Engineering Assignment Coversheet

Student Number(s)
906348
683557
Group Code (if applicable):

Please note that you:

- Must keep a full copy of your submission for this assignment
- Must staple this assignment
- Must NOT use binders or plastic folders except for large assignments

Assignment Title:	Sampling of analog signals and design of anti-aliasing filters
Subject Number:	ELEN90058
Subject Name:	Signal Processing
Student Name:	Weicheng Duan Shiyu Zhang
Lecturer/Tutor:	
Due Date:	22/AUG/2017

For Late Assignments Only

Has an extension been granted? Yes / No (circle)

A per-day late penalty may apply if you submit this assignment after the due date/extension. Please check with your Department/coordinator for further information.

Plagiarism

Plagiarism is the act of representing as one's own original work the creative works of another, without appropriate acknowledgment of the author or source.

Collusion

Collusion is the presentation by a student of an assignment as his or her own which is in fact the result in whole or in part of unauthorised collaboration with another person or persons. Collusion involves the cooperation of two or more students in plagiarism or other forms of academic misconduct.

Both collusion and plagiarism can occur in group work. For examples of plagiarism, collusion and academic misconduct in group work please see the University's policy on Academic Honesty and Plagiarism: http://academichonesty.unimelb.edu.au/

Plagiarism and collusion constitute cheating. Disciplinary action will be taken against students who engage in plagiarism and collusion as outlined in University policy. Proven involvement in plagiarism or collusion may be recorded on my academic file in accordance with Statute 13.1.18.

STUDENT DECLARATION

Please sign below to indicate that you understand the following statements:

I declare that:

- This assignment is my own original work, except where I have appropriately cited the original source.
- This assignment has not previously been submitted for assessment in this or any other subject.

For the purposes of assessment, I give the assessor of this assignment the permission to:

- Reproduce this assignment and provide a copy to another member of staff; and
- Take steps to authenticate the assignment, including communicating a copy of this assignment to a checking service (which may retain a copy of the assignment on its database for future plagiarism checking).

Student signature ………………………………………………………………… Date 22/AUG/2017 …………………………………

This page was left intentionally

Part A

a) Derive an expression for the spectrum Xc(j Ω), Plot xc(t) and Xc(j Ω) for some values of a and Ω_1

CTFT for the signal:

If $x_c(t) = \begin{cases} e^{-at}\cos(\Omega_1 t)\,, \ t \geq 0 & , \quad a > 0 \\ 0 & \quad t < 0 \end{cases}$

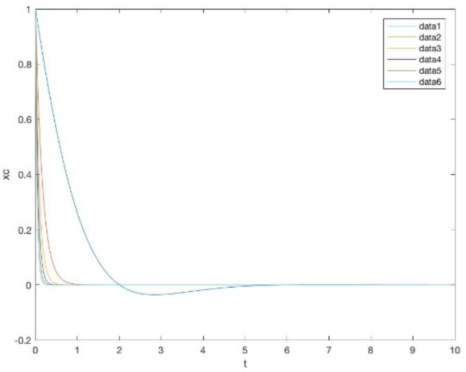
$$X_c(j\Omega) = \int_{-\infty}^{0} 0 \times e^{-j\Omega t} dt + \int_{0}^{\infty} e^{-at}\cos(\Omega_1 t)\, e^{-j\Omega t} dt$$

$$= \int_0^{\infty} e^{-at}\cos(\Omega_1 t)\, e^{-j\Omega t} dt$$

$$= \frac{(a+j\Omega)}{{\Omega_1}^2 + (a+j\Omega)^2}$$

Plots in Matlab

Fix Ω_1, change a. The diagram of $x_c(t)$ and $X_c(j\Omega)$ showed as below:



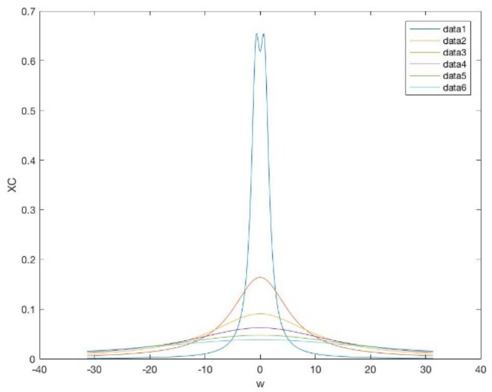
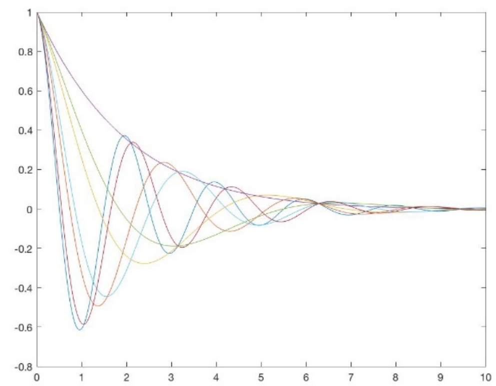


Figure: signal in time domain

Figure: signal in frequency domain

Fix a, changeΩ_1. The diagram of $x_c(t)$ and $X_c(j\Omega)$ showed as below:



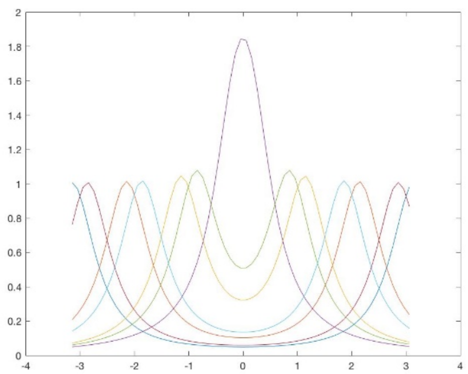


Figure: signal in time domain

Figure: signal in frequency domain

From the diagram, it can be observed that if keep the value of Ω_1, then changing a, for $x_c(t)$, the rise time and set time are changed with a, for $X_c(j\Omega)$, the peak value increases with a, but the number of peak decreases as a goes up. While if $\Omega_1$is changing, a is a constant.

From the graphs, they show that there is more vibration before the curve coms to a stable value in $x_c(t)$. Meanwhile, with the changing of Ω_1, the number of peak decreases as the value of Ω_1 close to 0, while the curve becomes lower with the increasing of Ω_1.

The code in Matlab to get the diagram

Fix Ω_1 , change a:

```
clc
clear all

%a = 0.12;

for a = (1:5:30);
   OMG = 0.25*pi;
   %xc = exp(-a.*t).*cos(OMG1.*t);
   t = (0:0.01:10);
   w=(-10*pi:0.1:10*pi);
   xc = exp(-a.*t).*cos(OMG.*t);
   XC = (a+1i*w)./((a+1i*w).^2+ OMG.^2);


   figure(1);
   plot(t, xc);
   xlabel('t');
   ylabel('xc')
   hold on

   figure(2);
   plot(w, XC);
   xlabel('w');
   ylabel('XC')
   hold on

end
```

Fix , changeΩ_1:

```
clc
clear all

a = 0.5;

for OMG = -pi:pi
   %OMG = 0.25*pi;
   %xc = exp(-a.*t).*cos(OMG1.*t);
   t  = [0:0.01:10];
   xc = exp(-a.*t).*cos(OMG.*t);
   figure(1);
   plot(t, xc);
   hold on

   figure(2);
   w = [-pi:0.1:pi];
   XC = (a+1i*w)./((a+1i*w).^2+ OMG.^2);
   plot(w, XC);
   hold on

end
```

b)

If $x[n] = x_c(nT) = \begin{cases} e^{-anT}\cos(\Omega_1 nT), nT \geq 0, a > 0. \\ 0, \qquad\qquad\qquad\qquad nT < 0. \end{cases}$

$$X(e^{j\omega}) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\omega}$$

$$= \sum_{-\infty}^{\infty} e^{-anT}\cos(\Omega_1 nT)e^{-j\omega n}$$

DTFT of e^{-anT} is

$$\sum_{-\infty}^{\infty}(e^{-(aT+j\omega)})^n = \frac{1}{1-e^{-aT}e^{-j\omega}}$$

If $\alpha = e^{-aT}$,

DTFT of e^{-aT} is $\frac{1}{1-\alpha e^{-j\omega}} = X_1(e^{j\omega})$

$$\therefore X\left(e^{j\omega}\right) = \frac{1}{2}[X_1\left(e^{j(\omega+\Omega_1 T)}\right) + X_2(e^{j(\omega-\Omega_1 T)})]$$

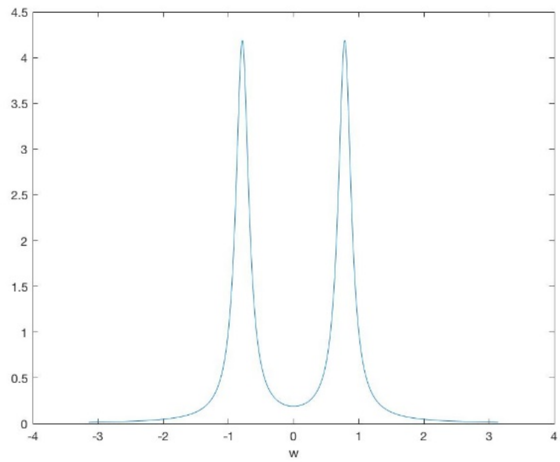
$$= \frac{1}{2}\left[\frac{1}{1-e^{-aT}e^{-j(\omega+\Omega_1 T)}} + \frac{1}{1-e^{-aT}e^{-j(\omega-\Omega_1 T)}}\right]$$

$$= \frac{1}{2}\left[\frac{2-e^{-aT-j(\omega+\Omega_1 T)} - e^{-aT+j(\omega-\Omega_1 T)}}{1-e^{-aT}e^{-j(\omega+\Omega_1 T)} * 1-e^{-aT}e^{-j(\omega-\Omega_1 T)}}\right]$$

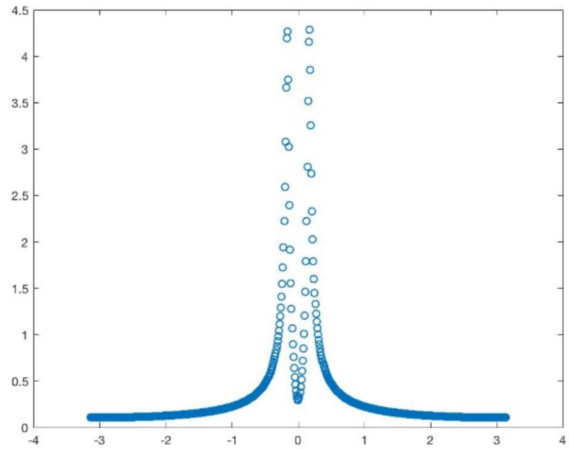
$$= \frac{\mathbf{1} - \boldsymbol{e^{-aT}cos(\Omega_1 T)}e^{-j\omega}}{\mathbf{1} - \mathbf{2}\boldsymbol{e^{-aT}cos(\Omega_1 T)}e^{-j\omega} + \boldsymbol{e^{-2aT}}e^{-2j\omega}}$$

c)

The diagram of $X_c(j\Omega)$



The diagram of $X(e^{j\omega})$



The command **freqz ,** [f w] = freqz (A , B , w), returns the frequency response vector f, and the corresponding angular frequency vector w, for the digital filter with numerator and denominator polynomial coefficients stored in B and A, respectively.

The code in Matlab to plot $X(e^{j\omega})$

```
%XC after sampling is
clear all
clc
close all

a = 0.12;
OMG = 0.25*pi;
T = 1/4.8;
%fundamental frequency range
w = -pi:0.01:pi;
XC = (a+1i*w)./((a+1i*w).^2+ OMG.^2);%continues time FT
%abs(xc) only plot the magnetude

%NOW plot the sampled DTFT
A = [1 -exp(-a*T)*cos(OMG*T) 0];
B = [1 -2*exp(-a*T)*cos(OMG*T) exp(-2*a*T)];
[f w] = freqz (A , B , w);
hold on;
%w/T=continuous time frequency
%abs(f)*T
figure(2);
plot ( w, abs(f)*T, 'o');
hold on

end
```

d) Stability of the filter

Proof:

If $H_{LP}(Z) = \frac{1-\alpha}{2}\frac{1+Z^{-1}}{1-\alpha Z^{-1}}$

$$H_{LP}(e^{j\omega}) = \frac{1-\alpha}{2}\frac{1+e^{-j\omega}}{1-\alpha e^{-j\omega}}$$

From Euler Formula $H_{LP}(e^{j\omega}) = \frac{1-\alpha}{2}\frac{1+\cos\omega - j\sin\omega}{1-\alpha\cos\omega+\alpha j\sin\omega}$

$$|H_{LP}(e^{j\omega})| = \frac{1-\alpha}{2}\frac{\sqrt{(1+\cos\omega)^2+\sin^2\omega}}{\sqrt{(1-\alpha\cos\omega)^2+\alpha^2\sin^2\omega}}$$

$$= \frac{1-\alpha}{2}\frac{\sqrt{2+2\cos\omega}}{\sqrt{1+\alpha^2-2\alpha\cos\omega}}$$

$$|H_{LP}(e^{j\omega})|^2 = \frac{(1-\alpha)^2}{2}\frac{1+\cos\omega}{1+\alpha^2-2\alpha\cos\omega}$$

$$= \frac{(1-\alpha)^2(1+\cos\omega)}{2(1+\alpha^2-2\alpha\cos\omega)}$$

Focus on the stability

Because $H_{LP}(Z) = \frac{1-\alpha}{2}\frac{1+Z^{-1}}{1-\alpha Z^{-1}}$

$$= \frac{1-\alpha}{2}\frac{Z+1}{Z-\alpha}$$

The only pole of the signal is $Z = \alpha$;

So $|\alpha| \leq 1$ and $1-\alpha \neq 0$;

$\therefore -1 < \alpha < 1$

$$|H_{LP}(e^{j\omega})| = \frac{1-\alpha}{2}\frac{\sqrt{2+2\cos\omega}}{\sqrt{1+\alpha^2-2\alpha\cos\omega}}$$

$$\angle H_{LP}(e^{j\omega}) = \cot\left(-\frac{\sin\omega}{1+\cos\omega}\right) - \cot\left(\frac{\alpha\sin\omega}{1-\alpha\cos\omega}\right)$$

Plot $|H_{LP}(e^{j\omega})|$ and $\angle H_{LP}(e^{j\omega})$

Diagram of $|H_{LP}(e^{j\omega})|$

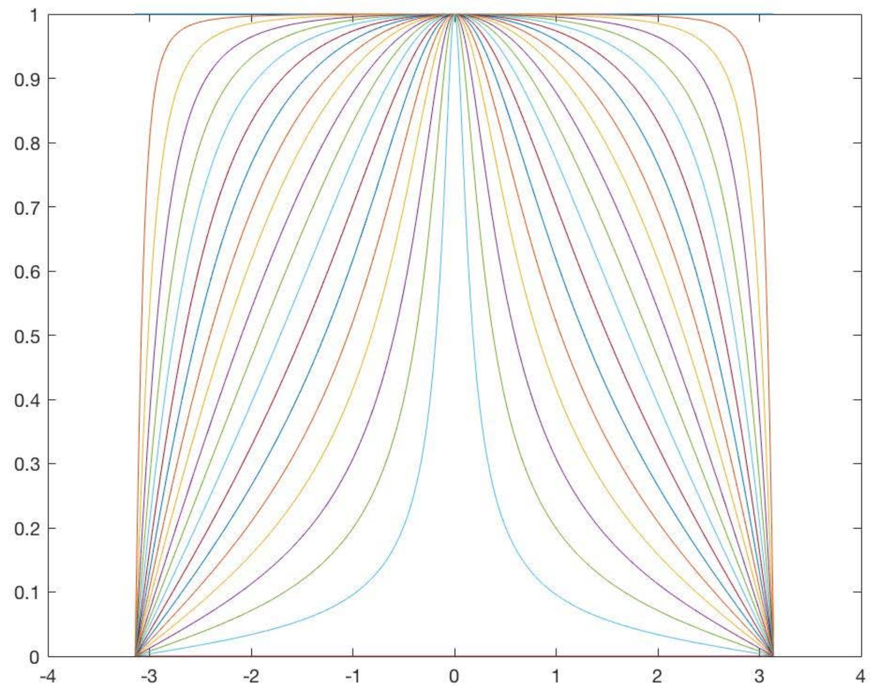
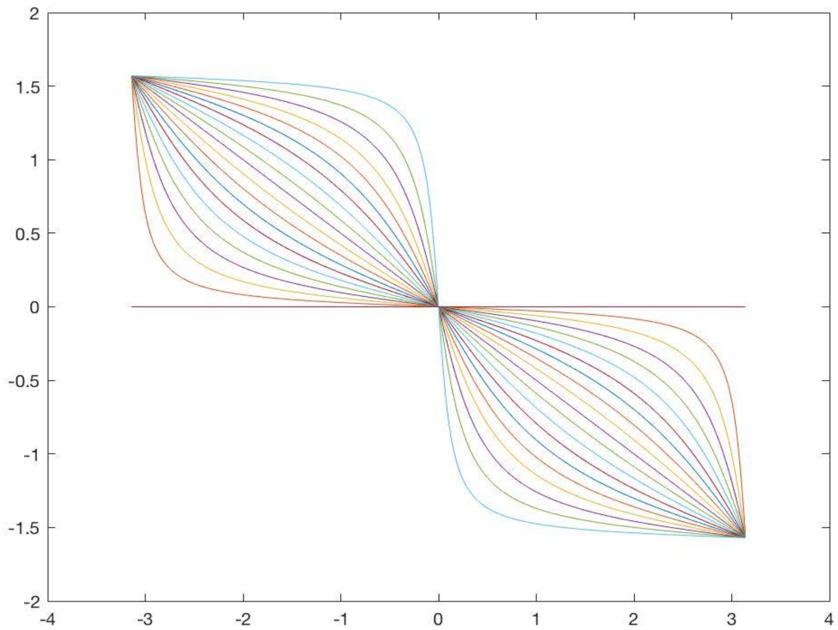


Diagram of $\angle H_{LP}\left(e^{j\omega}\right)$



From the diagram above it can be observed that as the value of α changing from -1 to 1, the peak of diagram of $\left|H_{LP}(e^{j\omega})\right|$ does not change, but the value of [-1,0] and [0,1] are closer to 0 as the increasing of α.

When talking to diagram of $\angle H_{LP}\left(e^{j\omega}\right)$, if the value of $\alpha = 0$, then the curve of $\angle H_{LP}\left(e^{j\omega}\right)$ is just like the function y=-x. When $-1 < \alpha < 0$, the relative curves of $\angle H_{LP}\left(e^{j\omega}\right)$ are all below the curve which has $\alpha = 0$. As the α becomes smaller, the curves are closer to real axis. When the $0 > \alpha > 1$, the relative curves are all above the curve which has $\alpha = 0$. With the increasing of value of α, these curves are closer to the imaginary axis.

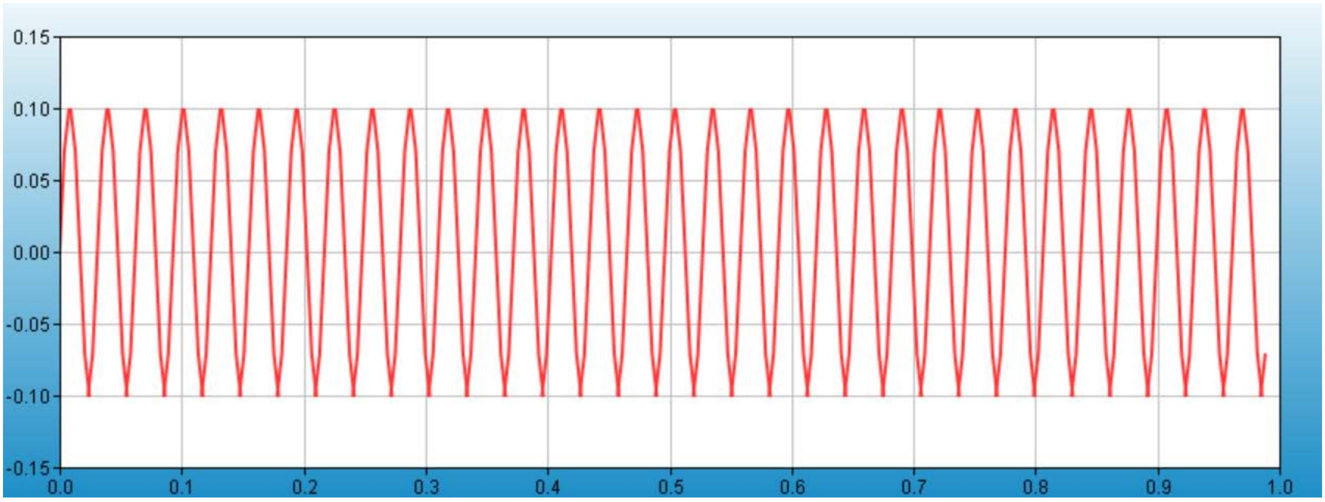
The code in Matlab to plot the diagram of $\left|H_{LP}(e^{j\omega})\right|$ and $\angle H_{LP}\left(e^{j\omega}\right)$

```
clear all
clc

for a = -1:0.1:1;
   H = zeros(0, 629);
   i = 1;
   for w = -pi :0.01: pi
      z = exp(j*w);
      H(i) = (1 - a) / 2 * (1 + 1/z) / (1 - a/z);
      i = i+1;
   end
    figure(1);
    plot(-pi :0.01: pi, abs(H));
    hold on;

   figure(2);
   plot(-pi :0.01: pi, angle(H));
   hold on;
```

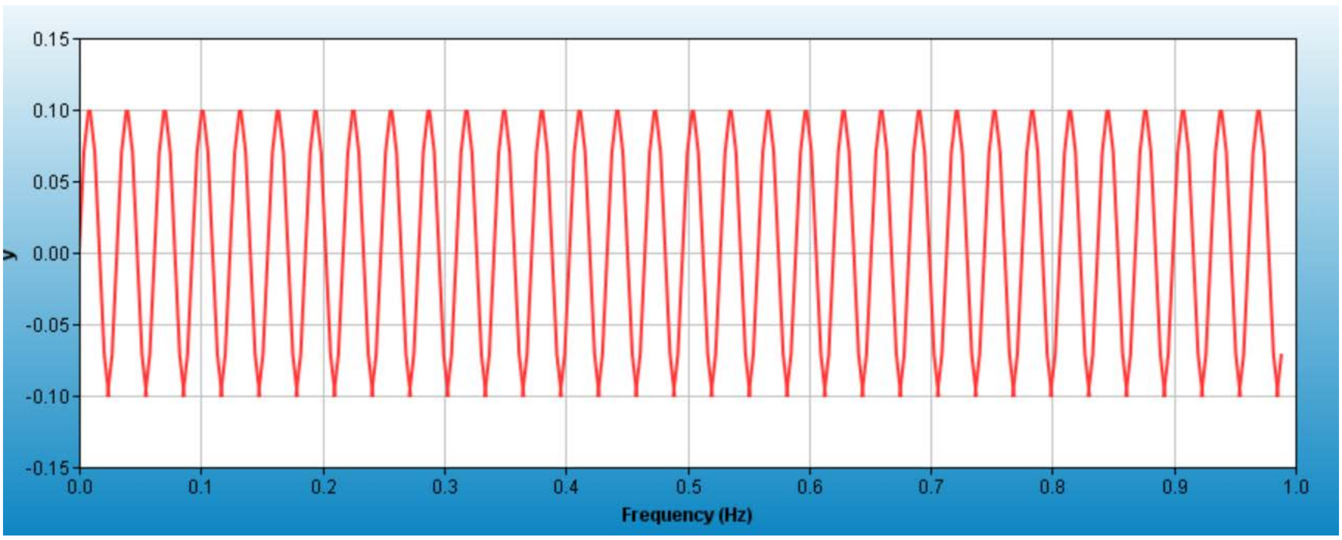
B. Implementation of digital anti-aliasing filters on a DSP

a) Plot of the sample signal x(n) = 0.1 * sin(0.25 * 1.0 * n);

The Time domain representation of the signal has been plotted in CCES as the figure below, where its x axis is time(s) and y axis is amplitude.



The frequency domain representation of the signal is shown in the figure below, where the x axis is frequency(Hz), y axis is amplitude.



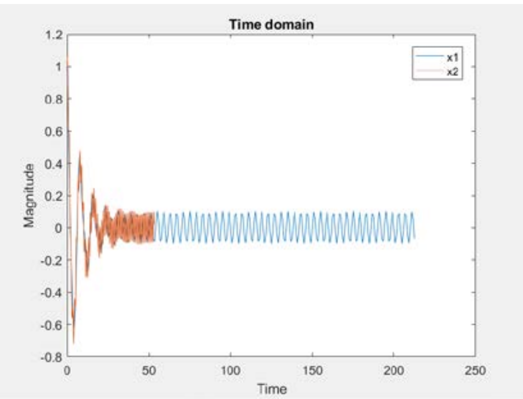
b) **Write a c-program which generates two sequences of 256 samples of x(t), using the sampling frequencies F1 = 1.2Hz and F2 = 4.8Hz.**

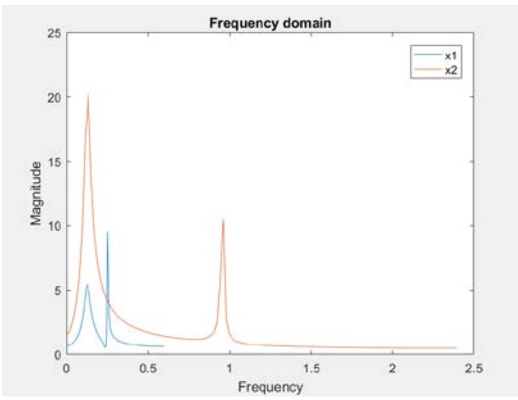
The code used to generate the 2 signals are attached below.

```
#include <stdio.h>
#include <math.h>

// Globals
#define N       256
#define         PI      3.1415

float x1[N]; // Sample at T1
float x2[N]; // Sample at T2
int main(void)
{
        int i;
        float omega1 = 0.25 * PI, omega2 = 1.9 * PI;
        float T2 = 1/4.8;
        float T1 = 1/1.2;
        float a=0.12;
        float alpha1 = 0.593, alpha2 = 0.464;

        x1[0]  = exp(-a*0*T1)*cos(omega1*0*T1) + 0.1*sin(omega2*0*T1);
        x2[0]  = exp(-a*0*T2)*cos(omega1*0*T2) + 0.1*sin(omega2*0*T2);

        for (i = 0; i < N; i++)
        {
                x1[i] = exp(-a*i*T1)*cos(omega1*i*T1) + 0.1*sin(omega2*i*T1);
                x2[i] = exp(-a*i*T2)*cos(omega1*i*T2) + 0.1*sin(omega2*i*T2);

        }
        printf("Done.\n");
        return 0;
}
```

c) **Use the plot facility within CCES to plot the sampled signals in the time domain and in the frequency domain. Comment on the results.**

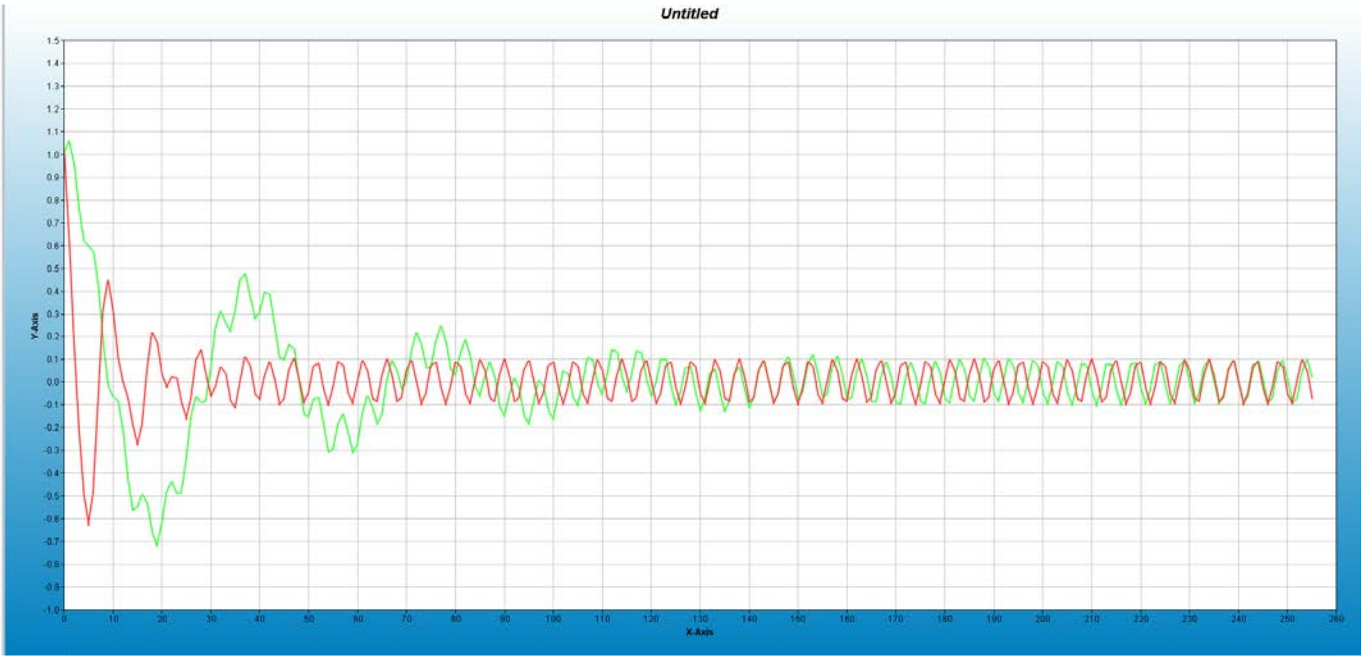
To predict how the signal is going to behave, Matlab was used to plot x1 and x2 in time and frequency domain. (Code attached in appendix).



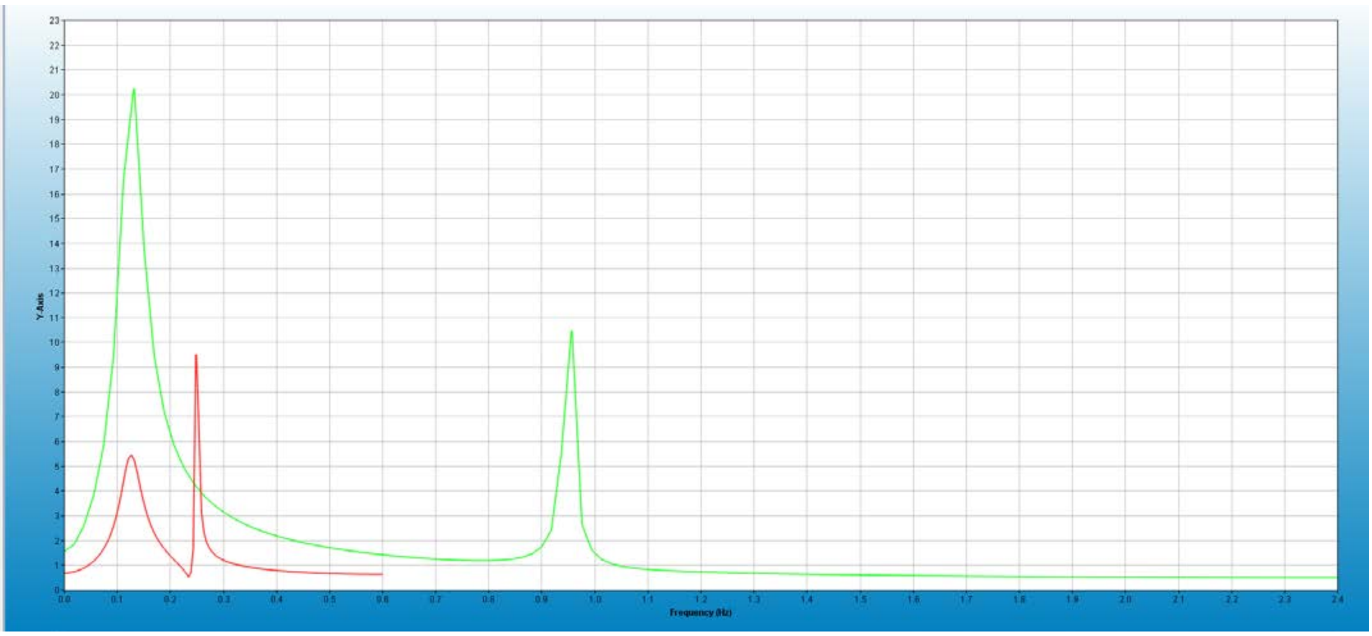


The result generated by CESS are shown as below

Time domain plot. x axis: sample time, y axis: magnitude(linear scale)



Frequency domain plot. x axis: frequency(Hz), y axis: magnitude(linear scale)



As the plots above show, the signal generated in the DSP board are identical to that simulated in Matlab.

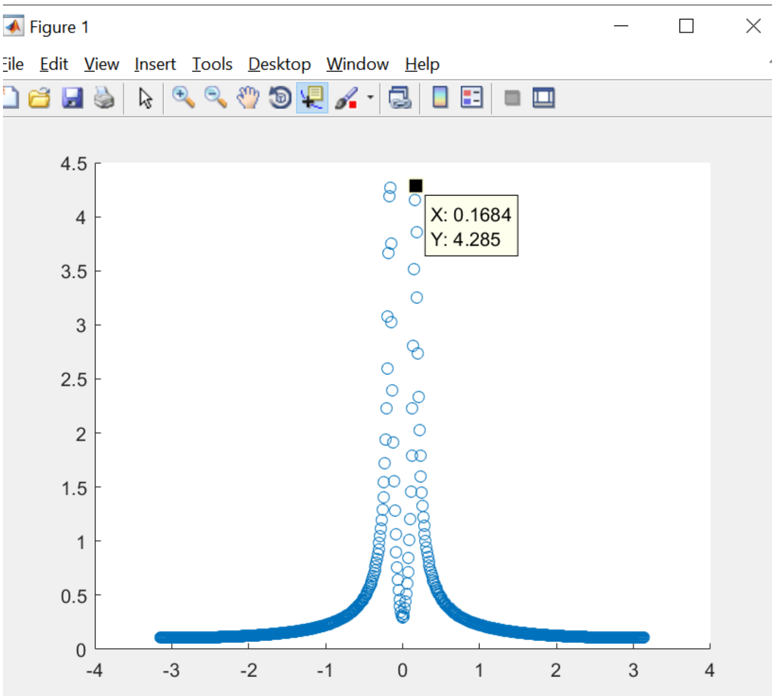
Comment on the result:

Theoretically speaking, the signal should generate two peaks at $\boldsymbol{f1} = \frac{\Omega_1}{2*pi} = \frac{0.25*pi}{2*pi} = 0.125Hz$, and $\boldsymbol{f}_2 = \frac{\Omega_2}{2*pi} = \frac{1.9*pi}{2*pi} = 0.95Hz$. Which is matching up with what signal x2(the green trace) in the CCES FFT plot.

However, there are two facts about the signal x1's FFT plot were noticed in the FFT plot: 1. The 0.95Hz peak was missing, 2. There is an "unpredicted" peak between the region of 0.2-0.3Hz. Fact 1 might due to the sampling angular velocity $\Omega_{sampling1} = 1.2 * 2\text{pi} < 2\Omega_2 = 2 * 1.9 * 2pi$. When the sampling frequency is less than $2\Omega_{max}$ of the signal, high frequency component of the signal will be distorted while sampling. Fact 2 is due to the fact that the sampling frequency is too low and it cause the folding effect around $\frac{f_{samling}}{2} = \frac{1.2}{2} = 0.6Hz$. The frequency component from $f_2 = \frac{\Omega_2}{2*pi} = \frac{1.9*pi}{2*pi} = 0.95Hz$ is distorted and folded back to 0.6-(0.95-0.6) = 0.25Hz, which fell into the 0.2-0.3Hz region we mentioned in the last paragraph.

d) Analysis of the first order and second order filter

After DTFT, the signal $x_1 = e^{-at}\cos(\Omega_1 t)\,, where\ a = 0.12, \Omega_1 = 0.25pi$, has been plotted in frequency domain in question part A(c), and $\omega_{max} = 0.1684\ rad/s$ can be obtained from the plot.



(i) Find $\boldsymbol{\alpha}$ of the first order filter

It has been proven that

$$|H_{LP}(e^{j\omega})|^2 = \frac{(1-\alpha)^2(1+\cos\omega)}{2(1+\alpha^2-2\alpha\cos\omega)}$$

The gain(magnitude)=0.95 of this filter at $\omega_{max} = 0.1684\ rad/s$,

can be calculated by

$$|H_{LP}(e^{j\omega})| = \sqrt{|H_{LP}(e^{j\omega})|^2} = \frac{1-\alpha}{\sqrt{2}} * \frac{\sqrt{1+\cos(w_{max})}}{\sqrt{1+\alpha^2-2\alpha\cos(w\_max)}} = 0.95$$

Solving this equation at Wolfram Alpha gives us $\alpha = 0.593$.

As the filter is designed to filter out 5% of the gain of x1, now we need to evaluate the gain of the signal after the filter.

Recall in discrete frequency domain, there will be a peak at $w_2 = \Omega_2 * T_{4.8Hz} = 1.9pi * \frac{1}{4.8} = 1.2435$ rad/s.

$$|H_{LP}(e^{j\omega_2})| = \frac{1-\alpha}{\sqrt{2}} * \frac{\sqrt{1+\cos(w_2)}}{\sqrt{1+\alpha^2 - 2\alpha\cos(w_2)}} = \frac{1-0.593}{\sqrt{2}} * \frac{\sqrt{1+\cos(1.2435)}}{\sqrt{1+0.593^2 - 2*0.593*\cos(1.2435)}}$$
$$= 0.3385$$

As the requirement of the filter is to get a gain less than 0.25 at ω_2, the calculation above has shown that this filter won't satisfy the design requirement.

(ii) Find $\boldsymbol{\alpha}$ of the second order filter

Since $H_2(e^{j\omega}) = H_{LP}(e^{j\omega})^2$, $H_2(e^{j\omega}) = |H_{LP}(e^{j\omega})|^2 = \frac{(1-\alpha)^2(1+\cos\omega)}{2(1+\alpha^2-2\alpha\cos\omega)}$

The gain(magnitude)=0.95 of this filter at $\omega_{max} = 0.1684\ rad/s$,

can be calculated by

$$|H_2(e^{j\omega})| = \frac{(1-\alpha)^2}{2} * \frac{1+\cos(w_{max})}{1+\alpha^2 - 2\alpha\cos(w\_max)} = 0.95$$

Solving this equation at Wolfram Alpha gives us $\alpha = 0.464$.

As the filter is designed to filter out 5% of the gain of x1, now we need to evaluate the gain of the signal after the filter.

$$|H_2(e^{j\omega_2})| = \frac{(1-\alpha)^2}{2} * \frac{1+\cos(w_2)}{1+\alpha^2-2\alpha\cos(w_2)} = \frac{(1-0.464)^2}{2} * \frac{1+\cos(1.2435)}{1+0.464^2 - 2*0.464*\cos(1.2435)} = 0.207$$

Gain = 0.207 is less than 0.25 therefore this filter will satisfy the design specification.

e) Implementation of the two filters in C

Firstly, we get the time domain of the filters.

For the first order filter,

We get

$$\mathrm{Y(z) = X(z) \cdot H(z)}$$
$$\mathrm{Y(z) - Y(z)}\frac{\alpha}{z} = X(z) \cdot \frac{1-\alpha}{2} \cdot (1 + \frac{1}{z})$$
$$\mathrm{Y(z) - Y(z)} \cdot \frac{\alpha}{z} = \frac{1-\alpha}{2}X(z) + \frac{1-\alpha}{2} \cdot \frac{X(z)}{z}$$
$$\mathrm{Y(z) - \alpha Y(z) \cdot z^{-1}} = \frac{1-\alpha}{2}X(z) + \frac{1-\alpha}{2}X(z)z^{-1}$$

Do inverse z transform on both sides, we get:

$$\mathrm{y_1[n] - \alpha y_1[n-1]} = \frac{1-\alpha}{2}x[n] + \frac{1-\alpha}{2}x[n-1]$$
$$\mathrm{y_1[n]} = \frac{1-\alpha}{2}x[n] + \frac{1-\alpha}{2}x[n-1] + \mathrm{\alpha y_1[n-1]}$$

For the second order filter,

We get

$$\mathrm{Y(z) = X(z) \cdot H^2(z)}$$
$$\mathrm{Y(z) = [X(z) \cdot H(z)] \cdot H(z)}$$

We notice that $\mathrm{X(z) \cdot H(z)}$ is the signal out of first order filter, so if we use $\mathrm{y_1[n]}$ replace of x[n], we get

$$\mathrm{y_2[n]} = \frac{1-\alpha}{2}\mathrm{y_1[n]} + \frac{1-\alpha}{2}\mathrm{y_1}[n-1] + \mathrm{\alpha y_2[n-1]}$$

i)

For the signal x[n], x[n]=x(nt)=$x_1(\mathrm{nt}_1) + x_2(nt_2)$=$e^{-0.12t} \times \cos(\omega_1 \mathrm{nt}) + 0.1 \times \sin(\omega_2 \mathrm{nt})$

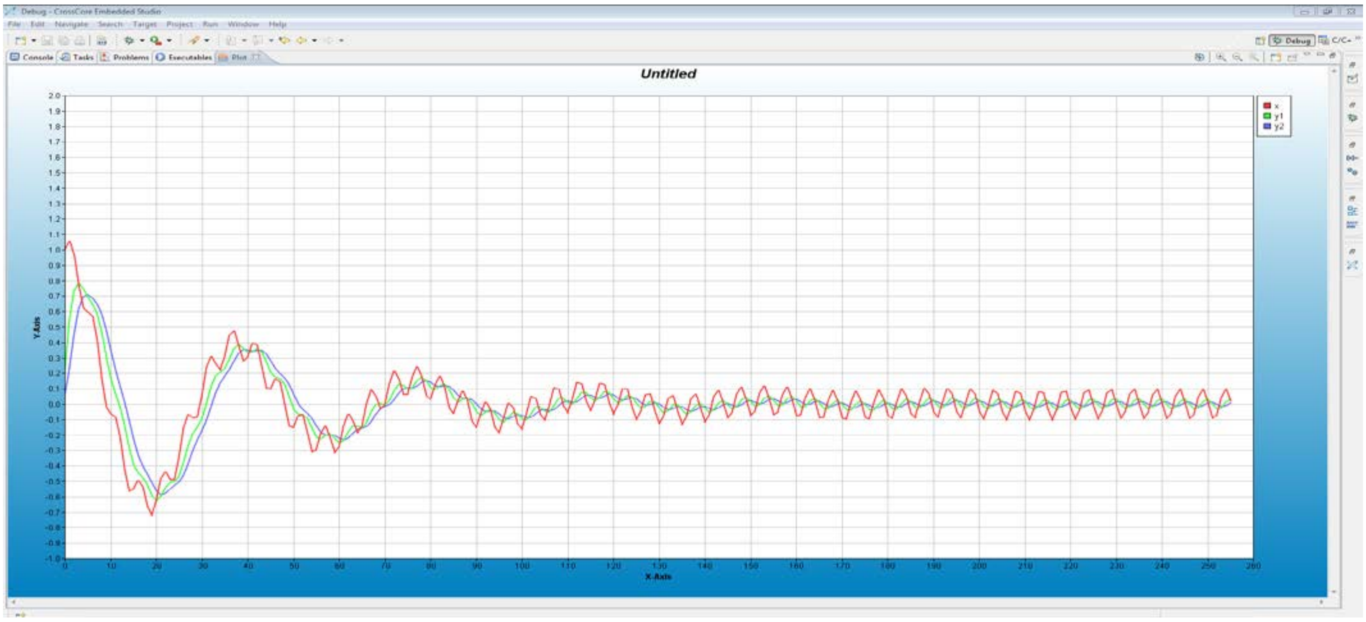
=$e^{-0.12t} \times \cos(0.25\pi \mathrm{nt}) + 0.1 \times \sin(1.9\pi \mathrm{nt})$

As we know:t=1/4.8

X[n]= $e^{-0.025} \times \cos(0.052\pi \mathrm{n}) + 0.1 \times \sin(0.396\pi \mathrm{n})$

After the signal go through two kinds of low pass filters, we get:

Time domain

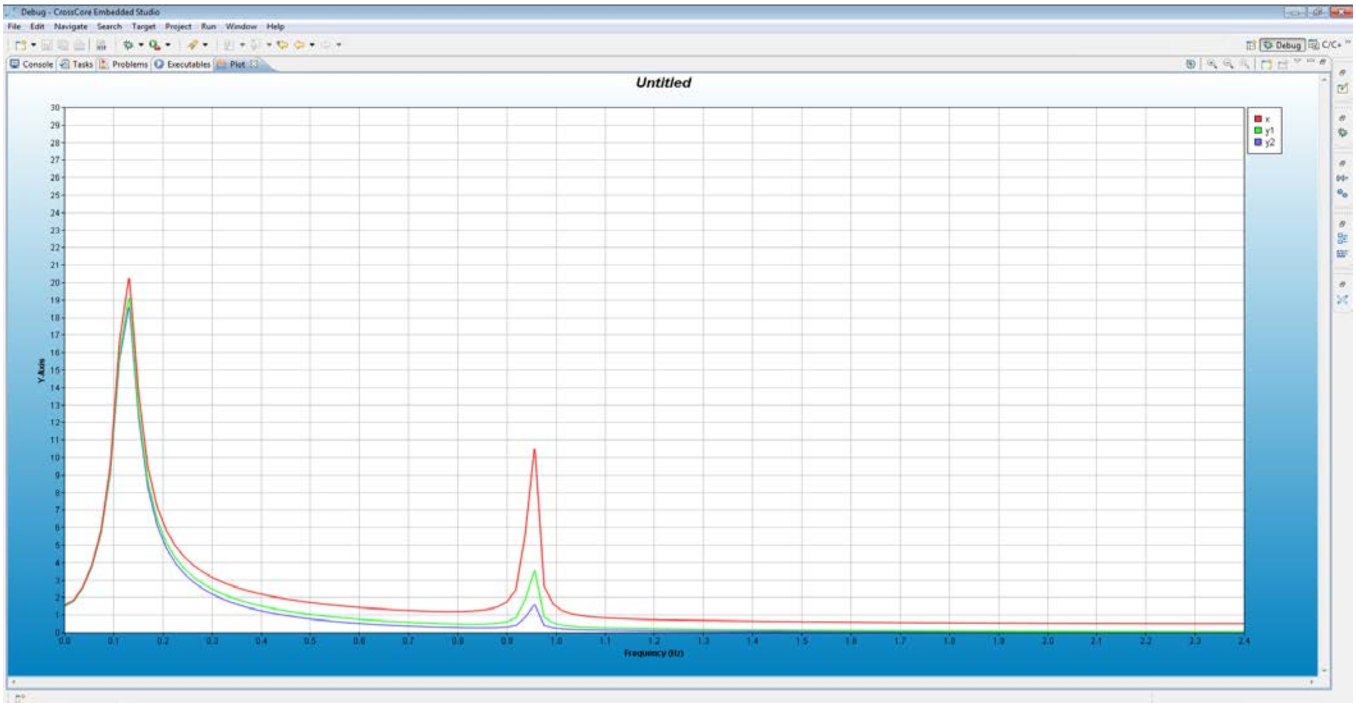


From the time domain plot, we checked that the gain of the implemented filters agrees with the gain of the designed filters at the frequency of the x_2[n] signal. When the signal is settled, the gain$\approx 0.3 \div 2.0 = 0.15$, which is lower than 0.25.

In the two images above, x (the red one) means x[n], the sequence without filters, y1 (the green one) means signal go through first order filter, and y2 (the blue one) means signal go through second order filter. From the diagram, we can see at the frequency of 0.95Hz,

the gain=1.7÷10.5=0.162, which is lower than 0.25, thus it suits the filter we designed.

Frequency domain

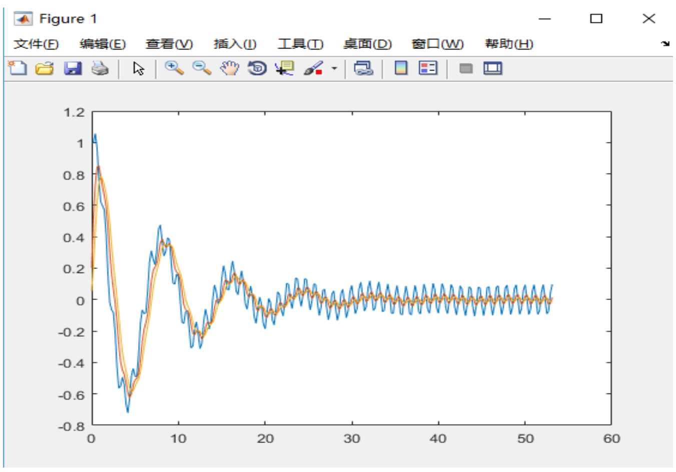


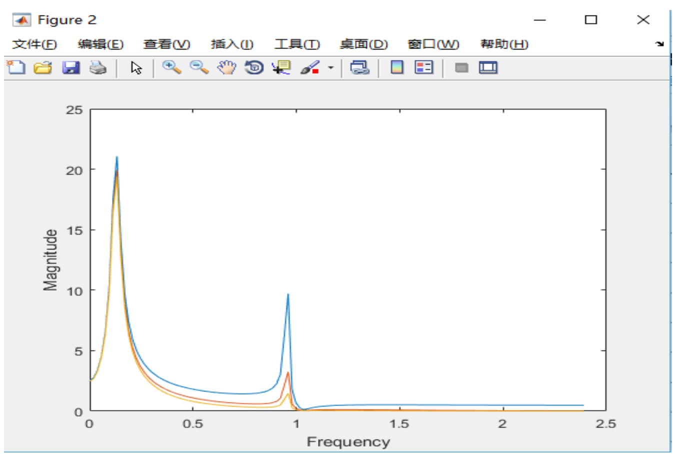
ii)

As seen from the frequency plot, at $f_{\max} = \frac{0.25pi}{2pi} = 0.125Hz$, the original signal has magnitude of 20, and the signal after the first order signal has magnitude of 19 and that in the second order filter has the same performance, $\frac{19}{20} * 100\% = 95\%$, both filter is satisdying the requirement which retain 95% of the magnitude of the signal.

Whereas at $f_2 = 0.95Hz$, the original signal has magnitude of 10.2, and the signal after the first order signal has magnitude of 3.2, which is $\frac{3.3}{10.2}\ \% =\ 32.3\%$, dissatisfying the design requirement, and that in the second order filter is $\frac{1.7}{10.2} * 100\% = 16.7\% < 25\%$.

We use Matlab to stimulate the signal before implementing on the DSP board. The code and the figures are showed below.



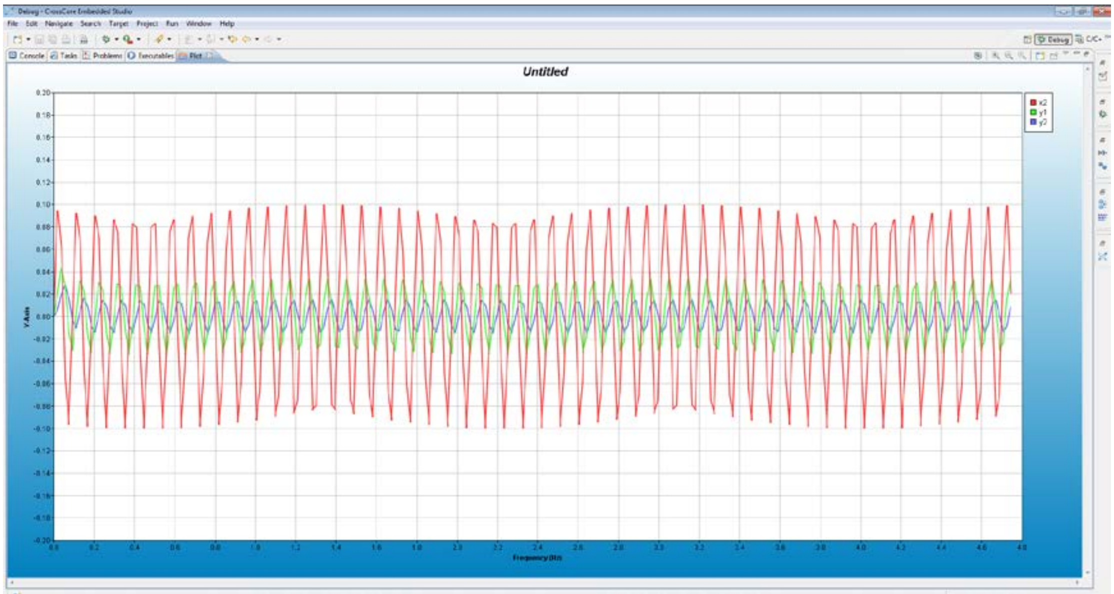


Time domain

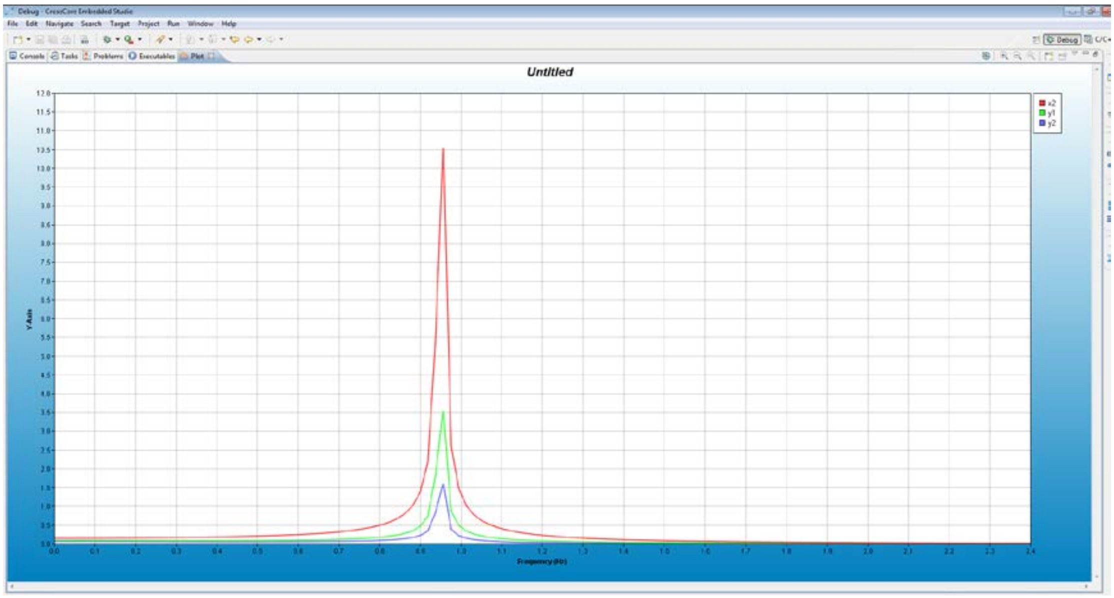
Frequency domain

f)

After signal $x_2[n]$ go through two kinds of filters, we get:



Time domain



Frequency domain

In the two images above, x (the red one) means $x_2[n]$, the sequence without filters, y1 (the green one) means signal go through first order filter, and y2 (the blue one) means signal go through second order filter. From the diagram, we can see at the frequency of 0.95Hz,
the gain=$\frac{1.7}{10.2} * 100\% = 16.7\%$ after the second order filter, which is lower than 0.25, thus it suits the filter we designed. However, the first order filter has percentage gain of $\frac{3.3}{10.2}\ \% =\ 32.3\%$. Moreover, After the signal go through the second order filter, it has 90 degrees delay.

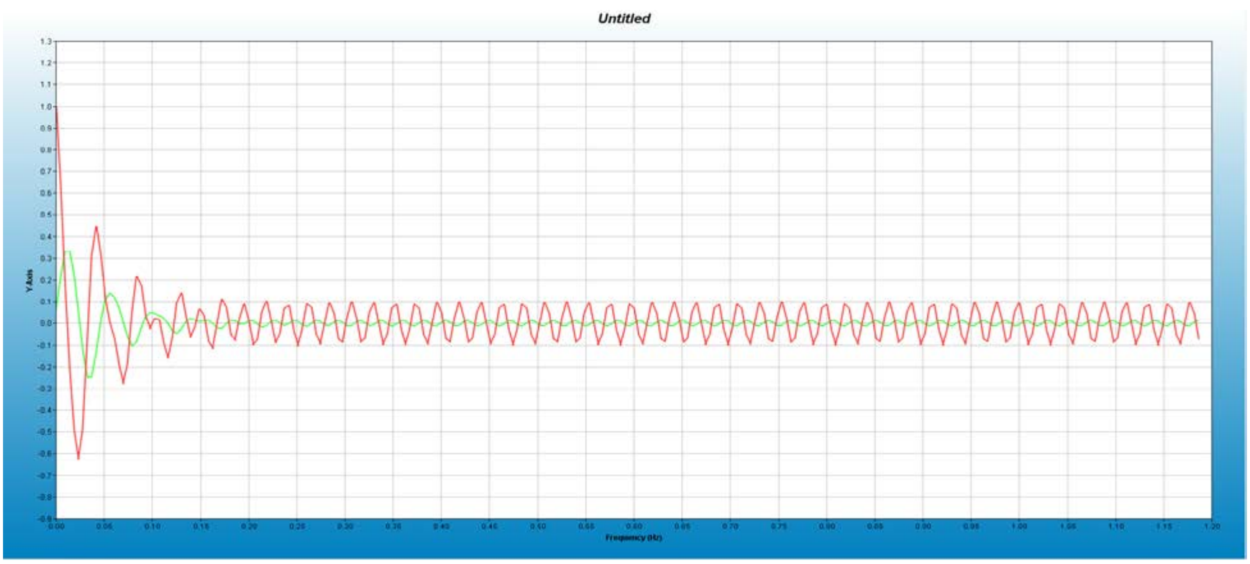
The code is attached below.

```
#include <math.h>
#include <stdio.h>
// Globals
#define N      256
```

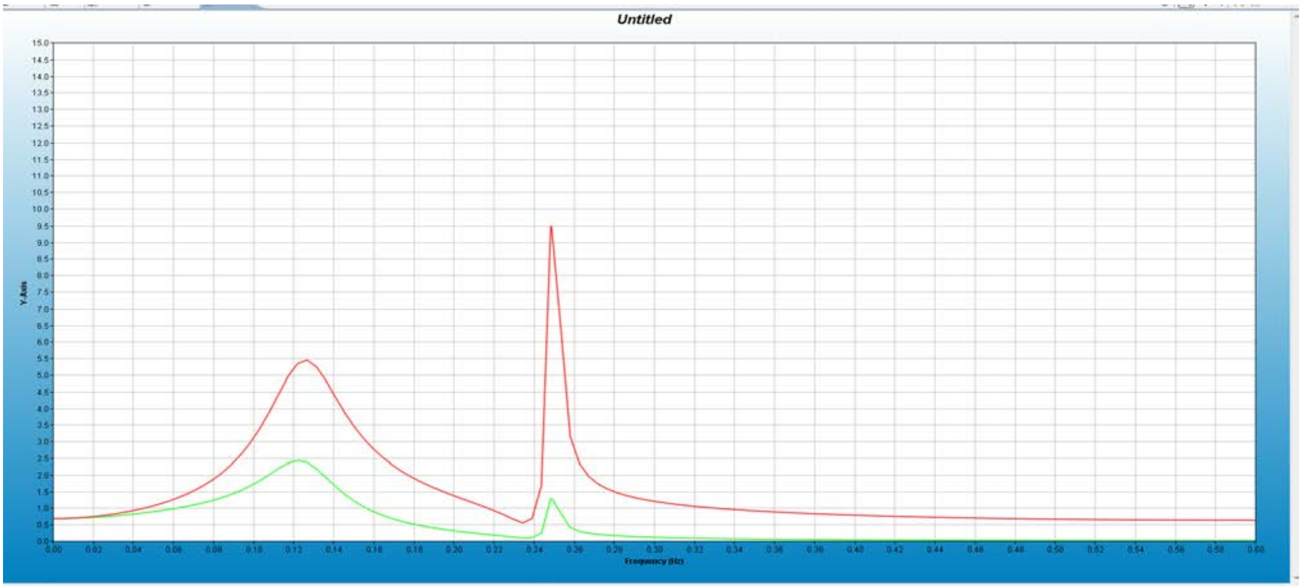
```
#define PI      3.1415
float x[N];
float x2[N];
float y1[N];
float y2[N];
int main(void)
{
        int i;
        float omega1 = 0.25 * PI, omega2 = 1.9 * PI;
        float T2 = 1/4.8;
        float T = 1.0;
        float a=0.12;
        float alpha1 = 0.593, alpha2 = 0.464;
        //x[0]  = exp(-a*0*T2)*cos(omega1*0*T2) + 0.1*sin(omega2*0*T2);
        x2[0]  = 0.1*sin(omega2*0*T2);
        y1[0] = ((1-alpha1)/2)*x2[0];
        y2[0] = ((1-alpha2)/2)*y1[0];
        for (i = 0; i < N; i++)
        {
                //x[i] = exp(-a*i*T2)*cos(omega1*i*T2) + 0.1*sin(omega2*i*T2);
                x2[i]  = 0.1*sin(omega2*i*T2);
                y1[i] = ((1-alpha1)/2)*x2[i]+((1-alpha1)/2)*x2[i-1]+alpha1*y1[i-1];
        y2[i] = ((1-alpha2)/2)*y1[i]+((1-alpha2)/2)*y1[i-1]+alpha2*y2[i-1];
        }
        /*for (i = 0; i < N; i++)
        {
                printf("x[%d] = %f\n", i, x[i]);
        }
*/
        printf("Done.\n");
        return 0;
}
```

g)

After the signal go through the second order low pass filter, we get:



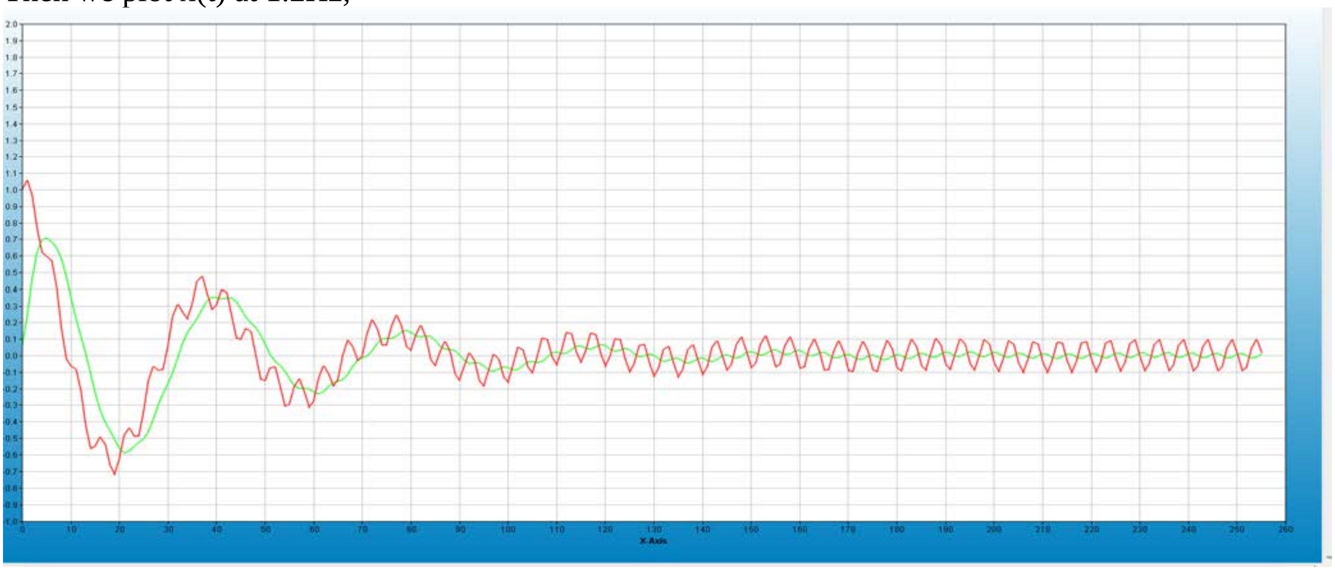
Time domain



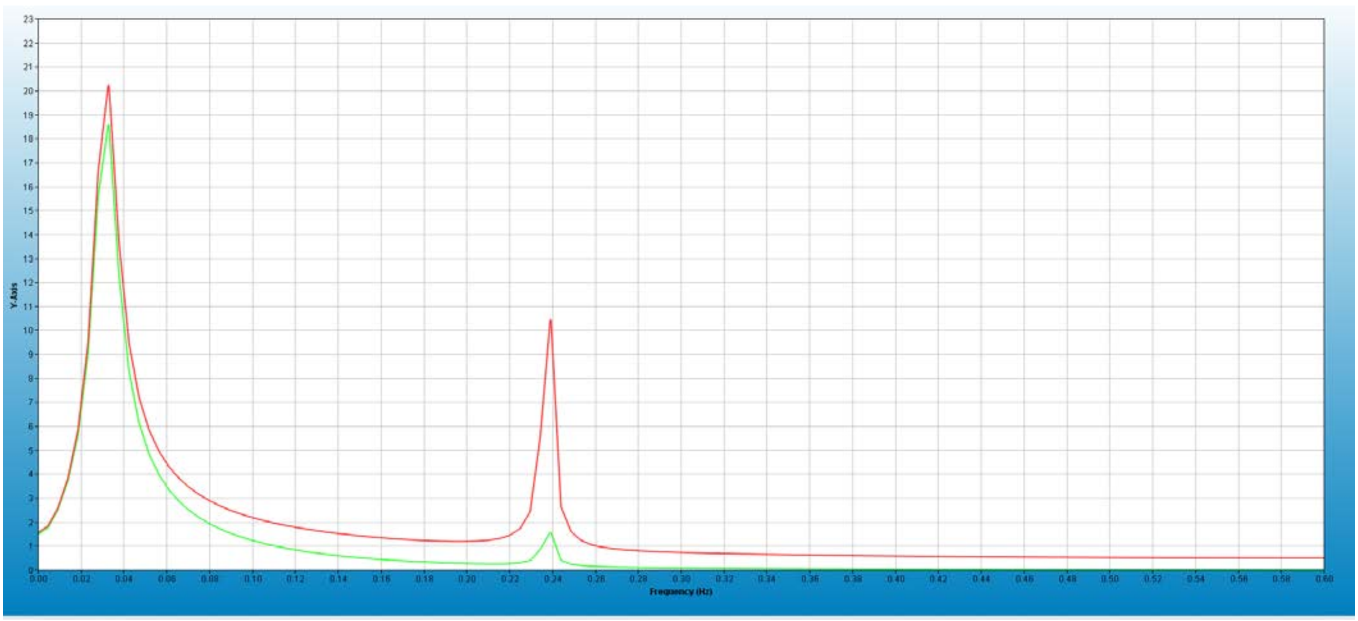
Frequency domain

In the two images above, x (the red one) means x[n], the sequence without filters, y1 (the green one) means signal go through the second order filter.

Then we plot x(t) at 1.2Hz,



Time domain



Frequency domain

First we analyse from time domain. Both of the signals attenuate after it goes through the filter. However, for the signal y[n], it attenuates very quickly. While the signal x[n] attenuates slowly.

For frequency domain, before y[n] goes through the filter, its main frequency is 0.25Hz, and another lower peat is at 0.13Hz. After y[n] goes through the filter, it has two main frequency peaks, at 0.13Hz and 0.25Hz. At 0.25Hz, the gain =2.3÷19.0=0.121; Then at 0.13Hz, the gain =5.0÷11.0=0.455. We can see that at both frequencies the signal attenuates, especially 0.25Hz.
For x(t), the frequency peaks are at 0.03Hz and 0.24Hz.when it goes through the second order filter, at 0.24Hz, the gain= 1.5÷10.5=0.143; at 0.03Hz, the gain=18.5÷20.3=0.911. From the analysis we see that the filter act well when signal is x(t).

The code of how to generate and plot y[n] is attached below.

```
#include <math.h>
#include <stdio.h>
#include <complex.h>
// Globals
#define N      256
#define PI     3.1415
float x[N];
float x2[N];
float y1[N];
float y2[N];
float n1[N];
int main(void)
{
       int i;
       float omega1 = 0.25 * PI, omega2 = 1.9 * PI;
       float T2 = 1/4.8;
       float T = 1/1.2;;
       float a=0.12;
       float alpha1 = 0.593, alpha2 = 0.464;
       x[0]  = exp(-a*0*T2)*cos(omega1*0*T2) + 0.1*sin(omega2*0*T2);
       //x2[0]  = 0.1*sin(omega2*0*T2);
       y1[0] = ((1-alpha1)/2)*x2[0];
       y2[0] = ((1-alpha2)/2)*y1[0];
       for (i = 0; i < N; i++)
       {   n1[i]=4*i;
              x[i] = exp(-a*i*T2)*cos(omega1*i*T2) + 0.1*sin(omega2*i*T2);
              //x2[i]  = 0.1*sin(omega2*i*T2);
              y1[i] = ((1-alpha1)/2)*x[i]+((1-alpha1)/2)*x[i-1]+alpha1*y1[i-1];
       y2[i] = ((1-alpha2)/2)*y1[i]+((1-alpha2)/2)*y1[i-1]+alpha2*y2[i-1];
       }
       /*for (i = 0; i < N; i++)
       {
              printf("x[%d] = %f\n", i, x[i]);
       }
*/
       printf("Done.\n");
       return 0;
}
```

Appendix:

Matlab Code for Part B (c)

```
clc
clear all
close all

T1 = 1/1.2;
T2 = 1/4.8;
omega1 = 0.25 * pi;
omega2 = 1.9 * pi;
T = 1.0;
i = 0;
a = 0.12;
for n = 1:1:256
        x1(n) = exp(-a*i*T1)*cos(omega1*i*T1) + 0.1*sin(omega2*i*T1);
    x2(n) = exp(-a*i*T2)*cos(omega1*i*T2) + 0.1*sin(omega2*i*T2);
    i = i+1;
end


sample = 0:1:255;
plot(T1*sample, x1);

hold on
plot(T2*sample, x2);
%legend();
xlabel('Time');
ylabel('Magnitude');
legend({'x1','x2'});
title('Time domain');


figure;
fs1 = 1.2;
fs2 = 4.8;
N=255;
X1_mags = abs(fft(x1));
X2_mags = abs(fft(x2));
%Y1_mags = abs(fft(y1));
%Y2_mags = abs(fft(y2));
fax_bins = [0 : N-1]; %frequency axis in bins
N_2 = ceil(N/2);
plot(fax_bins(1:N_2)*fs1/N, X1_mags(1:N_2));
legend('x1');
hold on;
plot(fax_bins(1:N_2)*fs2/N, X2_mags(1:N_2));
legend('x2');
xlabel('Frequency');
ylabel('Magnitude');
legend({'x1','x2'});
title('Frequency domain');
```

Matlab code in question Part B(e).
Code:

```
clc
clear all
close all

T1 = 1/1.2;
T2 = 1/4.8;
omega1 = 0.25 * pi;
omega2 = 1.9 * pi;
T = 1.0;
i = 0;
a = 0.12;

alpha1 = 0.593;
alpha2 = 0.464;

x(1) = exp(-a*0*T2)*cos(omega1*0*T2) + 0.1*sin(omega2*0*T2);
y1(1) = ((1-alpha1)/2)*x(1);
y2(1) = ((1-alpha2)/2)*y1(1);
for n = 2:1:256
    %only use fs=4.8Hz
            x(n) = exp(-a*i*T2)*cos(omega1*i*T2) + 0.1*sin(omega2*i*T2);
    y1(n) = ((1-alpha1)/2)*x(n)+((1-alpha1)/2)*x(n-1)+alpha1*y1(n-1);
    y2(n) = ((1-alpha2)/2)*y1(n)+((1-alpha2)/2)*y1(n-1)+alpha2*y2(n-1);
    i = i+1;
end

sample = 0:1:255;
plot(T2*sample, x);
hold on
plot(T2*sample, y1);
plot(T2*sample, y2);


figure;
fs1 = 1.2;
fs2 = 4.8;
N=255;
X_mags = abs(fft(x))
Y1_mags = abs(fft(y1));
Y2_mags = abs(fft(y2));
fax_bins = [0 : N-1]; %frequency axis in bins
N_2 = ceil(N/2);
plot(fax_bins(1:N_2)*fs2/N, X_mags(1:N_2))
hold on;
plot(fax_bins(1:N_2)*fs2/N, Y1_mags(1:N_2))
plot(fax_bins(1:N_2)*fs2/N, Y2_mags(1:N_2))
xlabel('Frequency')
ylabel('Magnitude');
```