ELCE 705 Simulation Project 3

M-B5-5515-2 Tang Qi

Problem 1: Sampling Theorem

The following source code is used to sampling the input signal $x(t)=\sin(2\pi t)+ 0.3\sin(6\pi t)$ with the sampling frequency fs=10Hz. The sampled discrete-time signal is then used to reconstruct the continuous-time signal. The input, sampled signal and the reconstructed signal are shown in one figure.

```
Ts=0.1;
t=-2:0.01:2;
x=sin(2*pi*t)+0.3*sin(6*pi*t);
subplot(4,1,1), plot(t, x);% original input

nt=-2:Ts:2;
xs=sin(2*pi*nt)+0.3*sin(6*pi*nt);
subplot(4,1,2), stem(nt, xs);

n=-2/Ts:1:2/Ts;
xn=sin(2*pi*n*Ts)+0.3*sin(6*pi*n*Ts);
subplot(4,1,3), stem(n, xn);

T_N=ones(length(n),1)*nt-n'*Ts*ones(1,length(n));
xc=xn*sinc(1/Ts*pi*T_N); % convolution between xn and sinc
function
subplot(4,1,4); plot(nt, xc);
```

(a) Run the above source codes. Include the result in your report. What are the four signals in the figure? Comment on them.

Solution:

Reading the source code, the first subplot in Fig.01 is the original continuous-time input, with t ranges from -2 to 2 and 1/fs to be 0.01. Because Matlab can only handle discrete time signals, the original signal here in fact was still processed as discrete-time but with 1/fs = 0.01 and plotted as continuous-time signal.

The second subplot is the sampled signal of the original input in continuous time axis with the sampling frequency fs =1/Ts=10Hz, ranges from -2 to 2.

The third subplot is still a sampled signal. The function expressions are the same but the domain changes to [-20,20] in discrete time interval and the sampling step to be 1 instead of 0.1. Sampling rate here is does not satisfying Nyquist Rule.

The forth subplot is the reconstruction of the continuous-time signal by the convolution between xn in the third subplot and sinc function.

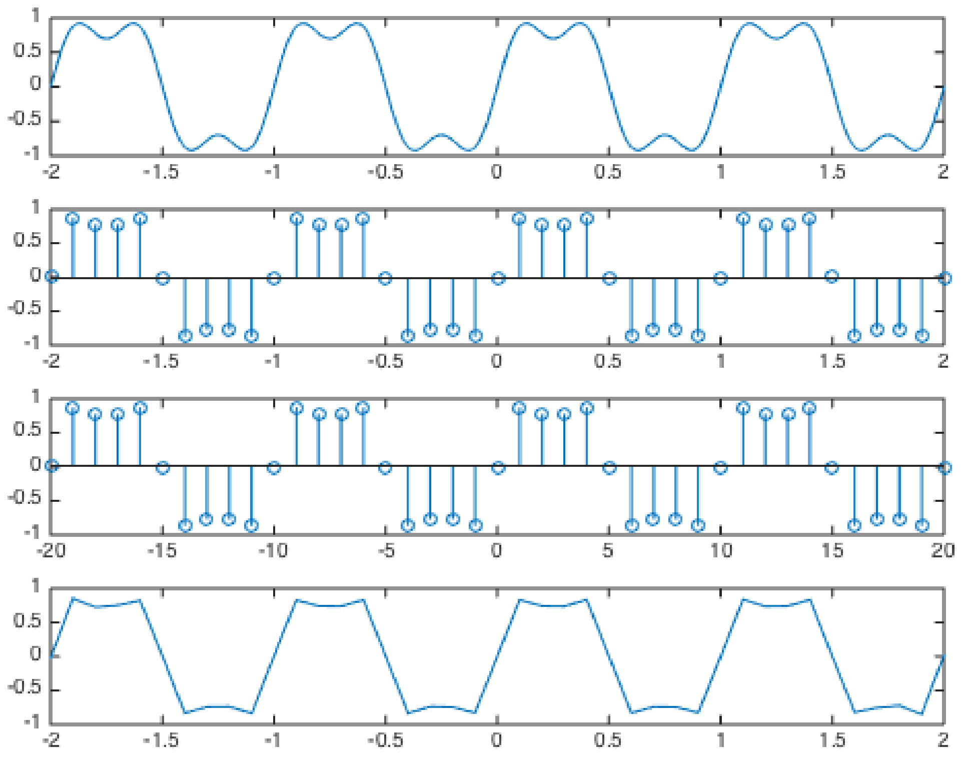


Fig.01 Original x, sampled xs, sampled xn, reconstructed xc (Ts=0.1)

(b) Based on the above source code, if the sampling frequency is changed to fs=4Hz and fs=20Hz. Repeat the sampling and reconstruction process. Show the corresponding waveforms in your report and discuss on them.

Solution:

<u>Changing the sampling frequency to be 4 Hz instead of 1Hz, Ts=1/4:</u>

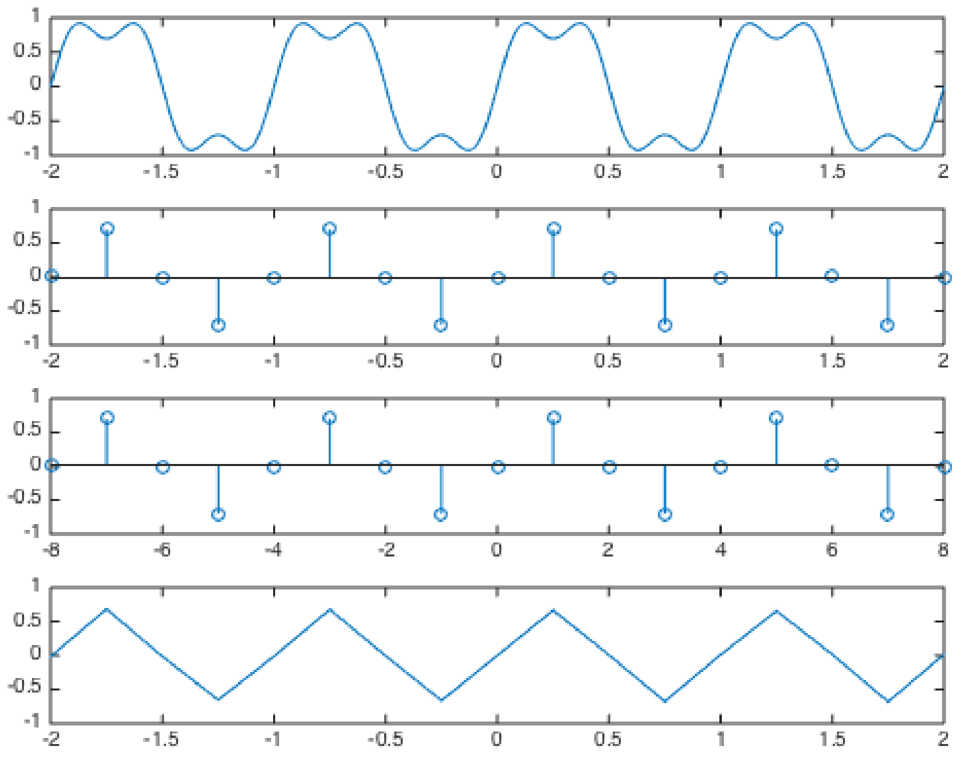


Fig.02 Original x, sampled xs, sampled xn, reconstructed xc (Ts=0.25)

The original input in subplot one remains the same. Because the sampling frequency has been deceased from 10 Hz to 4 Hz, the third subplot are sparse and simplified compared to corresponding subplot in Fig.01. The forth subplot reconstructs the continuous-time signal, because the fs does not greater than or equal to 2*fc, the reconstruction is not accurate.

<u>Changing the sampling frequency to be 4 Hz instead of 1Hz, Ts=1/20:</u>

The sampling frequency increased from 10Hz to 20Hz, the sampled discrete-time signal is more close to original form of the original continuous signal. according to the mathematical expression of the reconstruction, the reconstruction result is more acute. If we want to fully reconstruct the continuous signal, the sampling frequency should first satisfy the Nyquist Rule.

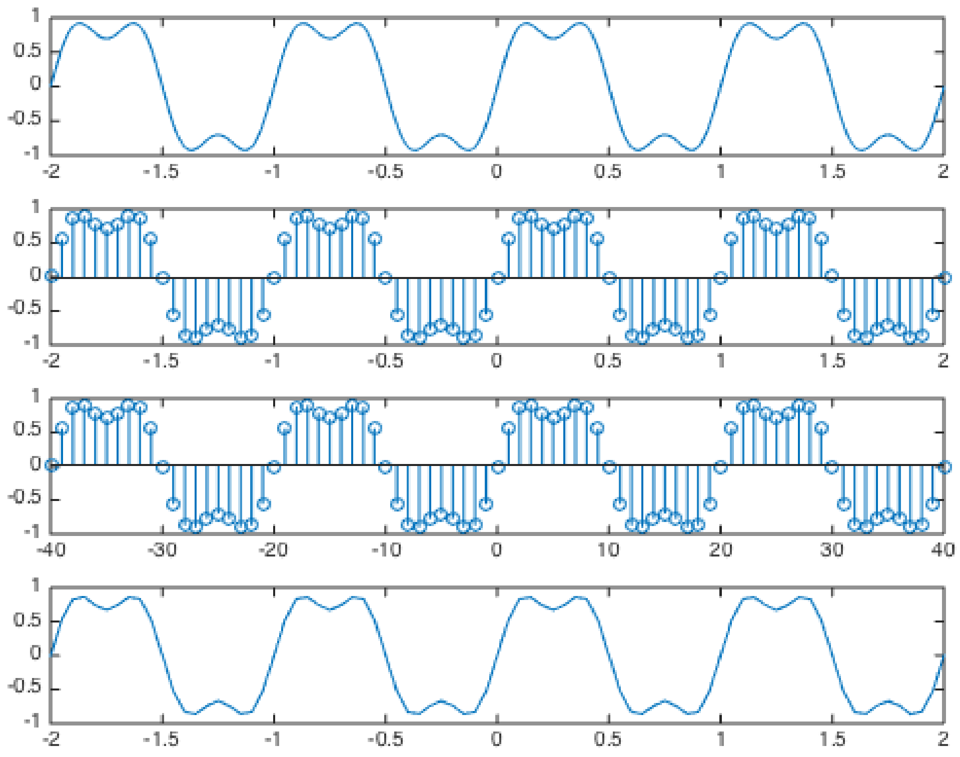
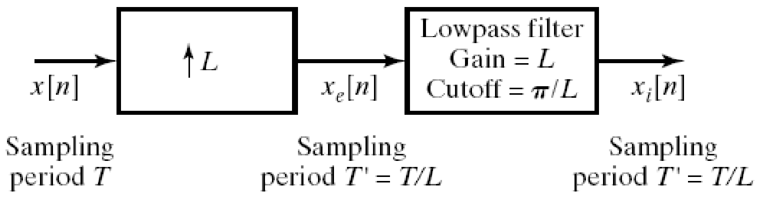


Fig.03 Results of source code (Ts=0.05)

Problem 2: Up-sampling and Down-sampling in Time Domain 1. Up-sampling

The following program can be used to up-sampling one discrete-time sequences.



```
n=0:50;
x=sin (2*pi*0.12*n);
xe=zeros(1, 3*length(x));
xe( [1: 3: length(xe)] ) = x;
```

(a) Plot sequences x and xe in the range of n=0~50. What is the up-sampling factor L?

Solution:

Reading the source code, the up-sampling factor L is set to 3.

```
n=0:50;
x=sin(2*pi*0.12*n);
xe=zeros(1, 3*length(x));
xe(1:3:length(xe)) = x;
figure
subplot(2,1,1),stem(n,x);
title('sequence x')
subplot(2,1,2),stem(n,xe(1:51));
title('sequence xe')
```

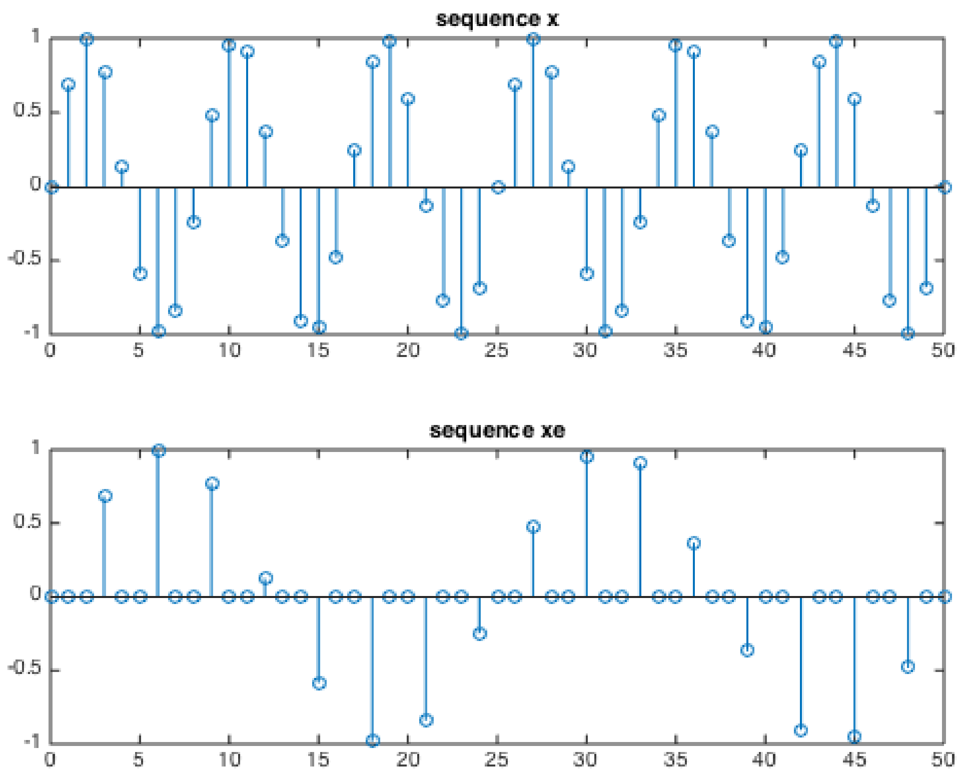


Fig.04 sequences x and xe in the range of n=0~50

(b) **The above program only implement the first block diagram in the up sampling. Using [b,a]=butter(4, 0.3) to design the 4th-order low-pass filter, and calculate the final output signal xi by using matlab function *filter* (Don't forget to compensate the attenuation in the amplitude). Draw the discrete-time signals x, xe and xi in the same figure. Comments on your results.**

Solution:

```
n=0:50;
x=sin(2*pi*0.12*n);
xe=zeros(1, 3*length(x));
xe(1:3:length(xe)) = x;
figure
subplot(3,1,1),stem(n,x);
title('sequence x')
subplot(3,1,2),stem(n,xe(1:51));
title('sequence xe')
%4th-order low-pass filter
[b,a]=butter(4, 0.3);
xi=3*filter(b,a,xe);%compensate attenuation
subplot(3,1,3),stem(n,xi(1:51));
title('sequence xi')
```

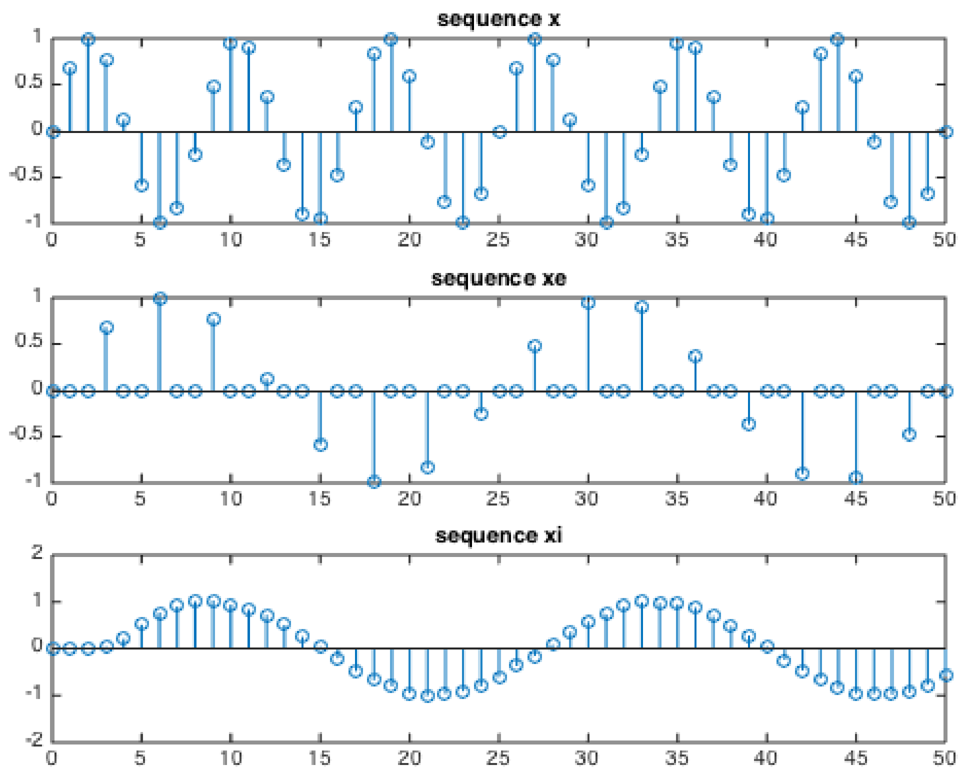


Fig.05 sequences x and xe AND xi

Up sampling is performed on a sequence of samples of a continuous function or signal and produces an approximation of the sequence that would have been obtained by sampling the signal at a higher rate. Up sampling consists of two steps, expander and interpolating (Butterworth low pass filter).

For the first step, according to the formula, where xe equal to x[n/L] at n=0, ∓L， ∓2L,… and else, 0. Then we get the DTFT of xe [n]. The output of the expander is frequency-scaled, ω = ΩTi and here in Fig.05, L=3. We can observe from the second subplot that xe = 0 when n is not equal to ∓3， ∓6, ∓9…

After DTFT of the desired interpolated signals and get extrapolator output, we need to apply the LPF and the cutoff frequency pi/L=pi/3. From Fig.05, we can observe that the sampling rate increased 3 times.

(c) Repeat above program for the up-sampling factor L=5. In this case, can we still use the same low-pass filter? If the low-pass filter in part b is not OK, modify the parameters to get the low-pass filter you required. Draw the discrete-time signals x, xe and xi for L=5. Comments on your results.

Solution:

```
n=0:50;
x=sin(2*pi*0.12*n);
xe=zeros(1, 5*length(x));
xe(1:5:length(xe)) = x;
figure
subplot(3,1,1),stem(n,x);
title('sequence x')
subplot(3,1,2),stem(n,xe(1:51));
title('sequence xe')
%4th-order low-pass filter
[b,a]=butter(4, 0.2);
xi=5*filter(b,a,xe);%compensate attenuation
subplot(3,1,3),stem(n,xi(1:51));
title('sequence xi')
```

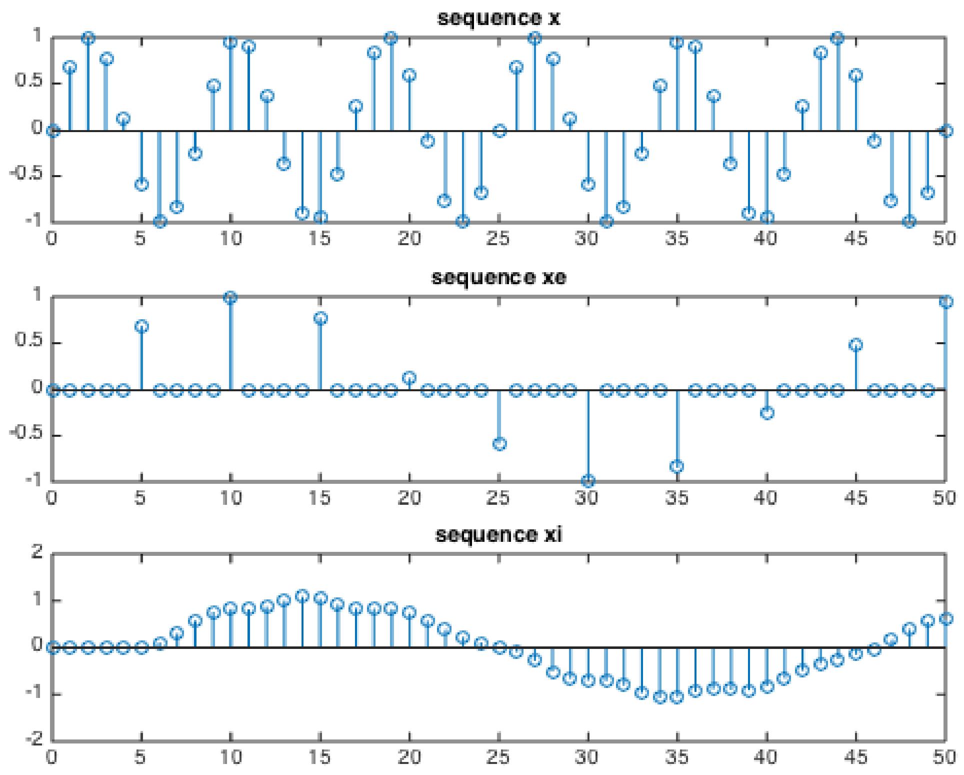


Fig.05 sequences x and xe and xi (L=5)

Observing the xi of the same low-pass filter but L changed to 5, after increase the up sampling factor, cutoff frequency changed from L/3 to L/5, the frequency spectrum of fundamental wave is narrowed. Thus, the low-pass filter in b is not ok here.

```
n=0:50;
x=sin(2*pi*0.12*n);
xe=zeros(1, 5*length(x));
xe(1:5:length(xe)) = x;
figure
subplot(3,1,1),stem(n,x);
title('sequence x')
subplot(3,1,2),stem(n,xe(1:51));
title('sequence xe')
%4th-order low-pass filter
[b,a]=butter(4, 0.15);
xi=5*filter(b,a,xe);%compensate attenuation
subplot(3,1,3),stem(n,xi(1:51));
title('sequence xi')
```

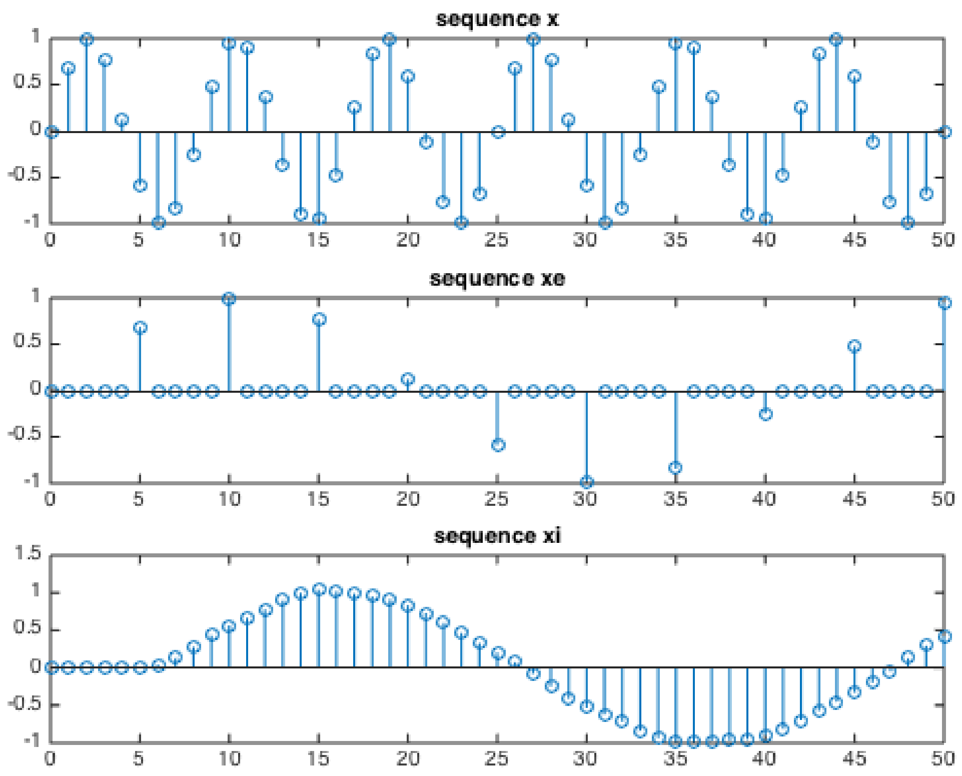


Fig.06 sequences x and xe and xi with wn to be 0.15

After several trials, we set the new cutoff frequency wn to be 0.15 to maintain in form of sin wave.

(d) Modify above program if linear interpolation is used for up-sampling. Choose the same up-sampling L as part (b).

```
n=0:50;
x=sin(2*pi*0.12*n);
xe=zeros(1,3*length(x));
xe(1:3:length(xe)) = x;

figure
subplot(3,1,1),stem(n,x);
title('sequence x')
subplot(3,1,2),stem(n,xe(1:51));
title('sequence xe')

[b,a]=butter(4, 0.3);
xi=3*filter(b,a,xe);%compensate attenuation
r=-3:3;
```

```
h=1-(abs(r))/3;
xi=conv(xe,h);
subplot(3,1,3),stem(n,xi(1:51));
title('sequence xi')
```

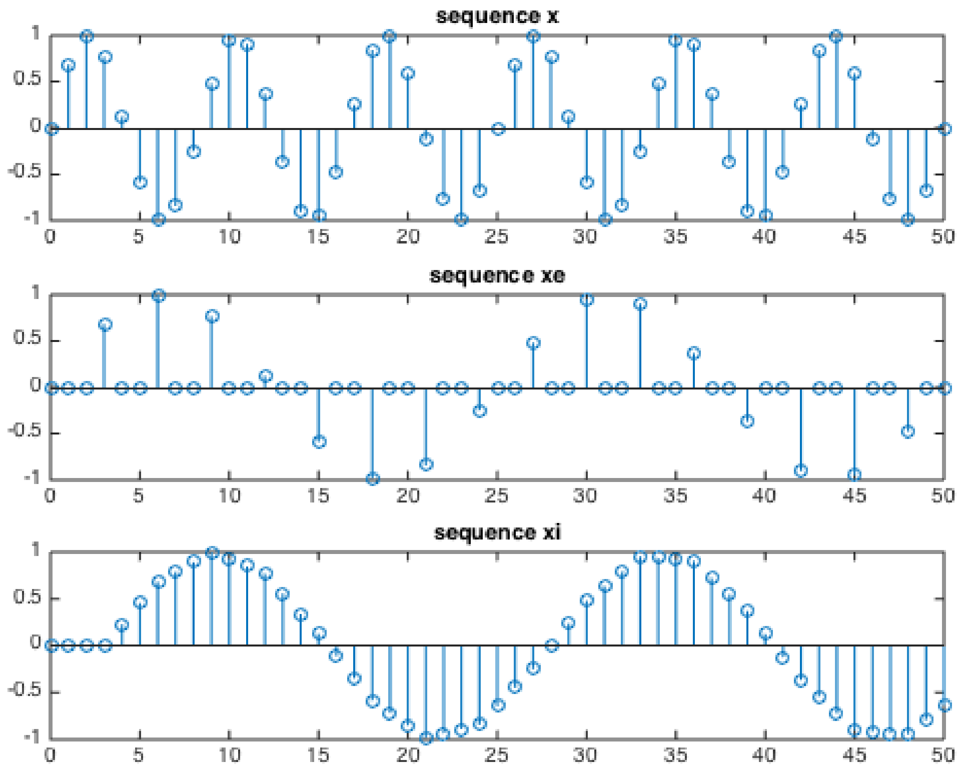


Fig.07 sequences x and xe and xi

We apply L=3. From the program, we can find that the linear interpolation can perform as Butterworth low pass filter here for the up sampling interpolation.

2. Down-sampling

The following program can be used to down-sampling a sequences.

n=0:49;   m=0:50*3-1 x=sin(2*pi*0.042*m); y=x([1:3:length(x)]);

(a) Plot sequences x and y, comments on your results.

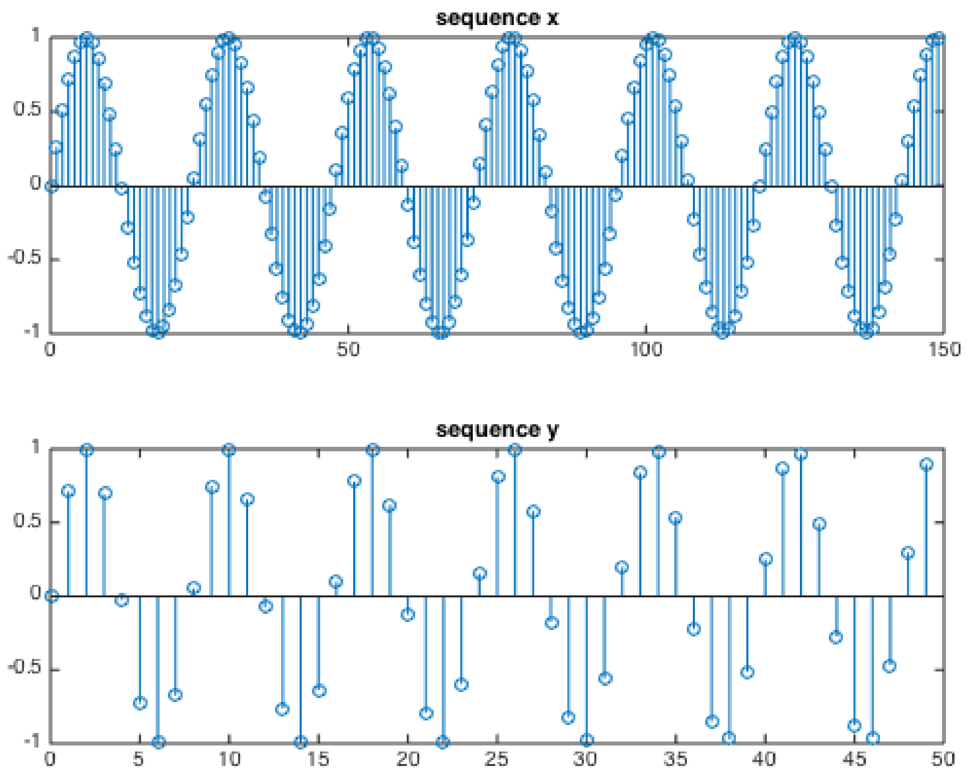


Fig.08 sequences of x and y, M=3

Because it is the down sampling, the normal sampling, thus, after doing sampling, the sampling points decrease. Observing Fig.08, the down sampling factor M is 3.

(b) Down-sampling the output sequences in part (a) with the down-sampling factor M=4. Comment on your results.

```
m=0:50*3-1;
x=sin(2*pi*0.042*m);
y=x(1:4:length(x));

[b,a]=butter(4, 0.25);
xe=filter(b,a,y);
xi=xe(1:4:length(xe));

figure
subplot(3,1,1), stem(y);
subplot(3,1,2),stem(xe);
subplot(3,1,3),stem(xi);
```

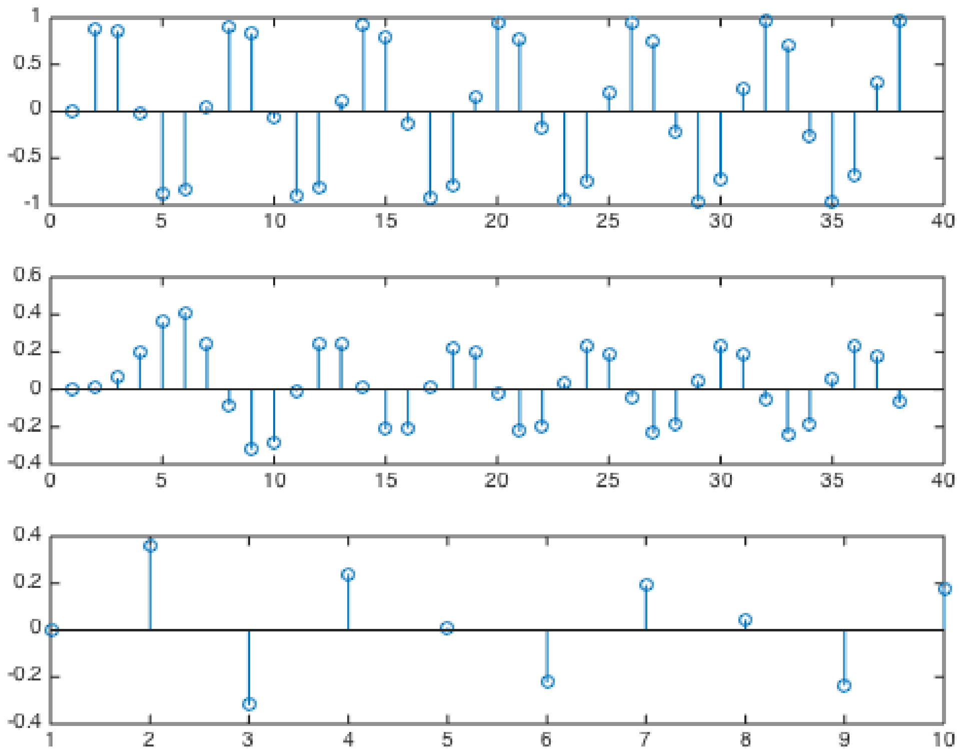


Fig.09 sequences of y, xe, xi (M=4)

M=4, after down sampled, sequence y contains less information of sequence x. For down sampling, we need a low pass filter (wn=pi/M=pi/4) for anti-aliasing. Then reduce the sampling rate of a sequence with the down sampling factor M=4.

Problem 3: Up-sampling and Down sampling in Frequency Domain

To demonstrate the effect of up-sampling and down-sampling in the frequency-domain we need to create a finite-length input sequence that is also band-limited in the frequency-domain. To this end we can utilize the matlab-function *fir2*.

A band-limited input sequence is created by following program:

```
freq = [0 0.42 0.48 1];
mag = [0 1 0 0];
   x = fir2 ( 99, freq, mag);
```

Its spectrum can be evaluated:

[Xz, w]=freqz (x, 1, 512, ‘whole’);

plot (w/pi, abs(Xz));

(a) Plot the input sequences x and its spectrum in the range of w/pi = 0 ~ 1.

Solution:

```
freq = [0 0.42 0.48 1];
mag = [0 1 0 0];
x = fir2 (99, freq, mag);
[Xz, w]=freqz (x, 1, 512, 'whole');
subplot(2,1,1),plot(w/pi,abs(Xz)); xlabel('w/pi'),ylabel('abs(Xz)');
title('spectrum')
subplot(2,1,2),stem(0:99,x); xlabel('a'),ylabel('x');
title('x');
```

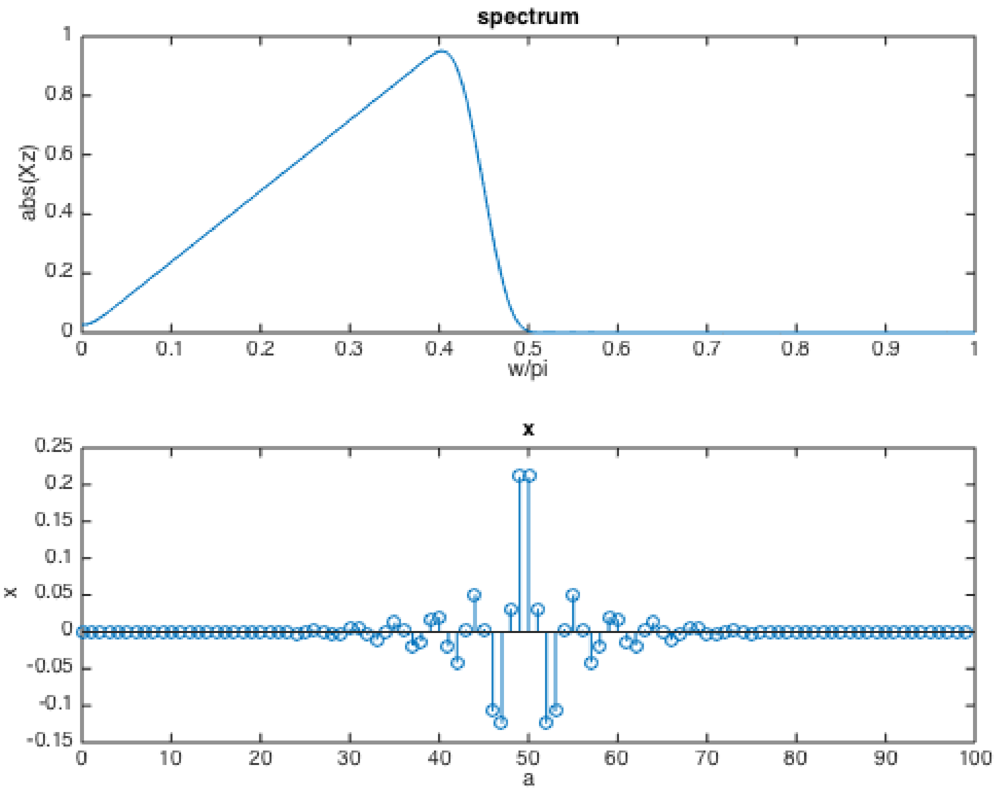


Fig.10 spectrum in range of w/pi and sequence x

(b) Up-sampling the sequence x by the factor L=2, 3 and 5. Draw the obtained sequences and its spectrum. Comments on your results.

L=2:

```
freq = [0 0.42 0.48 1];
mag = [0 1 0 0];
x = fir2 ( 99, freq, mag);
xe=zeros(1, 2*length(x));
xe(1:2:length(xe)) = x;
[b,a]=butter(4, 0.5);
xi=2*filter(b,a,xe);
figure
subplot(3,1,2);stem(0:99,x);ylabel('x');
subplot(3,1,3);stem(xi);ylabel('xi');
[Xz, w]=freqz (xi,1,512,'whole');
subplot(3,1,1);plot(w/pi,abs(Xz));ylabel('|Xz|');axis([0 1 0 2]);
```

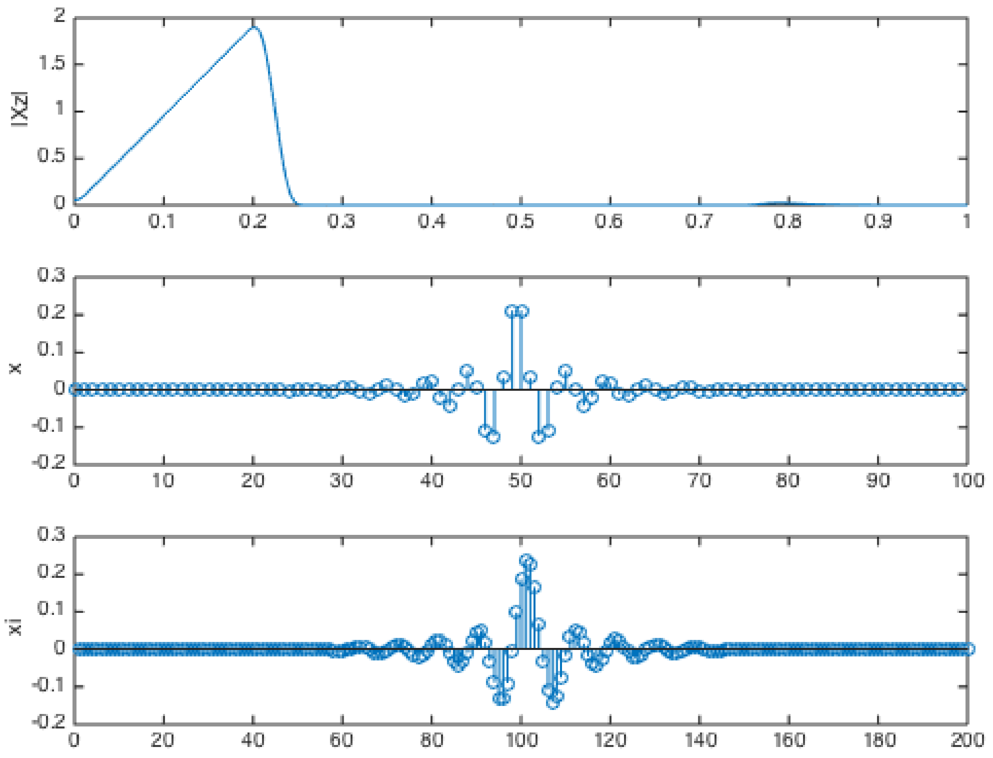


Fig.12 |X(z)|, x, xi (L=2)

L=3:

```
freq = [0 0.42 0.48 1];
mag = [0 1 0 0];
x = fir2 ( 99, freq, mag);
xe=zeros(1, 3*length(x));
xe(1:3:length(xe)) = x;
[b,a]=butter(4, 0.2);
xi=2*filter(b,a,xe);
figure
subplot(3,1,2);stem(0:99,x);ylabel('x');
subplot(3,1,3);stem(xi);ylabel('xi');
[Xz, w]=freqz (xi,1,512,'whole');
subplot(3,1,1);plot(w/pi,abs(Xz));ylabel('|Xz|');axis([0 1 0 3]);
```

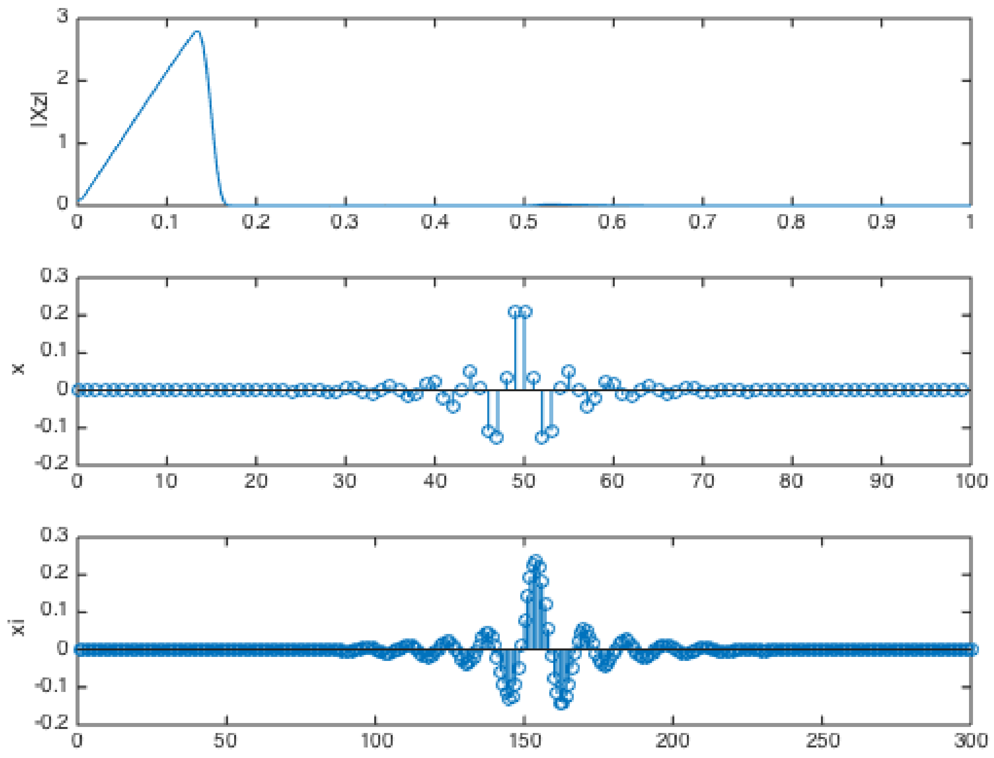


Fig.12 |X(z)|, x, xi (L=3)

L=5:

```
>> freq = [0 0.42 0.48 1];
mag = [0 1 0 0];
x = fir2 ( 99, freq, mag);
xe=zeros(1, 5*length(x));
xe(1:5:length(xe)) = x;
[b,a]=butter(4, 0.1);
```

```
xi=5*filter(b,a,xe);
figure
subplot(3,1,2);stem(0:99,x);ylabel('x');
subplot(3,1,3);stem(xi);ylabel('xi');
[Xz, w]=freqz (xi,1,512,'whole');
subplot(3,1,1);plot(w/pi,abs(Xz));ylabel('|Xz|');axis([0 1 0
5]);
```

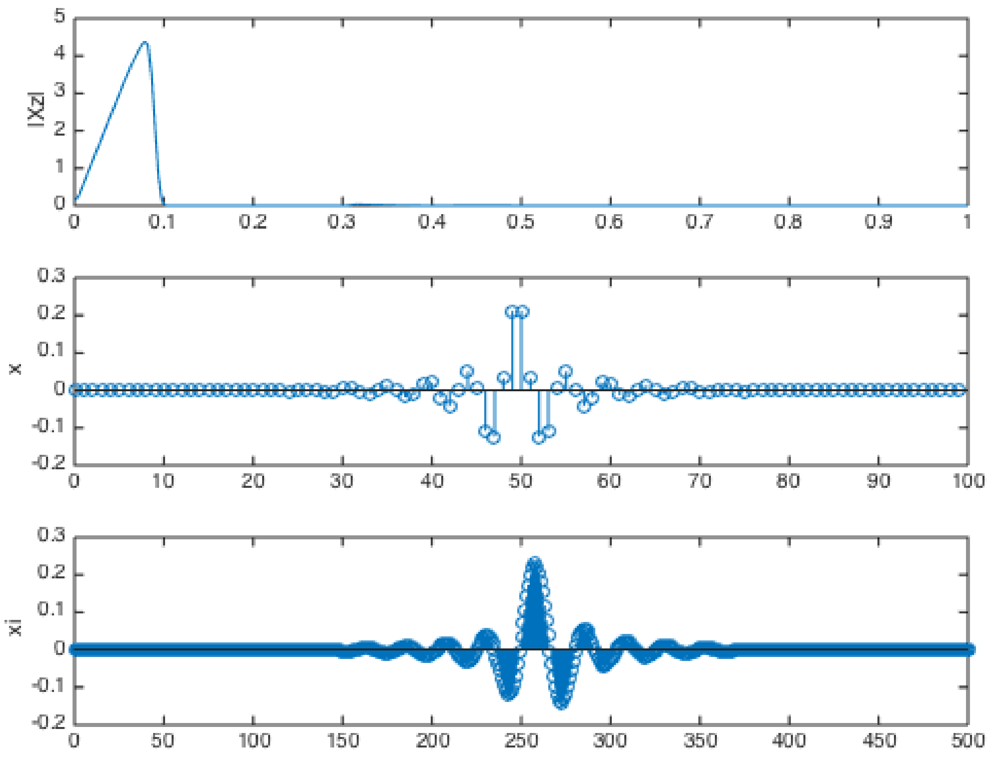


Fig.13 |X(z)|, x, xi (L=5)

Observing spectrums when L= 2,3,5, we can find that the spectrums after up sampling in inverse proportion of factors L. The magnitude of the spectrums of after up sampling is zero after 0.5/L. Besides, the amplitudes of the spectrums equal to the factor L because of the gains are set equal to L.

(c) Down-sampling the sequence x by factor M= 2, 3 and 5. Draw the obtained sequences and its spectrum. Comments on your results. What is the minimum value of M for which aliasing occurs?

```
freq=[0 0.42 0.48 1];
mag=[0 1 0 0];
x=fir2(99,freq,mag);
```

```
[Xz,w]=freqz(x,1,512,'whole');
[b,a]=butter(4,0.5);
xe=filter(b,a,x);
xi=xe(1:2:length(xe));

figure
subplot(3,1,3), stem(xi);ylabel('xi');
[Xz,w]=freqz(xi,1,512,'whole');
subplot(3,1,2),plot (w/pi, abs(Xz)); axis([0 2 0 0.5]);
subplot(3,1,1);stem(0:99,x);ylabel('x');
```

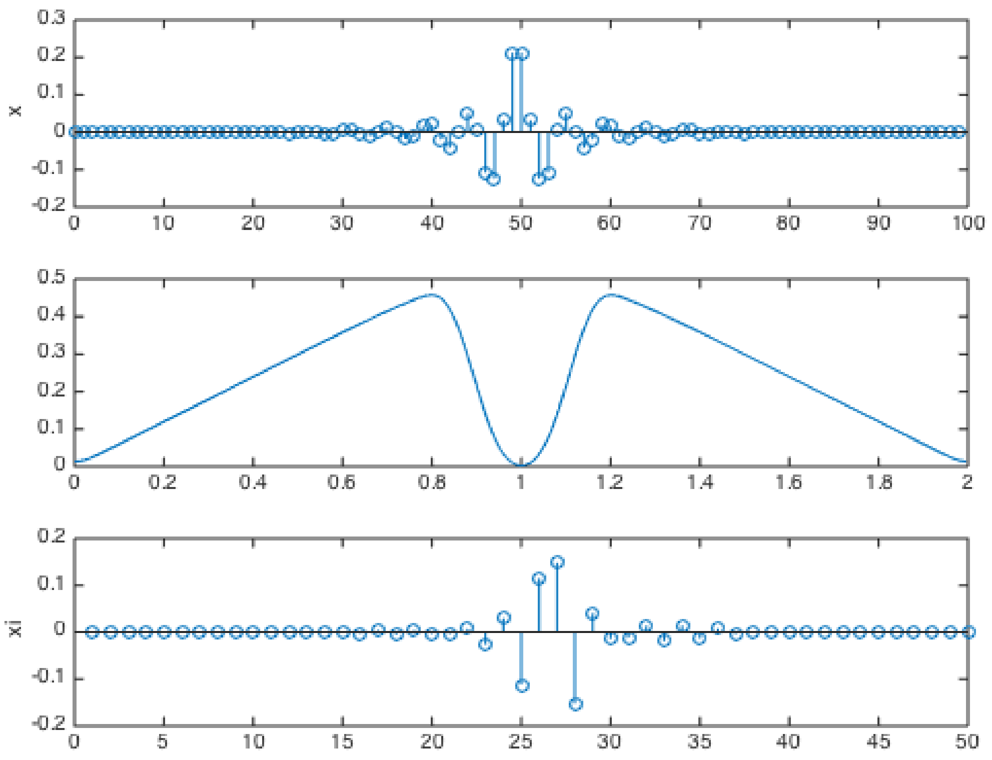


Fig.14 x, |xz|, xi (M=2)

```
freq=[0 0.42 0.48 1];
mag=[0 1 0 0];
x=fir2(99,freq,mag);
[Xz,w]=freqz(x,1,512,'whole');
[b,a]=butter(4,1/3);
xe=filter(b,a,x);
xi=xe(1:3:length(xe));

figure
subplot(3,1,3), stem(xi);ylabel('xi');
[Xz,w]=freqz(xi,1,512,'whole');
subplot(3,1,2),plot (w/pi, abs(Xz)); axis([0 2 0 0.2]);
subplot(3,1,1);stem(0:99,x);ylabel('x');
```

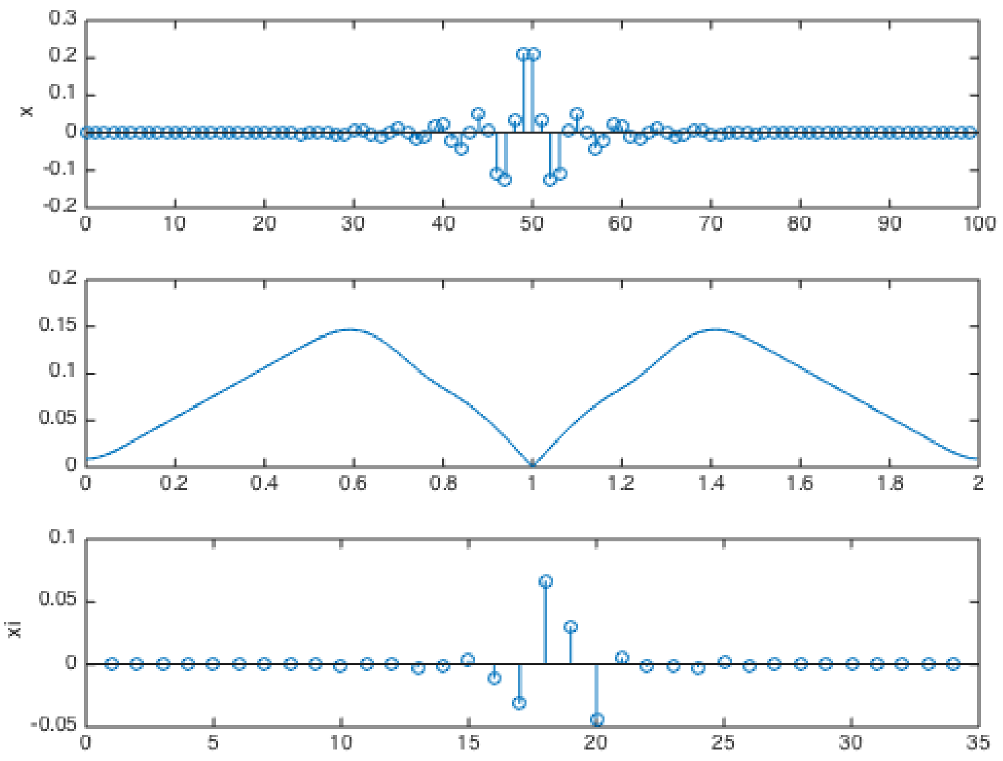


Fig.15 x, |xz|, xi (M=3)

```
freq=[0 0.42 0.48 1];
mag=[0 1 0 0];
x=fir2(99,freq,mag);
[Xz,w]=freqz(x,1,512,'whole');
[b,a]=butter(4,0.2);
xe=filter(b,a,x);
xi=xe(1:5:length(xe));

figure
subplot(3,1,3), stem(xi);ylabel('xi');
[Xz,w]=freqz(xi,1,512,'whole');
subplot(3,1,2),plot (w/pi, abs(Xz)); axis([0 2 0 0.15]);
subplot(3,1,1);stem(0:99,x);ylabel('x');
```

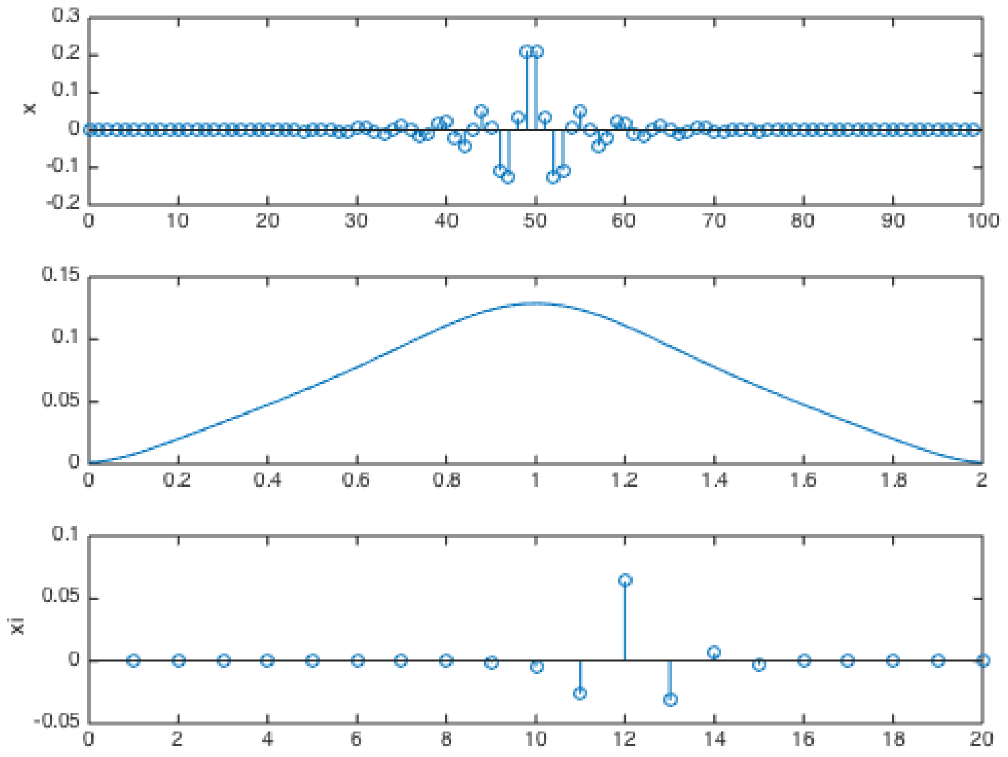


Fig.15 x, |xz|, xi (M=5)

To avoid aliasing, we should be sure that the cutoff frequency wn*M ⩽ π , Observing the spectrum, wn of the input sequence is about 0.48pi. Thus, the down sampling factor M should be ingteger not larger than 3 (2.0833), or we will need a LPF with wn= π /m.