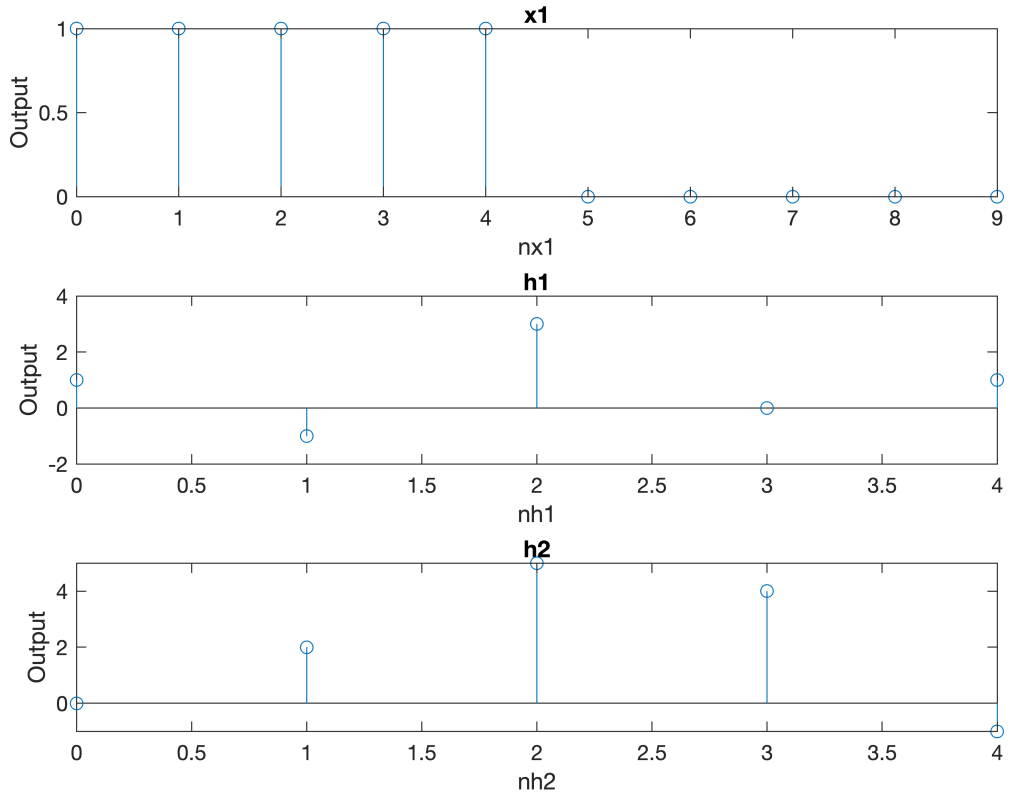
Report of Signals and Systems Lab Assignment 2

Written by HUANG Guanchao, SID 11912309 and GONG Xinrui, SID 11911233.

2.4

Basic Problem a

The plots of $x_1[n]$, $h_1[n]$ and $h_2[n]$ are shown below.



The MATLAB script is shown below.

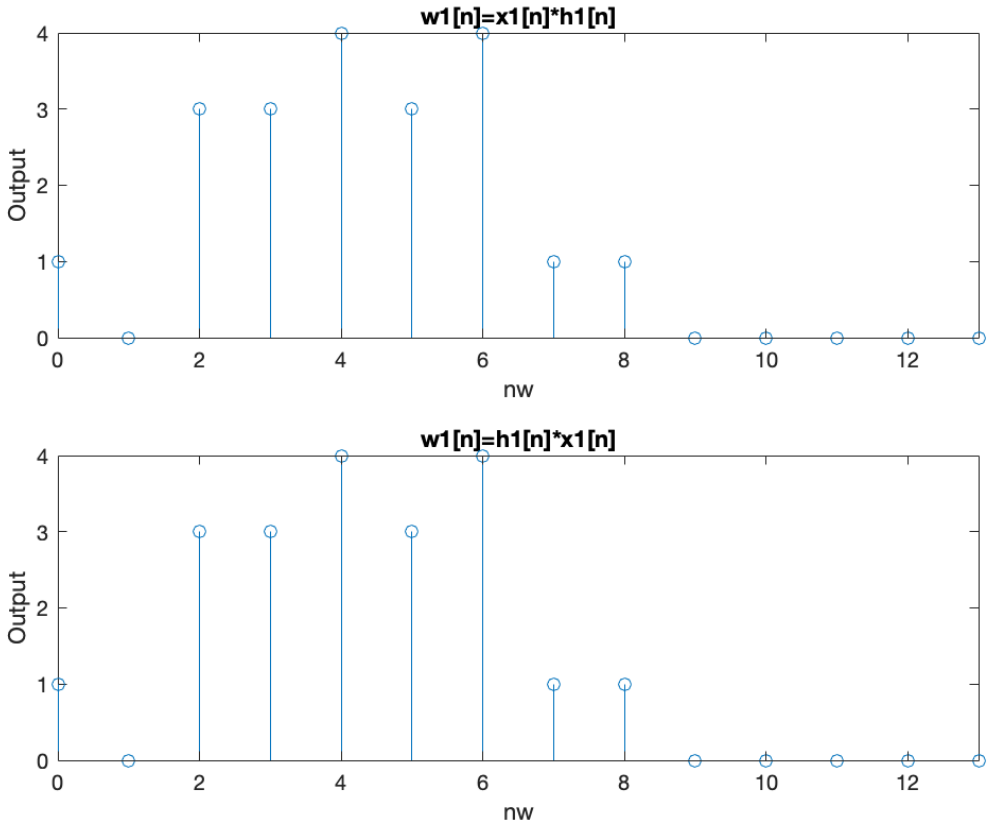
```
nx1 = 0:9;
nh1 = 0:4;
x1 = [ones(1, 5) zeros(1, 5)];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];

subplot(3, 1, 1)
stem(nx1, x1)
xlabel('nx1')
ylabel('Output')
title('x1')

subplot(3, 1, 2)
stem(nh1, h1)
xlabel('nh1')
ylabel('Output')
title('h1')
subplot(3, 1, 3)

stem(nh1, h2)
xlabel('nh2')
ylabel('Output')
title('h2')
```

Basic Problem b

The outputs are the same.

We first find convolved x1 with h1, which is $x_1[n] * h1[n]$, then we change $x_1[n]$ into the impulse response and $h_1[n]$ into the input, and make convolution again, which is $x_1[n] * h_1[n]$.

The plots for them are shown below, and we can easily figure out that $x_1[n] * h_1[n]$ equals to &h_1[n] * x_1[n]$.



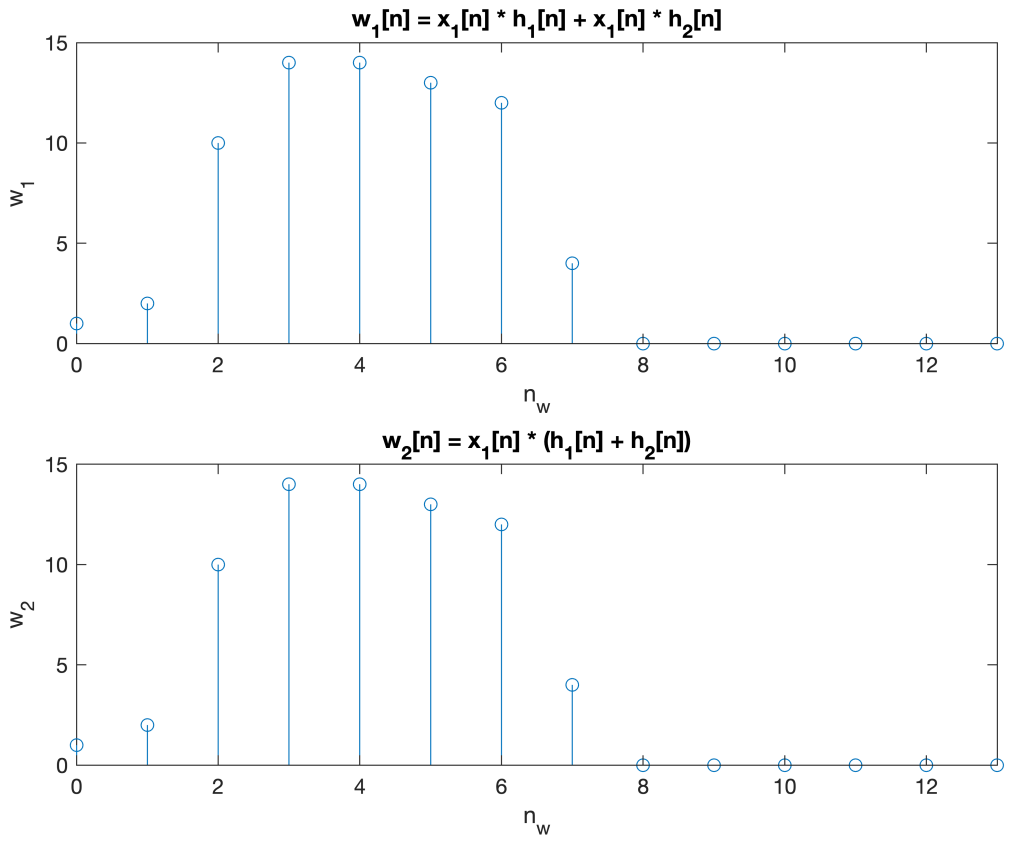
The MATLAB script is shown above.

```
nx1 = 0:9;
nh1 = 0:4;
x1 = [ones(1, 5) zeros(1, 5)];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];

nw = 0:13;
w1 = conv(x1, h1);
w2 = conv(h1, x1);

subplot(2, 1, 1)
stem(nw, w1)
xlabel('nw')
ylabel('Output')
title('w1[n]=x1[n]*h1[n]')
subplot(2, 1, 2)
stem(nw, w1)
xlabel('nw')
ylabel('Output')
title('w1[n]=h1[n]*x1[n]')
```

Basic Problem c

The two methods give the same results.

We first convolved `x1[n]*h1[n]` and `x1[n]*h2[n]` separately, and then add them together and find out the plot. After that, we try to add `h1[n]` and `h2[n]` first, and convolved `x1[n]*(h1[n]+h2[n])`.



The two plots are shown above, and we can see that they are exactly the same, which means they have the same results.

The MATLAB script is shown below.

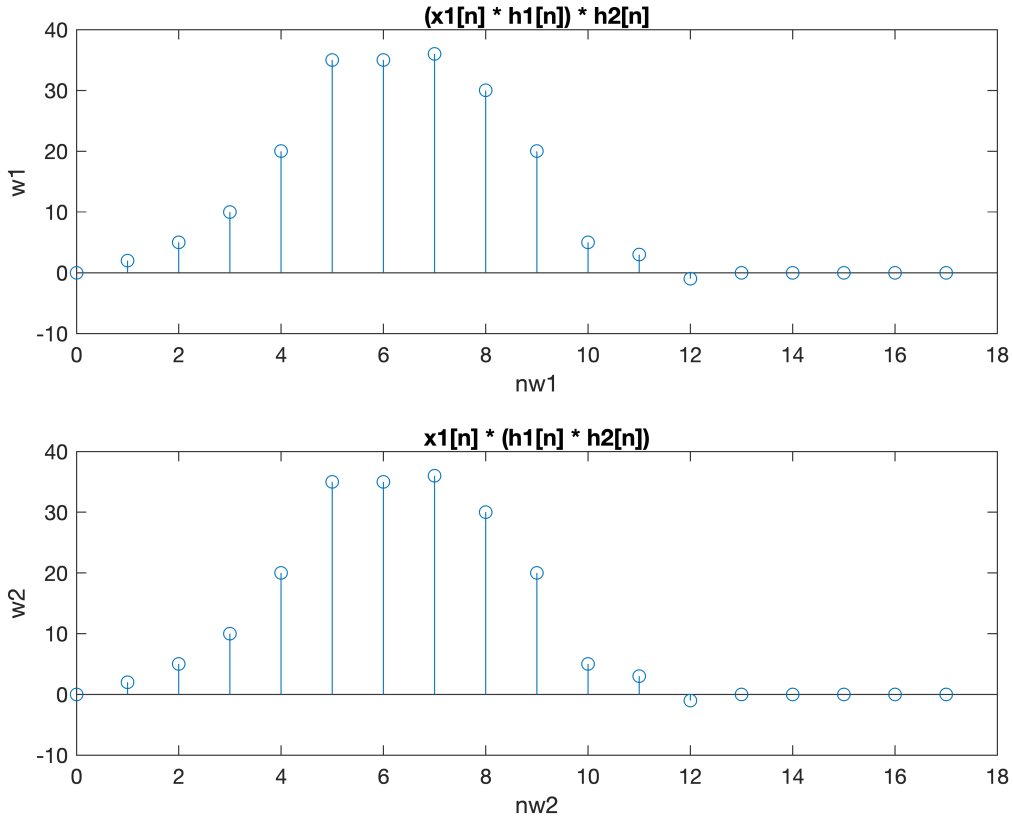
```
nx1 = 0:9;
nh1 = 0:4;
x1 = [ones(1, 5) zeros(1, 5)];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];
h3 = h1 + h2;

nw = 0:13;
w1 = conv(x1, h1) + conv(x1, h2);
w2 = conv(x1, h3);

subplot(2, 1, 1)
stem(nw, w1)
xlabel('n_w')
ylabel('w_1')
title('w_1[n] = x_1[n] * h_1[n] + x_1[n] * h_2[n]')
subplot(2, 1, 2)
stem(nw, w2)
```

```
axis([0, 13, ylim])
xlabel('n_w')
ylabel('w_2')
title('w_2[n] = x_1[n] * (h_1[n] + h_2[n])')
```

Basic Problem d

Compairing $y_{d1}[n]$ and $y_{d2}[n]$,we have the same results.

First we calculate $x_1[n] * h_1[n]$ and then we use this result to convolved with $h_2[n]$.Next we calculate the $h_1[n] * h_2[n]$ firstly and then use the output as the impulse response to convolved with $x_1[n]$.



The two plots are shown above, it can be apparently concluded that the output are the same.

The MATLAB script is shown below.

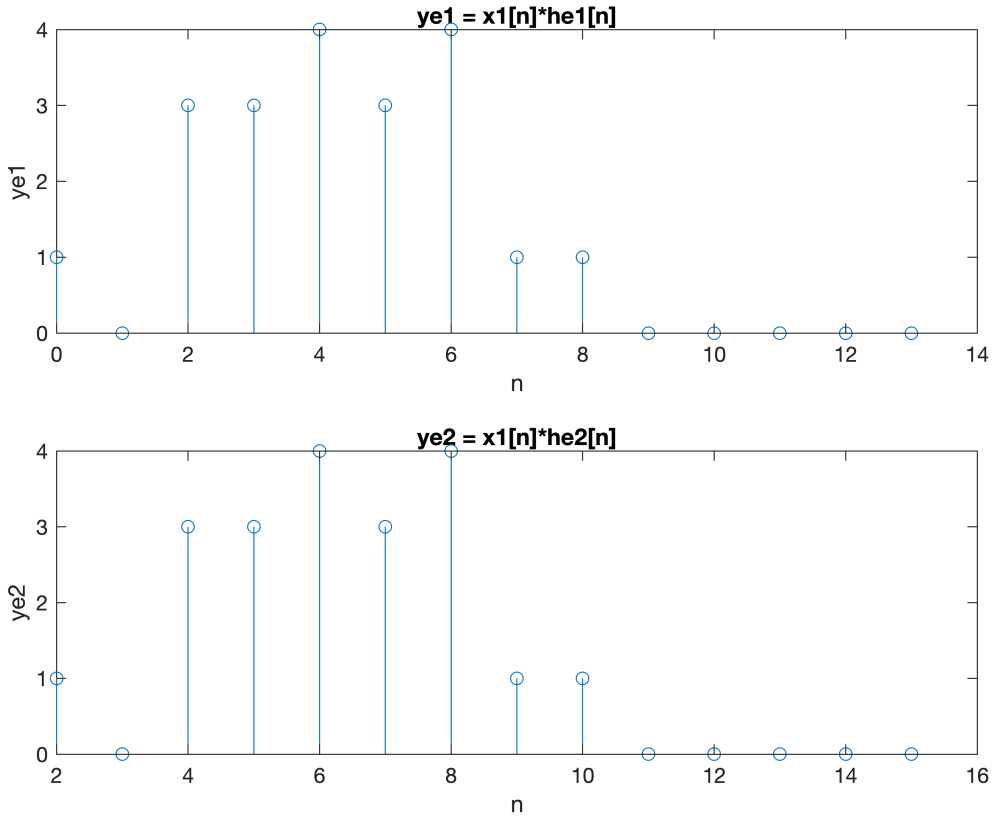
```
nx1 = 0:9;
nh1 = 0:4;
x1 = [ones(1, 5) zeros(1, 5)];
h1 = [1 -1 3 0 1];
h2 = [0 2 5 4 -1];

nw = 0:13;

h3 = conv(x1, h1);
h4 = conv(h1, h2);

w1 = conv(h3, h2);
nw1 = 0:17;
w2 = conv(x1, h4);
```

```
nw2 = 0:17;

subplot(2, 1, 1)
stem(nw1, w1)
xlabel('nw1')
ylabel('w1')
title('(x1[n] * h1[n]) * h2[n]')
subplot(2, 1, 2)
stem(nw2, w2)
xlabel('nw2')
ylabel('w2')
title('x1[n] * (h1[n] * h2[n])')
```

Intermediate Problem e

For this question, we convolved $x_1[n] * h_{e1}[n]$ and $x_2[n] * h_e 2[n]$, where $h_{e1}[n]$ and $h_{e2}[n]$ have the relationship $h_{e2}[n] = hg_{e1}[n - n_0]$.

The two plots are shown below.



We can easily conclude from the plots that $y_{e2}[n] = y_{e1}[n - n_0]$.

The MATLAB script is shown below.

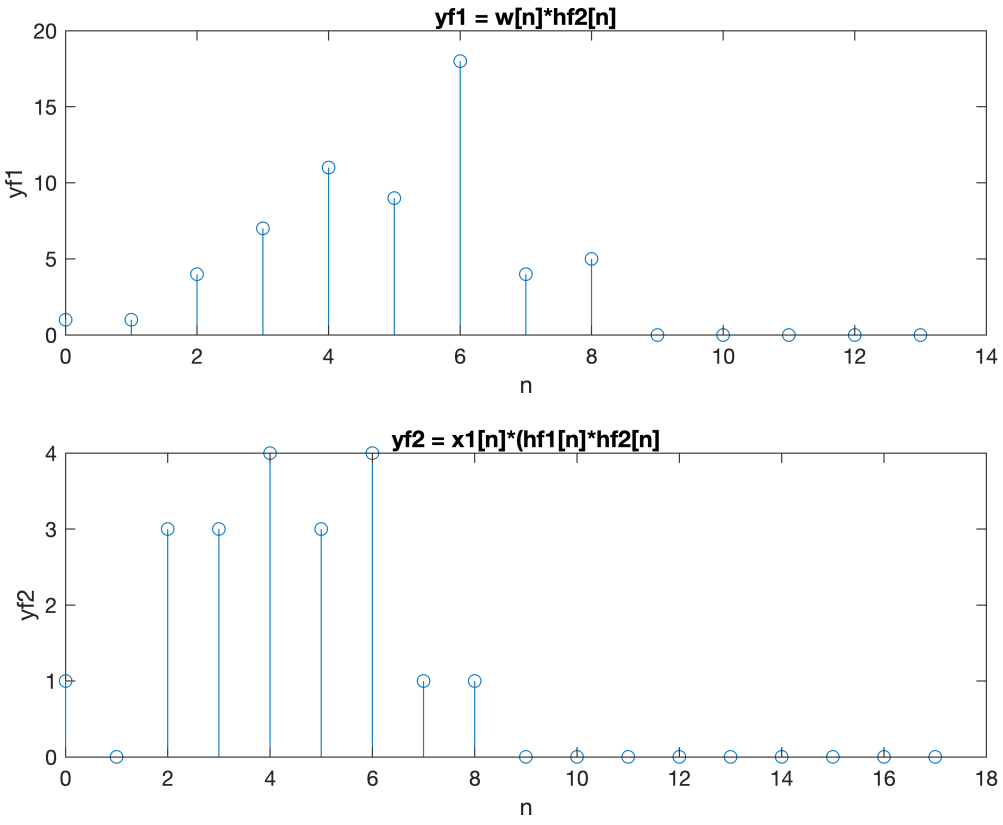
```
nx1 = 0:9;
nh1 = 0:4;
nh2 = 2:6;
x1 = [ones(1,5) zeros(1,5)];
h1 = [1 -1 3 0 1];
h2 = h1;
```

```
ny1 = nx1(1)+nh1(1):nx1(end)+nh1(end);
ny2 = nx1(1)+nh2(1):nx1(end)+nh2(end);
y1 = conv(x1,h1);
y2 = conv(x1,h2);

subplot(2,1,1)
stem(ny1,y1)
xlabel('n')
ylabel('ye1')
title('ye1 = x1[n]*he1[n]')

subplot(2,1,2)
stem(ny2,y2)
xlabel('n')
ylabel('ye2')
title('ye2 = x1[n]*he2[n]');
```

Intermediate Problem f

To finds out the two outputs, we should first figure out the output of system 1, which is `(nx1 + 1) .* x1`. Then we use the output as the input of System 2, and use $h_{f2}[n]$ as the impulse response to get the output $y_{f1}[n]$.

Another try is that we first convolved $h_{f1}[n]$ with $h_{f2}[n]$, then we use the result as the impulse response to convolved with $x_1[n]$, the convolution result is named as $y_{f2}[n]$.



We can easily conclude from the plots that they are not equal to each other.

The MATLAB script is shown below.

```
nx1 = 0:9;
```

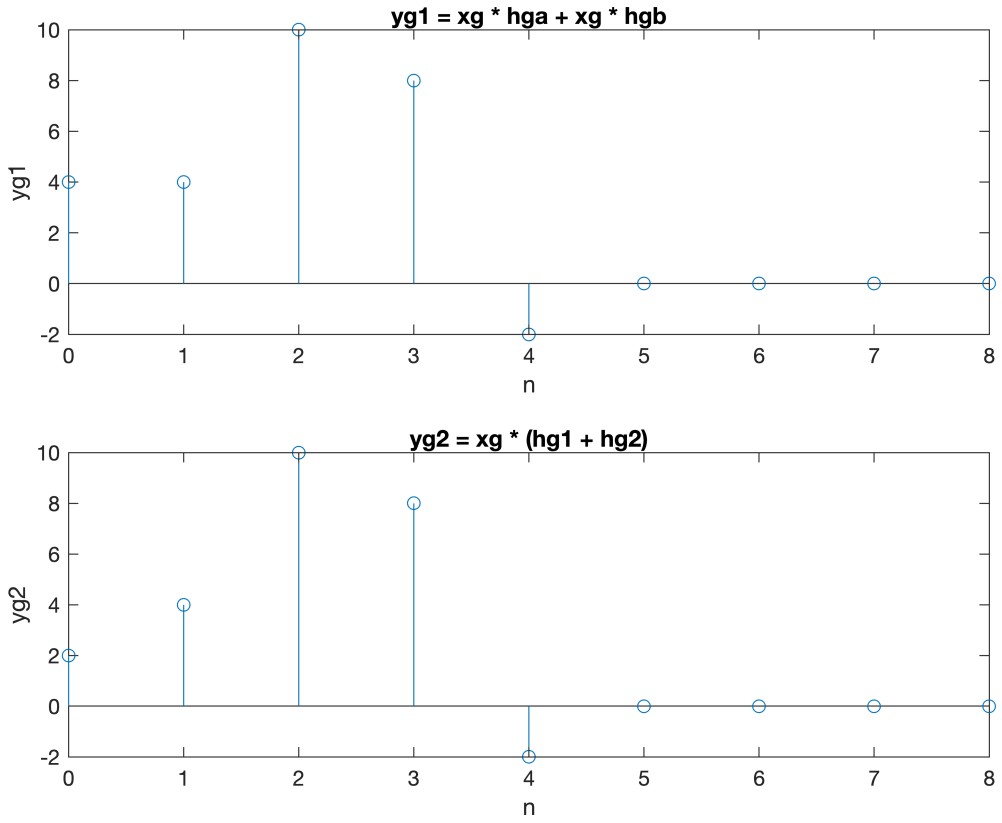
```
nhf2 = 0:4;
x1 = [ones(1, 5) zeros(1, 5)];
hf2 = [1 -1 3 0 1];

w = (nx1 + 1) .* x1;

yf1 = conv(w, hf2);
nyf1 = nx1(1) + nhf2(1):nx1(end) + nhf2(end);

xf1 = [1 zeros(1, 4)];
hf1 = (nhf2 + 1) .* xf1;

hseries = conv(hf1, hf2);
nhseries = nhf2(1) + nhf2(1):nhf2(end) + nhf2(end);

yf2 = conv(x1, hseries);
nyf2 = nx1(1) + nhseries(1):nx1(end) + nhseries(end);

subplot(2, 1, 1)
stem(nyf1, yf1)
xlabel('n')
ylabel('yf1')
title('yf1 = w[n]*hf2[n]')

subplot(2, 1, 2)
stem(nyf2, yf2)
xlabel('n')
ylabel('yf2')
title('yf2 = x1[n]*(hf1[n]*hf2[n]')
```

Intermediate Problem g

To find out the two output signals, we first need to find out the signal $x_g[n]$, and then use this signal as input signal for System 1 to find out the output signal $y_{ga}[n]$.Then we use the output of System 1 as the input signal for System 2. And the output $ygb[n]$ is the convolution of $y_{ga}[n]$ and $h_{g2}[n]$. $y_{g1}[n]$ is the result of add $y_{ga}[n]$ and $y_{gb}[n]$ together, which is $y_{g1}[n] = y_{ga}[n] + y_{gb}[n]$.

For another try, we first figure out the unit impulse response for System 1 by letting the input signal be unit impulse signal. Then we add this response signal $h_{g1}[n]$ with $h_{g2}[n]$. After that we convolved $x_g[n]$ with this result, so now we get the $y_{g2}[n]$ which is
$y_{g2}[n] = x_g[n] * (h_{g1}[n] + h_{g2}[n])$.

The two plots for $y_{g1}[n]$ and $y_{g2}[n]$ are shown below.



It is obvious that they are the same, which means $y_{g1}[n] = y_{g2}[n]$.

The MATLAB script is shown below.

```
n = 0:4;
x = [1 zeros(1, 4)];
hg2 = [0 2 5 4 -1];
xg = 2 .* x;

yga = xg.^2;
ygb = conv(xg, hg2);
nygb = 0:8;
yg1 = [yga zeros(1, 4)] + ygb;
nyg1 = 0:8;

hg1 = x.^2;
hparallel = hg1 + hg2;
yg2 = conv(xg, hparallel);
nyg2 = 0:8;

subplot(2, 1, 1)
stem(nyg1, yg1)
xlabel('n')
ylabel('yg1')
title('yg1 = xg * hga + xg*hgb')

subplot(2, 1, 2)
stem(nyg2, yg2)
xlabel('n')
ylabel('yg2')
title('yg2 = xg * (hg1 + hg2)')
```

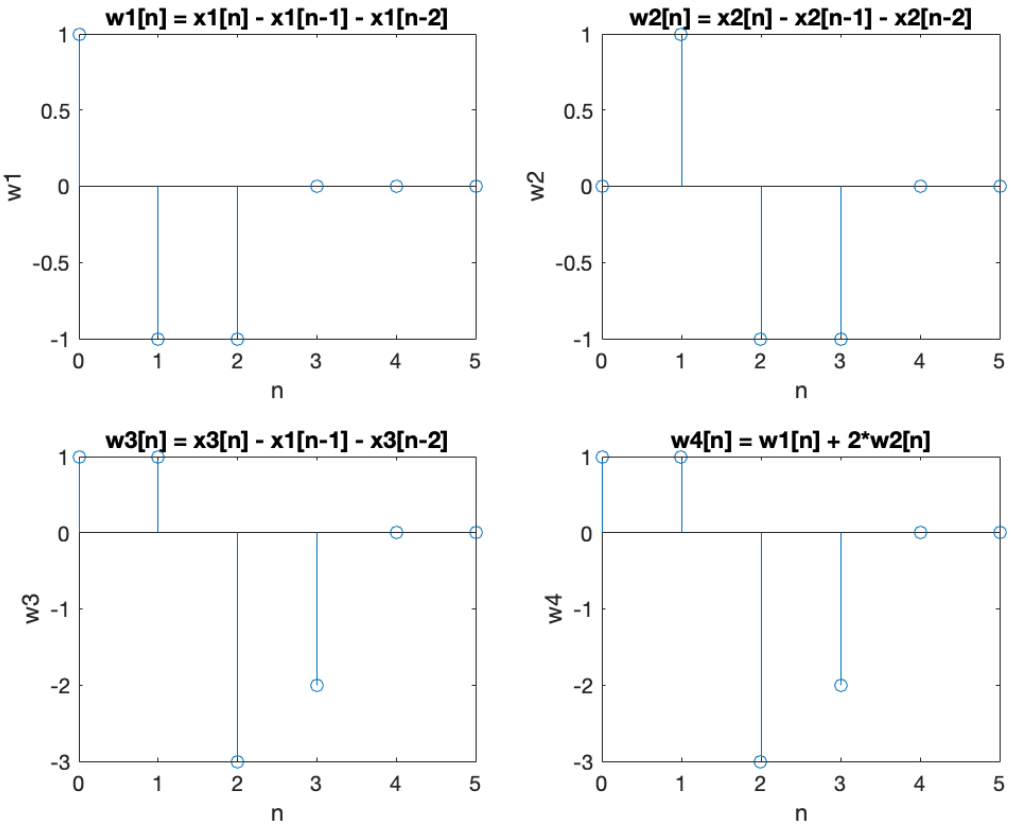
2.5

Basic Problem a

Basic problems a,b,c use the same Systems, so first we plot the Systems' outputs.

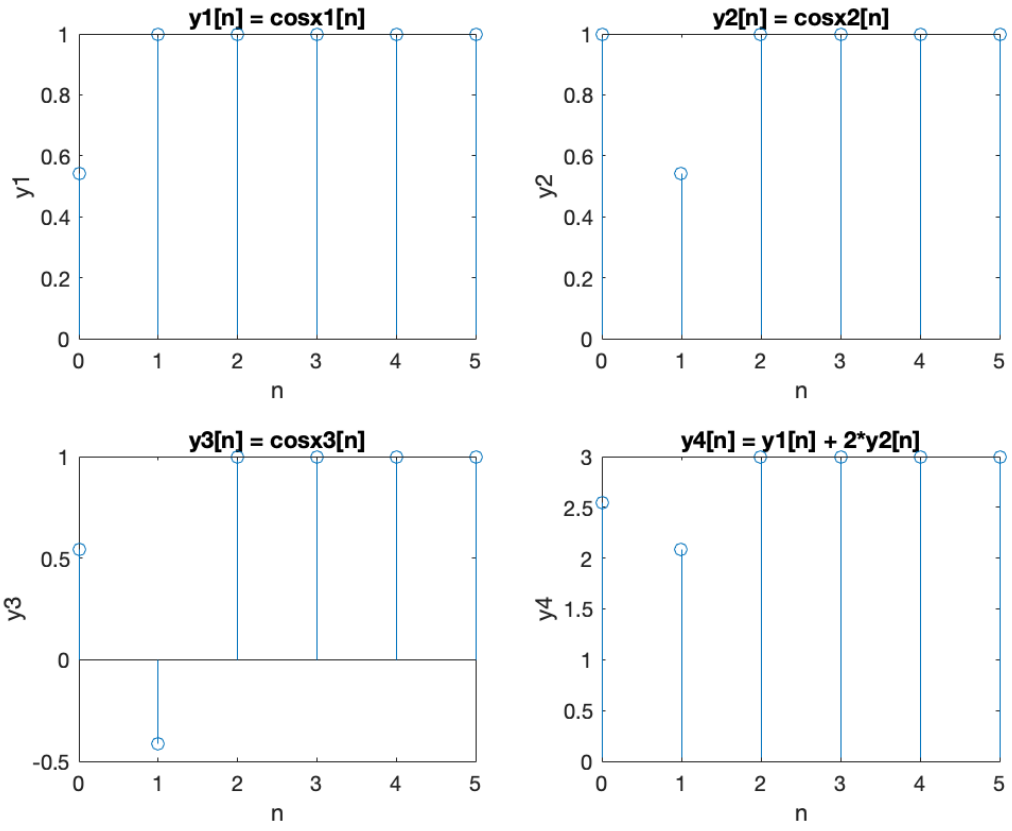
For System 1, it is not easy to figure out the expression for y using x quickly, so use vector to record their index and count the result using `filter()`.

The plot is shown below.



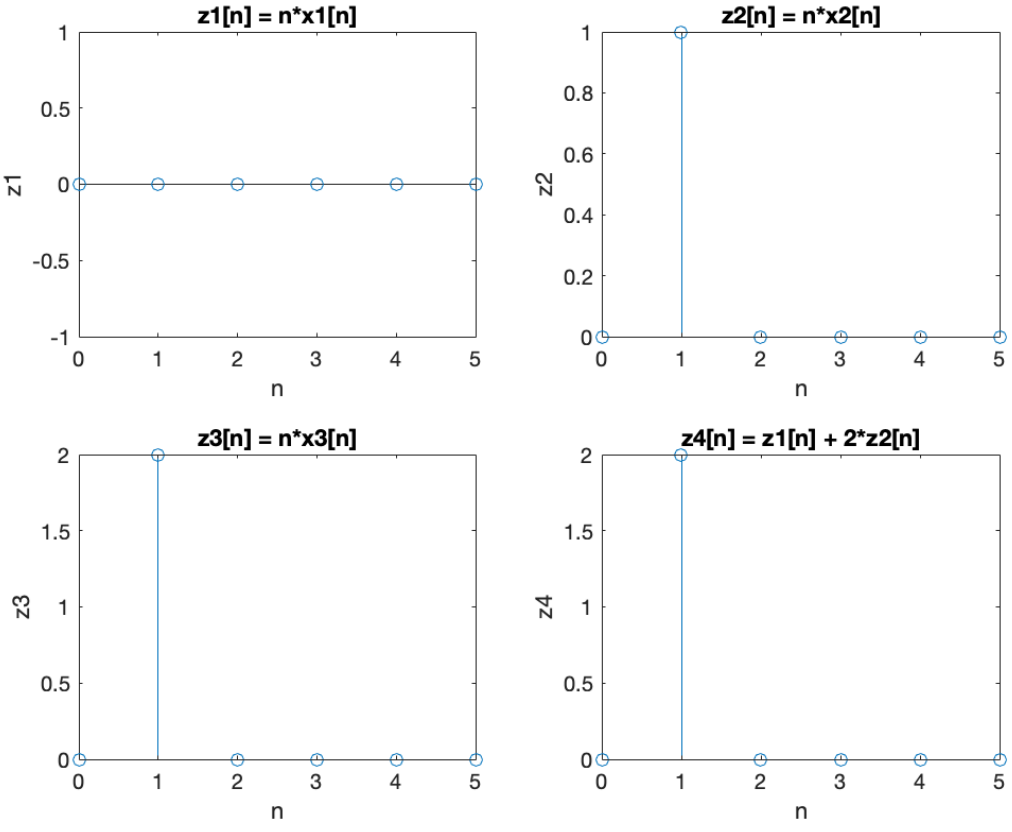
For System2, we simply use `cos()` to find out the result.

The plot is shown below.



For System 3, we calculated the results using point multiplication, which is `n .* x`.

The plot is shown below.



The MATLAB script is shown below.

```
n = 0:4;
```

```
x = [1 zeros(1, 4)];
hg2 = [0 2 5 4 -1];
xg = 2 .* x;

yga = xg.^2;
ygb = conv(xg, hg2);
nygb = 0:8;
yg1 = [yga zeros(1, 4)] + ygb;
nyg1 = 0:8;

hg1 = x.^2;
hparallel = hg1 + hg2;
yg2 = conv(xg, hparallel);
nyg2 = 0:8;

subplot(2, 1, 1)
stem(nyg1, yg1)
xlabel('n')
ylabel('yg1')
title('yg1 = xg*hga + xg*hgb')

subplot(2, 1, 2)
stem(nyg2, yg2)
xlabel('n')
ylabel('yg2')
title('yg2 = xg*(hg1 + hg2)')
```

Basic Problem b

Since for all three systems the input $x_3[n] = x_1[n] + 2 * x_2[n]$, and the output $y_4[n] = y_1[n] + 2 * y_2[n]$, so to analyse whether they are linear, we only need to check whether the plots for $y_3[n]$ and $y_4[n]$ are the same.

It is easy to see that System 1 and System 3 are linear but System 2 is not.

Basic Problem c

To figure out whether they are time invariant, we need to compare $y_1[n]$ and $y_2[n]$, since the input $x_1[n]$ and $x_2[n]$ have the relation $x_1[n] = x_2[n - n_0]$, so if $y_1[n] = y_2[n - n_0]$, then the system is time invariant, otherwise not.

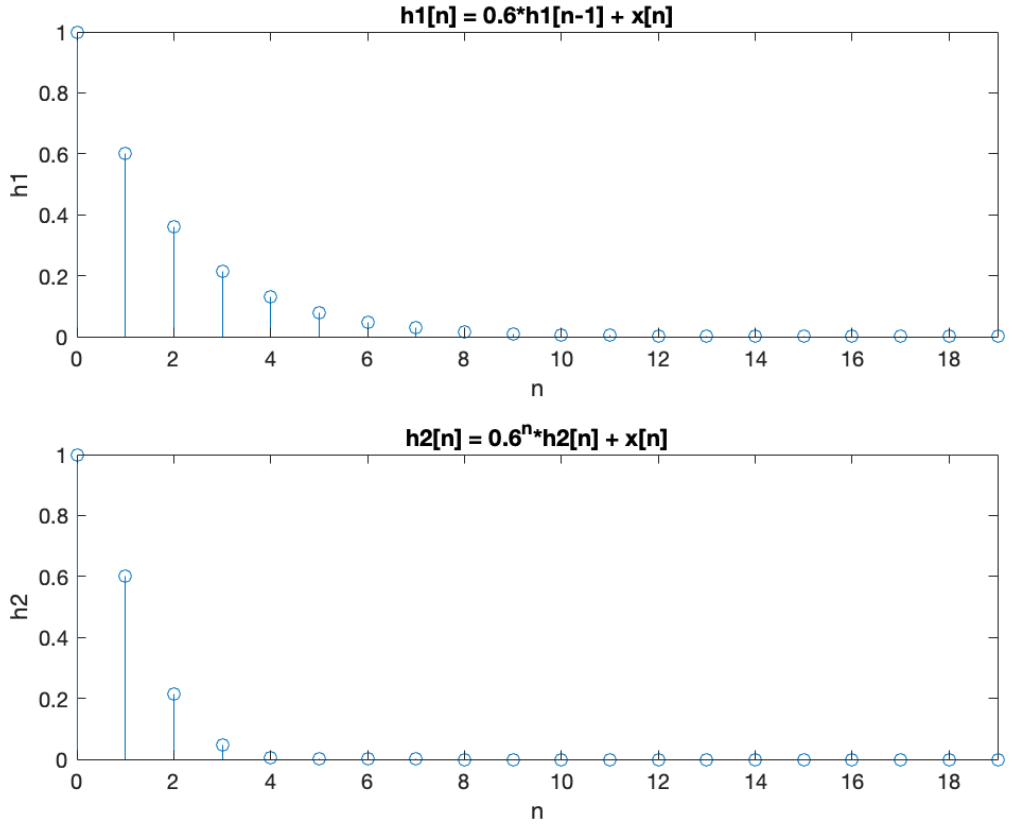
It is easy to see that System 1 and System 2 are time invariant while System 3 are not.

Intermediate Problem d

The System 1 output can be figured out by using `filter()`, and to find the unit impulse response we can use unit impulse as input.

But for System 2 we cannot use filter anymore because the variable n occurs outside the input signal itself. To deal with this problem, we should use a `for` loop to figure out all the members in the unit impulse response one by one.

The plots for $h_1[n]$ and $h_2[n]$ are shown below.



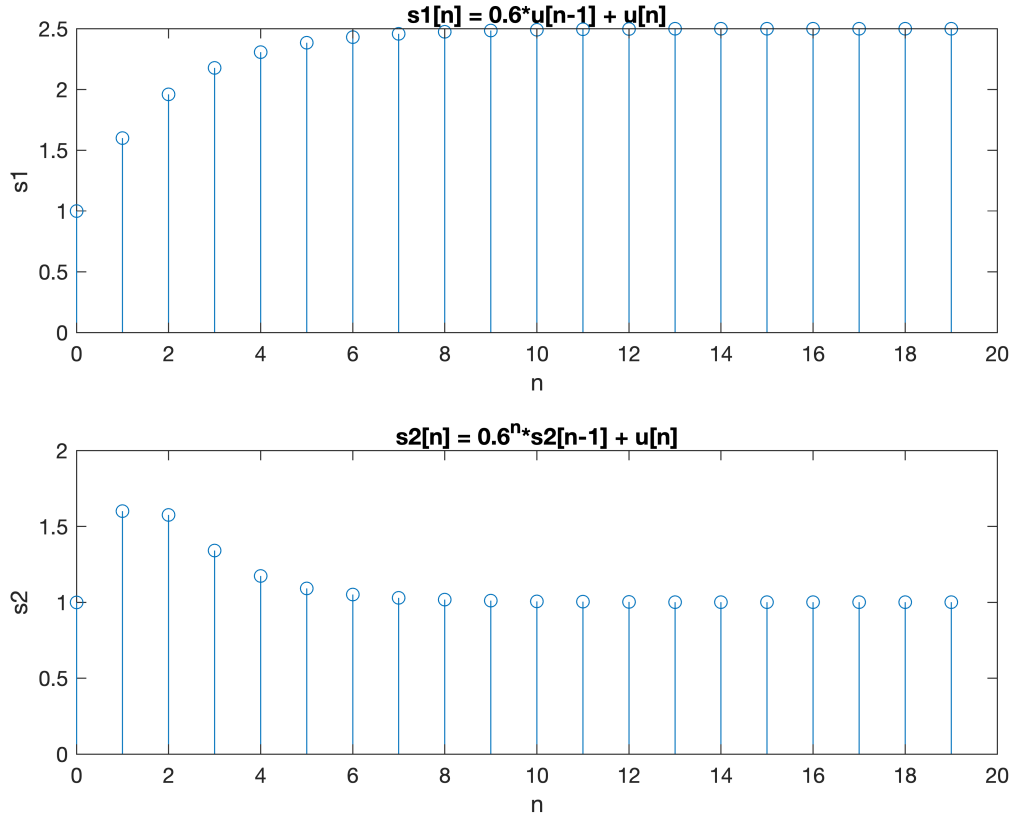
The MATLAB script is shown below.

```
n = 0:4;
x = [1 zeros(1, 4)];
hg2 = [0 2 5 4 -1];
xg = 2 .* x;

yga = xg.^2;
ygb = conv(xg, hg2);
nygb = 0:8;
yg1 = [yga zeros(1, 4)] + ygb;
nyg1 = 0:8;

hg1 = x.^2;
hparallel = hg1 + hg2;
yg2 = conv(xg, hparallel);
nyg2 = 0:8;

subplot(2, 1, 1)
stem(nyg1, yg1)
xlabel('n')
ylabel('yg1')
title('yg1 = xg*hga + xg*hgb');
subplot(2, 1, 2)
stem(nyg2, yg2)
xlabel('n')
ylabel('yg2')
title('yg2 = xg*(hg1 + hg2)');
```

Intermediate Problem e

From the Intermediate Problem d we have already found the unit impulse response, so for finding the unit step response we can use the same method but change input from $\delta[n]$ to $u[n]$.

The plots are shown below.



The MATLAB script is shown below.

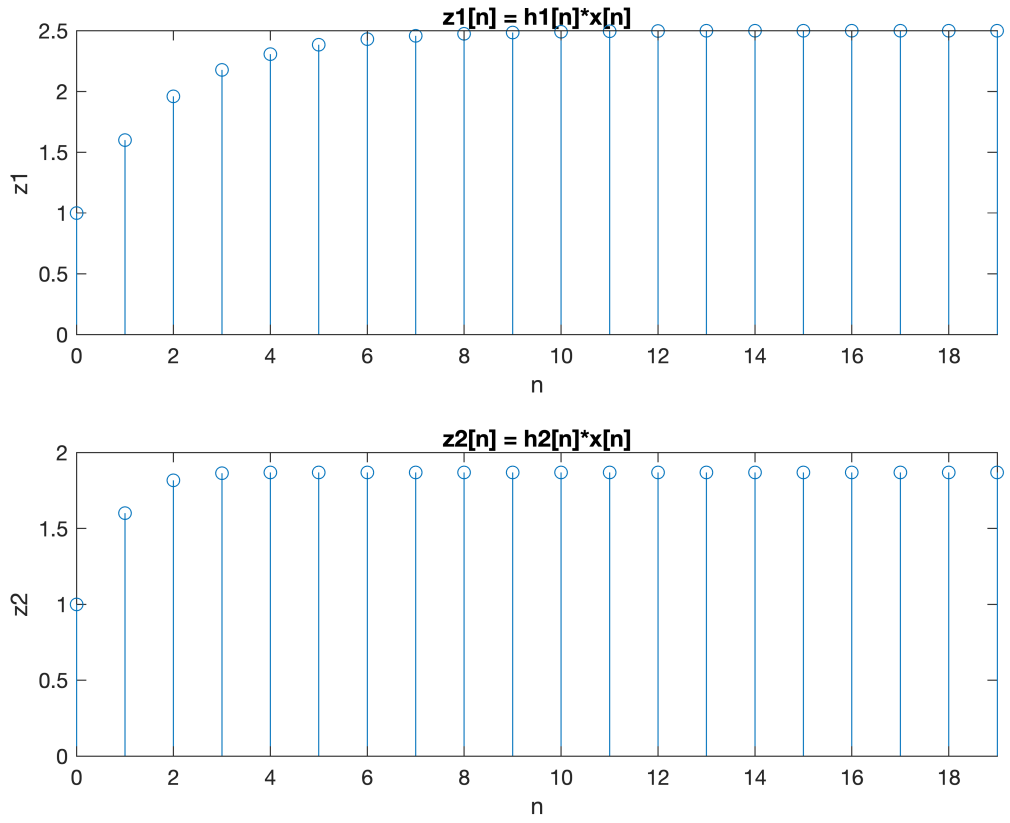
```
n = 0:19;
x1 = ones(1, 20);
%System1
a1 = [1 -0.6];
b1 = [1 0];
s1 = filter(b1, a1, x1);
%System2
s2 = zeros(1, 20);
s2(1) = 1; %cuz s2(i) = 0 if i <1, s2[0] = x[0] = 1, s2[1] = s2[0] + 0;

for i = 1:19
    s2(i + 1) = 0.6^i * s2(i) +x1(i + 1);
end

subplot(2, 1, 1)
stem(n, s1)
xlabel('n')
ylabel('s1')
title('s1[n] = 0.6*u[n-1] + u[n]')

subplot(2, 1, 2)
stem(n, s2)
xlabel('n')
```

```
ylabel('s2')
title('s2[n] = 0.6^n*s2[n-1] + u[n]')
```

Intermediate Problem f

To find out the results, all we need to do is to use $h_1[n]$ and $h_2[n]$ as inputs and convolve with unit step signal $u[n]$.

The plots are shown below.



The MATLAB script is shown below.

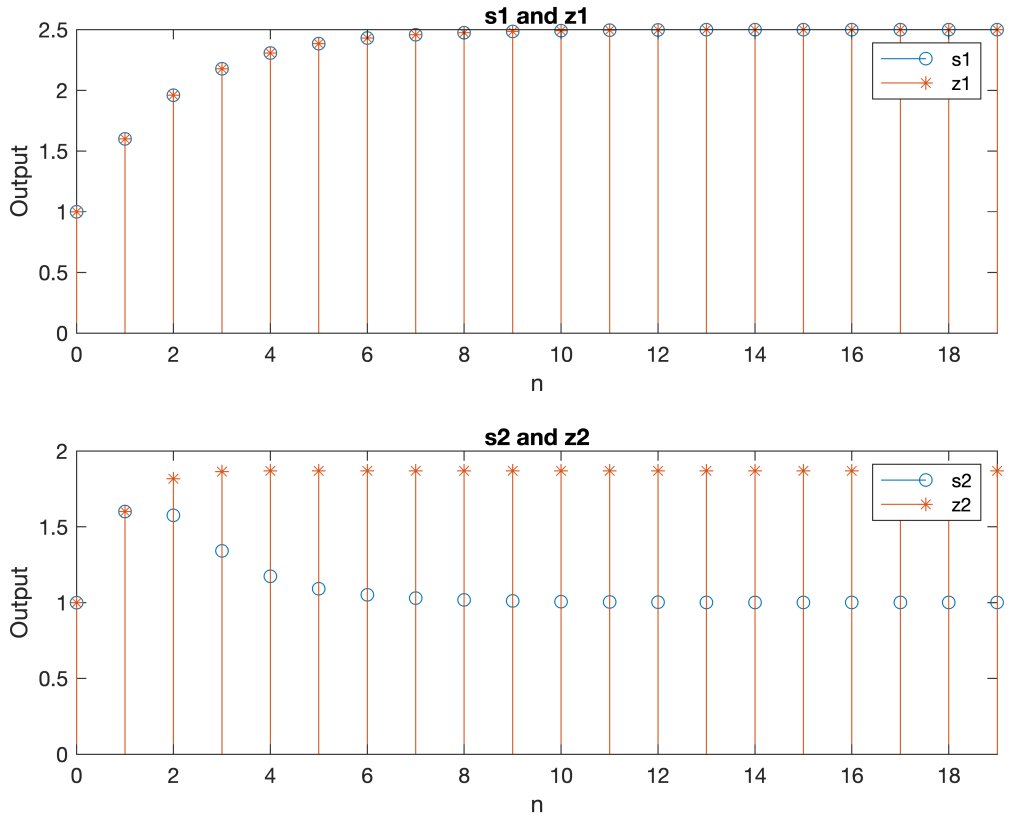
```
n = 0:19;
x = [1 zeros(1, 19)];
x1 = ones(1, 20);
%System1
a1 = [1 -0.6];
b1 = [1 0];
h1 = filter(b1, a1, x);

% System2
h2 = zeros(1, 20);
h2(1) = 1
% h2(i) = 0 if i <1, h2[0] = x[0] = 1, h2[1] = h2[0] + 0;

for i = 1:19
    h2(i + 1) = 0.6^i * h2(i) +x(i + 1);
end

z1 = conv(h1, x1);
nz1 = 0:38;
```

```
z2 = conv(h2, x1);
nz2 = 0:38;

subplot(2, 1, 1)
stem(nz1, z1)
axis([0, 19, ylim])
xlabel('n')
ylabel('z1')
title('z1[n] = h1[n]*x[n]')

subplot(2, 1, 2)
stem(nz2, z2)
axis([0, 19, ylim])
xlabel('n')
ylabel('z2')
title('z2[n] = h2[n]*x[n]')
```

Intermediate Problem g

THe plots for `s1`, `z1`, `s2` and `z2` have already been graphed in problem e and f, and now we place the two plots in one graph.

The plots are shown below.



It is obvious that s1 equals to z1 but s2 doesn't equal to z2. The reason is that System 1 is a LTI system but System 2 is not, so the convolution for input and impulse response will not equal to the exact output.

The MATLAB script is shown below.

```
n = 0:19;
```

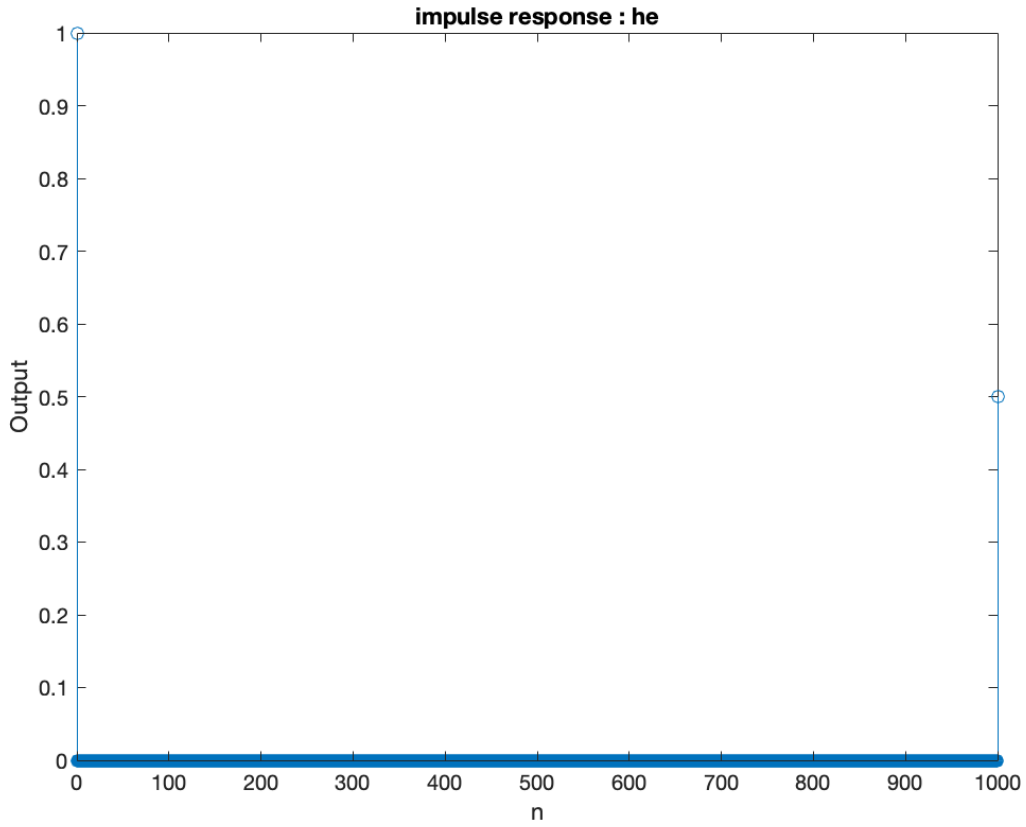
```
x = [1 zeros(1, 19)];
x1 = ones(1, 20);
% System1
a1 = [1 -0.6];
b1 = [1 0];
h1 = filter(b1, a1, x);
%System2
h2 = zeros(1, 20);
h2(1) = 1;
% h2(i) = 0 if i <1, h2[0] = x[0] = 1, h2[1] = h2[0] + 0;

for i = 1:19
    h2(i + 1) = 0.6^i * h2(i) +x(i + 1);
end

%The four outputs
z1 = conv(h1, x1);
nz1 = 0:38;
s1 = filter(b1, a1, x1);
z2 = conv(h2, x1);
nz2 = 0:38;
s2(1) = 1;

for i = 1:19
    s2(i + 1) = 0.6^i * s2(i) +x1(i + 1);
end

subplot(2, 1, 1)
stem(n, s1, 'o')
hold on;
stem(nz1, z1, '*')
axis([0, 19, ylim])
xlabel('n')
ylabel('Output')
title('s1 and z1')
legend('s1', 'z1')

subplot(2, 1, 2)
stem(n, s2, 'o')
axis([0, 19, ylim])
hold on;
stem(nz2, z2, '*')

xlabel('n');
ylabel('Output')
title('s2 and z2')
legend('s2', 'z2')
```

2.10

Basic Problem a

For this question we need to find out the unit impulse response for the System $y[n] = x[n] + \alpha x[n - N]$, and $\alpha = 0.5$ and $N = 1000$. So we use $\delta[n]$ as the input and sort the range $0 \leq n \leq 1000$, calculate the output and store it in vector `he`.

The plot is shown below.



The MATLAB script is shown below.

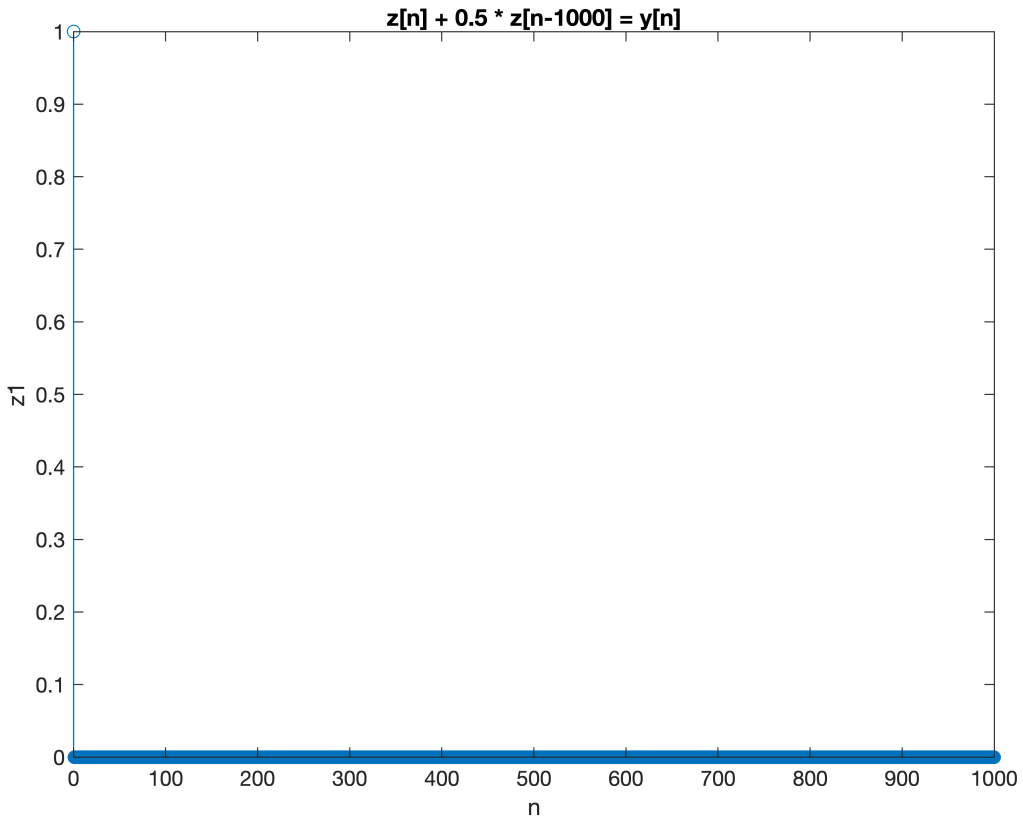
```
n = 0:1000;
x = [1 zeros(1, 1000)];
a1 = 1;
b1 = [1 zeros(1, 999) 0.5];

he = filter(b1, a1, x);

stem(n, he)
xlabel('n')
ylabel('Output')
title('impulse response : he')
```

Basic Problem b

We now should deal with $z[n] + \alpha z[n - N] = y[n]$, where $\alpha = 0.5$ and $N = 1000$, the same condition as problem a.However, because we cannot find out the expression of $z[n]$ directly, so we use the function `filter` to calculate `z` then, the index vector for output is `[1 0.5]` and the index vector for input is `[1]`. Also , the input signal is the result of problem a.

From mathematical sight we can easily know that $z[n] + \alpha z[n-N] = y[n] = x[n] + \alpha x[n-N]$, which is clear that $z[n] + \alpha z[n-N] = x[n] + \alpha x[n-N]$. So it is obvious now that $z[n] = x[n]$.

We can also prove it using the plot of $z[n]$.The plot is shown below.



The output is a unit impulse signal, the same as $x[n]$. So $z[n] = x[n]$.

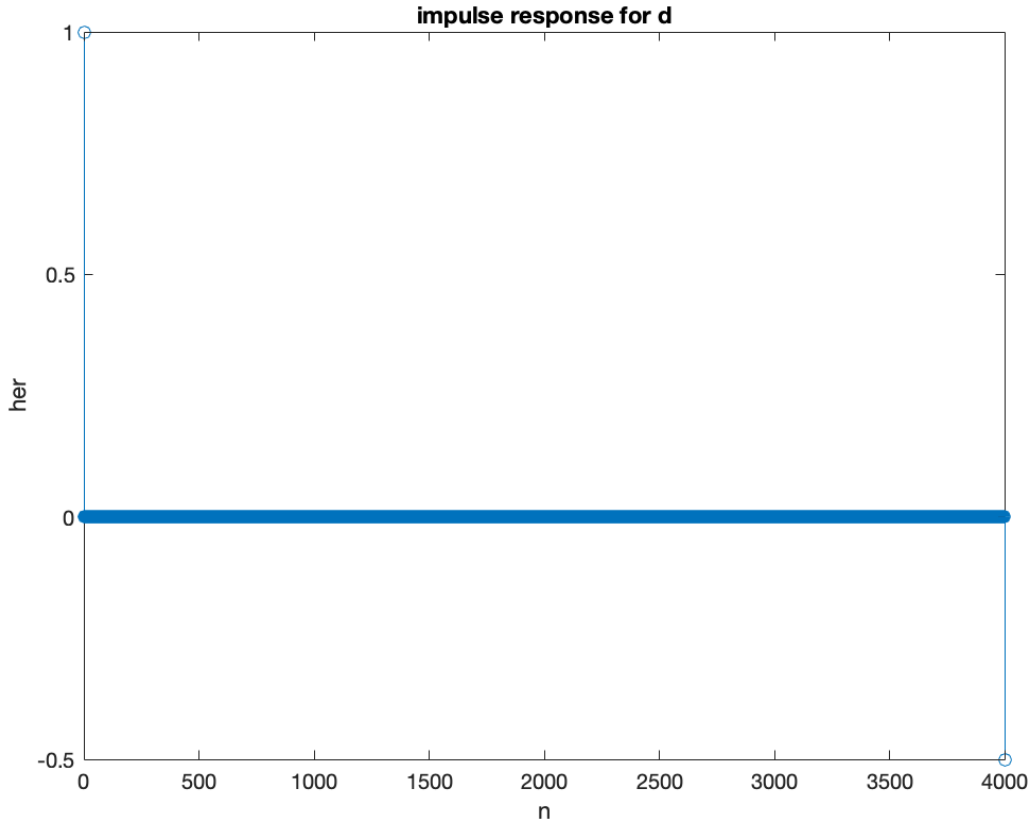
The MATLAB script is shown below.

```
n = 0:1000;
x = [1 zeros(1, 1000)];
a1 = 1;
b1 = [1 zeros(1, 999) 0.5];
y1 = filter(b1, a1, x);

a2 = [1 zeros(1, 999) 0.5];
b2 = 1;
z1 = filter(b2, a2, y1);

stem(n, z1)
xlabel('n')
ylabel('z1')
title('z[n] + 0.5 * z[n-1000] = y[n]')
```

Intermediate Problem c

This problem is similar to problem a and the only difference between them is the sorted range for unit impulse signal $\delta[n]$ is $0 \leq n \leq 4000$. We store this output in vector `her`.

The plot is shown below.



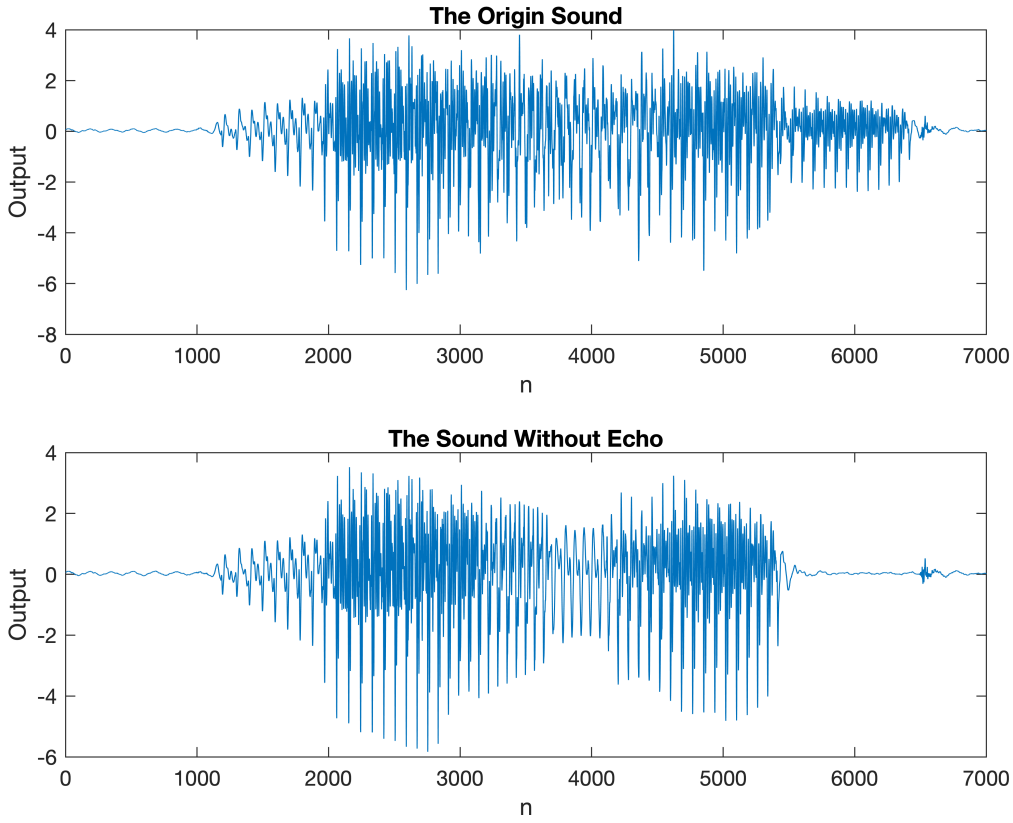
The MATLAB script is shown below.

```
n = 0:4000;
d = [1 zeros(1, 4000)];
a1 = [1 zeros(1, 3999) 0.5];
b1 = 1;

her = filter(b1, a1, d);

stem(n, her)
xlabel('n')
ylabel('her')
title('impulse response for d')
```

Intermediate Problem d

By using `z = filter(1, a, y)`, we can clearly see the differences between the original sound and the sound without echo.

The plot is shown below.



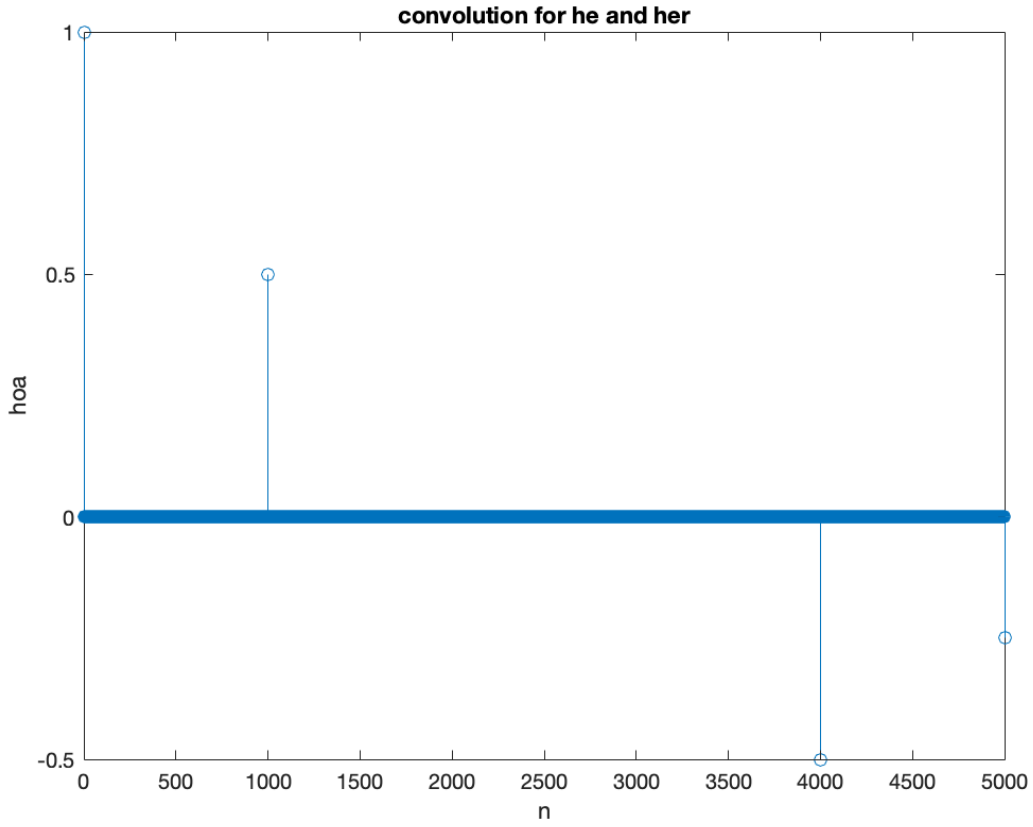
The MATLAB script is shown below.

```
load lineup.mat;
sound(y, 8192)

a = [1 zeros(1, 999) 0.5];

r = filter(1, a, y);

subplot(2, 1, 1)
plot(y)
xlabel('n')
ylabel('Output')
title('The Origin Sound')

subplot (2, 1, 2)
plot(r
xlabel('n')
ylabel('Output')
title('The Sound Without Echo')
```

Intermediate Problem e

We convolved the two impulse responses `he` and `her`, and named the output as `hoa`.

The plot is shown below.



It is not a unit impulse signal, because the zeros between the two impulses of `he` and `her` are not the same, for `he` is 1000 but for `her` is 4000.

The MATLAB script is shown below.

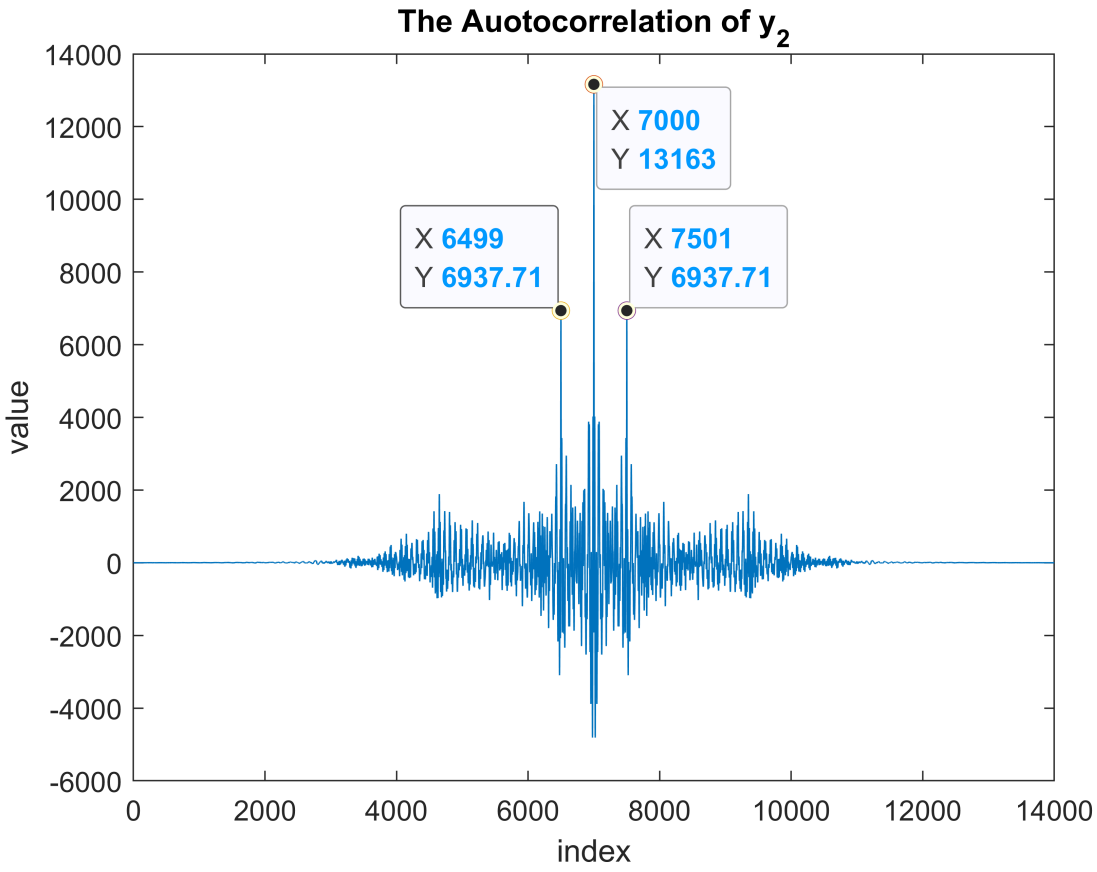
```
n = 0:1000;
x = [1 zeros(1, 1000)];
a1 = 1;
b1 = [1 zeros(1, 999) 0.5];
he = filter(b1, a1, x);

n2 = 0:4000;
d = [1 zeros(1, 4000)];
a2 = [1 zeros(1, 3999) 0.5];
b2 = 1;

her = filter(b2, a2, d);

hoa = conv(he, her)
nhoa = 0:5000;

stem(nhoa, hoa)
xlabel('n')
ylabel('hoa')
title('convolution for he and her')
```

Advanced Problem f

In audio signal `y2`, the wave satisfies the pattern

$$y[n] = x[n] + \alpha x[n - N].$$

The plot of `Rxx`, which is the autocorrelation signal of `y2` is shown below.



From the plot, we can conclude that there's a maximum peak in the center of the sampling points, which represents for the original audio signal without time delay. Next to the center peak there are two sub-maximum peaks symmetric with respect to the maximum peak, which represent the delayed audio signal.

```
% to obtain the maximum peak
[p1y, p1x] = max(Rxx);

% assign 0 to the sampling points around the maximum peak
Rxx2 = Rxx;
Rxx2(p1x - 100:p1x + 100, 1) = 0;

% to obtain the sub-maximum peak
[p2y, p2x] = max(Rxx2);
```

Firstly, we find the time latency by calculating the time difference between the peaks.

```
% calculate the time difference
N = abs(p1x - p2x);
```

Then, we determine the attenuation factor α. According to the property of autocorrelation, the magnitude in autocorrelation represents the energy power contained by the wave. Hence alpha should be the square root of the ratio of values of the peaks.

```
% calculate the attenuation factor alpha
alp = sqrt(p2y / p1y);
```

Therefore, the `y2` signal can be decomposed as

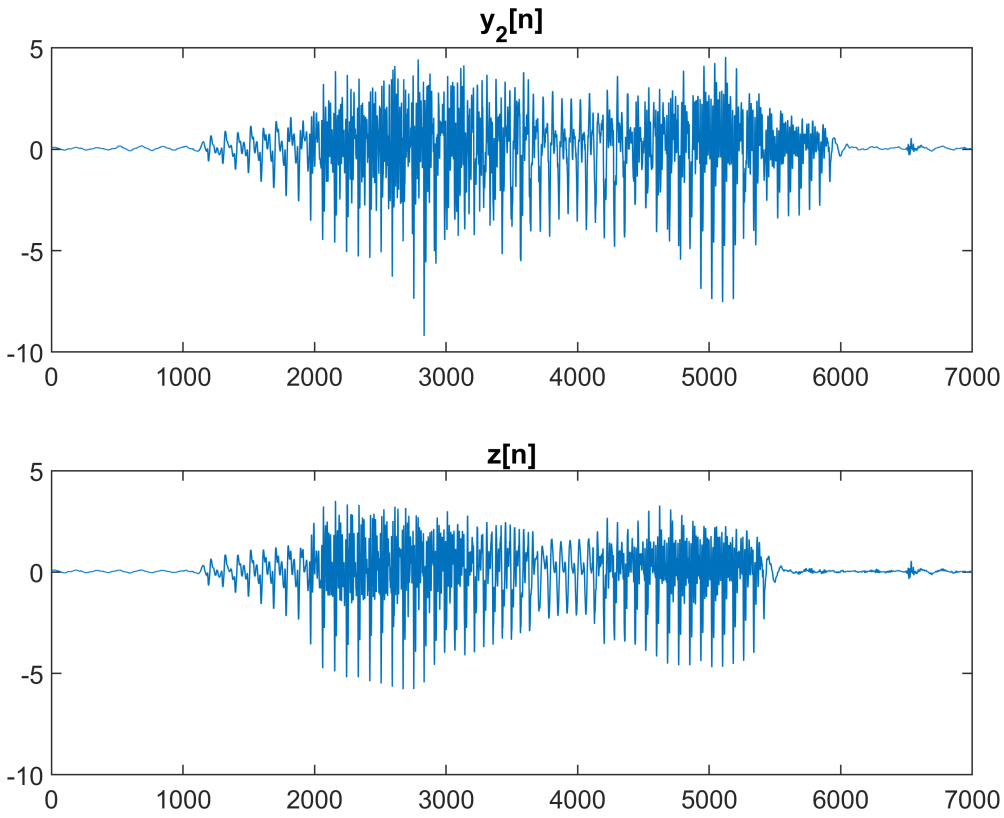
$$y_2[n] = z[n] + \alpha z[n - N],$$

in which $\alpha = 0.7260$, $N = 501$, and $z[n]$ is the original audio signal we require.

Using `filter()`, we can obtain $z[n]$.

```
% cancel the echo using filter
A2 = [1 zeros(1, N - 1) alp];
B2 = 1;
z = filter(B2, A2, y2);
```

By comparing the wave pattern of $y_2[n]$ and $z[n]$, as well as listening to the echo-canceled audio, the performance of the echo canceling process satisfies our expectation.



The complete MATLAB script of this part is shown below.

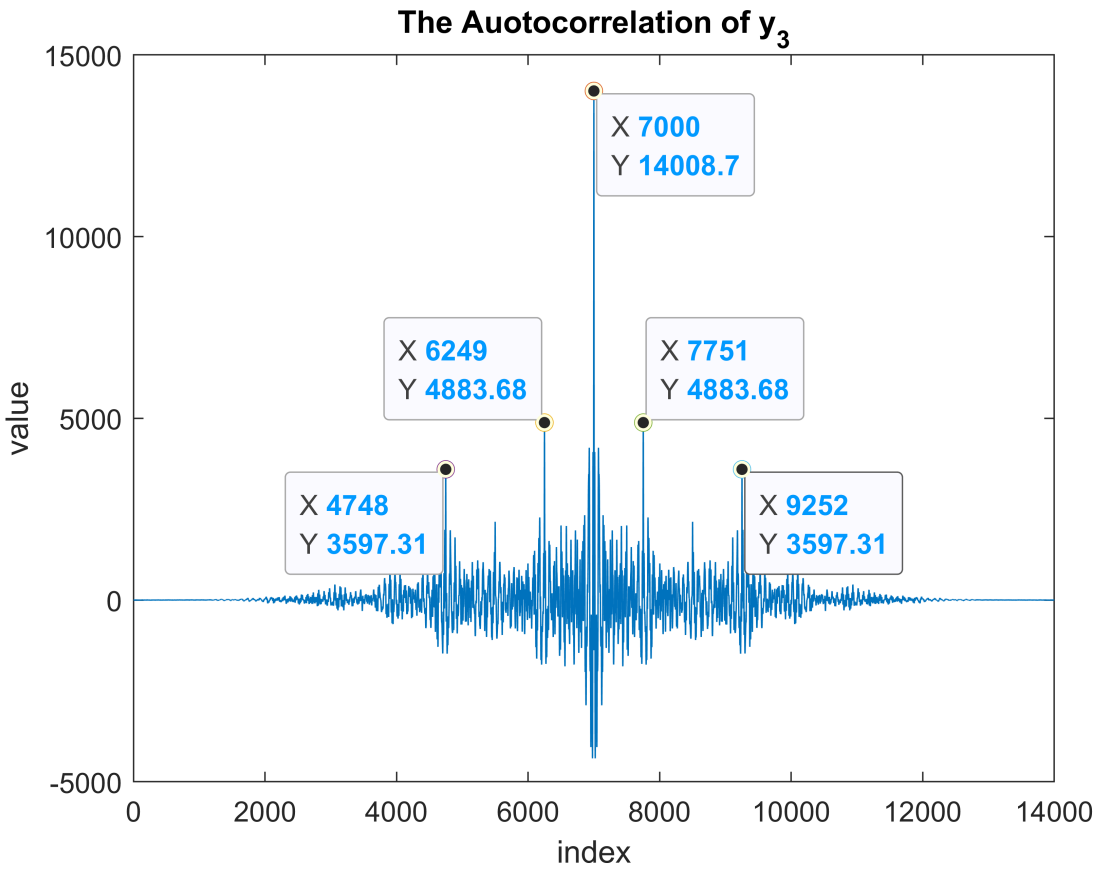
```
load lineup.mat

% obtain the autocorrelation of y2
Rxx = xcorr(y2);
figure
plot(Rxx)
title('The Auotocorrelation of y_2')
xlabel('index')
ylabel('value')
```

```
% to obtain the maximum peak
[p1y, p1x] = max(Rxx);

% assign 0 to the sampling points around the maximum peak
Rxx2 = Rxx;
Rxx2(p1x - 100:p1x + 100, 1) = 0;

% to obtain the sub-maximum peak
[p2y, p2x] = max(Rxx2);

% mark the peaks in the plot
hold on
plot(p1x, p1y, 'o')
plot(p2x, p2y, 'o')
plot(2 * p1x - p2x, p2y, 'o')

% calculate the time difference
N = abs(p1x - p2x);

% calculate the attenuation factor alpha
alp = sqrt(p2y / p1y);

% cancel the echo using filter
A2 = [1 zeros(1, N - 1) alp];
B2 = 1;
z = filter(B2, A2, y2);

% verifying the result
figure
subplot(2, 1, 1)
plot(y2)
title('y_2[n]')
subplot(2, 1, 2)
plot(z)
title('z[n]')
ylim([-10, 5])

sound(z)
```

In audio signal `y3`, the wave satisfies the pattern

$$y[n] = x[n] + \alpha_1 x[n - N_1] + \alpha_2 x[n - N_2]$$

The plot of `Rxx`, which is the autocorrelation signal of `y3` is shown below.



From the plot, we can conclude that there's a maximum peak in the center of the sampling points, which represents for the original audio signal without time delay. Next to the center peak there are two sub-maximum peaks and two third-maximum peaks symmetric with respect to the maximum peak, which represent the delayed audio signal.

```
% to obtain the maximum peak
[p1y, p1x] = max(Rxx);

% assign 0 to the sampling points around the maximum peak
Rxx2 = Rxx;
Rxx2(p1x - 100:p1x + 100, 1) = 0;

% to obtain the sub-maximum peak
[p2y, p2x] = max(Rxx2);

% assign 0 to the sampling points around the sub-maximum peak
Rxx3 = Rxx2;
Rxx3(p2x - 100:p2x + 100, 1) = 0;
Rxx3(2 * p1x - p2x - 100:2 * p1x - p2x + 100, 1) = 0;

% to obtain the third-maximum peak
[p3y, p3x] = max(Rxx3);
```

Firstly, we find the time latency by calculating the time difference between the peaks.

```
% calculate the time difference
N1 = abs(p1x - p2x);
N2 = abs(p1x - p3x);
```

Then, we determine the attenuation factors α_1 and α_1. According to the property of autocorrelation, the magnitude in autocorrelation represents the energy power contained by the wave. Hence alpha should be the square root of the ratio of values of the peaks.

```
% calculate the attenuation factor alpha
alpha1 = sqrt(p2y / p1y);
alpha2 = sqrt(p3y / p1y);
```

Therefore, the `y3` signal can be decomposed as

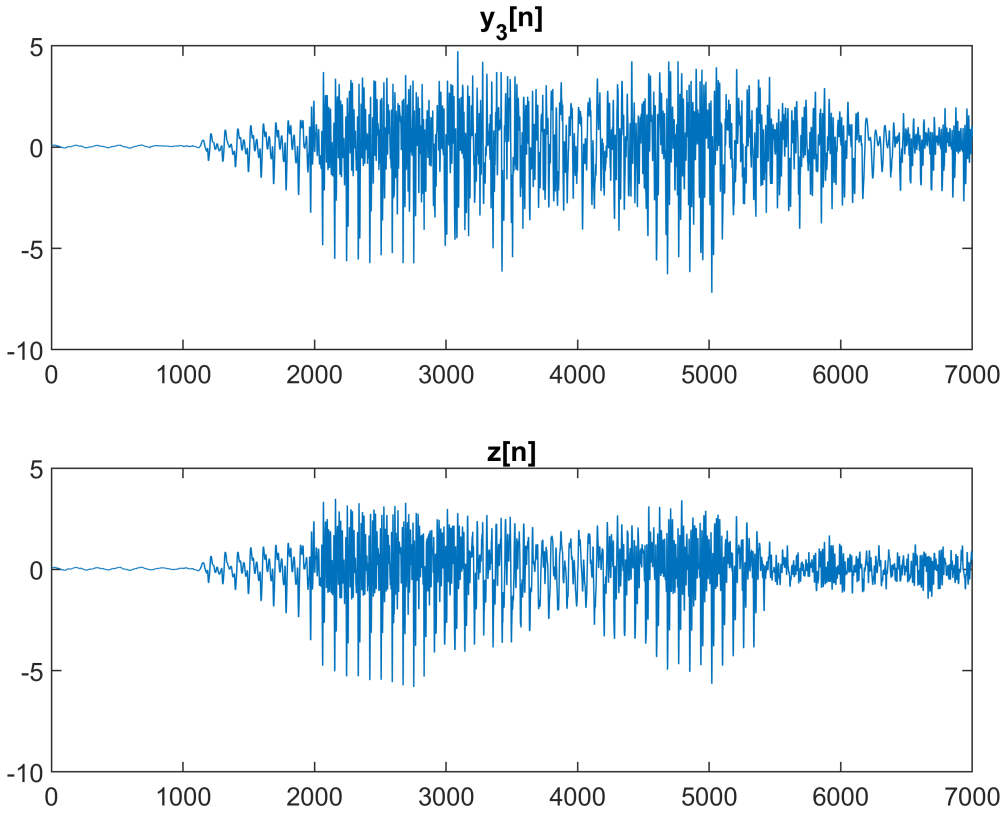
$$y_3[n] = z[n] + \alpha_1 z[n - N_1] + \alpha_2 z[n - N_2],$$

in which $\alpha_1 = 0.5904$, $\alpha_2 = 0.5067$, $N_1 = 751$, $N_2 = 2252$, and $z[n]$ is the original audio signal we require.

Using `filter()`, we can obtain $z[n]$.

```
% cancel the echo using filter
A2 = [1 zeros(1, N1 - 1) alpha1 zeros(1, N2 - N1 -1) alpha2];
B2 = 1;
z = filter(B2, A2, y3);
```

By comparing the wave pattern of $y_3[n]$ and $z[n]$, as well as listening to the echo-canceled audio, the performance of the echo canceling process satisfies our expectation.



The complete MATLAB script of this part is shown below.

```
load lineup.mat

% obtain the autocorrelation of y3
Rxx = xcorr(y3);
```

```
figure
plot(Rxx)
title('The Auotocorrelation of y_3')
xlabel('index')
ylabel('value')

% to obtain the maximum peak
[p1y, p1x] = max(Rxx);

% assign 0 to the sampling points around the maximum peak
Rxx2 = Rxx;
Rxx2(p1x - 100:p1x + 100, 1) = 0;

% to obtain the sub-maximum peak
[p2y, p2x] = max(Rxx2);

% assign 0 to the sampling points around the sub-maximum peak
Rxx3 = Rxx2;
Rxx3(p2x - 100:p2x + 100, 1) = 0;
Rxx3(2 * p1x - p2x - 100:2 * p1x - p2x + 100, 1) = 0;

% to obtain the third-maximum peak
[p3y, p3x] = max(Rxx3);

% mark the peaks in the plot
hold on
plot(p1x, p1y, 'o')
plot(p2x, p2y, 'o')
plot(p3x, p3y, 'o')
plot(2 * p1x - p2x, p2y, 'o')
plot(2 * p1x - p3x, p3y, 'o')

% calculate the time difference
N1 = abs(p1x - p2x);
N2 = abs(p1x - p3x);

% calculate the attenuation factor alpha
alpha1 = sqrt(p2y / p1y);
alpha2 = sqrt(p3y / p1y);

% cancel the echo using filter
A2 = [1 zeros(1, N1 - 1) alpha1 zeros(1, N2 - N1 -1) alpha2];
B2 = 1;
z = filter(B2, A2, y3);

% verifying the result
figure
subplot(2, 1, 1)
plot(y3)
title('y_3[n]')
subplot(2, 1, 2)
plot(z)
ylim([-10, 5])
title('z[n]')

sound(z)
```