浙江大學

本科实验报告

信号与系统实验 3

课程名称：	信号与系统
姓名：	szx
学院：	信息与电子工程学院
专业：	电子科学与技术
学号：	0000000000
指导老师：	张婷、郑斌

June 25, 2023

目 录

一、 Question 1: 离散时间卷积

1. 系统样值响应

系统单位样值为：

$$h[n] = 2\delta[n+1] - 2\delta[n-1] \tag{1}$$

而当输入为：

$$x[n] = \delta[n] + \delta[n-2] \tag{2}$$

可以通过卷积计算出输出响应。同时也可以得到其时间索引为：$ny = [-1:3]$。

通过 matlab 编程可得到结果，如图 1所示。

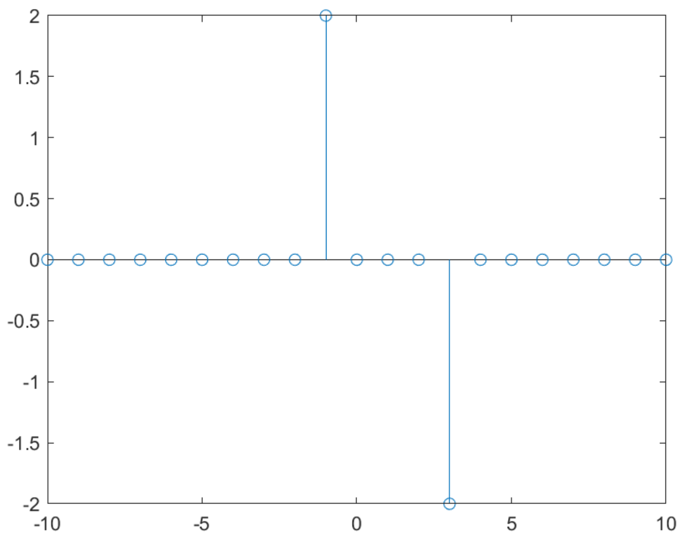

Figure 1: 单位样值响应

2. 一般情况下的样值响应

当 $a = 0, b = N-1, c = 0, d = M-1$ 时，系统响应为：

$$h[n] = \delta[n] + \delta[n-N+1] \tag{3}$$

输入为：

$$x[n] = \delta[n] + \delta[n-M+1] \tag{4}$$

卷积之后，输出 y 的范围为 $[0:M+N-2]$，其长度为 $M+N-1$。

matlab 的输出结果如图 2所示，可以看出，我们的计算是正确的。

3. n 取有限值与无限值时的输出对比

当输入为：

$$x[n] = (\frac{1}{2})^{n-2}u[n-2] \tag{5}$$

而系统函数为：

$$h[n] = u[n+2] \tag{6}$$

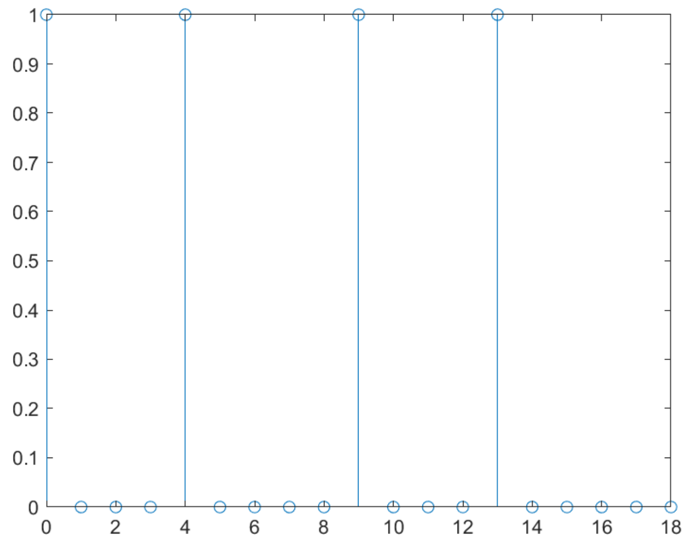

Figure 2: 一般化样值响应

从理论分析的角度，其输出应该为：

$$y[n] = x[n] * h[n] = 2 - (\frac{1}{2})^n \tag{7}$$

但实际上，我们并不能计算 n 从负无穷到正无穷的和，只能截取一部分进行求和计算，因此，实际数值计算的结果和理论分析的结果有不小的区别，导致数值计算结果只有一部分是有效的。

通过 matlab 的运算能更好的看出这一点，如图 3所示，两张图相同的部分是有效值。

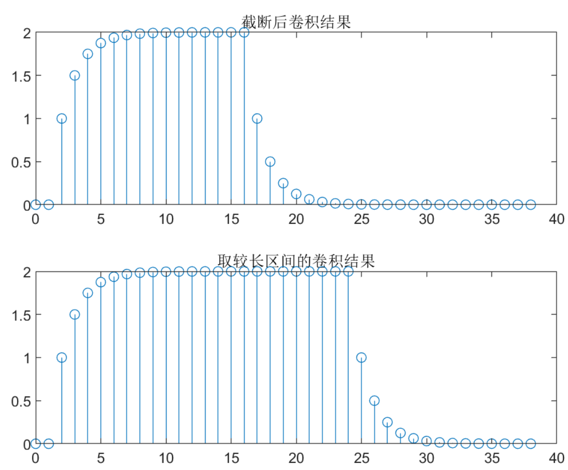

Figure 3: 截断求和的卷积结果

4. 分块卷积

系统函数为：

$$h[n] = 00.9^n(u[n] - u[n-10]) \tag{8}$$

输入为：

$$x[n] = \cos(n^2)sin(2\pi n/5) \tag{9}$$

在 $0 \leq n \leq 99$ 的范围内，取 L=50，可以将其分为两个部分分别卷积，再将其移位相加即得到卷积结果：

$$y[n] = y_0[n] + y_1[n-k] \tag{10}$$

在式中，k 应该取 L=50。

通过 matlab 编码，可以清晰地进行对比，如图所示。

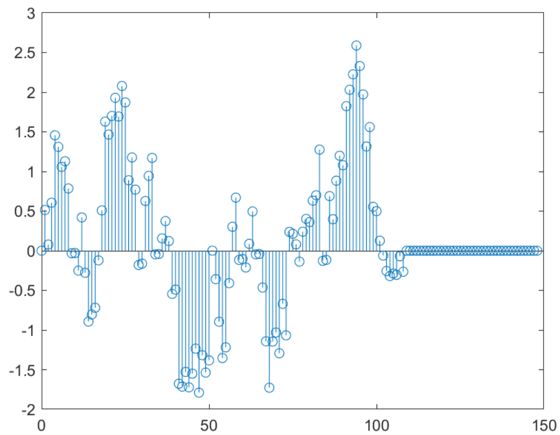

Figure 4: 分块卷积结果

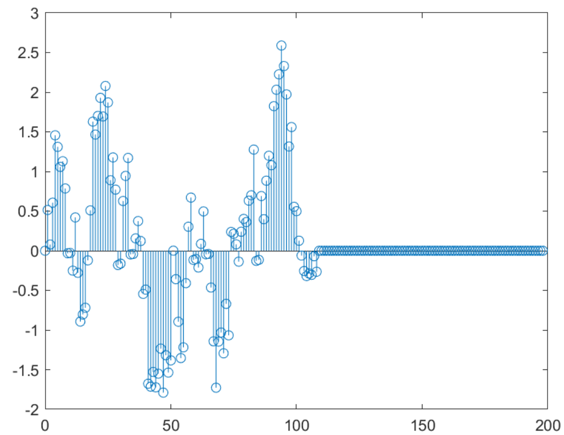

Figure 5: 直接卷积结果

二、 Question 2: 利用一维滤波器做图像处理

1. 三种滤波器的特性及效果

由以下代码构成了三个滤波器结构：

```
wc=0.4;
n1=10;n2=4;n3=12;

[b1,a1]=butter(n1, wc);
a2=1; b2=firpm(n2, [0 wc-0.04 wc+0.04 1],[1 1 0 0]);
a3=1; b3=firpm(n3, [0 wc-0.04 wc+0.04 1],[1 1 0 0]);
```

也可以由差分方程的形式表达出来。

其频率响应如图 6、7、8所示。

同时，我们可以通过 filter 函数得到这三个滤波器的阶跃响应，如图 9所示。从图中，我们可以看出，滤波器 2 的 overshoot 是最大的，而滤波器 1 的 ringing 现象最为明显。在进行图像滤波的过程中，overshoot 会导致滤波结果相较于输入会有所偏移，导致失真较为严重；而 ringring 则会导致吉布斯现象的产生。

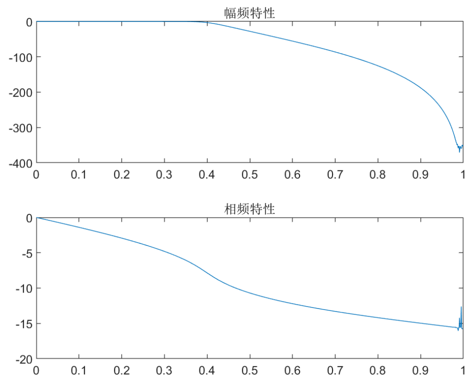

Figure 6: 滤波器 1 特性曲线

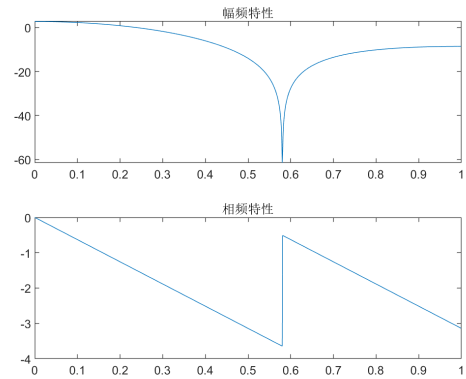

Figure 7: 滤波器 2 的特性曲线

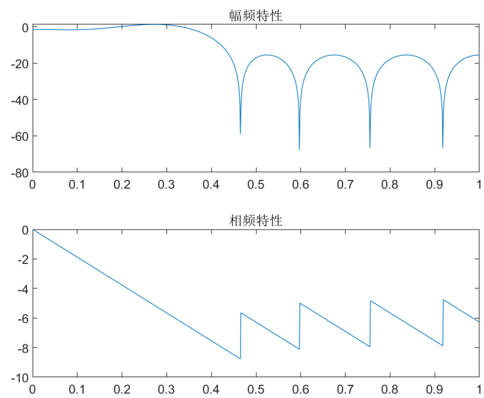

Figure 8: 滤波器 3 的特性曲线

图中坐标的横坐标经过了 $/\pi$ 的归一化，纵坐标单位为 dB。

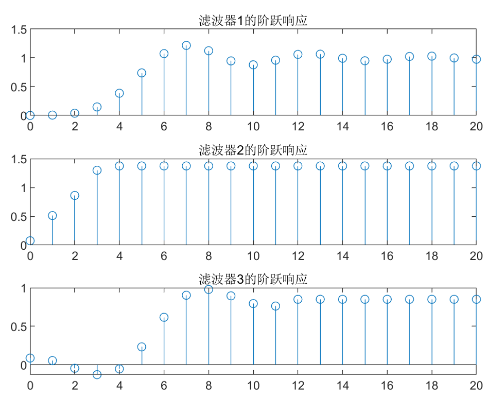

Figure 9: 三个滤波器的阶跃响应

2. 二维图像非因果滤波

在使用 filter 函数进行非因果滤波时，要进行偏移，而针对提供的 plus.mat 图像，可以设置偏移量为 $d = 6$，从而可以使平滑的不连续位置应对应图像不连续位置。实际以图像第 16 列进行滤波，其结果如图10所示。

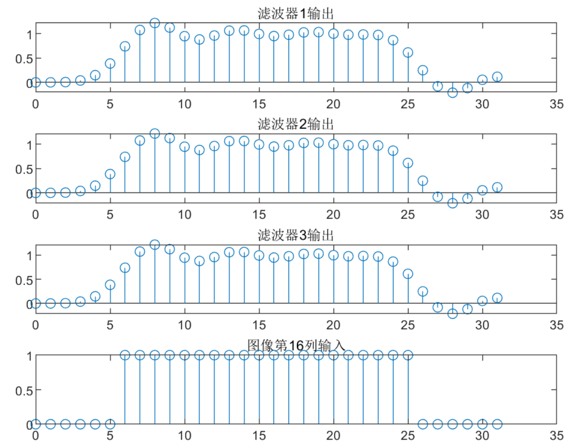

Figure 10: 图像第 16 列的输入和输出

根据以上分析及结果，我们可以通过循环遍历图像的每一列来进行二维图像的滤波，其滤波函数如下所示：

```
function y=filt2d(b, a, d, x)

[m,n]=size(x);

y=zeros(m,n);

for i=1:n

  w=filter(b,a,[x(:,i);zeros(d,1)]);

  y(:,i)=w(d:end-1);

end

end
```

通过该函数对 plus.mat 进行二维图像滤波，我们用之前搭建的三个滤波器进行滤波，可以得到如图 11、12、13所示的滤波结果。为使得滤波结果的对比效果更加明显，我对图像中每一个像素点进行 $\times 10$ 操作。

从三个图像中，我们可以看出，每个图像经过滤波之后都有所失真，而其中滤波器 2 的重影是最明显的，所以它的失真应是最为明显的，其主要原因是其 overshoot 最大。如此，我们便通过一维数字滤波器搭建起了二维的图像滤波器，只要选取合适的滤波器参数和偏移量就可以实现二维图像的滤波。

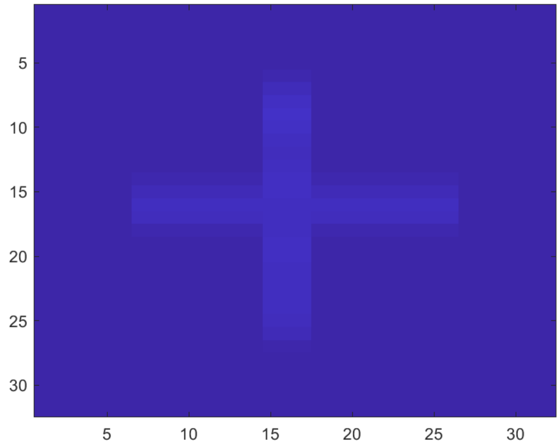

Figure 11: 滤波器 1 的二维图像滤波

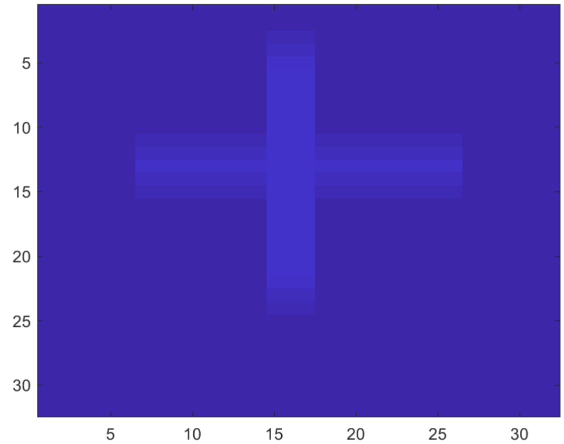

Figure 12: 滤波器 2 的二维图像滤波

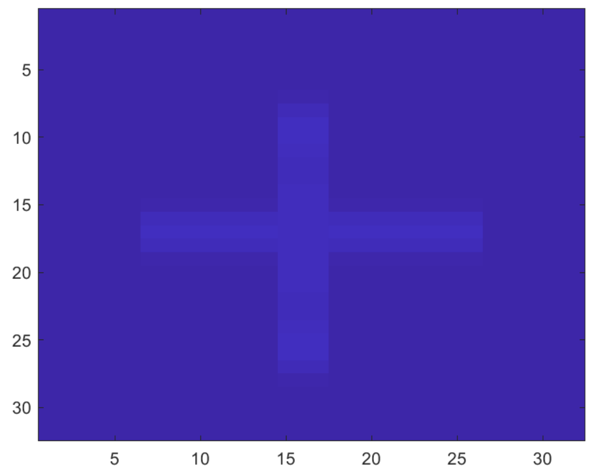

Figure 13: 滤波器 3 的二维图像滤波

三、 Question 3: 周期信号傅里叶级数表示

1. DTFS 系数合成离散时间信号

由离散傅里叶级数的计算式：

$$x[n]=\sum_{n=-\infty}^{+\infty}a_ke^{jk\omega_0n},\omega_0=\frac{2\pi}{N}\tag{11}$$

可以计算出 x[n] 的时域形式。matlab 的计算结果如图 14所示。

由于 a_k 与 a_{-k} 是共轭的关系，所以在计算的过程中，虚部会被抵消，所以最终还原出的 x[n] 应当是实数序列，虚部应为 0。而通过 matlab 的计算，我们也可以验证这一点。从图像上看，x[n] 的虚部全为 0，所以 x[n] 是全实数序列。

2. 三组周期信号的 DTFS 系数计算

在理论计算时，我们可以通过 DTFS 系数公式进行计算：

$$a_K=\sum_{k=<N>}x[n]e^{-jk\omega_0n}\tag{12}$$

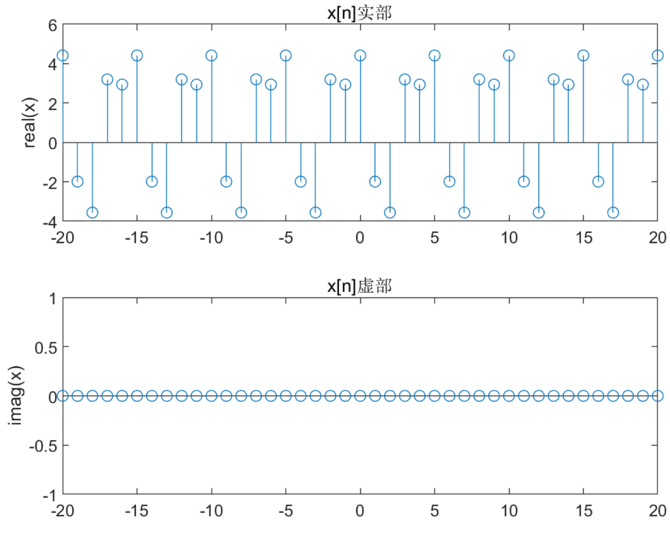

Figure 14: x 还原结果

而在计算机中，我们可以利用快速傅里叶变换来进行降阶运算，从而得到 DTFS 系数：

$$a_K = \frac{1}{N} fft(x) \tag{13}$$

在这一计算中，x 应带入一个周期内的序列，得到 N 个 a_k。

在本题中，我们选取了三个实数序列, 周期分别为 $N_1 = 8, N_2 = 16, N_3 = 32$，这三个函数的图像如图15所示：

$$x_1(x) = \begin{cases} 1, & 0 \le n \le 7; \end{cases} \tag{14}$$

$$x_1(x) = \begin{cases} 1, & 0 \le n \le 7; \\ 0, & 8 \le n \le 15 \end{cases} \tag{15}$$

$$x_1(x) = \begin{cases} 1, & 0 \le n \le 7; \\ 0, & 8 \le n \le 31 \end{cases} \tag{16}$$

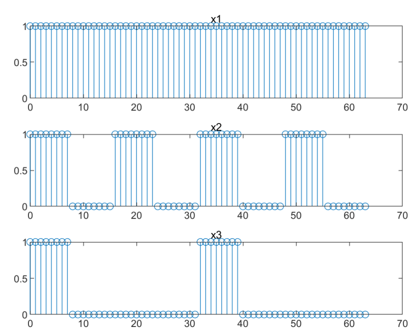

Figure 15: x[n] 图像

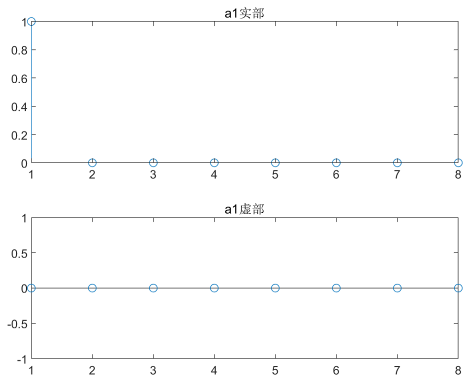

Figure 16: x1 的 DTFS 系数

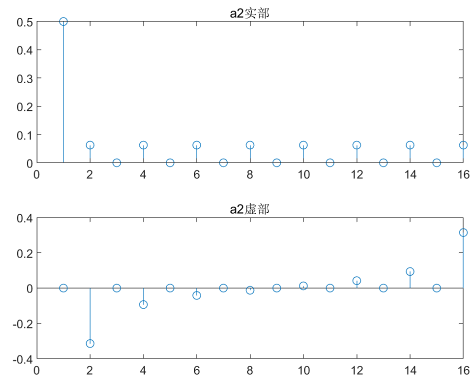

Figure 17: x2 的 DTFS 系数

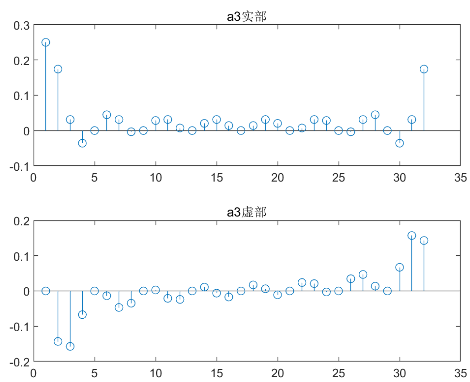

Figure 18: x3 的 DTFS 系数

在此基础上，通过 fft 函数计算得到三个序列的 DTFS 系数，其 matlab 计算结果如图 16、17、18所示。信号的 DC 值指的是信号一周期内的平均值，这三个序列的 DC 值为：

$$DC_1 = \frac{1}{N_1}\sum_{n=0}^{7} x_1[n] = 1 \tag{17}$$

$$DC_2 = \frac{1}{N_2}\sum_{n=0}^{15} x_2[n] = 0.5 \tag{18}$$

$$DC_3 = \frac{1}{N_3}\sum_{n=0}^{31} x_3[n] = 0.25 \tag{19}$$

将信号的 DC 值同其 DTFS 系数进行比较，可以发现，信号 DC 值和 DTFS 系数的实部 $n=0$ 时的值是相同的。在理论上也是如此，DC 值相当于 k 取 0 时的 DFS 运算，即：

$$a_0 = \frac{1}{N}\sum_{n=<N>} x[n]e^{-j0\omega_0 n} = \frac{1}{N}\sum_{n=<N>} x[n] = DC \tag{20}$$

3. 不同 k 取值下，x_3 的还原结果

当取不同个数的 DTFS 系数时，还原的原函数是右很大差别的。这里我们以 x_3 作为例子，分别选取 5 个、17 个、25 个及全部 DTFS 系数，观察其还原情况，通过 matlab 运算，我们得到其还原结果如图 19所

示。

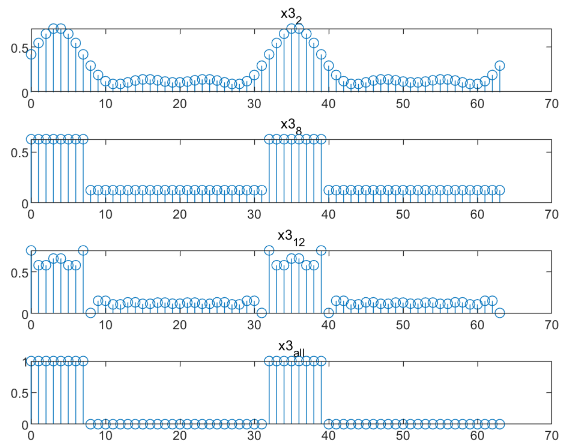

Figure 19: 不同 k 取值下 x3 还原结果

从图中我们可以看出，当 k 的取值较小时，还原结果会严重偏离原序列，且会有吉布斯现象的产生；而当 k 取所有值时（在此处取 x_3 对应的 0 到 31），能够得到同原序列相同的值。

并且理论上，由于原序列是实数序列，所以当 k 取全部的值时，还原的序列也应该是实数序列，其验证结果如图 20所示。

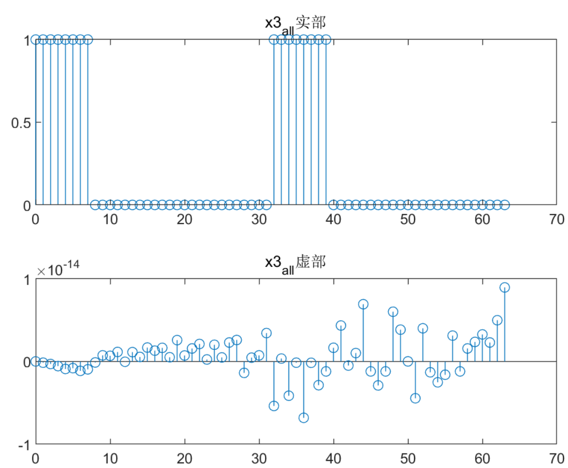

Figure 20: k 取全值时 x3 还原结果实虚部

在图中，x_3 的虚部的数量级在 10^{-14}，所以可以认为 x_3 还原结果是无虚数部分的，而这部分虚数小量的产生主要来源于数值计算的精度误差。故可以验证 k 取全部值时，x_3 的还原结果为实数序列。