x,lags]=xcorr(y1,y2,'coeff');
figure;plot(lags,x)
title('两路同源信号的互相关，峰值处最相关');
[mx,indx] = max(x);
```

```
TLag = abs(num - indx);
disp(['经计算相关函数，估计延迟点数为：',num2str(TLag),'。'])
TsLag=TLag/Fs1*1000;
disp(['估计延迟时间为：',num2str(TsLag),'ms。'])
%% 进行参数计算
noise=y2(1:TLag);
m = zeros(1,4);              %生成[0 0 0 0]
for i = 1 : TLag
    m(1) = m(1) + noise(i);      % 均值
    m(2) = m(2) + noise(i)^2;    % 二阶矩
    m(3) = m(3) + noise(i)^3;    %三阶
    m(4) = m(4) + noise(i)^4;    %四阶
end
m=m/TLag;
disp(['生成数据的数字特征'])
disp(['均值 = ',num2str(m(1))] );
disp(['均方值 = ',num2str(m(2))] );
disp(['三阶原点矩 = ',num2str(m(3))] );
disp(['四阶原点矩 = ',num2str(m(4))] );
meanValue=mean(noise);
stdValue=std(noise);
disp(['方差 = ',num2str(stdValue^2)] );
%% 和预设值对比
disp(['预设参数，均值为：',num2str(0),',方差为：',num2str(0.11)]);
meanErr=meanValue;
stdErr   = (stdValue^2 - 0.11)/(0.11)*100;
disp(['均值绝对误差为：',num2str(meanErr),' 方差相对误差为：',num2str(stdErr),' %'])
[f,xi]=ksdensity(noise);
figure,subplot(3,1,1),plot(xi,f);
title('利用 MATLAB 函数 ksdensity 估计的概率密度');
```

```
%%
t=-3:1/1000:3;
y=1/sqrt(2*pi*0.1)*exp((-5)*t.^2);
[f,xi]=ksdensity(noise);
subplot(3,1,2),plot(xi,f);
title('预设参数为 0.11 的高斯分布理论概率图');
suptitle('估计数据概率图与理论概率比较');
%% 使用傅里叶变换来判断高斯分布(1)
% Fs3=Fs1;
% f=fft(noise);
% a=abs(f/TLag);
% p1=a(1:TLag/2+1);
% p1(2:end-1)=2*p1(2:end-1);
% f1=Fs3*(0:(TLag/2))/TLag;
% subplot(3,1,3),plot(f1,p1);
%% 使用傅里叶变换来判断高斯分布(2)
fs=Fs1;
N=2*Fs1;
n=0:N-1;
x=fft(noise,N);
m=abs(x);
n=-N/2:N/2-1;
f=n*fs/N;
subplot(3,1,3),plot(f,m);
%% 使用假设检验验证噪声的分布
% A=noise;
% alpha=0.05;
% [mu,sigma]=normfit(A);
% p1=normcdf(A,mu,sigma);
% [H1,s1]=kstest(A,[A,p1],alpha);
```

```
% if H1==0
%     disp('该数据服从正态分布。')
% else
%     disp('该数据不服从正态分布。')
% end
%% 使用模块化假设检验验证噪声的分布
p_judge(noise,0.05);
```

函数 4　p_judge：

模块化噪声分布估计

```
function p_judge(A,alpha)
[mu,sigma]=normfit(A);
p1=normcdf(A,mu,sigma);
[H1,s1]=kstest(A,[A,p1],alpha);
n=length(A);
if H1==0
disp('该数据源服从正态分布。')
else
disp('该数据源不服从正态分布。')
end
if H1==1
[mu,sigma]=unifit(A);
p1=unifcdf(A,mu,sigma);
[H6,s6]=kstest(A,[A,p1],alpha);
n=length(A);
if H6==0
disp('该数据源服从均匀分布。')
else
disp('该数据源不服从均匀分布。')
```

```
end
if H6==1
phat=gamfit(A,alpha);
p2=gamcdf(A,phat(1),phat(2));
[H2,s2]=kstest(A,[A,p2],alpha);
if H2==0
disp('该数据源服从γ分布。')
else
disp('该数据源不服从γ分布。')
end
if H2==1
lamda=poissfit(A,alpha);
p3=poisscdf(A,lamda);
[H3,s3]=kstest(A,[A,p3],alpha);
if H3==0
disp('该数据源服从泊松分布。')
else
disp('该数据源不服从泊松分布。')
end
if H3==1
mu=expfit(A,alpha);
p4=expcdf(A,mu);
[H4,s4]=kstest(A,[A,p4],alpha)
if H4==0
disp('该数据源服从指数分布。')
else
disp('该数据源不服从指数分布。')
end
if H4==1
[phat, pci] = raylfit(A, alpha);
```

```
p5=raylcdf(A,phat);
[H5,s5]=kstest(A,[A,p5],alpha);
if H5==0
disp('该数据源服从 rayleigh 分布。')
else
disp('该数据源不服从 rayleigh 分布。')
end
end
end
end
end
end
```

函数 5　ycorr：

```
function [c, lag] = ycorr ( x,y,option)
%YCORR  相关函数估计。
%     [C LAG] = YCORR(X,Y),
%     [C LAG] = YCORR(X,Y，OPTION),
%     OPTION 可选，且为‘none’、‘coeff’、‘biased’、‘unbiased’之一,
%     X，Y 是行向量，返值 C 是  2N-1 长度的相关函数，N 是 X，Y
%     中最长者的长度。除了输入参数不同，其余与 XCORR 一致。
%     LAG =-(N-1):(N-1)给出 C 的标号

%     2016.11.1 aleko.

%规定输入有参数形式为（X，Y）或（X，Y，OPTION）
if (nargin ~=2 && nargin ~=3)
    disp('2 or 3 nargin needed.');
```

```
    c=0;lag=0;
    return
else
    if nargin == 2
        option ='none';
    end
end
%如果输入向量不等长，补 0 使等长
N = max(length(x),length(y));
X = [x zeros(1,N-length(x))];
Y = [y zeros(1,N-length(y))];
%default
if isequal(option,'none')
    c0 = sum(X.*Y);
    for m = 1:N-1
        cleft(m) =sum([X(m+1:end),zeros(1,m)].* Y);
        cright(m) =sum(X .* [Y(m+1:end),zeros(1,m)]);
    end
    c=[fliplr(cright) c0,cleft];
    lag = -(N-1):N-1;
    return;
end
%归一化，自相关结果与 XCORR 一致，互相关有微小差异，互相关不会归一化，无关
if isequal(option,'coeff')
    if ~isequal(x,y)
        disp('coeff for autocorelation only');
        c=0;lag=0;
        return
    end
    c0 = sum(X.*Y);
```

```
    for m = 1:N-1
        cleft(m) =sum([X(m+1:end),zeros(1,m)].* Y);
        cright(m) =sum(X .* [Y(m+1:end),zeros(1,m)]);
    end
    c=[fliplr(cright) c0,cleft]/c0;
    lag = -(N-1):N-1;
    return;
end
%有偏估计
if isequal(option,'biased')
    c0 = sum(X.*Y);
    for m = 1:N-1
        cleft(m) =sum([X(m+1:end),zeros(1,m)].* Y);
        cright(m) =sum(X .* [Y(m+1:end),zeros(1,m)]);
    end
    c=[fliplr(cright) c0,cleft]/N;
    lag = -(N-1):N-1;
    return;
end
%无偏估计
if isequal(option,'unbiased')
    c0 = sum(X.*Y)/N;
    for m = 1:N-1
        cleft(m) =sum([X(m+1:end),zeros(1,m)].* Y)/(N-m);
        cright(m) =sum(X .* [Y(m+1:end),zeros(1,m)])/(N-m);
    end
    c=[fliplr(cright) c0,cleft];
    lag = -(N-1):N-1;
    return;
end
```