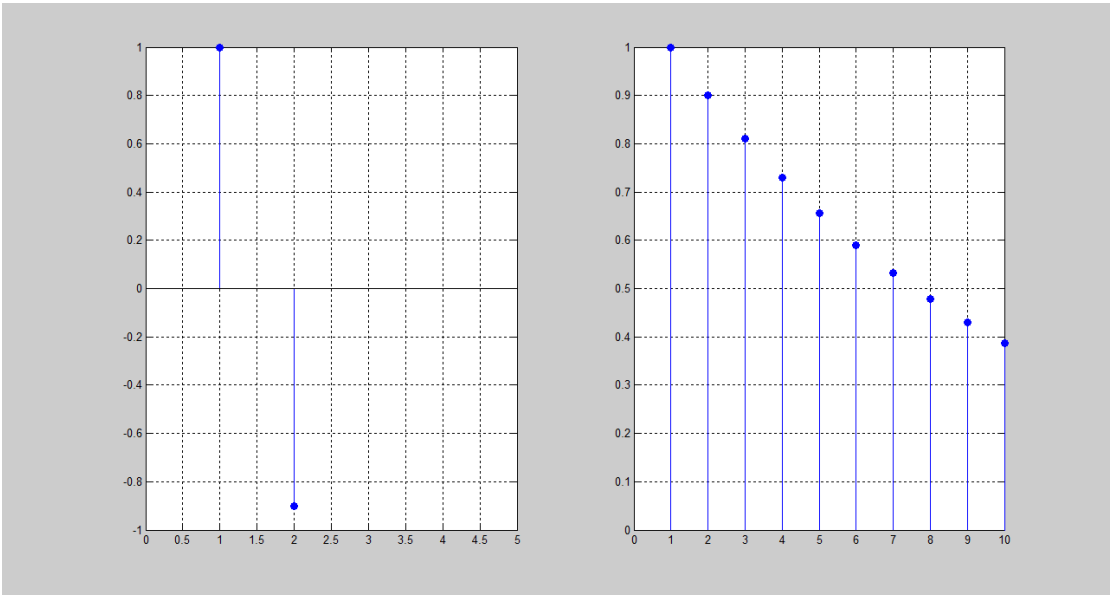
Sampling, Convolution, & FIR Filtering

1. Deconvolution Experiment for 1-D Filter

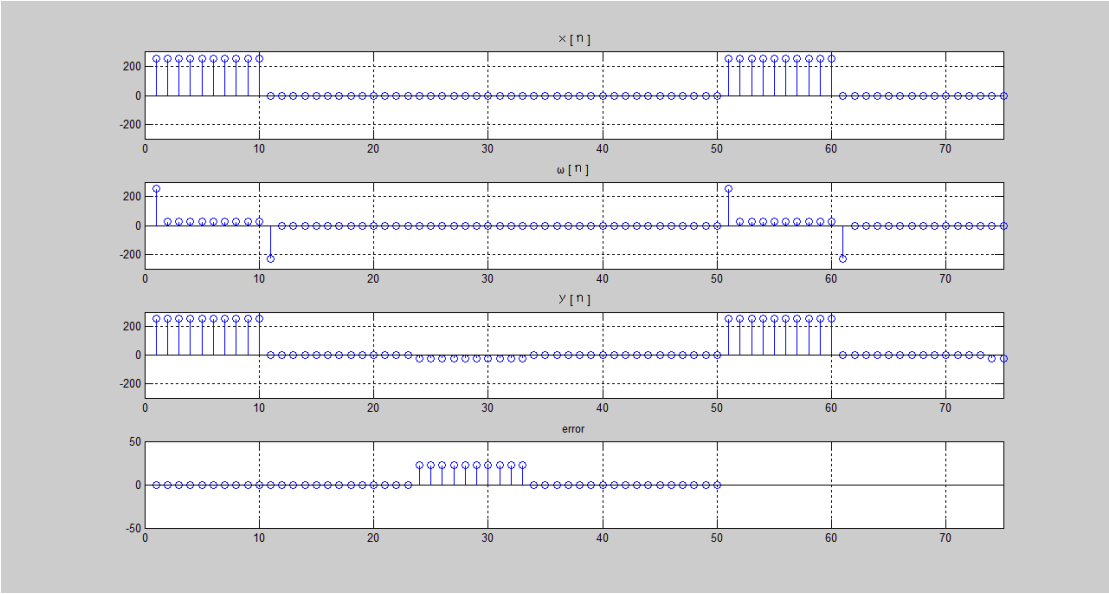
a. Plot both the input and output waveforms:

```
xx=256*(rem(0:100,50)<10);
bb=[1,-0.9];
ff=firfilt(bb,xx);

b1=ones(1,23);
for i=1:23
    b1(i)=b1(i)*0.9^(i-1);
end
f1=firfilt(b1,ff);
figure,subplot(1,2,1);stem(bb,'filled');axis([0 5 -1 1]);grid on;
subplot(1,2,2);stem(b1,'filled');axis([0 10 0 1]);grid on;

figure,subplot(4,1,1);stem(xx);grid on;axis([0 75 -300 300]);title('x[n]');
subplot(4,1,2);stem(ff);grid on;axis([0 75 -300 300]);title('ω  [n]');
subplot(4,1,3);stem(f1);grid on;axis([0 75 -300 300]);title('y[n]');
```

The FIR filter:



The filter signals:



As the mathematical calculating methods:

$$y[n] = \sum_{k=0}^{M} b_k \cdot x[n-k]$$

Therefore, we know the result is just like the way which the plot presented.

b. Note that w[n] and x[n] are not the same length.

According to the length of the convolution calculation:

$$length(yy) = length(xx) + length(bb) - 1$$

And it indicts that the length of the signals:

$$length(\omega[n]) = length(x[n]) + length(\text{FIRfilter}) - 1.$$

2. Restoration Filter

$$y[n]=\sum_{l=0}^{M} r^{l}\cdot\omega[n-l]$$

① ***Processing the signal $\omega[n]$ with Filter to obtain the output signal $y[n]$. Put the plots together and make a comparison. Between the 0 and 50, find out the error.***

```
error=ones(0,50);
for j=1:50
    error(j)=xx(j)-f1(j);
end
subplot(4,1,4);stem(error);grid on;axis([0 75 -50 50]);title('error');
```

The plots have been shown previously.

② ***Evaluate the worst-case error by the MATLAB function max().***

```
max_err=max(error);
```

From the MATLAB command interface, we could find out the Max error is 22.6891. Combining the max error and the plot that indicates the difference between the original signal and the filtered signal, we could judge the quality of the restoration of x[n] is almost the same, expect the flat area in the plot. I think the error which would be within 3% to 5% is acceptable, and we could not notice it; therefore, we ignore it.

3. An echo filter

The echo filter could be explained like the formula:

$$y_1[n]=x_1[n]+\gamma\cdot x_1[n-P]$$

1. The sample rate is 8000 fs, according to the demand for the delay 0.2 s, and the amplitude should be the 90% of the original. We could determine that :

   γ = -0.1; P = 1600.

2. Describe the filer coefficients of this FIR filter:

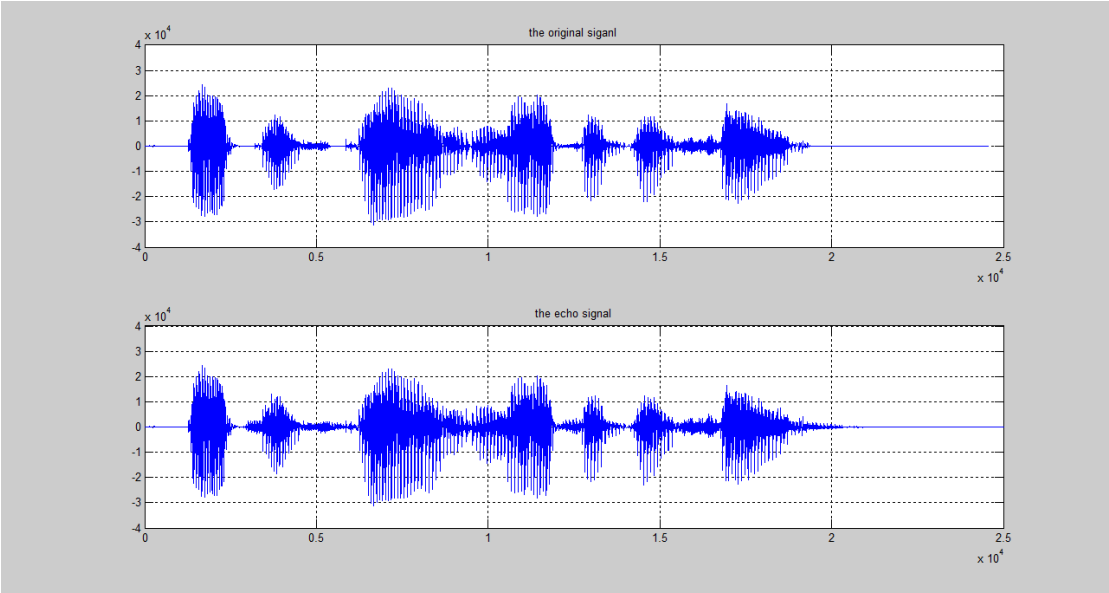
```
b2=zeros(1,1600);
b2(1)=1;b2(1600)=-0.1;
```

Therefore, the length of the FIR's length is 1600.

3. Implement the echo filter :

```
yecho=firfilt(b2,x2);
figure,subplot(2,1,1);plot(x2);axis([0 25000 -40000 40000]);
title('the original siganl');grid on
subplot(2,1,2);plot(yecho);axis([0 25000 -40000 40000]);
title('the echo signal');grid on;
soundsc(yecho,8000);
```

Implement that FIR filter in the signal which found in the data file **labdat.mat**, and it will sounds like there is an echo in every syllable. The plots that show the audio signal are presented below:



4. An echo filter

A. Overall impulse response

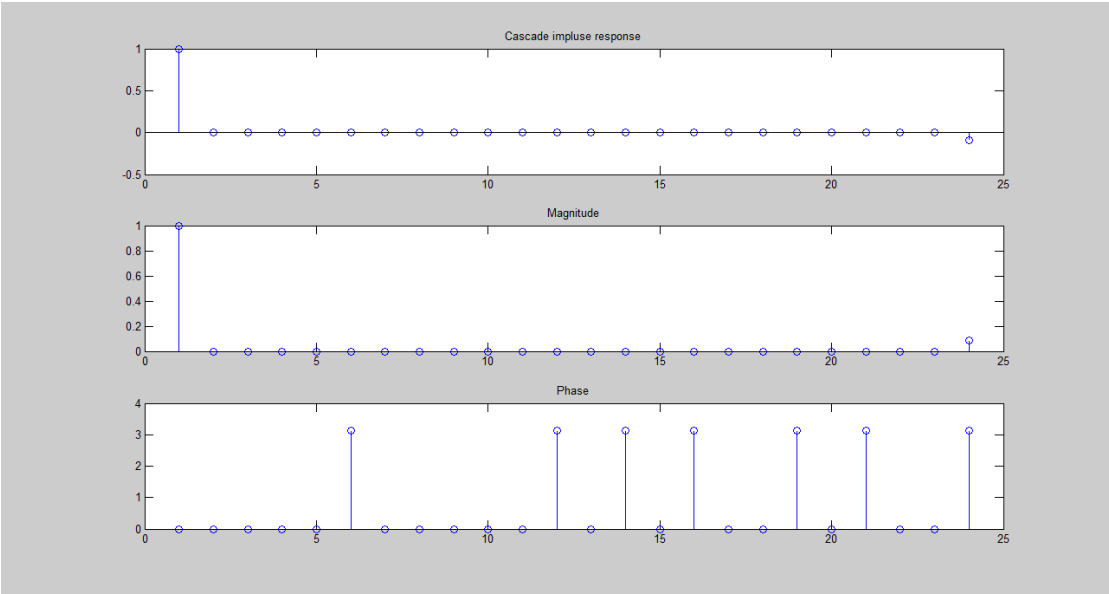
1. Two FIR filter are described to be connected in cascade:

$$w[n] = x[n] - q \cdot x[n-1] \qquad \text{(FIR filter 1)}$$

$$y[n]=\sum_{l=0}^{M} r^{l}\cdot\omega[n-l] \qquad \text{(FIR filter 2)}$$

```
h=firfilt(b1,bb);
mag=abs(h);
pha=angle(h);
figure,subplot(3,1,1);stem(h);title('Cascade impluse response')
subplot(3,1,2);stem(mag);title('Magnitude');
subplot(3,1,3);stem(pha);title('Phase')
```

And the impulse response is shown below:



2. According to the method which solved the previously homework, and I calculated the convolution to verify the result, and the result is correct.

3. For the de-convolution applications:

   To achieve the de-convolution, the system should be $h_1[n]*h_2[n]$, while the impulse response of each system are $h_1[n]$ and $h_2[n]$, respectively.

B. Distorting and Restoring Images

1. Load the image *echart.mat* with *load* command.

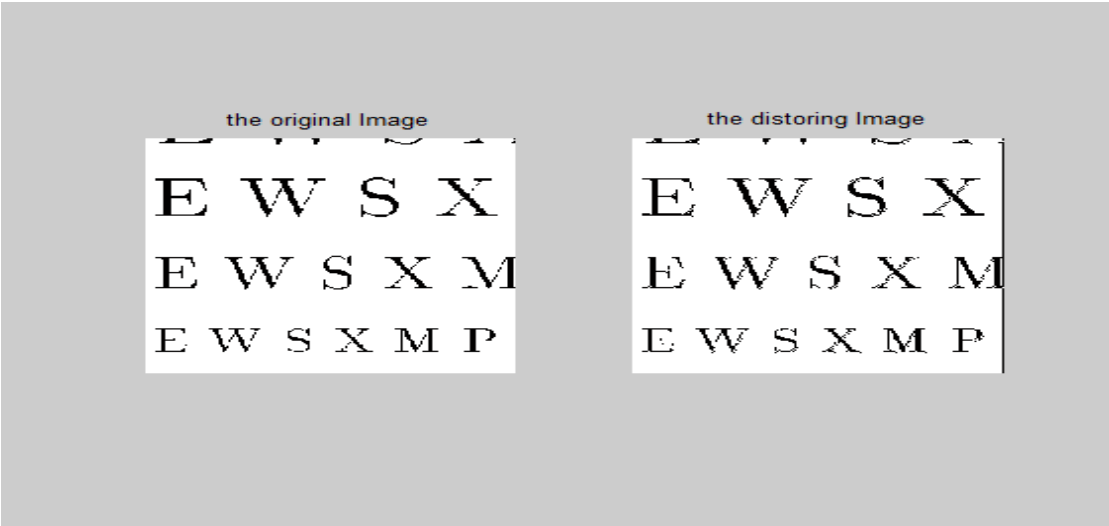
2. Set the values for the Filter-1. And filter the image at both the vertical and horizontal

directions:

```
load('C:\Users\Administrator\Desktop\ELG4177\LABs\Lab4\echart.mat');
bbh=[1,-0.9];
bbv=bbh';
e=conv2(echart,bbh);
ech90=conv2(e,bbv);
figure,subplot(1,3,1);imshow(echart); title('the original Image')
subplot(1,3,2);imshow(ech90); title('the distoring Image')
```

3. De-convolution filter with the Filter-2.

```
rb=ones(1,23);
for i=1:23
    rb(i)=rb(i)*0.9^(i-1);
end
back=firfilt(rb,ech90);
subplot(1,3,3);imshow(back);title('the restoring Image')
```



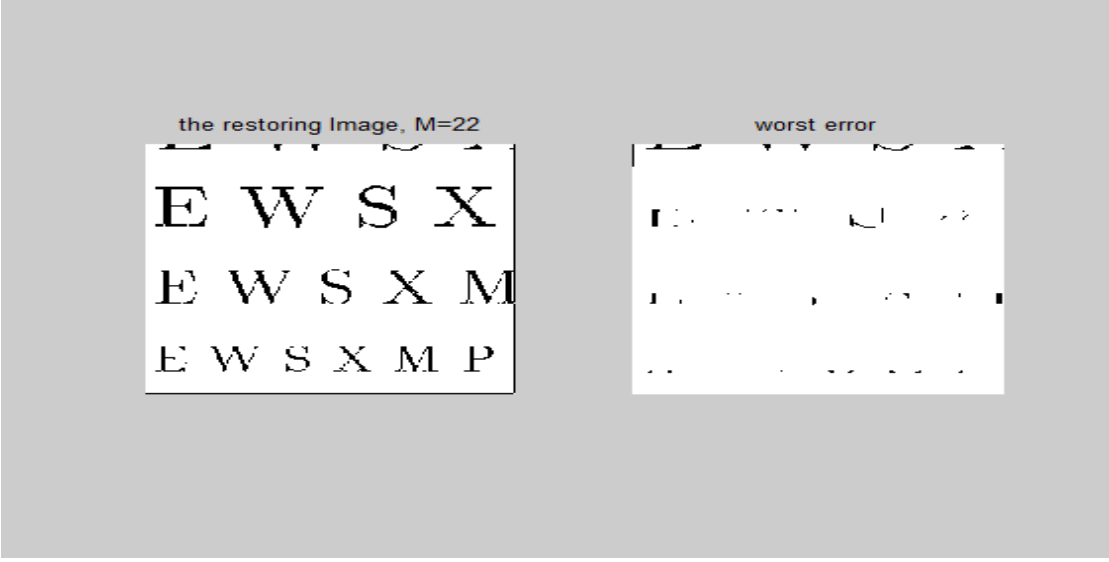
The “ghost” is an effect that the convolution causes. And we calculate the worst error, then we make an analysis based on that:

```
for m=1:256
    for n=1:1:256
        img_err(m,n) =-echart(m,n)+back(m,n);
    end
end
figure, imshow(img_err);
```

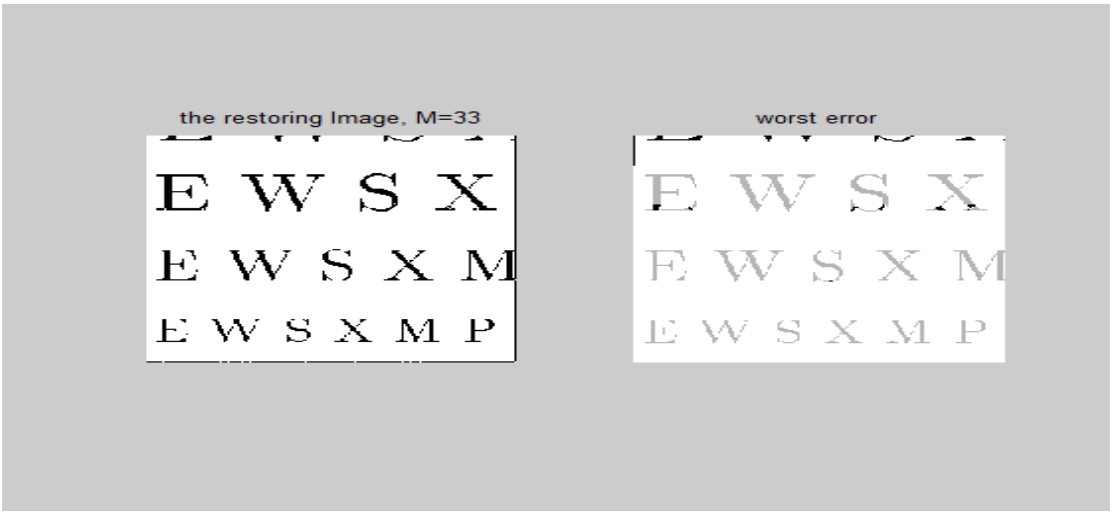
C. A second Restoration Experiment

1. M=11:

2. M=22;



3. M=33;



According to the result, we could get the conclusion that: if the value of M is great enough, the “ghost” is ignored that we could not notice the difference between the reconstructed pictures. I verified that by using the *dconvdemo* file to generate the impulse response of the sequences, and the results would better explain my conclusion.

The worst-case of each cases (part a) were shown in the previous plots. If the pictures are 256 levels gray- scale, the worst-error would be $256 \cdot x\%$, where $x = 0.08$ in the previous section. Therefore, the worst-error would be 22. Human’s eye cannot perceive a gray scale change of one level.