Lab 2：Linear Time-Invariant Systems

Author	Name：毕云天 唐心宇	Student ID：12112501 11911817

Introduction

1.Get impulse response of systems and check the properties of calculation of convolution.

2.learning to use MATLAB function to verity whether system is linear or time-invariant

3.Use autocorrelation to get alpha and N of echo systems.

Lab results & Analysis：

2.4

■ 2.4 Properties of Discrete-Time LTI Systems

In this exercise, you will verify the commutative, associative and distributive properties of convolution for a specific set of signals. In addition, you will examine the implications of these properties for series and parallel connections of LTI systems. The problems in this exercise will assume that you are comfortable and familiar with the `conv` function described in Tutorial 2.1. Although the problems in this exercise solely explore discrete-time systems, the same properties are also valid for continuous-time systems.

Basic Problems

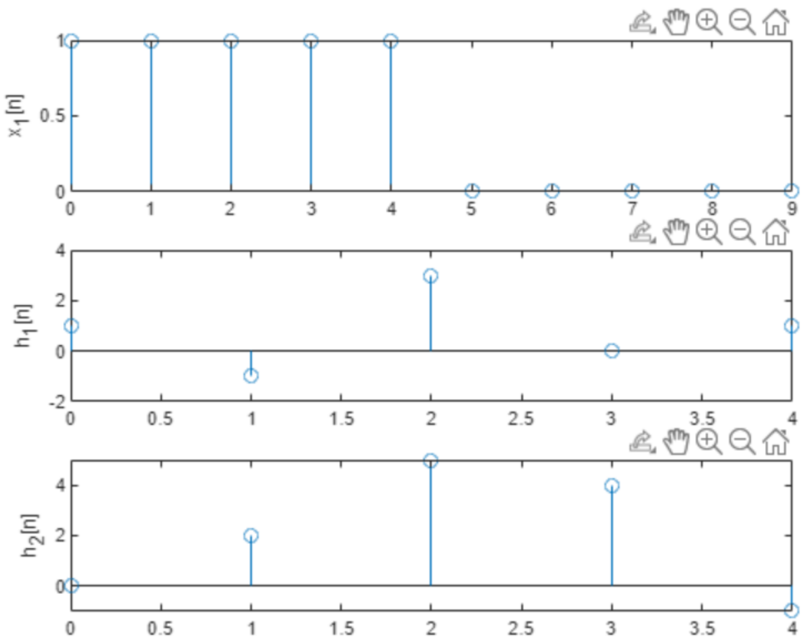
(a). Many of the problems in this exercise will use the following three signals:

$$x_1[n] = \begin{cases} 1, & 0 \le n \le 4, \\ 0, & \text{otherwise}, \end{cases}$$

$$h_1[n] = \begin{cases} 1, & n = 0, \\ -1, & n = 1, \\ 3, & n = 2, \\ 1, & n = 4, \\ 0, & \text{otherwise}, \end{cases}$$

$$h_2[n] = \begin{cases} 2, & n = 1, \\ 5, & n = 2, \\ 4, & n = 3, \\ -1, & n = 4, \\ 0, & \text{otherwise}. \end{cases}$$

Define the MATLAB vector **x1** to represent $x_1[n]$ on the interval $0 \le n \le 9$, and the vectors **h1** and **h2** to represent $h_1[n]$ and $h_2[n]$ for $0 \le n \le 4$. Also define **nx1** and **nh1** to be appropriate index vectors for these signals. Make appropriately labeled plots of all the signals using **stem**.



(b). The commutative property states that the result of a convolution is the same regardless of the order of the operands. This implies that the output of an LTI system with impulse response $h[n]$ and input $x[n]$ will be the same as the output of an LTI system with impulse response $x[n]$ and input $h[n]$. Use **conv** with **h1** and **x1** to verify this property. Is the output of **conv** the same regardless of the order of the input arguments?

```
ans = isequal(y1, y2)
```

```
ans = logical
   1
```

y1 and y2 are the same.

(c). Convolution is also distributive. This means that

$$x[n] * (h_1[n] + h_2[n]) = x[n] * h_1[n] + x[n] * h_2[n] .$$

This implies that the output of two LTI systems connected in parallel is the same as one system whose impulse response is the sum of the impulse responses of the parallel systems. Figure 2.8 illustrates this property.

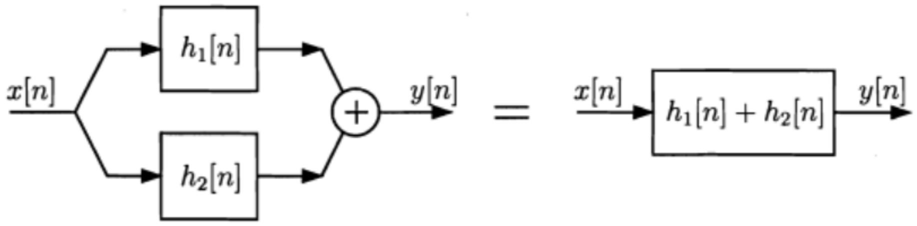


Figure 2.8. Distributive property of convolution.

Verify the distributive property using **x1**, **h1** and **h2**. Compute the sum of the outputs of LTI systems with impulse responses $h_1[n]$ and $h_2[n]$ when $x_1[n]$ is the input. Compare this with the output of the LTI system whose impulse response is $h_1[n] + h_2[n]$ when the input is $x_1[n]$. Do these two methods of computing the output give the same result?

let y1 be the output of the left picture, and y2 be the other.

```
ans = isequal(y1, y2)
```

```
ans = logical
   1
```

y1 and y2 are the same.

(d). Convolution also possesses the associative property, i.e.,

$$(x[n] * h_1[n]) * h_2[n] = x[n] * (h_1[n] * h_2[n]) \,.$$

This property implies that the result of processing a signal with a series of LTI systems is equivalent to processing the signal with a single LTI system whose impulse response is the convolution of all the individual impulse responses of the connected systems. Figure 2.9 illustrates this property for the case of two LTI systems connected in series.

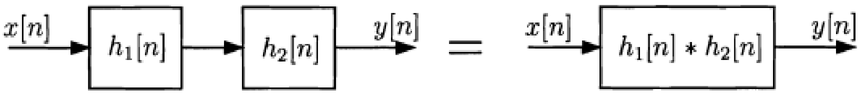


Figure 2.9. Associative property of convolution.

Use the following steps to verify the associative property using **x1**, **h1** and **h2**:

- Let $w[n]$ be the output of the LTI system with impulse response $h_1[n]$ shown in Figure 2.9. Compute $w[n]$ by convolving $x_1[n]$ and $h_1[n]$.
- Compute the output $y_{d1}[n]$ of the whole system by convolving $w[n]$ with $h_2[n]$.
- Find the impulse response $h_{\text{series}}[n] = h_1[n] * h_2[n]$.
- Convolve $x_1[n]$ with $h_{\text{series}}[n]$ to get the output $y_{d2}[n]$.

Compare $y_{d1}[n]$ and $y_{d2}[n]$. Did you get the same results when you process $x_1[n]$ with the individual impulse responses as when you process it with $h_{\text{series}}[n]$?

```
ans = isequal(yd1, yd2)
```

```
ans = logical
   1
```

$y_{d1}[n] = y_{d2}[n]$

Intermediate Problems

(e). Suppose two LTI systems have impulse responses $h_{e1} = h_1[n]$ and $h_{e2}[n] = h_1[n - n_0]$, where $h_1[n]$ is the same signal defined in Part (a) and n_0 is an integer. Let $y_{e1}[n]$ and $y_{e2}[n]$ be the outputs of these systems when $x[n]$ is the input. Use the commutative property to argue that the outputs will be the same if you interchange the input and impulse response of each system. Notice that once you have done this the two systems have the same impulse response and the inputs are delayed versions of the same signal. Based on this observation and time-invariance, argue that $y_{e2}[n] = y_{e1}[n - n_0]$. Use MATLAB to confirm your answer for the case when $n_0 = 2$ and the input $x[n]$ is the signal $x_1[n]$ defined in Part (a).

according to commutative property.

conv($h_1[n]$, x[n]) = conv(x[n], $h_1[n]$)

conv(h_1 [n - n_0], x[n]) = conv(x[n], h_1 [n - n_0])

so the output is the same after exchanging input and impulse response.

because the system is time invariant.

and h_{e2}[n] = h_{e1}[n - n_0]

so y_{e2}[n] = y_{e1}[n - n_0]

```
ye1 = conv(x1, he1)
```

```
ye1 = 1×16
     1     0     3     3     4     3     4     1     1     0     0     0     0     0     0     0
```

```
ye2 = conv(x1, he2)
```

```
ye2 = 1×16
     0     0     1     0     3     3     4     3     4     1     1     0     0     0     0     0
```

```
ans = isequal(ye1(1:end-2), ye2(3:end))
```

```
ans = logical
   1
```

the answer is y_{e2}[n] = y_{e1}[n - n_0]

(f). Consider two systems connected in series; call them System 1 and System 2. Suppose System 1 is a memoryless system and is characterized by the input/output relationship $y[n] = (n+1)x[n]$, and System 2 is LTI with impulse response $h_{f2}[n] = h_1[n]$ as defined in Part (a). Suppose you decide to investigate whether or not the associative property of convolution holds for the series connection of these two systems by following the steps:

- Let $w[n]$ be the output of System 1 when the input is $x_1[n]$ as defined above. Use **nx1** and **x1** with the termwise multiplication operator **.*** to define a MATLAB vector **w** to represent $w[n]$.
- Use $w[n]$ as the input to System 2, and let the output of that system be $y_{f1}[n]$. Compute **yf1** in MATLAB using **w** and **h1**.
- Let $h_{f1}[n]$ be the output of System 1 when the input to the system is $\delta[n]$. Define a vector **hf1** to represent this signal over the interval $0 \le n \le 4$.
- Let $h_{\text{series}}[n] = h_{f1}[n] * h_{f2}[n]$. Compute a vector **hseries** to represent this signal.
- Let $y_{f2}[n]$ be the output of an LTI system whose impulse response is $h_{\text{series}}[n]$ when the input is $x_1[n]$. Compute **yf2** in MATLAB using **hseries** and **x1**.

Does $y_{f1}[n] = y_{f2}[n]$? If so, why would you expect it to? If not, this means the result of processing a signal with the series connection of Systems 1 and 2 is not equal to the result of processing the signal with a single system whose impulse response is the convolution of the impulse response of System 1 with the impulse response of System 2. Does this violate the associative property of convolution as discussed in Part (d)?

```
yf1 = conv(w, hf2)
```

```
yf1 = 1x14
     1     1     4     7    11     9    18     4     5     0     0     0     0     0
```

```
impu = [1 0 0 0 0];
hf1 = (nh1+1).*impu;

hseries = conv(hf1, hf2);
yf2 = conv(x1, hseries)
```

```
yf2 = 1x18
     1     0     3     3     4     3     4     1     1     0     0     0     0     0     0     0     0     0
```

```
ans = isequal(yf1, yf2(1:14))
```

```
ans = logical
   0
```

yf1[n] != yf2[n]. It dosen't violate the associative property. Beacuse system 1 is not time invariant which is simple to find. So w != conv(x1, hf1).

(g). Consider the parallel connection of two systems; call them System 1 and System 2. System 1 is a memoryless system characterized by the input/output relationship $y[n] = x^2[n]$. System 2 is an LTI system with impulse response $h_{g2}[n] = h_2[n]$ as defined in Part (a). Suppose you were to use the steps below to investigate whether or not the distributive property of convolution held for the parallel connection of these systems:

- Let $y_{ga}[n]$ be the output of System 1 when the input is the signal $x_g[n] = 2\delta[n]$. Define **xg** to represent this input over the interval $0 \leq n \leq 4$, and use **xg** and the termwise exponentiation operator `.^` to define a MATLAB vector **yga** to represent $y_{ga}[n]$.
- Let $y_{gb}[n]$ be the output of System 2 when $x_g[n]$ is the input, and define **ygb** to represent this signal.
- Let $y_{g1}[n]$ be the sum of $y_{ga}[n]$ and $y_{gb}[n]$, the outputs of the parallel branches. Define the vector **yg1** to represent $y_{g1}[n]$. Note that because **yga** is shorter in length than **ygb**, you will have to extend **yga** with some zeros before you can add the vectors.
- Let $h_{g1}[n]$ be the output of System 1 when the input is $\delta[n]$. Define **hg1** to represent this signal on the interval $0 \leq n \leq 4$.
- Let $h_{\text{parallel}}[n]$ be $h_{g1}[n] + h_{g2}[n]$. Define **hparallel** to represent this signal.
- Let $y_{g2}[n]$ be the output of the LTI system with impulse response $h_{\text{parallel}}[n]$ when the input is $x_g[n]$. Define a vector **yg2** to represent this signal.

Are **yg1** and **yg2** equal? If so, why would you expect this? If not has the distributive property of convolution been violated ?

```
yg1
```

```
yg1 = 1×9
     4     4    10     8    -2     0     0     0     0
```

```
yg2
```

```
yg2 = 1×9
     2     4    10     8    -2     0     0     0     0
```

```
ans = isequal(yg1, yg2)
```

```
ans = logical
   0
```

yg1 and yg2 is not equal. This dosen't violate distributive property. Because system 1 is not LTI. Impluse of system1 can not be used to calculate output.

2.5

Basic Problems

Consider the following three systems:

$$\begin{aligned} &\text{System 1:} && w[n] = x[n] - x[n-1] - x[n-2]\,, \\ &\text{System 2:} && y[n] = \cos(x[n])\,, \\ &\text{System 3:} && z[n] = n\,x[n]\,, \end{aligned}$$

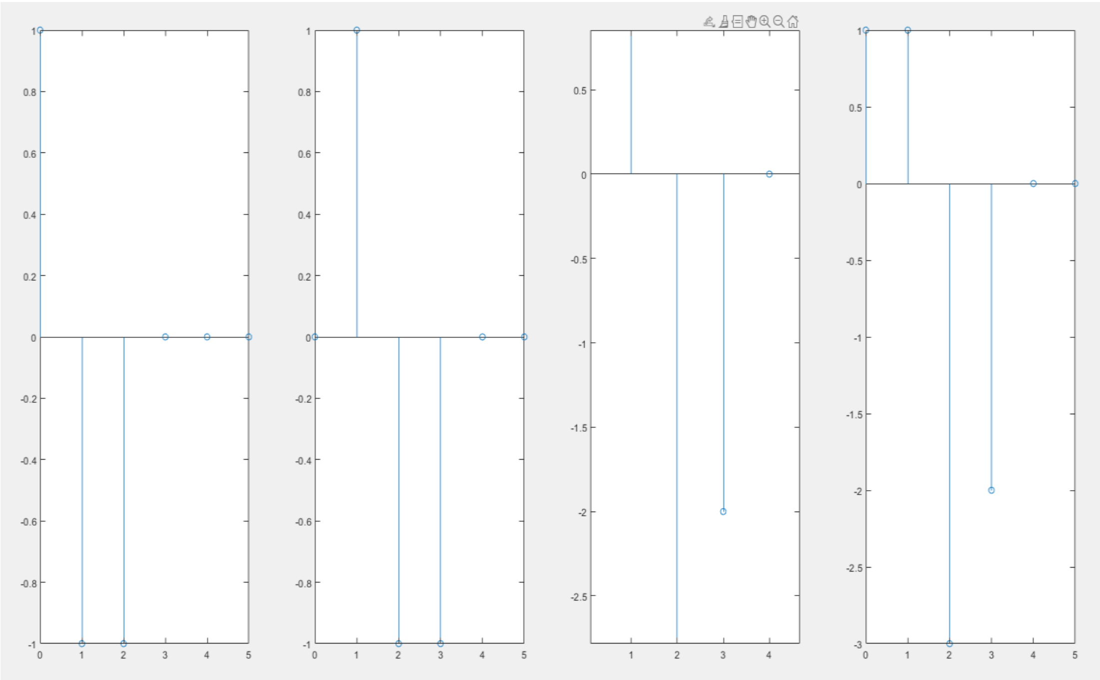
where $x[n]$ is the input to each system, and $w[n]$, $y[n]$, and $z[n]$ are the corresponding outputs.

(a). Consider the three inputs signals $x_1[n] = \delta[n]$, $x_2[n] = \delta[n-1]$, and $x_3[n] = (\delta[n] + 2\,\delta[n-1])$. For System 1, store in `w1`, `w2`, and `w3` the responses to the three inputs. The vectors `w1`, `w2`, and `w3` need to contain the values of $w[n]$ only on the interval $0 \le n \le 5$. Use `subplot` and `stem` to plot the four functions represented by `w1`, `w2`, `w3`, and `w1+2*w2` within a single figure. Make analogous plots for Systems 2 and 3.

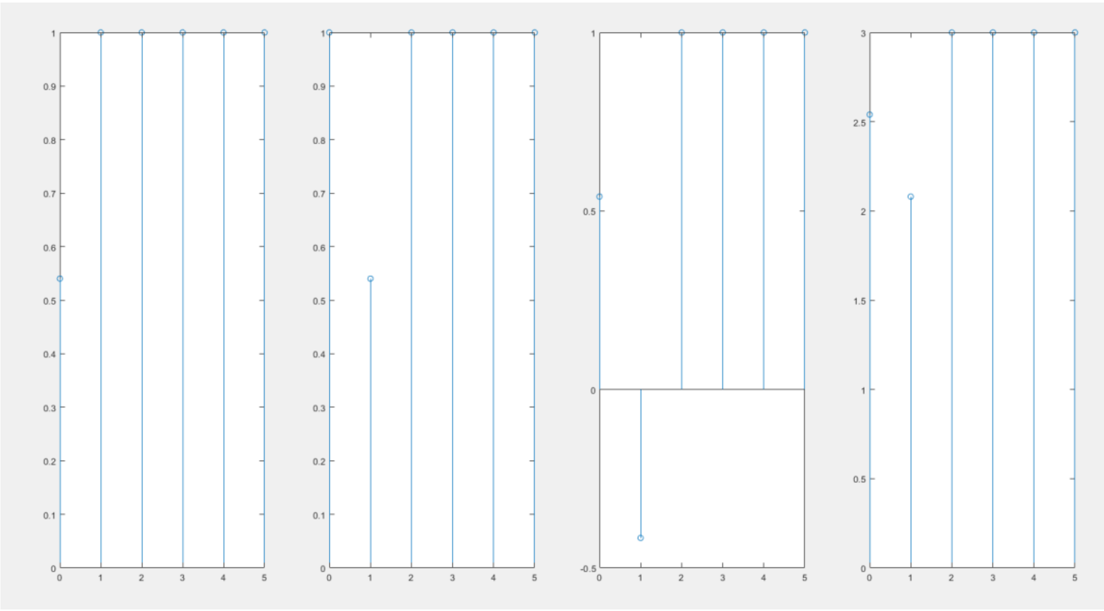
(b). State whether or not each system is linear. If it is linear, justify your answer. If it is not linear, use the signals plotted in Part (a) to supply a counter-example.

(c). State whether or not each system is time-invariant. If it is time-invariant, justify your answer. If it is not time-invariant, use the signals plotted in Part (a) to supply a counter-example.

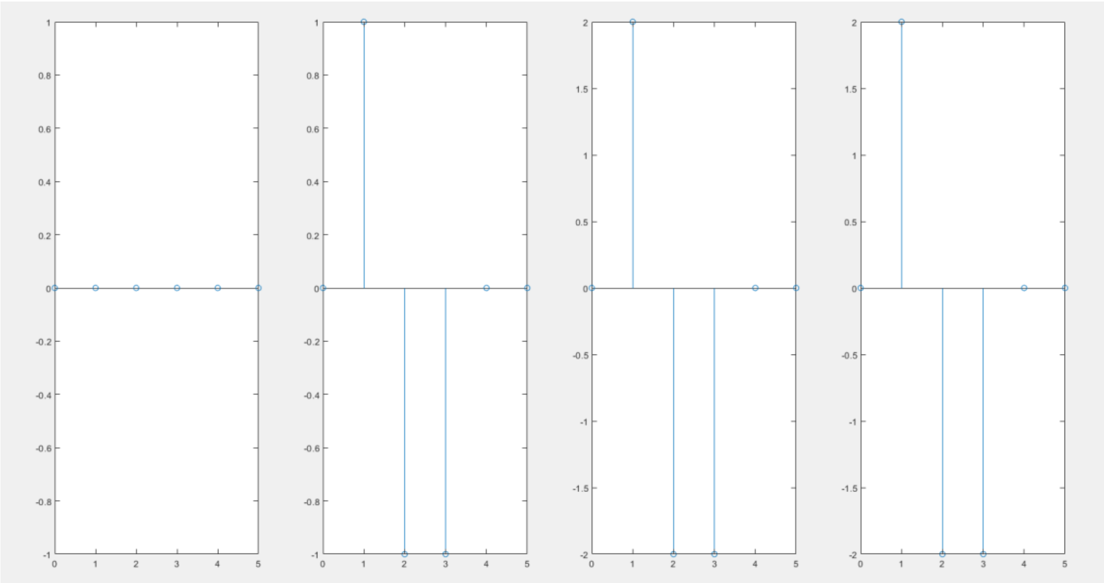
A(1):



A(2):



A(3):



b.

use the figure with the difference between w3(f.3)and w1+2*w2(f.4)in a could find that system 1 and system 3 is linear but system 2 is not linear. because f.4 in a(2) show that it is not linear.

c.

use the figure with the difference between w1(f.1)and w2(f.2)in a could find that system 1 and system 2 is linear but system 3 is not linear. The reason of that is when n=0 y3n will output zero instead of 1 that make it not time-invariant (d). Calculate $h_1[n]$ and $h_2[n]$ on the interval $0 \le n \le 19$, and store these responses in **h1** and **h2**. Plot each response using **stem**. Hint: The **filter** function can be used to calculate **h1**. However, System 2 is described by a difference equation with non-constant coefficients; therefore, you must either determine **h2** analytically or use a **for** loop rather than **filter** to calculate **h2**.

(e). For each system, calculate the unit step response on the interval $0 \le n \le 19$, and store the responses in **s1** and **s2**. Again, **filter** can be used only to calculate the step response of System 1. Use a **for** loop to calculate **s2**.

(f). Note that $h_1[n]$ and $h_2[n]$ are zero for $n \ge 20$ for all practical purposes. Thus **h1** and **h2** contain all we need to know about the response of each system to the unit impulse. Define $z_1[n] = h_1[n] * u[n]$ and $z_2[n] = h_2[n] * u[n]$, where $u[n]$ is the unit step function. Use **conv** to calculate $z_1[n]$ and $z_2[n]$ on the interval $0 \le n \le 19$, and store these calculations in the vectors **z1** and **z2**. You must first define a vector containing $u[n]$ over an appropriate interval, and then select the subset of the samples produced by **conv(h1,u)** and **conv(h2,u)** that represent the interval $0 \le n \le 19$. Since you have truncated two infinite-length signals, only a portion of the outputs of **conv** will contain valid sequence values. This issue was also discussed in Exercise 2.7 Part (c).

(g). Plot **s1** and **z1** on the same set of axes. If the two signals are identical, explain why you could have anticipated this similarity. Otherwise, explain any differences between the two signals. On a different set of axes, plot **s2** and **z2**. Again, explain how you might have anticipated any differences or similarities between these two signals.

D

```
clear;
clc;
n=[0:19];
x1=[1,zeros(1,19)];
a1=[1,-0.6];
b1=[1];
h1=filter(b1,a1,x1);
subplot(1,2,1);
stem(h1);
h2(1)=1;
for i=2:20
    h2(i)=0.6^i*h2(i-1);
end
subplot(1,2,2);
stem(h2);
```

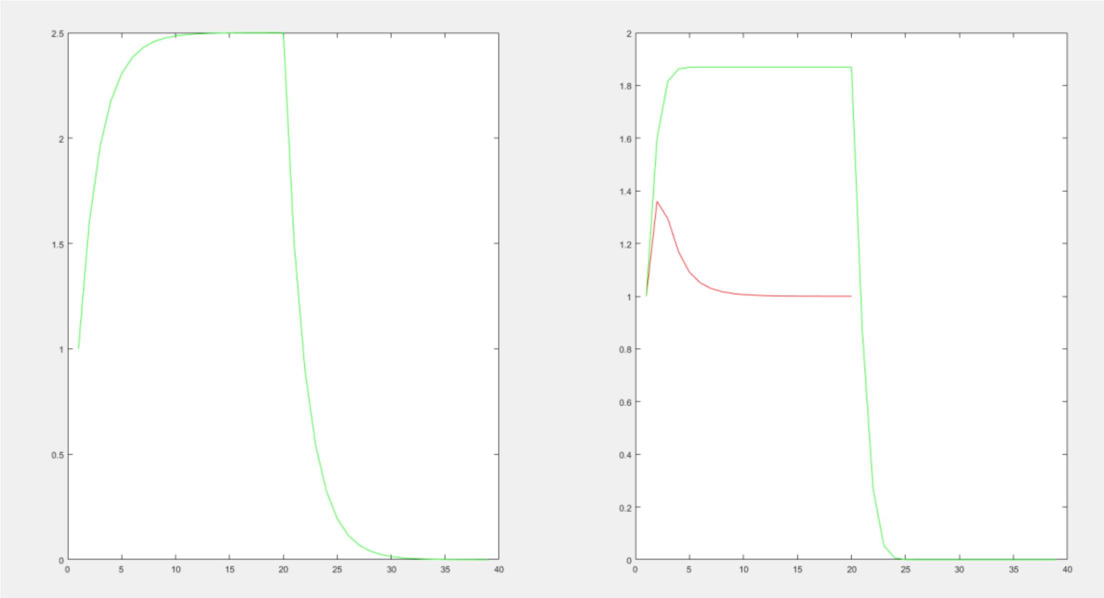
E

```
clear;
clc;
n=[0:19];
x1=[1,zeros(1,19)];
for i=1:20
    x1(i)=1;
end
a1=[1,-0.6];
b1=[1];
s1=filter(b1,a1,x1);
subplot(1,2,1);
stem(s1);
s2(1)=1;
for i=2:20
    s2(i)=0.6^i*s2(i-1)+x1(i);
end
subplot(1,2,2);
stem(s2);
```

F

```
n=[0:19];
x1=[1,zeros(1,19)];
un1=[1];
for i=1:20
    un1(i)=1;
end
a1=[1,-0.6];
b1=[1];
h1=filter(b1,a1,x1);
h2(1)=1;
for i=2:20
    h2(i)=0.6^(i-1)*h2(i-1);
end
z1=conv(h1,un1);
z2=conv(h2,un1);
subplot(1,2,1);
plot(s1,'r');
hold on;
plot(z1,'g');
subplot(1,2,2);
plot(s2,'r');
hold on;
plot(z2,'g')
```

G



In f.1(left s1 and z1) you could find that it is totally similar because of the exchange law of convolution that un*x1*y1=x1*y1*un .

Inf.2(right s2 and z2) the (0.6)^n break the similarity and make difference in the figure, which also made by for loop.

2.10

■ 2.10 Echo Cancellation via Inverse Filtering

In this exercise, you will consider the problem of removing an echo from a recording of a speech signal. This project will use the audio capabilities of MATLAB to play recordings of both the original speech and the result of your processing. To begin this exercise you will need to load the speech file `lineup.mat`, which is contained in the Computer Explorations Toolbox. The Computer Explorations Toolbox can be obtained from The MathWorks at the address provided in the Preface. If this speech file is already somewhere in your MATLABPATH, then you can load the data into MATLAB by typing

```
>> load lineup.mat
```

You can check your MATLABPATH, which is a list of all the directories which are currently accessible by MATLAB, by typing `path`. The file `lineup.mat` must be in one of these directories.

Once you have loaded the data into MATLAB, the speech waveform will be stored in the variable `y`. Since the speech was recorded with a sampling rate of 8192 Hz, you can hear the speech by typing

```
>> sound(y,8192)
```

You should hear the phrase "line up" with an echo. The signal $y[n]$, represented by the vector `y`, is of the form

$$y[n] = x[n] + \alpha x[n-N], \tag{2.21}$$

where $x[n]$ is the uncorrupted speech signal, which has been delayed by N samples and added back in with its amplitude decreased by $\alpha < 1$. This is a reasonable model for an echo resulting from the signal reflecting off of an absorbing surface like a wall. If a microphone is placed in the center of a room, and a person is speaking at one end of the room, the recording will contain the speech which travels directly to the microphone, as well as an echo which traveled across the room, reflected off of the far wall, and then into the microphone. Since the echo must travel further, it will be delayed in time. Also, since the speech is partially absorbed by the wall, it will be decreased in amplitude. For simplicity ignore any further reflections or other sources of echo.

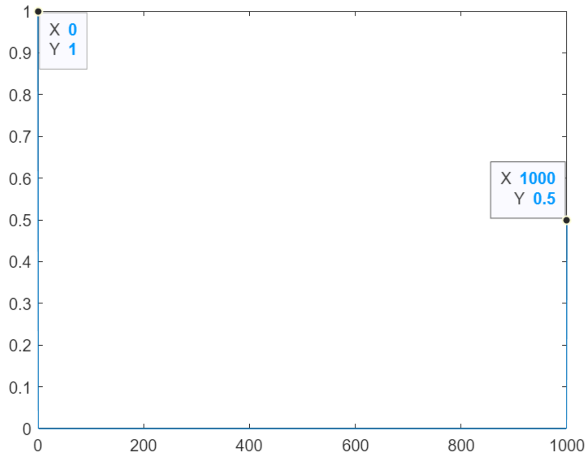
For the problems in this exercise, you will use the value of the echo time, $N = 1000$, and the echo amplitude, $\alpha = 0.5$.

Basic Problems

(a). In this exercise you will remove the echo by linear filtering. Since the echo can be represented by a linear system of the form Eq. (2.21), determine and plot the impulse

response of the echo system Eq. (2.21). Store this impulse response in the vector `he` for $0 \leq n \leq 1000$.

这一点可以直接看出：



(b). Consider an echo removal system described by the difference equation he=delta(n)+0.5delta(n-1000)

$$z[n] + \alpha z[n-N] = y[n], \qquad (2.22)$$

where $y[n]$ is the input and $z[n]$ is the output which has the echo removed. Show that Eq. (2.22) is indeed an inverse of Eq. (2.21) by deriving the overall difference equation relating $z[n]$ to $x[n]$. Is $z[n] = x[n]$ a valid solution to the overall difference equation?

看z(n)*he(n)?=y(n)

z[n] + 0.5z[n-1000] = y[n]

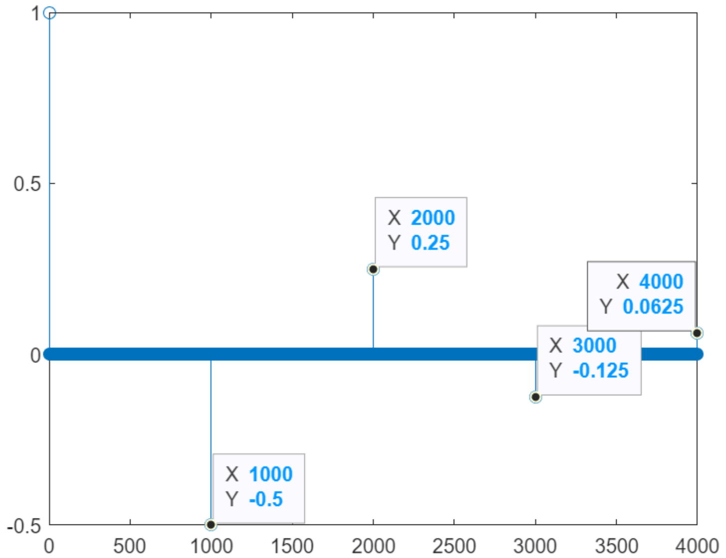
y[n] = x[n] + 0.5x[n-1000]

z[n] + 0.5z[n-100] = x[n] + 0.5x[n-1000]

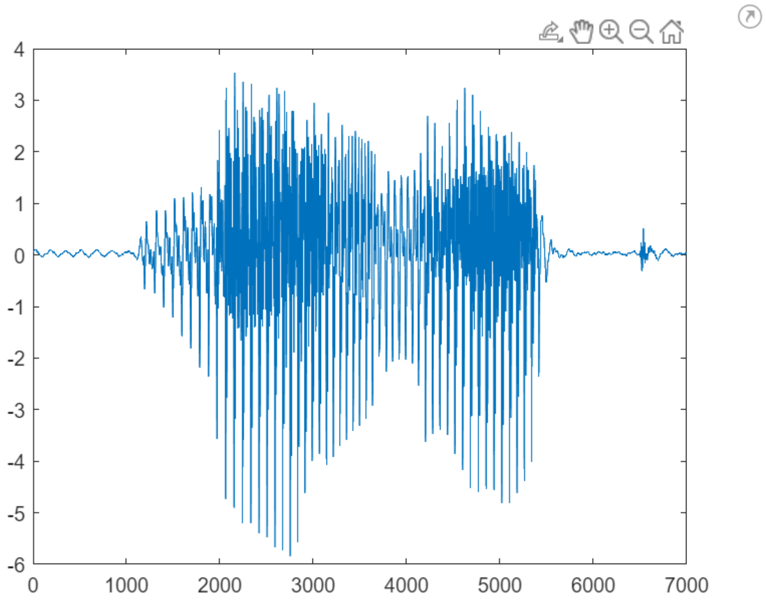
z[n] should be x[n]

(c). The echo removal system Eq. (2.22) will have an infinite-length impulse response. Assuming that $N = 1000$, and $\alpha = 0.5$, compute the impulse response using **filter** with an input that is an impulse given by **d=[1 zeros(1,4000)]**. Store this 4001 sample approximation to the impulse response in the vector **her**. 看IIR冲击响应=her

$$z[n] + \alpha z[n-N] = y[n], \qquad (2.22)$$

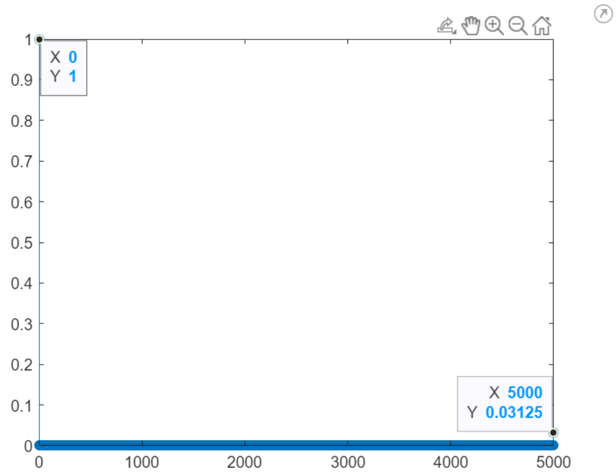


(d). Implement the echo removal system using **z=filter(1,a,y)**, where **a** is the appropriate coefficient vector derived from Eq. (2.22). Plot the output using **plot**. Also, listen to the output using **sound**. You should no longer hear the echo. 利用Y，直接计算X



(e). Calculate the overall impulse response of the cascaded echo system, Eq. (2.21), and echo removal system, Eq. (2.22), by convolving **he** with **her** and store the result in **hoa**. Plot the overall impulse response. You should notice that the result is not a unit impulse. Given that you have computed **her** to be the inverse of **he**, why is this the case?

he和her卷积hoa，观察hoa



The result is not a unit impluse. Output at n=5000 is 0.03125. Because the range of unit impluse input is not infinite. It is just an analog.

The output is equal to input. The cascade system is identical system. So the impluse response should be unit impluse. delta = conv(he, her).

So, he is inverse of her.

Advanced Problem

(f). Suppose that you were given $y[n]$ but did not know the value of the echo time, N, or the amplitude of the echo, α. Based on Eq. (2.21), can you determine a method of estimating these values? Hint: Consider the output **y** of the echo system to be of the form:

$$y[n] = x[n] * (\delta[n] + \alpha\delta[n-N])$$

and consider the signal,

$$R_{yy}[n] = y[n] * y[-n].$$

This is called the autocorrelation of the signal $y[n]$ and is often used in applications of echo-time estimation. Write $R_{yy}[n]$ in terms of $R_{xx}[n]$ and also plot $R_{yy}[n]$. You will have to truncate $y[n]$ before your calculations to keep $R_{yy}[n]$ within the limitations of the Student Edition of MATLAB. You will find that many of the properties of the autocorrelation will still hold when **y** is truncated. Also try experimenting with simple echo problems such as

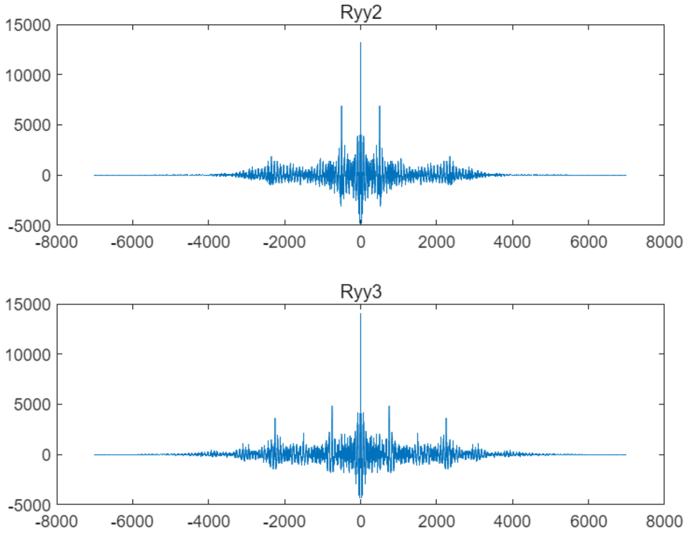
计算Y的自相关

```
>> NX=100;
```



```
>> x=randn(1,NX);
>> N=50;
>> alpha=0.9;
>> y=filter([1 zeros(1,N) alpha],1,x);
>> Ryy=conv(y,fliplr(y));
>> plot([-NX+1:NX-1],Ryy)
```

by varying `N,alpha`, and `NX`. Also, when you loaded `lineup.mat`, you loaded in two additional vectors. The vector `y2` contains the phrase "line up" with a different echo time N and different echo amplitude α. The vector `y3` contains the same phrase with two echoes, each with different times and amplitudes. Can you estimate N and α for `y2`, and N_1, α_1, N_2, and α_2 for `y3`?




```
find(Ryy2==max(Ryy2(7010:end)))
```

```
ans = 1×2
        6499        7501
```

N for y2 is 501

```
find(Ryy3==max(Ryy3(7010:end)))
```

```
ans = 1×2
        6249        7751
```

```
find(Ryy3==max(Ryy3(7761:end)))
```

```
ans = 1×2
        4748        9252
```

N1 N2 for y3 is 751 2252

$$R_{yy}[n] = x[n] * (\delta[n] + \alpha\delta[n-N]) * x[-n] * (\delta[n] + \alpha\delta[n+N])$$

$$= R_{xx} * ((1+\alpha^2)\delta[n] + \alpha\delta[n-N] + \alpha\delta[n+N])$$

for y2

$$R_{yy}[0] = (1+\alpha^2)R_{xx}[0] + \alpha R_{xx}[-501] + \alpha R_{xx}[501]$$

$$R_{yy}[0] = 13163$$

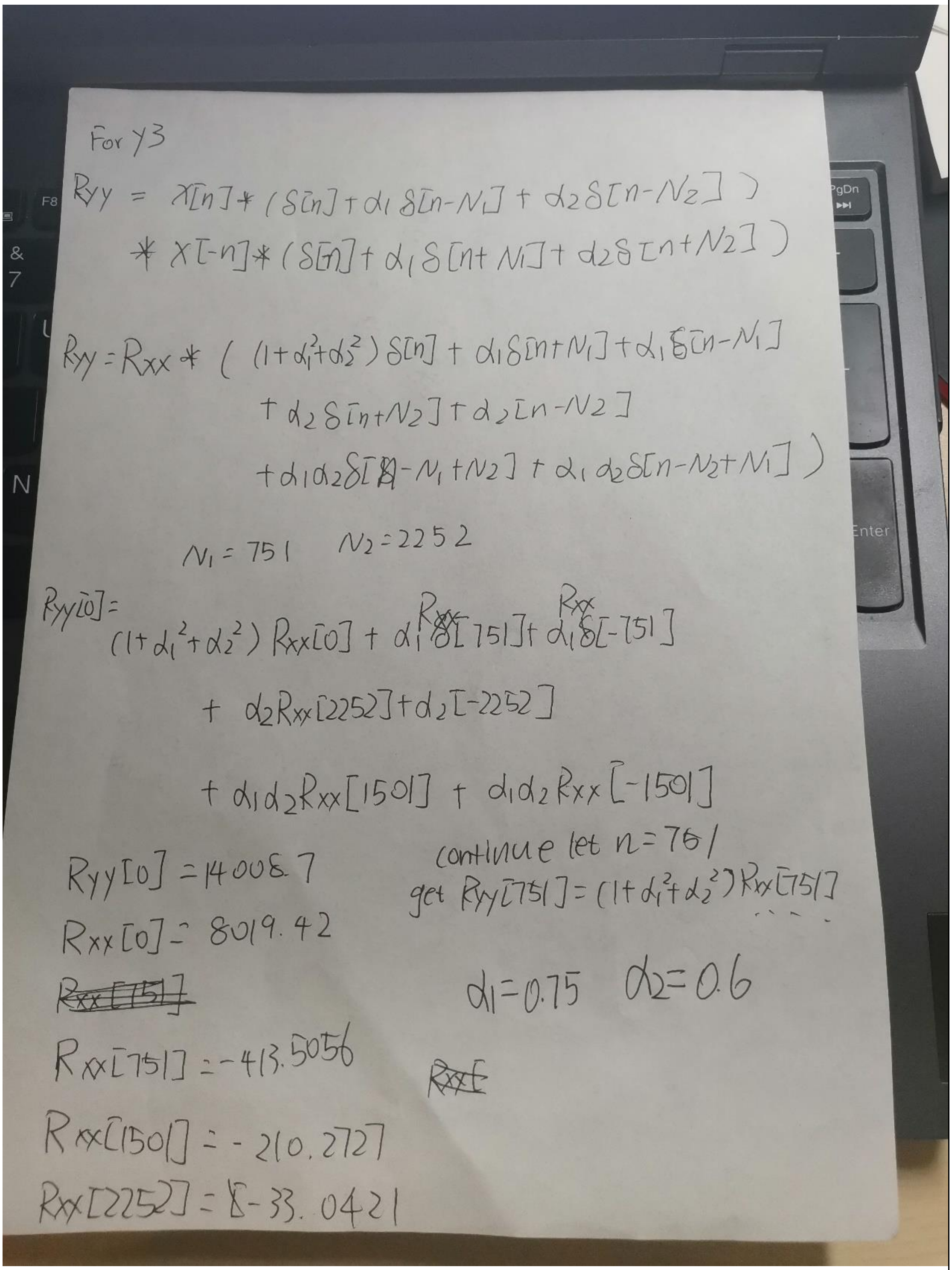
$$R_{xx}[0] = 8019.42$$

$$R_{xx}[-501] = R_{xx}[501] = \cancel{0.1173}\ 868.3714$$

$$13163 = (1+\alpha^2)\,8019.42\ \cancel{+2\alpha\ 1136.7\alpha}$$

$$\alpha = \cancel{\sqrt{\frac{13163}{8019.42} - 1}}\quad 0.8009$$

$$\cancel{\approx 0.8009}$$



For y3

$R_{yy} = x[n] * (\delta[n] + \alpha_1\delta[n-N_1] + \alpha_2\delta[n-N_2])$

$* x[-n] * (\delta[n] + \alpha_1\delta[n+N_1] + \alpha_2\delta[n+N_2])$

$R_{yy} = R_{xx} * ((1+\alpha_1^2+\alpha_2^2)\delta[n] + \alpha_1\delta[n+N_1] + \alpha_1\delta[n-N_1]$

$+ \alpha_2\delta[n+N_2] + \alpha_2[n-N_2]$

$+ \alpha_1\alpha_2\delta[\cancel{n}-N_1+N_2] + \alpha_1\alpha_2\delta[n-N_2+N_1])$

$N_1 = 751 \quad N_2 = 2252$

$R_{yy}[0] =$

$(1+\alpha_1^2+\alpha_2^2) R_{xx}[0] + \alpha_1 \overset{R_{xx}}{\cancel{\delta}}[751] + \alpha_1 \overset{R_{xx}}{\delta}[-751]$

$+ \alpha_2 R_{xx}[2252] + \alpha_2[-2252]$

$+ \alpha_1\alpha_2 R_{xx}[1501] + \alpha_1\alpha_2 R_{xx}[-1501]$

$R_{yy}[0] = 14008.7$

$R_{xx}[0] = 8019.42$

~~$R_{xx}[751]$~~

$R_{xx}[751] = -413.5056$

$R_{xx}[1501] = -210.2727$

$R_{xx}[2252] = \cancel{8}-33.0421$

continue let $n = 751$

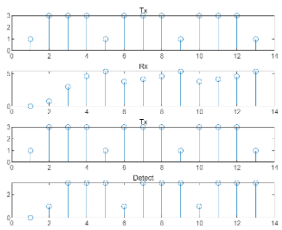
get $R_{yy}[751] = (1+\alpha_1^2+\alpha_2^2)R_{xx}[751]$ ....

$\alpha_1 = 0.75 \quad \alpha_2 = 0.6$

~~$R_{xx}[$~~

Experience

```
x = [1 3 3 3 1 3 3 3 1 3 3 3 1];
A1 = 1;
B1 = [0 0.8:-0.2:0.4];
y = filter(B1, A1, x);
subplot(4,1,1), stem(x), title('Tx');
subplot(4,1,2), stem(y), title('Rx');
A2 = [0.8 0.6 0.4];
B2 = 1;
z = filter(B2, A2, y);
subplot(4,1,3), stem(x), title('Tx');
subplot(4,1,4), stem(z), title('Detect');
```



1. Learn to use conv to analyze echo system and find its inverse.
2. Calculate and combine conv to check the properties of conv.

3. function as conv and filter is easy to use but sometimes, they do not work well, you need to pay attention to signal and use code to solve the question.

4. linear and time invariant could be easy find with figure .

Score	2.4 10/10 2.5 9/10 2.10 7/10

Code for 2.4

(a)

```
clear; clc;
x1 = [1 1 1 1 1 0 0 0 0 0];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];
nx1 = 0:9;
nh1 = 0:4;
f24 = figure;
figure(f24);

subplot(3,1,1);
stem(nx1, x1);
ylabel('x_1[n]');

subplot(3,1,2);
stem(nh1, h1);
ylabel('h_1[n]');

subplot(3,1,3);
stem(nh1, h2);
ylabel('h_2[n]');
```

(b)

```
clear; clc;
x1 = [1 1 1 1 1 0 0 0 0 0];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];
nx1 = 0:9;
nh1 = 0:4;

y1 = conv(x1, h1);
y2 = conv(h1, x1);
ans = isequal(y1, y2)
```

(c)

```
clear; clc;
x1 = [1 1 1 1 1 0 0 0 0 0];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];
nx1 = 0:9;
nh1 = 0:4;

y1 = conv(x1, h1) + conv(x1, h2);
y2 = conv(x1, (h1+h2));
ans = isequal(y1, y2)
```

(d)

```
clc; clear;
x1 = [1 1 1 1 1 0 0 0 0 0];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];
nx1 = 0:9;
nh1 = 0:4;

w = conv(x1, h1);
yd1 = conv(w, h2);

h_series = conv(h1, h2);
yd2 = conv(x1, h_series);

ans = isequal(yd1, yd2)
```

(e)

```
clc; clear;
x1 = [1 1 1 1 1 0 0 0 0 0];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];
nx1 = 0:9;
nh1 = 0:4;

he1 = [h1 0 0];
he2 = [0 0 h1];

ye1 = conv(x1, he1)
ye2 = conv(x1, he2)
ans = isequal(ye1(1:end-2), ye2(3:end))
```

(f)

```
clc; clear;
x1 = [1 1 1 1 1 0 0 0 0 0];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];
nx1 = 0:9;
nh1 = 0:4;
hf2 = h1;

w = (nx1+1).*x1;
yf1 = conv(w, hf2)

impu = [1 0 0 0 0];
hf1 = (nh1+1).*impu;

hseries = conv(hf1, hf2);
yf2 = conv(x1, hseries)

ans = isequal(yf1, yf2(1:14))
```

(g)

```
clc; clear;
x1 = [1 1 1 1 1 0 0 0 0 0];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];
nx1 = 0:9;
nh1 = 0:4;
hf2 = h1;
xg = [2 0 0 0 0];
yga = xg.^2;
ygb = conv(h2,xg);
nygb = 0:8;
yga = [yga 0 0 0 0];
yg1 = yga+ygb;
d = [1 0 0 0 0];
hg1 = d.^2;
hparallel = hg1+h2;
yg2 = conv(hparallel, xg);

yg1
yg2
ans = isequal(yg1, yg2)
```

Code in 2.5

```
clear;
clc;
x1n=[1 ,zeros(1,5)];
x2n=[0 1 ,zeros(1,4)];
x3n=[1 2 ,zeros(1,4)];
n=[0:5];
ss1=[1];
ss2=[1 -1 -1];
w1=filter(ss2,ss1,x1n);
w2=filter(ss2,ss1,x2n);
w3=filter(ss2,ss1,x3n);
figure
title('system 1')
subplot(1,4,1);
stem(n,w1);
subplot(1,4,2);
stem(n,w2);
subplot(1,4,3);
stem(n,w3);
subplot(1,4,4);
```

```
stem(n,w1+2*w2);

clear;
clc;
x1n=[cos(1) 1 1 1 1 1];
x2n=[1 cos(1) 1 1 1 1];
x3n=[cos(1) cos(2) 1 1 1 1];
n=[0:5];
ss1=[1];
ss2=[1];
w1=filter(ss2,ss1,x1n);
w2=filter(ss2,ss1,x2n);
w3=filter(ss2,ss1,x3n);
figure
title('system 1')
subplot(1,4,1);
stem(n,w1);
subplot(1,4,2);
stem(n,w2);
subplot(1,4,3);
stem(n,w3);
subplot(1,4,4);
stem(n,w1+2*w2);

clear;
clc;
x1n=[0 ,zeros(1,5)];
x2n=[0 1 ,zeros(1,4)];
x3n=[0 2 ,zeros(1,4)];
n=[0:5];
ss1=[1];
ss2=[1 -1 -1];
w1=filter(ss2,ss1,x1n);
w2=filter(ss2,ss1,x2n);
w3=filter(ss2,ss1,x3n);
figure
title('system 1')
subplot(1,4,1);
stem(n,w1);
subplot(1,4,2);
stem(n,w2);
subplot(1,4,3);
stem(n,w3);
subplot(1,4,4);
```

```
stem(n,w1+2*w2);

clear;
clc;
n=[0:19];
x1=[1,zeros(1,19)];
a1=[1,-0.6];
b1=[1];
h1=filter(b1,a1,x1);
subplot(1,2,1);
stem(h1);
h2(1)=1;
for i=2:20
    h2(i)=0.6^i*h2(i-1);
end
subplot(1,2,2);
stem(h2);

clear;
clc;
n=[0:19];
x1=[1,zeros(1,19)];
for i=1:20
    x1(i)=1;
end
a1=[1,-0.6];
b1=[1];
s1=filter(b1,a1,x1);
subplot(1,2,1);
stem(s1);
s2(1)=1;
for i=2:20
    s2(i)=0.6^i*s2(i-1)+x1(i);
end
subplot(1,2,2);
stem(s2);
n=[0:19];
x1=[1,zeros(1,19)];
un1=[1];
for i=1:20
    un1(i)=1;
end
a1=[1,-0.6];
b1=[1];
```

```
h1=filter(b1,a1,x1);
h2(1)=1;
for i=2:20
    h2(i)=0.6^(i-1)*h2(i-1);
end
z1=conv(h1,un1);
z2=conv(h2,un1);
subplot(1,2,1);
plot(s1,'r');
hold on;
plot(z1,'g');
subplot(1,2,2);
plot(s2,'r');
hold on;
plot(z2,'g')
```

Code for 2.10

(a)

```
clear; clc;
y = load("lineup.mat");
sound(y.y, 8192)
impu = zeros(1, 1001);
impu(1) = 1;
A = [1];
B = zeros(1, 1001);
B(1) = 1; B(1001) = 0.5;
he = filter(B, A, impu);
f210a = figure;
figure(f210a);
plot(0:1000, he);

ax = gca;
chart = ax.Children(1);
datatip(chart,0,1);
datatip(chart,1000,0.5);
```

(b)

No need

(c)

```
clear; clc;
d = [1 zeros(1, 4000)];
A = [1 zeros(1, 999) 0.5];
B = [1];
her = filter(B, A, d);
f210c = figure;
figure(f210c);
stem(0:4000, her);

ax2 = gca;
chart2 = ax2.Children(1);
datatip(chart2,3000,-0.125);
datatip(chart2,2000,0.25);
datatip(chart2,1000,-0.5);
datatip(chart2,4000,0.0625);
```

(d)

```
clear; clc;
y = load("lineup.mat");
A = [1 zeros(1, 999) 0.5];
z = filter(1, A, y.y);
sound(z, 8192);
f210d = figure;
figure(f210d);
plot(1:length(z), z);
```

(e)

```
clear; clc;
y = load("lineup.mat");
sound(y.y, 8192)
impu = zeros(1, 1001);
impu(1) = 1;
A = [1];
B = zeros(1, 1001);
B(1) = 1; B(1001) = 0.5;
he = filter(B, A, impu);
nhe = 0:1000;

d = [1 zeros(1, 4000)];
A = [1 zeros(1, 999) 0.5];
B = [1];
her = filter(B, A, d);
nher = 0:4000;

nw = nher(1)+nhe(1):nhe(end)+nher(er
hoa = conv(he, her);
f210e = figure;
figure(f210e);
stem(nw, hoa);
ax3 = gca;
chart3 = ax3.Children(1);
datatip(chart3,5000,0.03125);
datatip(chart3,0,1);
```
(f)

```
clear; clc;
y = load("lineup.mat");
NX = 7000;

A = [1 zeros(1, 999) 0.5];
z = filter(1, A, y.y);

y2 = y.y2;
Ryy2 = conv(y2', fliplr(y2'));
f210f = figure;
figure(f210f);
subplot(2,1,1);
plot(-NX+1:NX-1, Ryy2);
title('Ryy2')
y3 = y.y3;
Ryy3 = conv(y3', fliplr(y3'));
subplot(2,1,2);
plot(-NX+1:NX-1, Ryy3);
title('Ryy3')
find(Ryy2==max(Ryy2(7010:end)))
```

N for y2 is 501

```
find(Ryy3==max(Ryy3(7010:end)))
find(Ryy3==max(Ryy3(7761:end)))
```

N1 N2 for y3 is 751 2252

```
f210f2 = figure;
figure(f210f2);
Rzz = conv(z', fliplr(z'));
plot(-NX+1:NX-1, Rzz);
title('Rzz')
```