Signals and Systems Laboratory Report

Name：凌智城

Class: 通信工程 1803

Student No.：201806061211

Submit Time：2019.12.27

Contents

Experiment Ⅰ

1. Objection

Learn how to generate signals in MATLAB,use the basic control statements,plot figures in MATLAB and write MATLAB file to complete some simple missions.

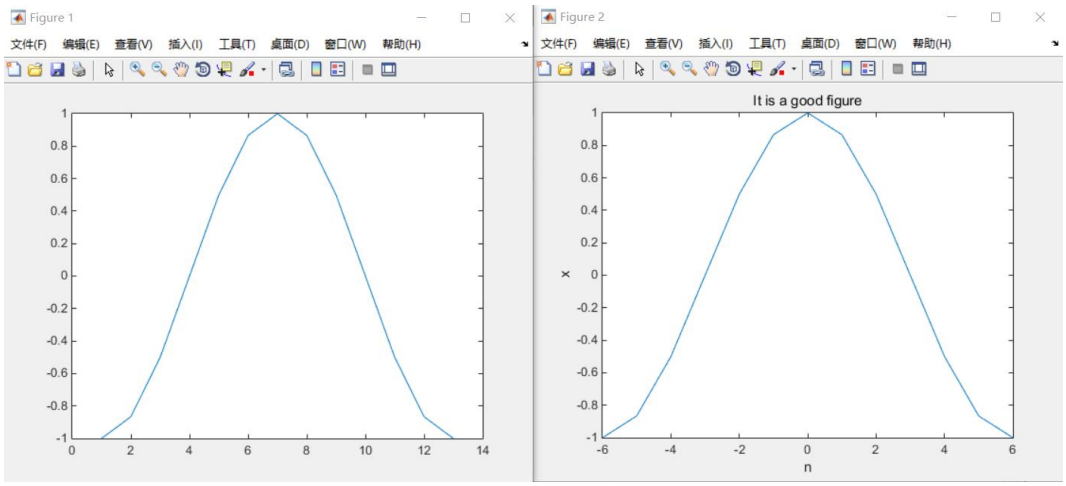
2. Experimental Content and Results

(1) 学习定义了复数和 cos，plot 和 figure 的使用，绘制图像的一些操作

仿真程序：

```
a = 1.25+2.36*1i;
b = cos(pi/3);
n = -6:1:6;
x = cos(pi/6*n);
plot(x);
figure;
plot(n,x);
xlabel('n');
ylabel('x');
title('It is a good figure');
```

仿真结果：

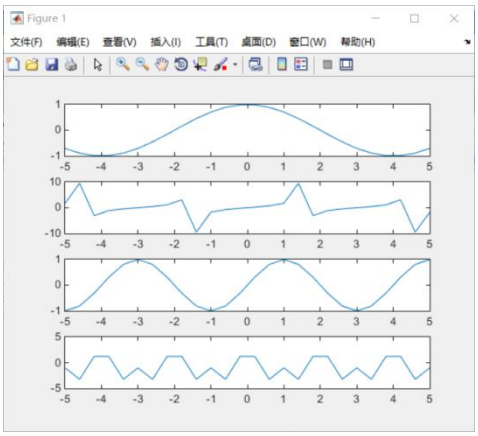


(2) 尝试了一些 sin，cos，tan 和 cot 的函数，学习 subplot 在一个 figure 上绘制多张图

仿真程序：

```
n = -5:0.4:5;
x1 = cos(pi/4*n);
x2 = tan(pi/3*n);
x3 = sin(pi/2*n);
x4 = sec(pi/1*n);
subplot(4,1,1);plot(n,x1);
subplot(4,1,2);plot(n,x2);
subplot(4,1,3);plot(n,x3);
subplot(4,1,4);plot(n,x4);
```

仿真结果：



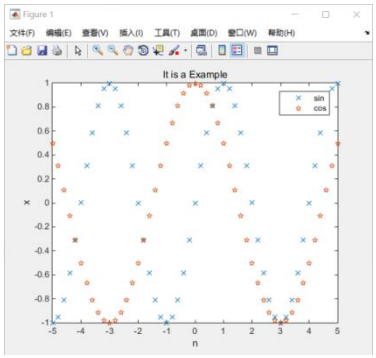
Matlab 可以直接调用一些比较简单的三角函数以及 sec，csc，cot 这些，subplot 是在一个 figure 中绘制多幅图，三个参数，第一个参数是有几行，第二个参数是有几列，第三个参数是表示这是第几幅。

(3) 学习在一张表上画多条线 hold on/hold off，以及 legend 的使用

仿真程序：

```
n = -5:0.2:5;
x1 = sin(pi/2*n);
x2 = cos(pi/3*n);
plot(n,x1,'x');
hold on;
plot(n,x2,'p');
hold off;
xlabel('n');
ylabel('x');
title('It is a Example');
legend('sin','cos');
```

仿真结果：



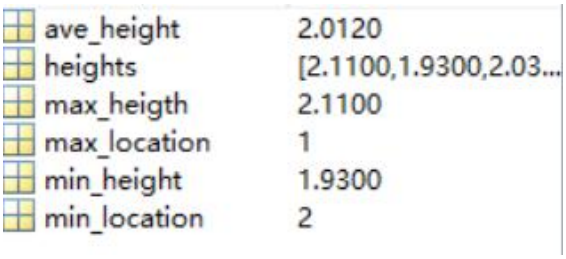
Hold on 继续在这张表上画，hold off 结束绘制，用 legend 来标注区分，并且在 plot 时用第三个变量可以自定义坐标点的样式。

(4) 学习使用 max，min 和 mean 函数求最大值最小值平均值

仿真程序：

```
heights = [2.11,1.93,2.03,2.03,1.96];
[max_heigth,max_location] = max(heights);
[min_height,min_location] = min(heights);
ave_height = mean(heights);
```

仿真结果：



增加参数参数 max_location, min_location 就可以返回最大值和最小值在数组中的位置。

(5) 自己编写第一个函数并且尝试在来另一个 script 中调用此函数

仿真程序：

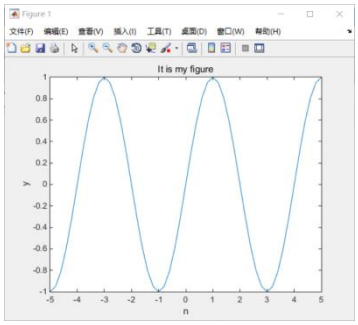
函数代码：

```
function [y] = MyFigure(n,p)
y = sin(pi/p*n) ;
plot(n,y);
xlabel('n');
ylabel('y');
title('It is my figure');
```

程序代码：

```
n = [-5:0.2:5];
p = 2;
[y] = MyFigure(n,p);
```

仿真结果：



(6) 输入一个原始信号 signal 和噪声信号 noise 叠加后，经过函数处理后不同的 alpha 值对应不同输出

仿真程序：

函数代码：

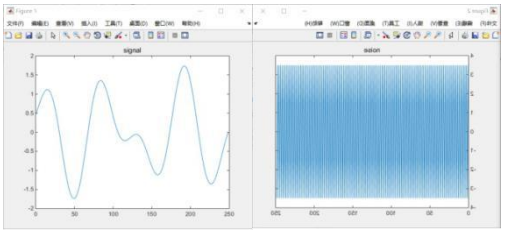
```
function [y]=MyProblem_fu(x,n,alpha)
N=length(n);
y=zeros(N,1);
y(1)=0+(1-alpha)/(1+alpha)*(x(1)+0);

for loop=2:N
   y(loop)=alpha*y(loop-1)+(1-alpha)/(1+alpha)*
(x(loop)+alpha*x(loop-1));
end
figure
plot(n,x)
title('x')
figure
plot(n,y)
title('y')
```

程序代码：

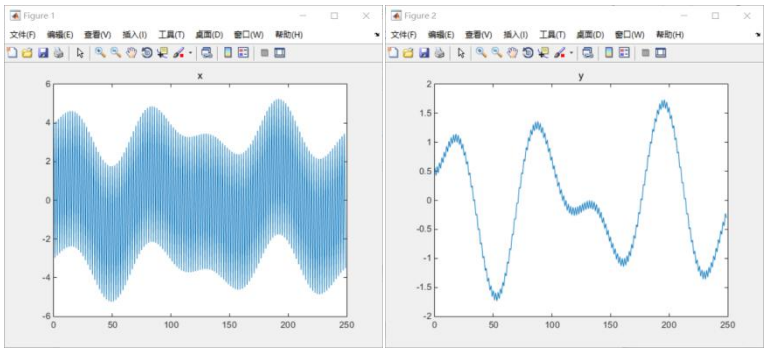
```
n = 0:1:249 ;
s = cos(0.021*pi*n)-0.75*cos(0.035*pi*n+pi/4);
e = 3.5*cos(pi*n);
x = s+e;
alpha = 0.75;
[y] = MyProblem_fu(x,n,alpha);
```

仿真结果：

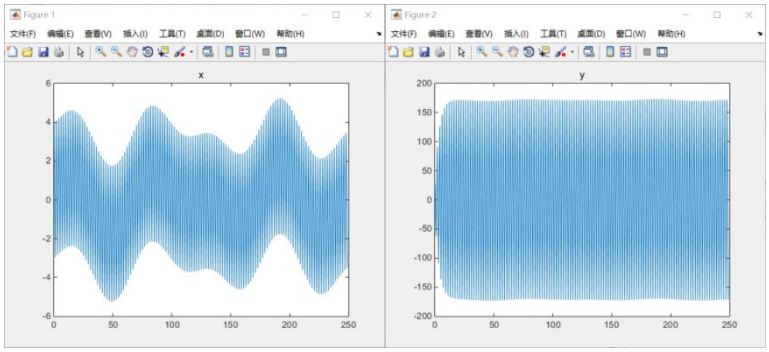
原始输入图像：



Alpha=0.75：



Alpha=-0.75：



3. Experimental Conclusion

通过这次实验，了解了 MATLAB 的一些最为基础的内容，学会了如何创建保存 script 等操作，熟悉了一些基本的 function 如 abs,cos,exp,figure,plot 以及 subplot 等，学会了如何编写和调用自己的 function，为以后进一步学习 MATLAB 奠定了良好的基础。

Experiment Ⅱ

1. Objection

Enhance the understanding of the properties of LTI systems.Gain more knowledge on the concepts of convolution and impulse response.

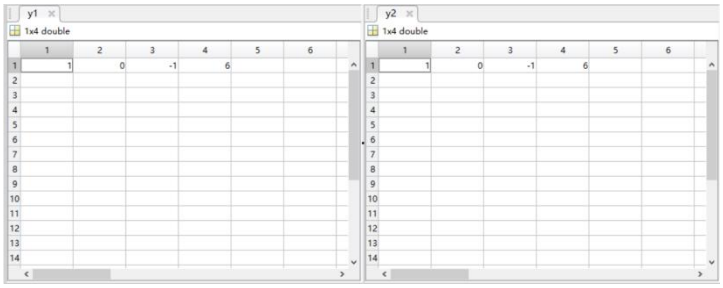
2. Experimental Content and results

（1）使用 conv 函数卷积并观察输出结果，验证卷积的交换律和线性规律

仿真程序：

```
x1 = [1 -2 3];
x2 = [1 2];
y1 = conv(x1,x2);
y2 = conv(x2,x1);
x11 = [1 0 0];
x12 = [0 -2 0 ];
x13 = [0 0 3];
y11 = conv(x11,x2);
y12 = conv(x12,x2);
y13 = conv(x13,x2);
y3 = y11+y12+y13;
```

仿真结果：



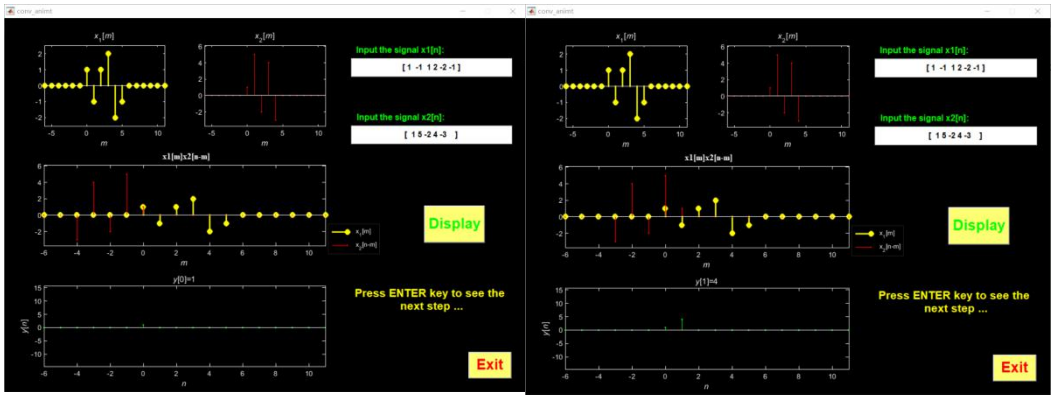
X1 与 X2 的卷积 和 X2 与 X1 的卷积相同

y3=y11+y12+y13 && y1=y2=y3

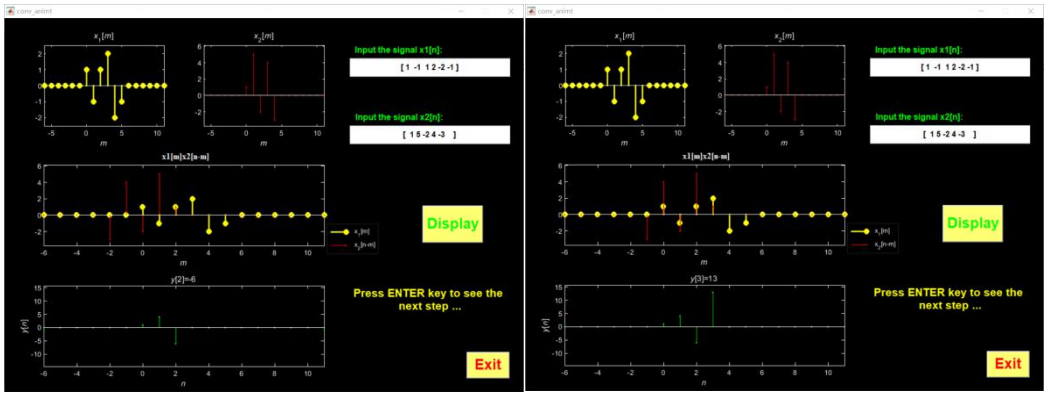
conv(x1,x2) = conv(x11,x2)+conv(x12,x2)+conv(x13,x2).

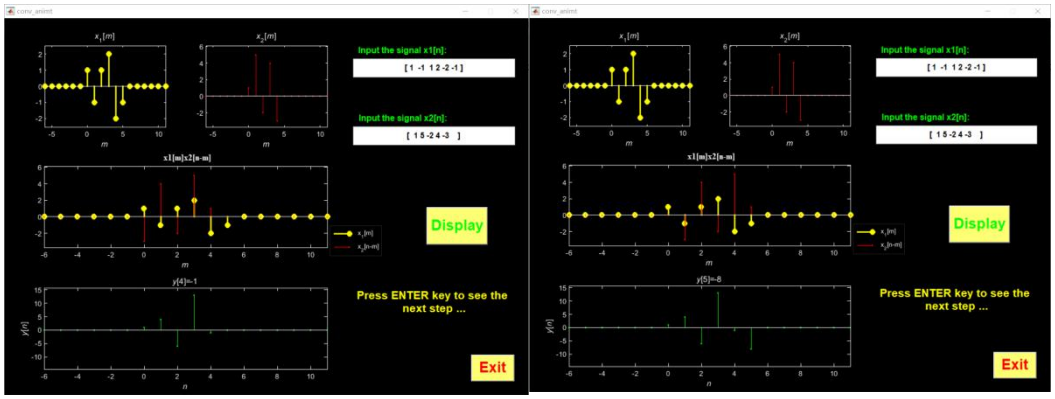
（2）用 conv_animt 来进一步分析卷积的过程

$$y[n] = \sum_{m=-\infty}^{\infty} x_1[m]x_2[n-m]$$

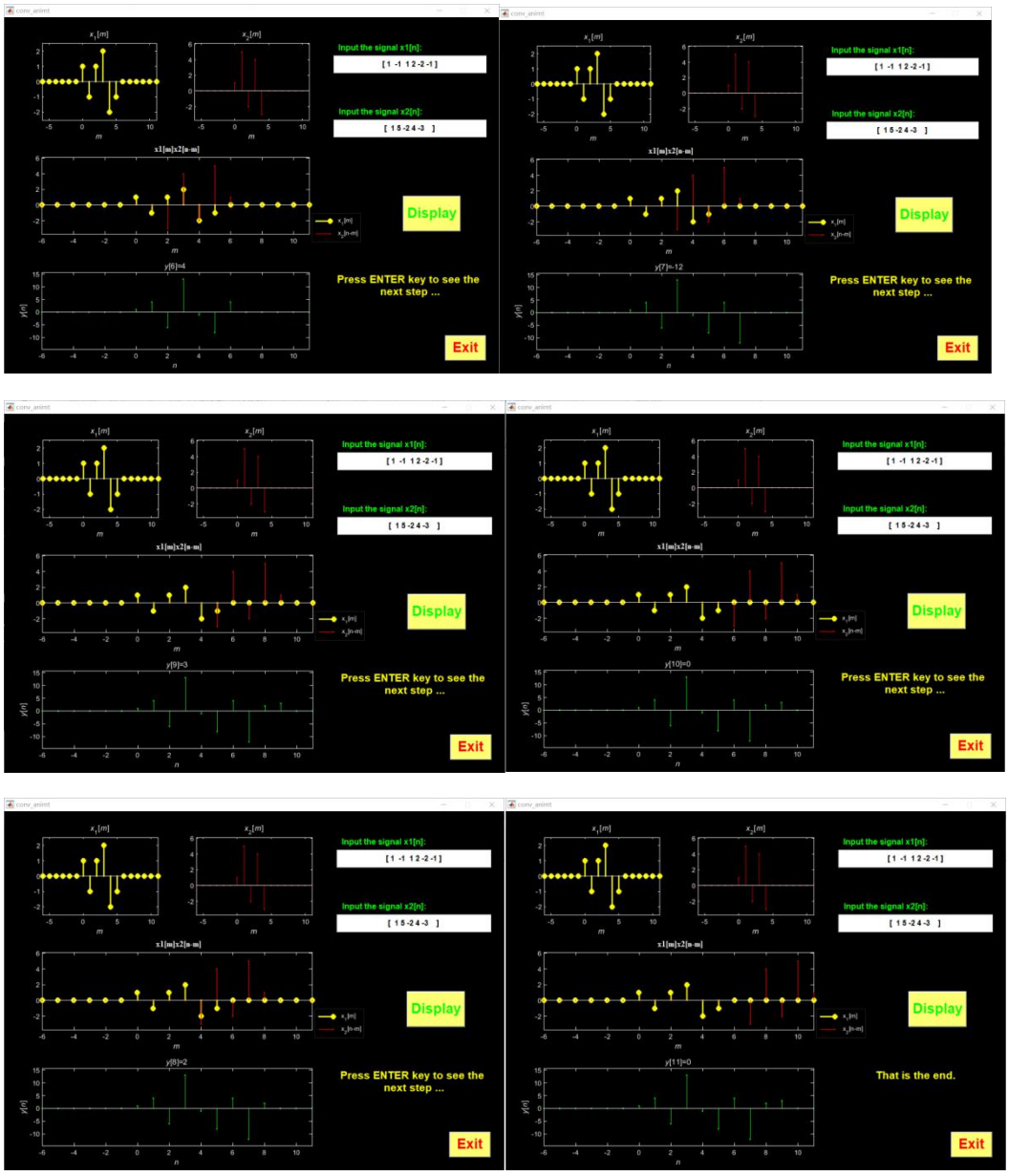


x1 和 x2 还没相遇时，不会产生响应





x2[n-m]随着 n 的变化会与 x1[m]相遇从而产生响应



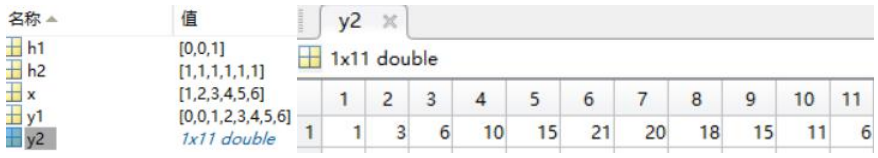
随着 x2[n-m]的 n 慢慢变大，与 x1[m]不再有交点后，随即不再产生响应。

（3）验证单位冲激响应

仿真程序：

```
x = [1 2 3 4 5 6];
h1 = [0 0 1];
y1 = conv(x,h1);
h2 = ones(1,length(x));
y2 = conv(x,h2);
```

仿真结果：



（4）观察不同的音乐均衡器，对不同频率的声音的过滤作用

仿真程序：

```
①Load the audio signal:
>>[origin,fs,nbits]wavread('mymusic.mat');
②listen the origin music:
>>wavplay(origin,fs);
③load the unit impulse response and make it a row vector:
>>load speech;    >>b = speech';
④now can transform it by using different equalizers:
>>eqlzed(:,1) = conv(origin(:,1),b);
>>eqlzed(:,2) = conv(origin(:,2),b);
⑤listen the equalized audio signal:
>>wavplay(eqlzed,fs);
```

msceq.m:

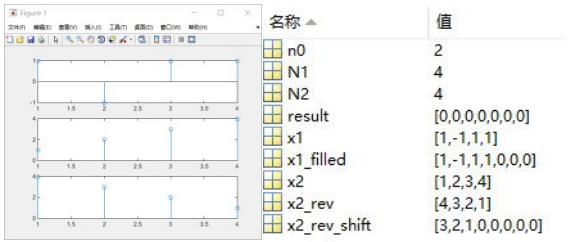
原始声音信号主要集中在低频，使用 low pass 可以滤去高频杂音，
使用 high pass 几乎将所有人声滤去，只剩下乐器的声音，
使用 speech 均衡可以得到更为清晰纯净的人声。

（5）初步认识图形卷积以及 reverse&&time shift

仿真程序：

```
x1=[1 -1 1 1];
x2=[1 2 3 4];
N1=length(x1);
N2=length(x2);
result=zeros(1,N1+N2-1);
x1_filled=[x1 zeros(1,N2-1)];
n0 = 2;
x2_rev=fliplr(x2) ;
x2_rev_shift=[x2_rev(N2-n0:N2) zeros(1,N1+N2-n0-1)];
```

仿真结果：



名称	值
n0	2
N1	4
N2	4
result	[0,0,0,0,0,0,0]
x1	[1,-1,1,1]
x1_filled	[1,-1,1,1,0,0,0]
x2	[1,2,3,4]
x2_rev	[4,3,2,1]
x2_rev_shift	[3,2,1,0,0,0,0,0]

fliplr 是逆转一个 vector

3. Experimental Conclusion

通过这次实验，对卷积有了更加深入的了解，对卷积的每一步分解在看了图形化的展示之后印象更加深刻，了解了 LTI 系统的单位冲激响应并且对 music equalizer 的函数结构有了初步认识，最后，也对图形化的卷积的 matlab 代码有了一些了解。

Experiment Ⅲ

1. Objection

To learn the Fourier transform of signals,filtering and its applications in MATLAB environment.

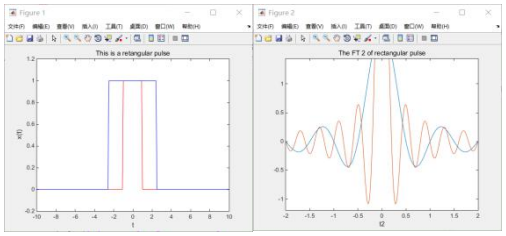
2. Experimental Content and Results

（1）学习了定义时域信号以及其傅里叶变换，并且做出其波特图

仿真程序：

```
tarray = -10:0.1:10;
T1 = 2;
S1 = rectpuls(tarray,T1);
T2 = 5;
S2 = rectpuls(tarray,T2);
plot(tarray,S1,'r');
hold on;
plot(tarray,S2,'b');
axis([-10 10 -0.2 1.2]);
title('This is a retangular pulse');
xlabel('t');
ylabel('x(t)');

figure;
syms t1 f1;
F1 = int(exp(-1i*2*pi*f1*t1),t1,-T1/2,T1/2);
ezplot(F1,-2,2);
xlabel('t1')
title('The FT 1 of rectangular pulse');

hold on;
syms t2 f2;
F2 = int(exp(-1i*2*pi*f2*t2),t2,-T2/2,T2/2);
ezplot(F2,-2,2);
xlabel('t2')
title('The FT 2 of rectangular pulse');
```

仿真结果：



y=rectpuls(t,width)，用以产生一个幅值为 1，宽度为 width，相对于 t=0 点左右对称的矩形波信号，该函数的横坐标范围由向量 t 决定，是以 t=0 为中心向左右各展开 width/2 的范围，width 的默认值为 1。

exp(-j*2*pi*f*t)来表示$e^{-j2\pi ft}$，plot 绘制较为一般的图像，ezplot 可以绘制隐函数等图像。

Suggestion extension:

when the width (in time) of the rectangular pulse is increased,what happened to the width of the frequency spectrum?

初始的方波信号的宽度增加之后，频谱的密度反而会增加增加，宽度越大，频谱密度越大

（2）在不同的输入信号频率下测试输出幅度以及幅度响应

仿真程序：

```
fs = 1000;
t = 0:1/fs:2;
f = 90;
x = cos(2*pi*f*t);
plot(t,x);
axis([0 1 -1.2 1.2]);
fc = 50;
N = 20;
[b] = filter_sys(N,fc,0);
y = filter(b,1,x);
plot(y);
output_magnitude = max(y);
```

（不断的改变 f 从 0~90hz）

仿真结果：

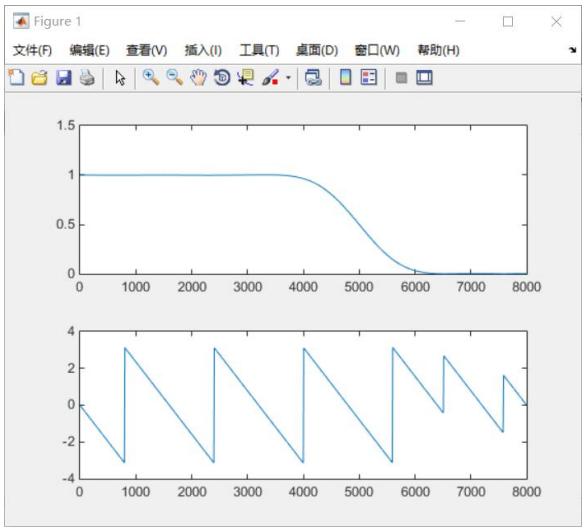
Frequency（Hz）	0	10	20	25	35	40	45	50	55	60	65	70	75	80	90
Input magnitude	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Output magnitude	1	0.98	0.92	0.88	0.78	0.72	0.66	0.59	0.53	0.47	0.41	0.35	0.29	0.24	0.16
Magnitude response	1	0.98	0.92	0.88	0.78	0.72	0.66	0.59	0.53	0.47	0.41	0.35	0.29	0.24	0.16

（3）学习对一个简单的滤波器

仿真程序：

```
fs = 16000;
fc = 5000;
wc = 2*fc/fs;
b = fir1(20,wc,'low');
[H,w] = freqz(b,1,2^10,fs);
subplot(2,1,1);
plot(w,abs(H));
subplot(2,1,2);
plot(w,angle(H));
```

仿真结果：



（4）Filtering of speech

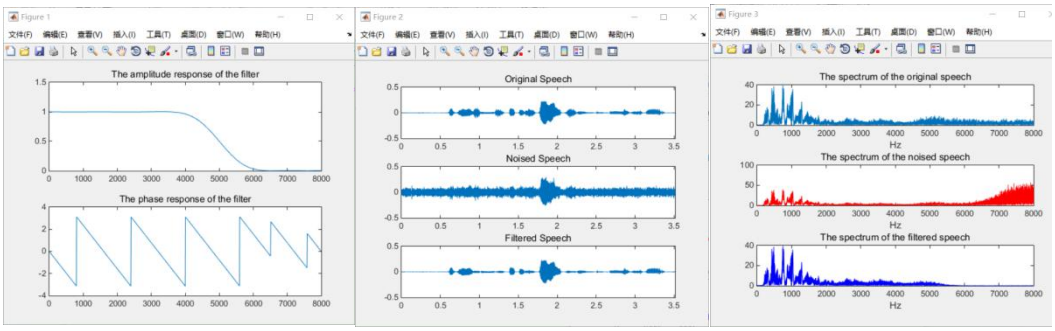
仿真程序：

```
function [S_in, S_out]=E3_speech_Filt( )
fs=16000;
N =input('the order of the filter:');
para=input('select the filter type:0---low pass\1---high pass:\2---bandpass\3---stoppass:');
switch para
    case 0
        fc= input('the cutoff frequency of the filter Hz:');
        wc=2*fc/fs;
        b=fir1(N,wc,'low');
    case 1
        fc= input('the cutoff frequency of the filter Hz:');
        wc=2*fc/fs;
        b=fir1(N,wc,'high');
    case 2
        fc= input('the central frequency of the filter Hz:');
```

```
            fw= input('the passband width of the filter Hz:');
            w1=2*(fc-fw/2)/fs;
            w2=2*(fc+fw/2)/fs;
            b=fir1(N,[w1 w2]);
        case 3
            fc= input('the central frequency of the filter Hz:');
            fw= input('the stopband width of the filter Hz:');
            w1=2*(fc-fw/2)/fs;
            w2=2*(fc+fw/2)/fs;
            b=fir1(N,[w1 w2],'stop');
        otherwise
            disp('No such filter. Use the configuration at last
time');
    end

    [H,w]=freqz(b,1,2^10,fs);
    subplot(2,1,1);
    plot(w,abs(H));
    title('The amplitude response of the filter');
    subplot(2,1,2);
    plot(w,angle(H));
    title('The phase response of the filter');
    fs=16000;
    ts=1/fs;
    y=audioread('f1_16.wav');
    S_in=audioread('f1_16noisespeech.wav');
    [r c]=size(S_in);
    t=0:r-1;
    t=t*ts;
    S_out=filter(b,1,S_in);
    NL=r*c; L=fix(NL/2);
    f=0:1:L-1;
    farray=f*fs/NL;
    y1=fft(y);
    y1=y1(1:L);
    y2=abs(y1);
    X1=fft(S_in);
    X1=X1(1:L);
    X2=abs(X1);
    Y1=fft(S_out);
    Y1=Y1(1:L);
    Y2=abs(Y1);
    figure;
    subplot(3,1,1);
```

```
plot(t,y);
title('Original Speech');
axis([t(1) t(r) -0.5 0.5]);
subplot(3,1,2);
plot(t,S_in);
title('Noised Speech');
axis([t(1) t(r) -0.5 0.5]);
subplot(3,1,3);
plot(t,S_out);
title('Filtered Speech');
axis([t(1) t(r) -0.5 0.5]);

figure;
subplot(3,1,1);
plot(farray,y2);
title('The spectrum of the original speech');
xlabel('Hz');
subplot(3,1,2);
plot(farray,X2,'r');
title('The spectrum of the noised speech');
xlabel('Hz');
hold on;
subplot(3,1,3);
plot(farray,Y2,'b');
title('The spectrum of the filtered speech');
xlabel('Hz');
```

仿真结果：



(5) Filtering of Image

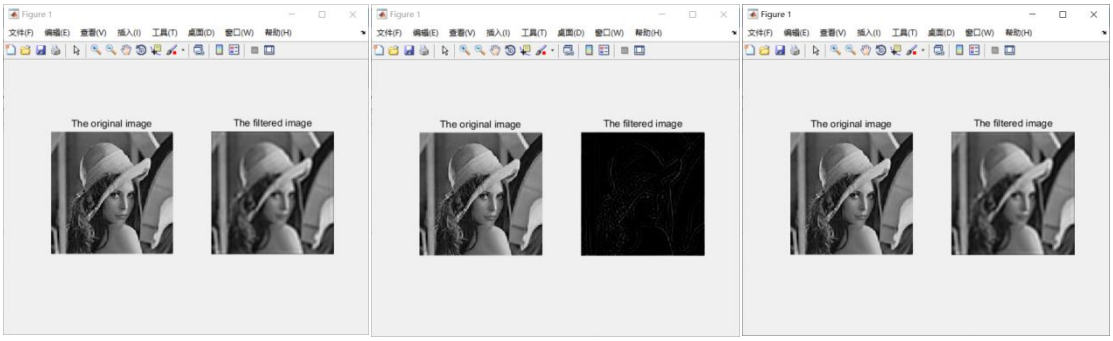
仿真程序：

```
function E3_image()
clc;
clear;
inp2=imread('Lena.tif');
myfilter=fir1(20,0.4,'low');
myfilter2=ftrans2(myfilter);
```

```
outp2=imfilter(inp2,myfilter2);
subplot(1,2,1);
imshow(inp2);
title('The original image');
hold on;
subplot(1,2,2);
imshow(outp2);
title('The filtered image');
```

（在 myfilter 中改变参数，low，high，band）

仿真结果：



依次为 low pass，high pass，band pass 滤波后的结果。

3. Experimental Conclusion

通过这次 MATLAB 实验，对傅里叶变换信号以及一些滤波的过程有了较为详细的了解，对学习信号与系统有一定的帮助，同幅度不同输入频率的幅度响应并不相同，并且学会了 exp，int（）等函数的使用，对后续 matlab 的学习有较大的帮助。

Experiment Ⅳ

1. Objection

Learn the sampling theorem for signals.Know how to reconstruct the continuous-time signals in MATLAB environment.

2. Experimental Content and Results

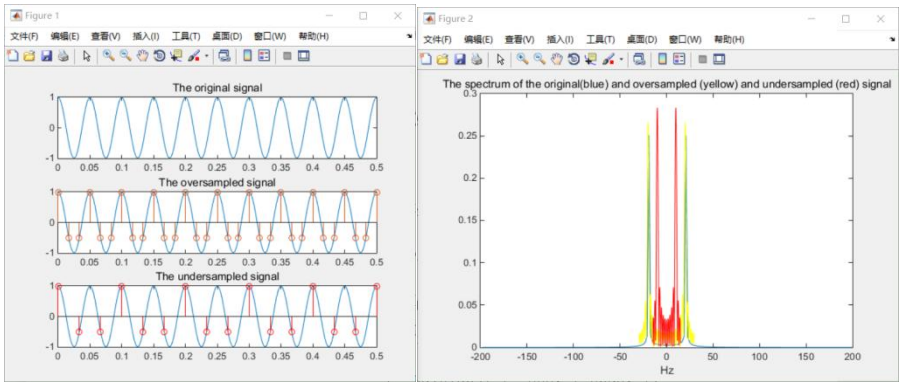
(1) 对欠采样和过采样的分析

仿真程序：

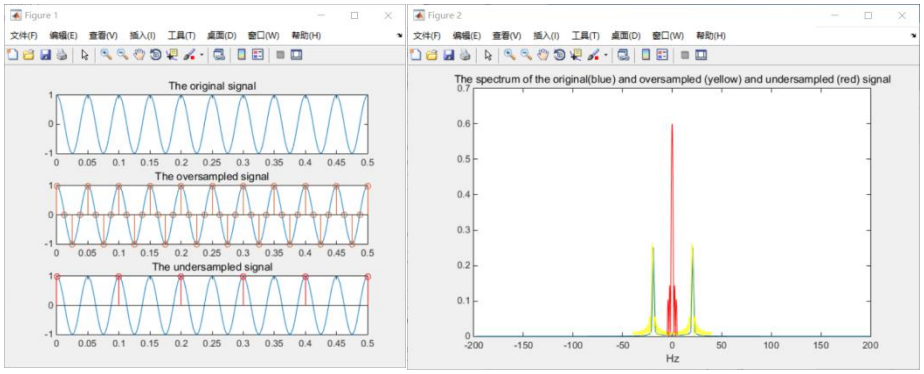
```
function sampling(signal_f, over_f,under_f)
. . . . . .
```

仿真结果：

sampling(20,60,30)



sampling(20,80,10)



大于两倍固有频率的是过采样，可以较好地恢复出原信号；

效于两倍固有频率采样的是欠采样，不能较好地恢复出原信号，会产生信号的混叠。

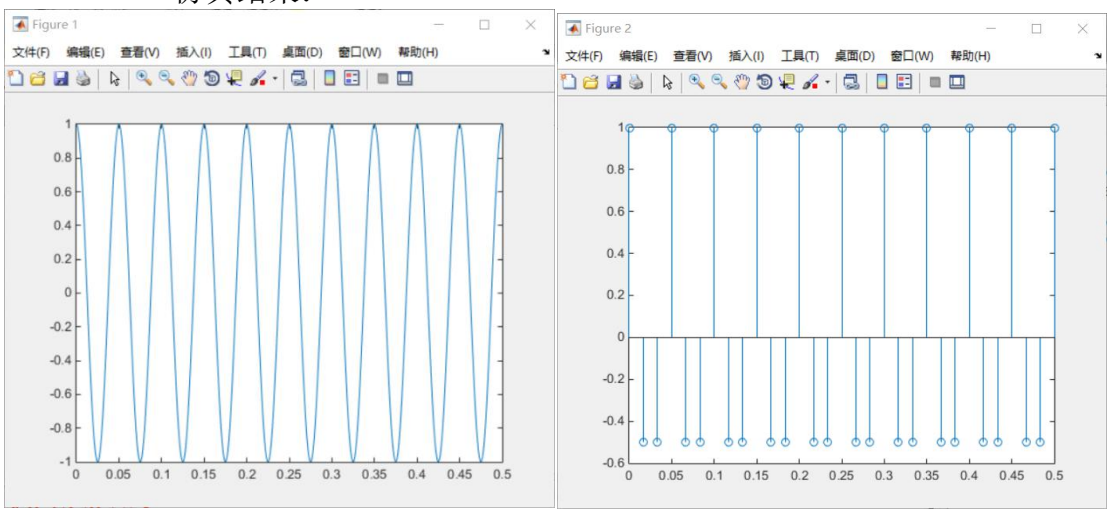
(2) 采样前连续时间信号与采样后的离散时间信号对比，更进一步理解什么是采样

仿真程序：

```
signal_f=20;
```

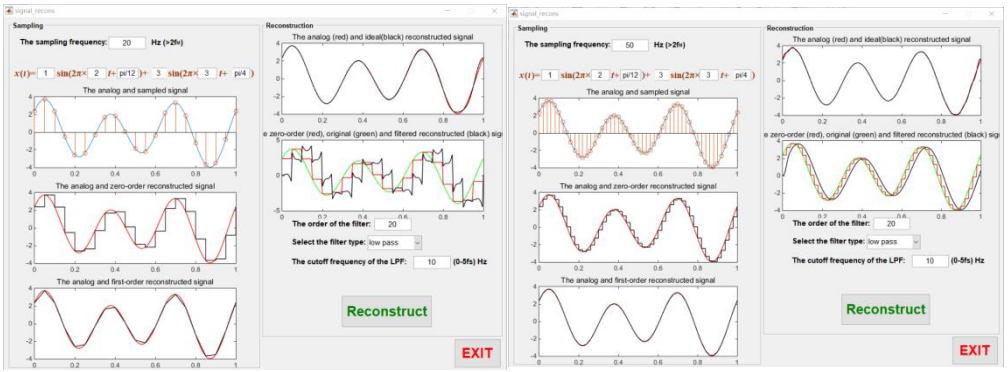
```
t=0:1/(50*signal_f):0.5;
x=cos(2*pi*signal_f*t);
plot(t,x);    % 画出采样前的信号
sampling_f=60;  %采样频率 60Hz
tss=0:1/sampling_f:0.5; %定义离散的时间
xs=cos(2*pi*signal_f*tss);
figure;
stem(tss,xs); %画出采样后离散的信号
T = 1/sampling_f;
X=fftshift(fft(x))*T;
```

仿真结果：



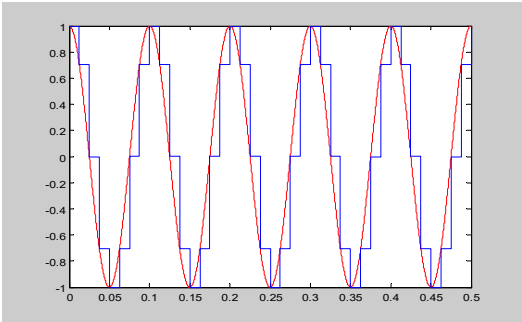
采样就是对一个已有的连续时间信号，以一个采样频率 sampling_f 对信号进行离散时间下的数据收集，采集原信号的样本，然后重新汇集成一个新的信号。

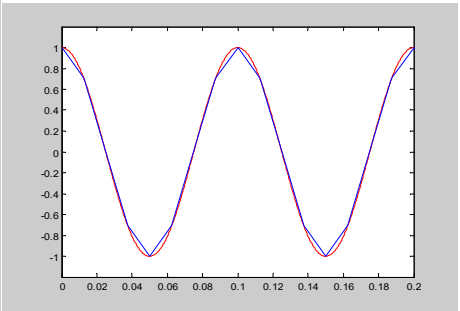
(3) 通过已有的 sampling_recons.m 对采样的零阶保持和一阶保持分析，以及滤波后的信号对比。



采样频率越大，对初始信号的真实性保留程度越好，恢复出的信号也就越接近原信号，但同时也会大大增加采样的难度，故选取一个何是的采样频率格外重要。

(4) 分析零阶保持和一阶保持的区别以及 smooth this plot





零阶保持使用的是矩形函数，一阶保持使用的是三角形函数，一阶保持在原信号的恢复上较零阶保持更为平滑与真实，但任然存在着一些失真部分。

3. Experimental Conclusion

通过以上所有的信号与系统 MATLAB 仿真实验，对 MATLAB 有了一个初步的了解，学会了 MATLAB 的基本操作方法，加深了对 LTI 系统中的卷积，傅里叶变换，滤波，采样等知识点的印象，更好地理解了相关知识点，对信号与系统课程的学习有很大的帮助。