Encoding and Decoding Touch tone Signals

1. Simple Band Pass Filter Design.

a. Devise the code to make the magnitude of the frequency response will be one.

$$\beta = 1/\max(magnitude);$$

And the MATLAB strategy:

```
n=0:L-1;
omega=0:pi/500:pi;
hh=[];
for i=1:length(fb)
    h=cos(2*pi*(fb(i)/fs)*n);
    H=freqz(h,1,omega);
    beta=1/max(abs(H));
    h=beta*cos(2*pi*(fb(i)/fs)*n);
    hh=[hh h'];
end
```

It makes sure that the peak value of frequency response will be one.

b. Complete the file and produce eight band pass filters.

We input the vector of the centered frequency in the interface of the MATLAB:

```
fb=[697, 770, 852, 941, 1209, 1336, 1477, 1633];
hh=dtmfdesign(fb, 50, 8000);
```

and the output *hh* is a (50 * 8) matrix, whose size is correspondent with (L*8).

c. Exhibit a correct set of BPFs.

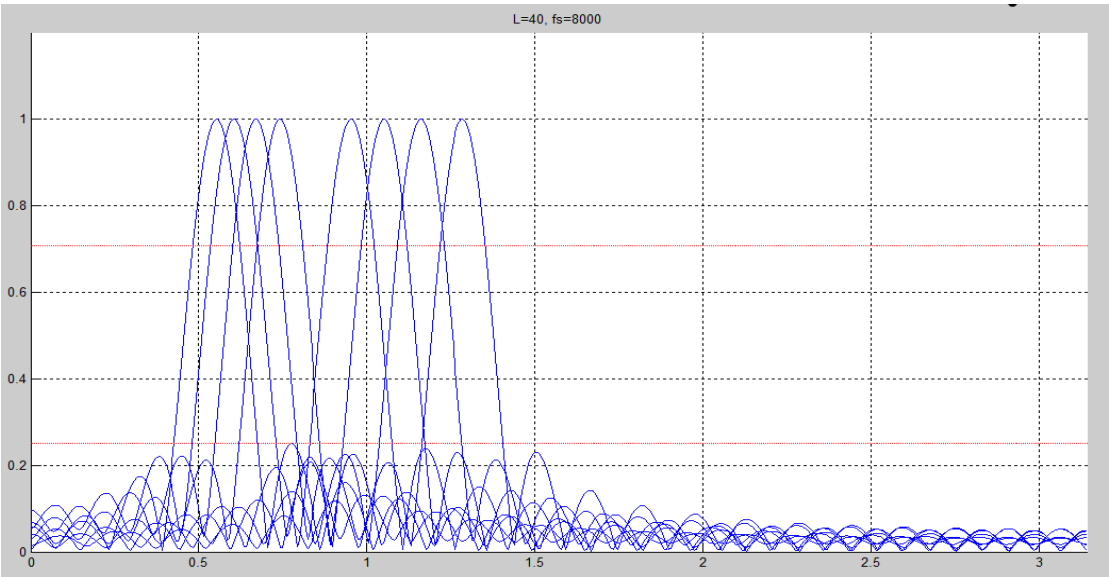
```
fb=[697, 770, 852, 941, 1209, 1336, 1477, 1633];
hh=dtmfdesign(fb, 40, 8000);
hh=dtmfdesign(fb, 80, 8000);
```

d. Generate the eight (scaled) band pass filter.

In the case that L=40, and fs=8000:

```
fb=[697, 770, 852, 941, 1209, 1336, 1477, 1633];
hh=dtmfdesign(fb, 40, 8000);
omega=0:pi/500:pi;
hold on;plot(omega,0.707,'r');axis([0,3.14,0,1.2]);
title('L=40, fs=8000');hold off;grid on; hold on;plot(omega,0.25.'r');hold off;
```

and the plot is shown below:



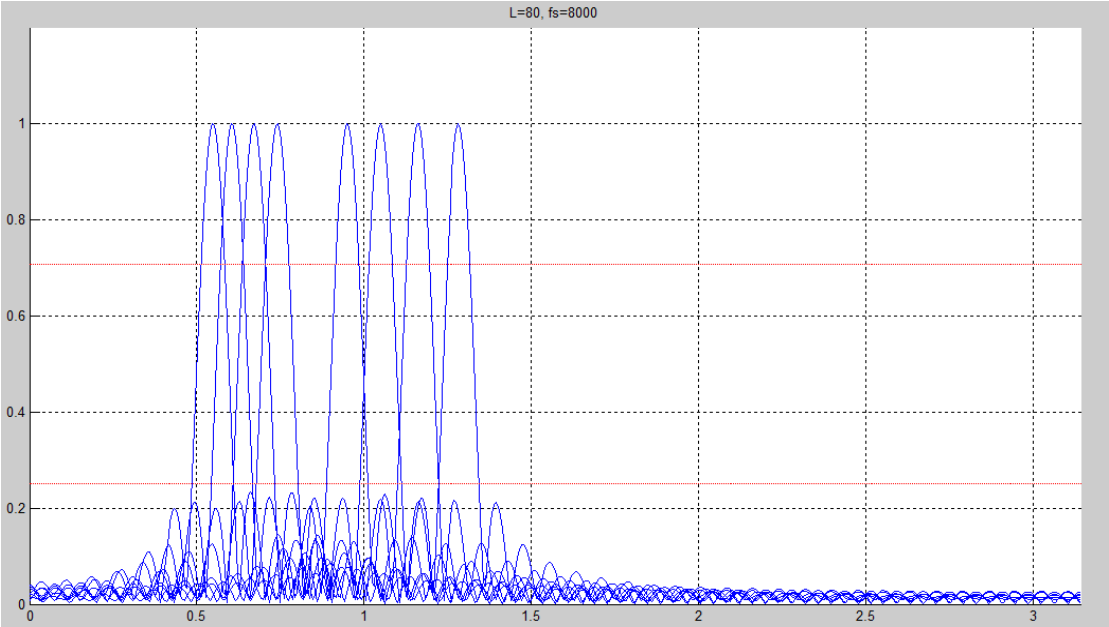
From the previous plot, obviously, there would be overlapped parts between near frequency response. The pass bands are not narrow enough to separate the DTMF frequency components.

e. Generate the eight (scaled) band pass filter.

In the case that L=80, and fs=8000:

```
fb=[697, 770, 852, 941, 1209, 1336, 1477, 1633];
hh=dtmfdesign(fb, 80, 8000);
omega=0:pi/500:pi;
hold on;plot(omega,0.707,'r');axis([0,3.14,0,1.2]);
title('L=80, fs=8000');hold off;grid on; hold on;plot(omega,0.25.'r');hold off;
```

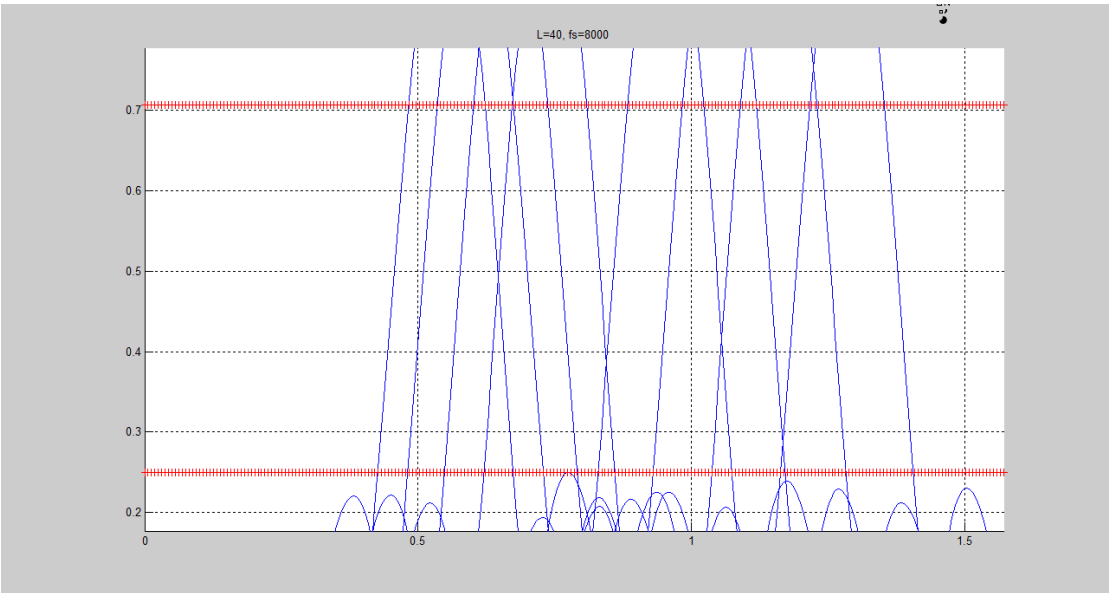
and the plot is shown below:



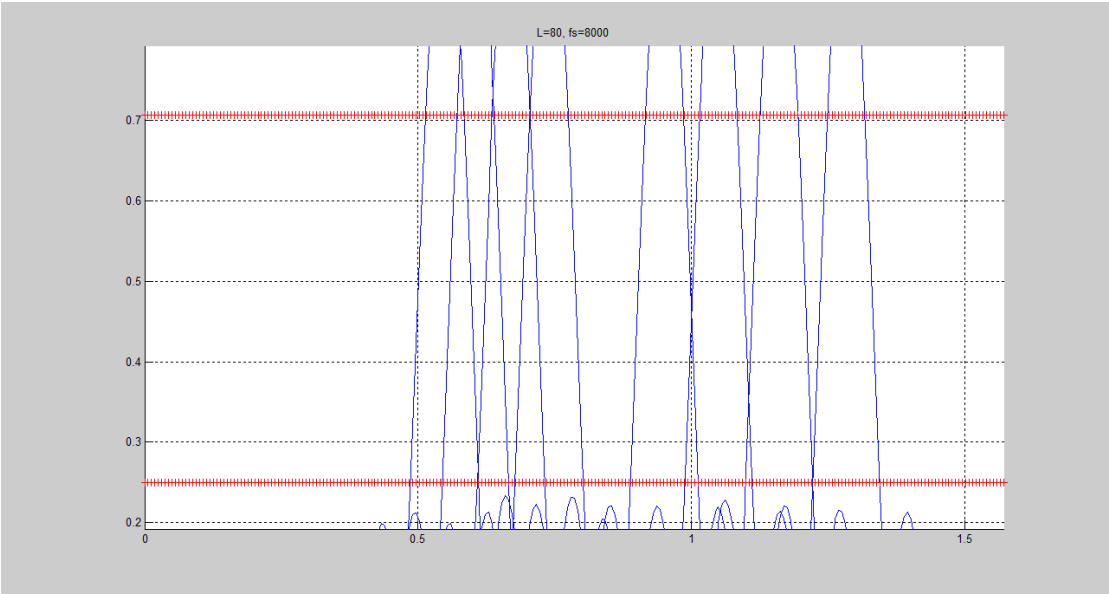
From the previous plot, when L=80, the overlapped parts are eliminated partly, it is better than that when L=40, though there would still be overlapped during the low frequencies. And it is true that the width of the band of pass is supposed to vary inversely with the filter length L.

f. Filter design specifications.

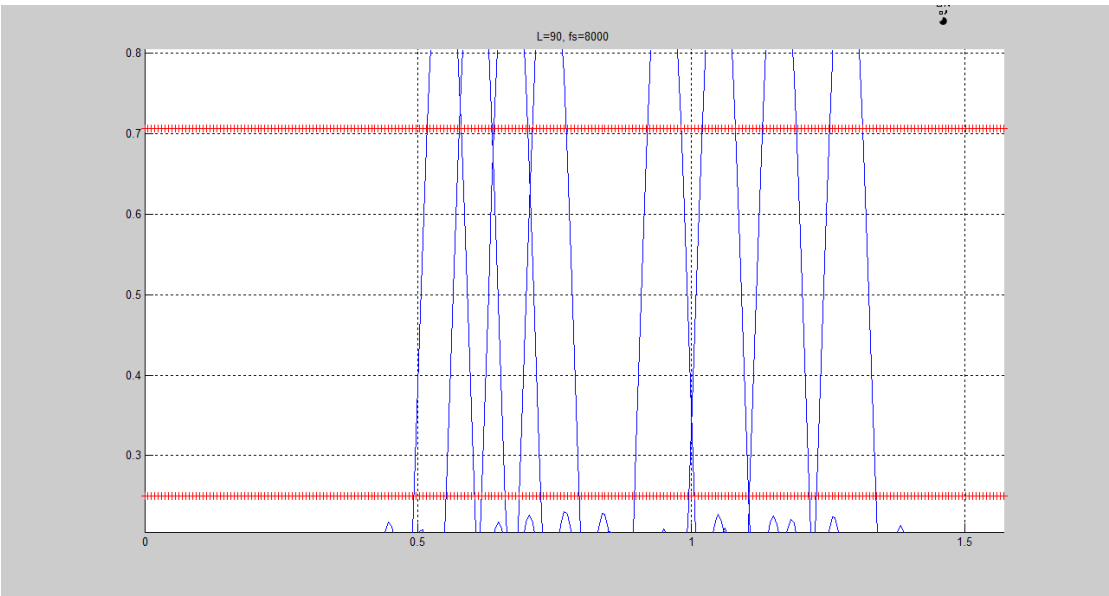
When L=40:



When L=80:



Comment: as I commented in previously parts: when L=40 or 80, there would be overlapped parts between near frequency response, even though L=80 is better, but still not perfect. As the experience, that a certain frequency component is in the band pass, while the rest frequency components are in the stop band, (it is the reason how the filter pass one component while rejecting the others), we select L=90. Here is the frequency response of in this case (L=90):



As the plot give me the insights that fb=697Hz and 770Hz are hard to meet the specifications for the chosen value of L.

2. A Scoring Function.

1. Complete the function.

Here is the MATLAB code demonstration:

```
xx = xx*(2/max(abs(xx)));
filted=firfilt(xx,hh);
if max(abs(filted))<0.59
    sc=0;
else
    sc=1;
end
```

2. **Use the rule for scoring.**

$$\max_n |y_i[n]| \geq 0.59, \quad sc = 1;$$

$$otherwise, \qquad sc = 0.$$

3. **Scaling process prior to filtering and scoring.**

```
xx = xx*(2/max(abs(xx)));
```

4. **The scoring rule depends on proper scaling of the frequency response of the BPF.**

The maximum magnitude for $H(e^{j\omega})$ must be equal to one for each filter, because after the filtering processing, the magnitude for each signal would not be changed; therefore, the amplitude of the output would easily to be predicted while we change the frequency response and the amplitude of the input.

5. **Useful plot command inside the file.**

```
n=200:500; plot(n,filted);
```

Plot the first 200-500points of the filtered output.

3. DTMF Decode Function.

1. **Design the eight band pass filter.**
2. **Break the input signal down into individual segments.**
3. **Score each BPF output of segments.**
4. **Determine the key for the segment.**
5. **Final output is the list of decoded keys.**

And the codes demonstrated that processes:

```
dtmf.keys = ['1','2','3','A';'4','5','6','B';'7','8','9','C';'*','0','#','D'];
center_freqs = [697, 770, 852, 941, 1209, 1336, 1477, 1633];
hh = dtmfdesign( center_freqs,L,fs );

[nstart,nstop] = dtmfcut(xx,fs);

Keys = [];
for kk=1:length(nstart)
    x_seg = xx(nstart(kk):nstop(kk));
    vscore=[];
    a=[];
    for i=1:8
        sc=dtmfscore(x_seg,hh(:,i)');
        if sc==1
            vscore=[vscore 1];
        else
            vscore=[vscore 0];
        end
    end

a=find(vscore==1);
Keys=[Keys dtmf.keys(a(1),a(2)-4)];
end
```

4. Telephone Numbers

All the M-files are on the MATLAB path, to test the DTMF system.

```
fs=8000;
tk=['4','0','7','*','8','9','1','3','2','#','B','A','D','C'];
xx=dtmfdial(tk,fs);
specgram(xx,fs);
soundsc(xx,fs);
L=90;
dtmfrun(xx,L,fs);
```

as the result in the interface of MATLAB, the original keys would be reconstructed:

```
Command Window
>> Keys

Keys =

407*89132#BADC

fx >>
```

And the spectrogram is displayed below: