MODULE 09 HOMEWORK

4/09/18

EN.525.718.81.SP18 MULTIRATE SIGNAL PROCESSING

SPRING 2018

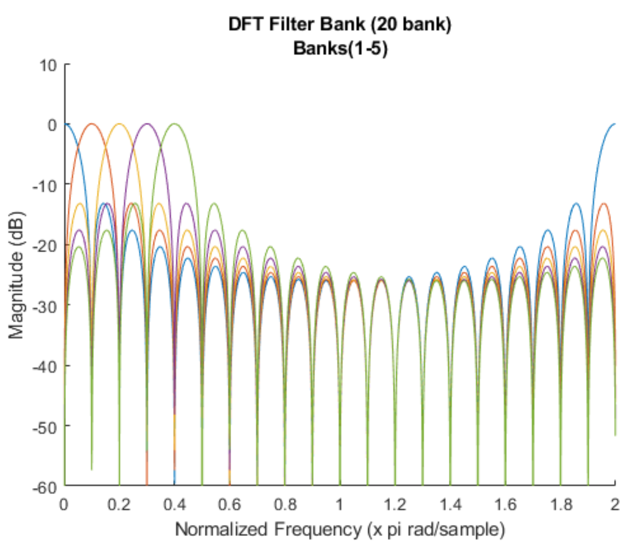
By: Zachary H. Sweep

Contents

Problem 1

Design and implement a 20-band DFT filter bank in MATLAB. Plot the frequency response magnitude for the first 5 sub bands.

```
% Number of banks in filter bank
M = 20;

% M bank DFT filter bank
l = (0:M-1);
k = (0:M-1).';
b_dft = exp(1j*2*pi*l.*k/M);

% Normalize the gain of the dft filter bank impulse responses
b_dft = b_dft/sum(b_dft(:));

% Compute frequency response for first 5 banks
H0 = fft(b_dft(1,:),2048);
H1 = fft(b_dft(2,:),2048);
H2 = fft(b_dft(3,:),2048);
H3 = fft(b_dft(4,:),2048);
H4 = fft(b_dft(5,:),2048);

% Plot magnitude response for first 5 banks
figure(1)
hold on
w = (0:2047)*2*pi/2048;
plot(w/pi,20*log10(abs(H0)));
plot(w/pi,20*log10(abs(H1)));
plot(w/pi,20*log10(abs(H2)));
plot(w/pi,20*log10(abs(H3)));
plot(w/pi,20*log10(abs(H4)));
axis([0 2 -60 10]);
title({'DFT Filter Bank (20 bank)','Banks(1-5)'});
xlabel('Normalized Frequency (x pi rad/sample)');ylabel('Magnitude (dB)');
```



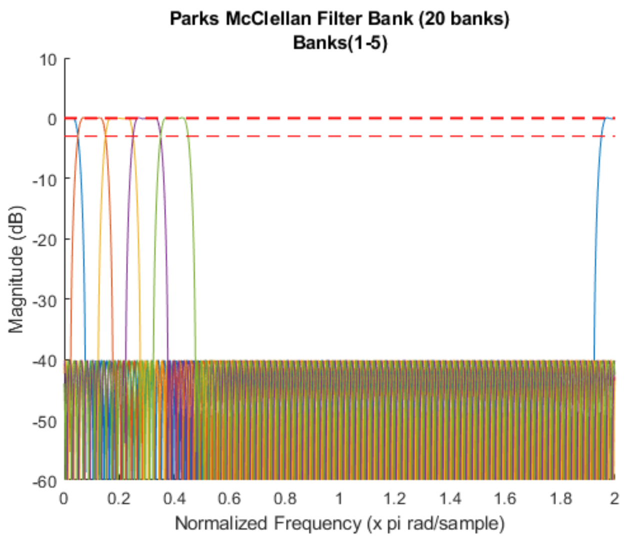
Problem 2

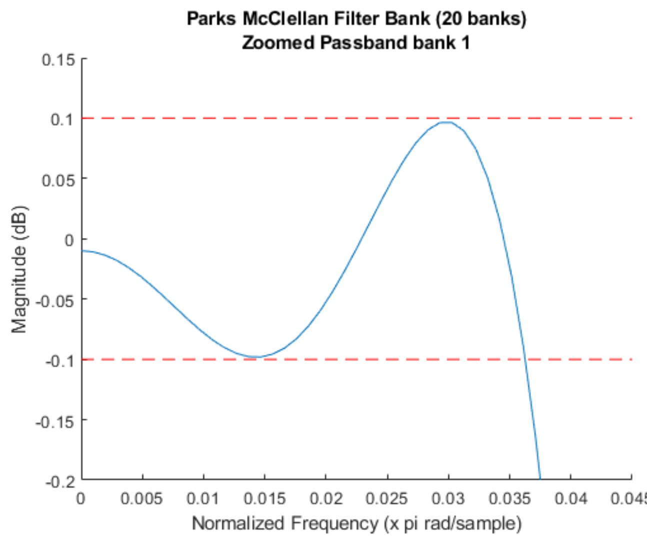
Design and implement a 20-band filter bank using the Parks-McClellan algorithm for the prototype filter. The passband and stopband specifications are as follows: R_p=0.1dB and R_s=40dB. Choose the passband and stopband cutoff frequencies such that each subband overlaps its neighbor at the 3dB crossover point (±0.1dB). Some iteration will likely be required to achieve this. Plot the frequency response magnitude for the first 5 subbands.

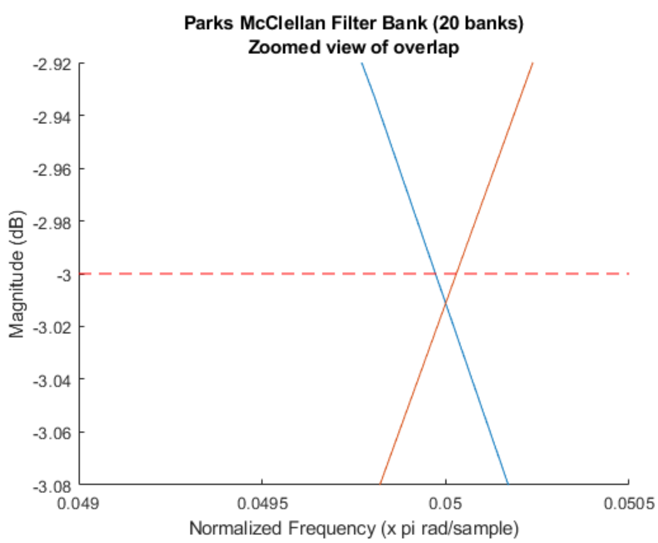
```
% Number of banks in filter bank
M = 20;

% Prototype filter passband and stopband ripple specificaitons
% Rp is assumed to be the peak maximum passband ripple in dB
Rp = 0.1;   %dB
Rs =  40;   %dB

% Find linear value of Rp
Rp_linear = min([(1-10^(-Rp/20)) (10^(Rp/20)-1)]);

% Choose cutoff such that each bank overlaps the neighbor at 3dB.
% This required iterating through different transition band widths as well
% as the the transition band center
offset = pi/50;
cent = (pi/M) +0.0196;
wp = cent - offset;
ws = cent + offset;

% Compute order of prototype filter needed to meet specifications
[n,fo,mo,w] = firpmord([wp/pi ws/pi], [1 0], [Rp_linear 10^(-Rs/20)]);

% Order was surprisingly over estimated for this filter
n = n-1;
pm_order = n;
fprintf('\nParks McClellan Prototype Filter Order: %d\n',pm_order);

% Compute PM prototype filter impulse response
b = firpm(n,fo,mo,w);

% Compute each impulse response for each bank by modulating the prototype
% filter impulse response
b_pm = zeros(M,n+1);
for i=0:M-1
    b_pm(i+1,:) = b.*exp(1j*2*pi*(0:length(b)-1)*i/M);
end

% Compute FFT for first 5 filter banks
H1 = fft(b_pm(1,:),2048);
H2 = fft(b_pm(2,:),2048);
H3 = fft(b_pm(3,:),2048);
H4 = fft(b_pm(4,:),2048);
H5 = fft(b_pm(5,:),2048);

w = (0:2047)*2*pi/2048;

% Plot magnitude response for first 5 banks
figure(2)
hold on;
plot(w/pi,20*log10(abs(H1)));
plot(w/pi,20*log10(abs(H2)));
```

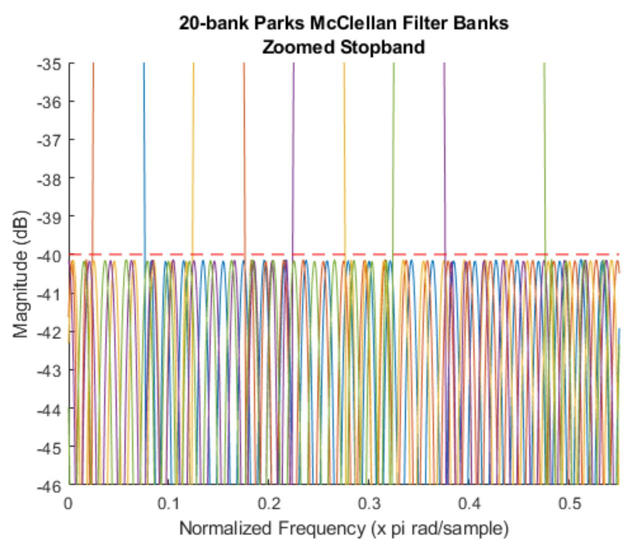
```
plot(w/pi,20*log10(abs(H3)));
plot(w/pi,20*log10(abs(H4)));
plot(w/pi,20*log10(abs(H5)));
line([0 2],[0.1 0.1],'color','red','LineStyle','--');
line([0 2],[-0.1 -0.1],'color','red','LineStyle','--');
line([0 2],[-3 -3],'color','red','LineStyle','--');
axis([0 2 -60 10]);
title({'Parks McClellan Filter Bank (20 banks)','Banks(1-5)'});
xlabel('Normalized Frequency (x pi rad/sample)');ylabel('Magnitude (dB)');

% Plot zoomed view of passband for first 5 banks
figure(3)
hold on;
plot(w/pi,20*log10(abs(H1)));
plot(w/pi,20*log10(abs(H2)));
plot(w/pi,20*log10(abs(H3)));
plot(w/pi,20*log10(abs(H4)));
plot(w/pi,20*log10(abs(H5)));
line([0 2],[0.1 0.1],'color','red','LineStyle','--');
line([0 2],[-0.1 -0.1],'color','red','LineStyle','--');
line([0 2],[-3 -3],'color','red','LineStyle','--');
axis([0 0.045 -0.2 0.15]);
title({'Parks McClellan Filter Bank (20 banks)','Zoomed Passband bank 1'});
xlabel('Normalized Frequency (x pi rad/sample)');ylabel('Magnitude (dB)');

% Plot zoomed view of overlap for banks 1 and 2
figure(4)
hold on;
plot(w/pi,20*log10(abs(H1)));
plot(w/pi,20*log10(abs(H2)));
line([0 2],[0.1 0.1],'color','red','LineStyle','--');
line([0 2],[-0.1 -0.1],'color','red','LineStyle','--');
line([0 2],[-3 -3],'color','red','LineStyle','--');
axis([0.049 0.0505 -3.08 -2.92]);
title({'Parks McClellan Filter Bank (20 banks)','Zoomed view of overlap'});
xlabel('Normalized Frequency (x pi rad/sample)');ylabel('Magnitude (dB)');

% Plot zoomed view of stopband
figure(5)
hold on;
plot(w/pi,20*log10(abs(H1)));
plot(w/pi,20*log10(abs(H2)));
plot(w/pi,20*log10(abs(H3)));
plot(w/pi,20*log10(abs(H4)));
plot(w/pi,20*log10(abs(H5)));
line([0 2],[0.1 0.1],'color','red','LineStyle','--');
line([0 2],[-0.1 -0.1],'color','red','LineStyle','--');
line([0 2],[-3 -3],'color','red','LineStyle','--');
line([0 2],[-40 -40],'color','red','LineStyle','--');
axis([0 0.55 -46 -35]);
title({'20-bank Parks McClellan Filter Banks','Zoomed Stopband'});
xlabel('Normalized Frequency (x pi rad/sample)');ylabel('Magnitude (dB)');
```

```
Parks McClellan Prototype Filter Order: 94
```


Parks McClellan Filter Bank (20 banks)
Banks(1-5)
Magnitude (dB)
10
0
-10
-20
-30
-40
-50
-60
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Normalized Frequency (x pi rad/sample)


Parks McClellan Filter Bank (20 banks)
Zoomed Passband bank 1
Magnitude (dB)
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
0
0.005
0.01
0.015
0.02
0.025
0.03
0.035
0.04
0.045
Normalized Frequency (x pi rad/sample)


Parks McClellan Filter Bank (20 banks)
Zoomed view of overlap
Magnitude (dB)
-2.92
-2.94
-2.96
-2.98
-3
-3.02
-3.04
-3.06
-3.08
0.049
0.0495
0.05
0.0505
Normalized Frequency (x pi rad/sample)


20-bank Parks McClellan Filter Banks
Zoomed Stopband
Magnitude (dB)
-35
-36
-37
-38
-39
-40
-41
-42
-43
-44
-45
-46
0
0.1
0.2
0.3
0.4
0.5
Normalized Frequency (x pi rad/sample)

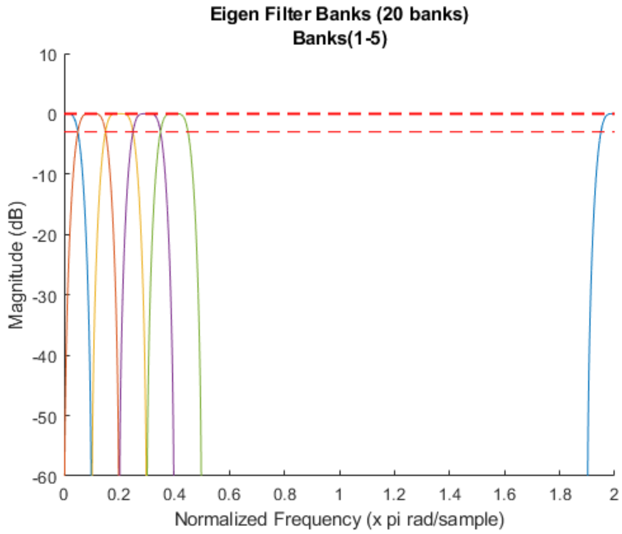
Problem 3

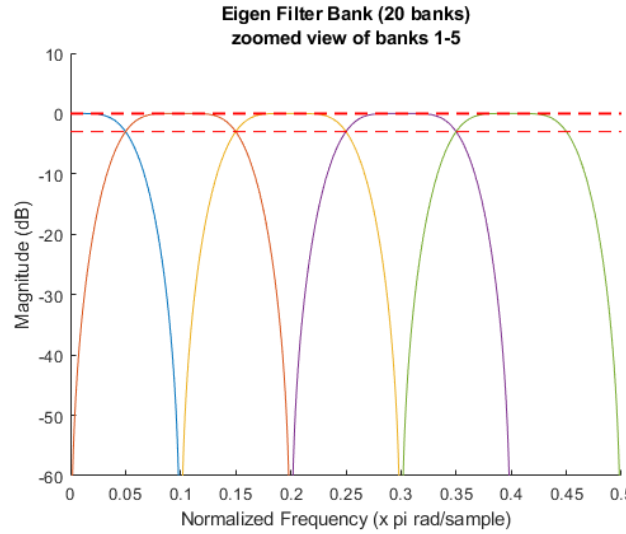
Repeat using the eigenfilter design method for the prototype filter. Fix the filter order at the same value you determined for the Parks-McClellan filter design in part 2. Set the stopband weight value α=0.2. Choose the passband and stopband cutoff frequencies such that each subband overlaps its neighbor at the 3dB crossover point (±0.1dB). Some iteration will likely be required to achieve this. Plot the frequency response magnitude for the first 5 subbands.

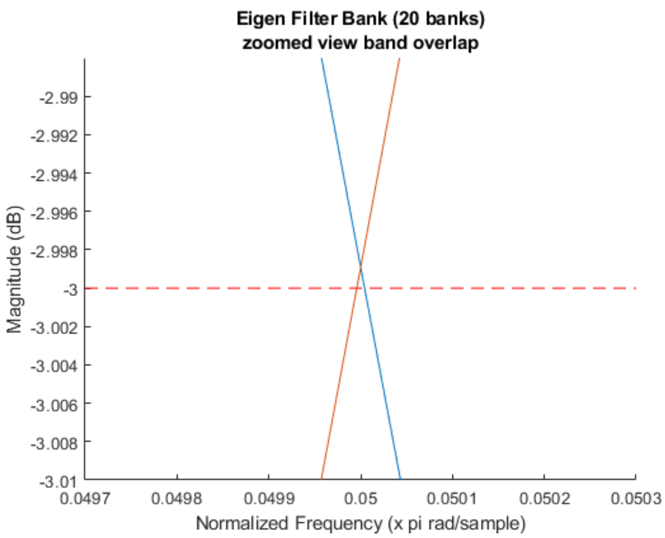
```
% Number of banks in filter bank
MM = 20;

% Choose cutoff such that each bank overlaps the neighbor at 3dB.
% The design of the filter that met the given specifications required
% iterating throught the design process with different values for the
% passband/stopband frequencies such that (wp + ws)/2 = pi/M was satisfied.
offset = pi/24.5;
cent = (pi/MM) + 0.0196;
wp = cent - offset;
ws = cent + offset;

% Fix eigne filter order to be the same as the Parks McClellan filter order
% from problem 2
N = pm_order;

% Since the order of the prototype filter in problem 2 has an even order
% (impulse response has odd length) and even symetry this is a type II
% linear phase FIR filter.
M = N/2;

% Stopband weight alpha = 0.2
alpha = 0.2;

% Pre-allocating impulse response vector
h = zeros(1,N+1);

% Compute c(w) and c(w)*c(w).'
syms w;
cw = cos((0:M)*w).';
cw = cw*cw.';

% Compute Ps Matrix
Ps = zeros(M+1,M+1);
w = linspace(ws,pi,1000);

for m=0:M
    for n=0:M
        % Evaluate cw(m,n) at each value of w
        c = eval(cw(m+1,n+1));

        % Handling cases where c = 1 or 0
        if c==1
            c = ones(length(w),1);
        elseif c==0
            c = zeros(length(w),1);
        end

        % Compute integral
        Ps(m+1,n+1) = (1/pi)*trapz(w,c);
    end
```

```
end

% Compute c(w) and (1-c(w))*(1-c(w)).'
syms w;
cw = cos((0:M)*w).';
cw = (1 - cw)*(1 - cw).';

% Compute Pp Matrix
Pp = zeros(M+1,M+1);
w = linspace(0,wp,1000);

for m=0:M
    for n=0:M
        % Evaluate cw(m,n) at each value of w
        c = eval(cw(m+1,n+1));

        % Handling cases where c = 1 or 0
        if c==1
            c = ones(length(w),1);
        elseif c==0
            c = zeros(length(w),1);
        end

        % Compute Integral
        Pp(m+1,n+1) = (1/pi)*trapz(w,c);
    end
end

% Compute P matrix with weights of alpha = 0.2 for the stop band and
% (1-alpha) for the passband
P = alpha*Ps + (1-alpha)*Pp;

% Compute Eigen Vectors/Values of P
[V,D] = eig(P,'vector');

% Find index of smallest Eigen value in the Eigen value column vector
ind = find(D==min(D));

% Find Eigen Vector containing smallest Eigen value using the index
b = V(:,ind);

% Re-organize bn to get h(n)
% h(M) = b(0)
h(M+1) = b(1);

% h(n) = b(n)/2 for n = 1 to M
h(M+2:end) = b(2:M+1).'/2;
h(1:M) = flip(b(2:M+1).'/2);

% Normalizing the impulse response such that the gain of the filter is
% unity
h = h/sum(h);

% Create the 20-bank filter bank by modulating the eigen filter prototype
b_eig = zeros(MM,N+1);
for i=0:MM-1
    b_eig(i+1,:) = h.*exp(1j*2*pi*(0:length(h)-1)*i/MM);
end

% Compute FFT for first 5 banks
```

```matlab
H1 = fft(b_eig(1,:),8192);
H2 = fft(b_eig(2,:),8192);
H3 = fft(b_eig(3,:),8192);
H4 = fft(b_eig(4,:),8192);
H5 = fft(b_eig(5,:),8192);
w = (0:8191)*2*pi/8192;

% Plot magnitude response for first 5 banks
figure(6)
hold on;
plot(w/pi,20*log10(abs(H1)));
plot(w/pi,20*log10(abs(H2)));
plot(w/pi,20*log10(abs(H3)));
plot(w/pi,20*log10(abs(H4)));
plot(w/pi,20*log10(abs(H5)));
line([0 2],[0.1 0.1],'color','red','LineStyle','--');
line([0 2],[-0.1 -0.1],'color','red','LineStyle','--');
line([0 2],[-3 -3],'color','red','LineStyle','--');
axis([0 2 -60 10]);
title({'Eigen Filter Banks (20 banks)','Banks(1-5)'});
xlabel('Normalized Frequency (x pi rad/sample)');ylabel('Magnitude (dB)');

% Plot zoomed view of first 5 banks
figure(7)
hold on;
plot(w/pi,20*log10(abs(H1)));
plot(w/pi,20*log10(abs(H2)));
plot(w/pi,20*log10(abs(H3)));
plot(w/pi,20*log10(abs(H4)));
plot(w/pi,20*log10(abs(H5)));
line([0 2],[0.1 0.1],'color','red','LineStyle','--');
line([0 2],[-0.1 -0.1],'color','red','LineStyle','--');
line([0 2],[-3 -3],'color','red','LineStyle','--');
axis([0 0.5 -60 10]);
title({'Eigen Filter Bank (20 banks)','zoomed view of banks 1-5'});
xlabel('Normalized Frequency (x pi rad/sample)');ylabel('Magnitude (dB)');

% Plot passband at overlap of banks 1 and 2
figure(8)
hold on;
plot(w/pi,20*log10(abs(H1)));
plot(w/pi,20*log10(abs(H2)));
line([0 2],[0.1 0.1],'color','red','LineStyle','--');
line([0 2],[-0.1 -0.1],'color','red','LineStyle','--');
line([0 2],[-3 -3],'color','red','LineStyle','--');
axis([0.0497 0.0503 -3.01 -2.988]);
title({'Eigen Filter Bank (20 banks)','zoomed view band overlap'});
xlabel('Normalized Frequency (x pi rad/sample)');ylabel('Magnitude (dB)');
```


Eigen Filter Banks (20 banks)
Banks(1-5)
Magnitude (dB)
Normalized Frequency (x pi rad/sample)
10
0
-10
-20
-30
-40
-50
-60
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2


Eigen Filter Bank (20 banks)
zoomed view of banks 1-5
Magnitude (dB)
Normalized Frequency (x pi rad/sample)
10
0
-10
-20
-30
-40
-50
-60
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5


Eigen Filter Bank (20 banks)
zoomed view band overlap
Magnitude (dB)
Normalized Frequency (x pi rad/sample)
-2.99
-2.992
-2.994
-2.996
-2.998
-3
-3.002
-3.004
-3.006
-3.008
-3.01
0.0497
0.0498
0.0499
0.05
0.0501
0.0502
0.0503

Problem 4

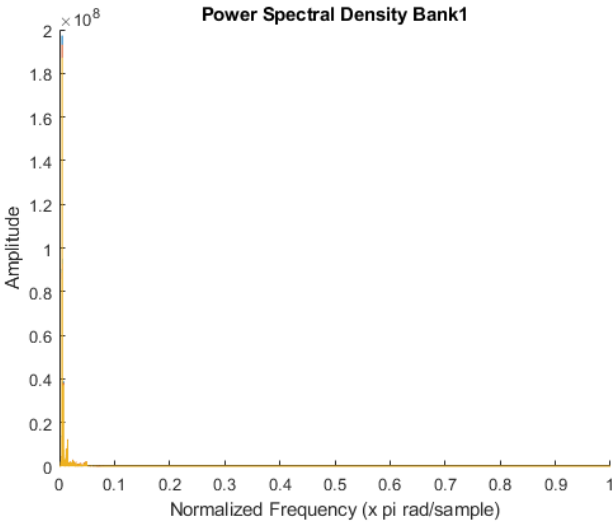
Process the supplied WAV file using the 3 filter banks (DFT, Parks-McClellan and Eigenfilter) you have designed and generate output WAV files for the first 5 subbands. By listening to the output WAV files, compare the 3 filter bank designs. In particular, comment on the degree of isolation between the subbands for each of the filter bank designs. Which one achieves the best isolation? Why? You will find the MATLAB functions *wavread* and *wavwrite* useful for this portion of the homework.

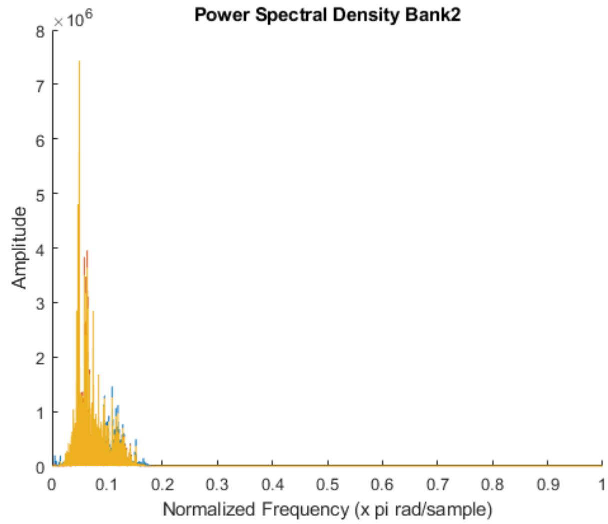
After filtering the provided audio file with the first 5 sub-bands of each filter bank and listening to it, it is very clear that the eigen filter bank has the highest isolation between sub-bands. The eigen filters higher degree of isolation stands out the most when listening to the bank 5 output audio signal. The output of the eigen filter was almost inaudible in the sense that the remaining frequencies in the filtered signal were so high that they could hardly be recognized.

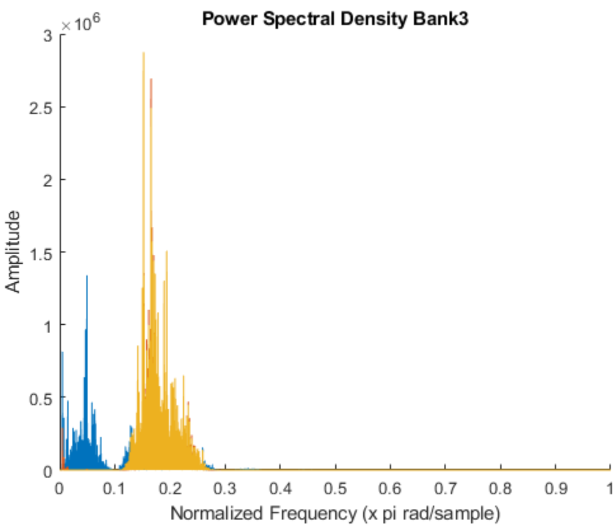
The eigen filter bank has a noticeably higher degree of isolation due to its decaying stopband attenuation. This means that the farther a frequency is from wc, the more it will be attenuated. The degree of isolation can be observed in the power spectral density plots below, especially in the outputs of banks 3, 4, and 5. Notice the power present in the out of band frequencies for the DFT and PM filter outputs compared to the EIG filter output.

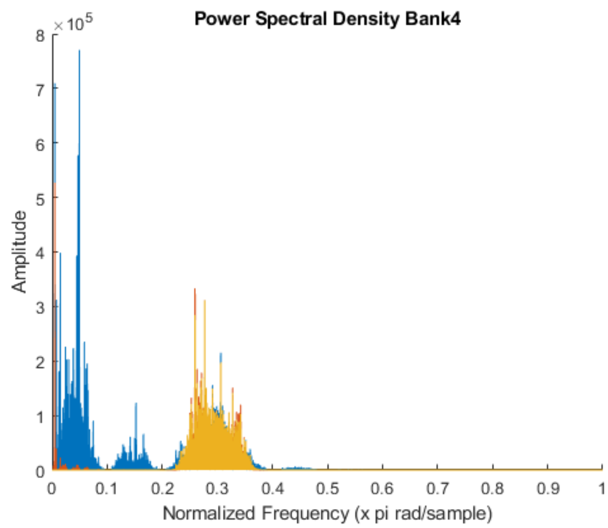
```
% Load the provide audio file to be filtered
[y,fs] = audioread('Misty Mountain Hop Snippet.wav');

% Find length of audio file
len = length(y);

% Create base file path names
f1 = 'recordings/dft_bank_';
f2 = 'recordings/pm_bank_';
f3 = 'recordings/eig_bank_';

% Apply banks 1-5 from each filter bank to the provided audio signal
for i=1:5

    % Apply dft filter bank(i) to input signal y. Normalize the filtered
    % signal between -1 and 1. Write filtered signal to file
    fname = strcat(f1,int2str(i),'.wav');
    y_dft = real(conv(y,b_dft(i,:),'same'));
    y_dft = y_dft./(max(abs(y_dft)));
    audiowrite(fname,y_dft,fs);

    % Apply Parks McClellan filter bank(i) to input signal y. Normalize the
    % filtered signal between -1 and 1. Write filtered signal to file
    fname = strcat(f2,int2str(i),'.wav');
    y_pm = real(conv(y,b_pm(i,:),'same'));
    y_pm = y_pm./(max(abs(y_pm)));
    audiowrite(fname,y_pm,fs);

    % Apply Eigen filter bank(i) to input signal y. Normalize the filtered
    % signal between -1 and 1. Write filtered signal to file
    fname = strcat(f3,int2str(i),'.wav');
    y_eig = real(conv(y,b_eig(i,:),'same'));
    y_eig = y_eig./(max(abs(y_eig)));
    audiowrite(fname,y_eig,fs);

    % Compute FFT for each filter bank output
    H_dft = fft(y_dft);
    H_pm = fft(y_pm);
    H_eig = fft(y_eig);
```

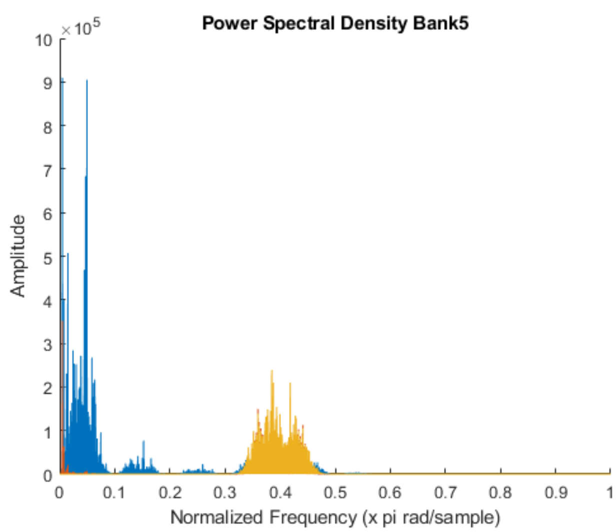
```
    w = (0:len-1)*2*pi/len;
    w = w(1:len/2);

    % Compute power spectral density of signal after
    PSD_dft = H_dft.*conj(H_dft);

    % Compute power spectral density of signal after applying pm filter
    PSD_pm = H_pm.*conj(H_pm);

    % Compute power spectral density of signal after applying eig filter
    PSD_eig = H_eig.*conj(H_eig);

    % Plot power spectral density of filtered signals to see how much
    % power is preset outside the bandwidth of each filter bank. The less
    % power outside of the filter bandwidth means less presence of those
    % frequencies within the filtered signal
    figure(i+10)
    hold on
    plot(w/pi,PSD_dft(1:len/2));
    plot(w/pi,PSD_pm(1:len/2));
    plot(w/pi,PSD_eig(1:len/2));
    hold off
    text = strcat('Power Spectral Density Bank ',int2str(i));
    title(text);xlabel('Normalized Frequency (x pi rad/sample)');
    ylabel('Amplitude');
end

% Computing and plotting magnitude response for bank 3 of each filter bank
% to compare the half power bandwidth.

% Compute magnitude response for bank 3 of each filter bank
H_dft = fft(b_dft(3,:),8192);
H_pm = fft(b_pm(3,:),8192);
H_eig = fft(b_eig(3,:),8192);
w = (0:8191)*2*pi/8192;

% Plot bank 3 magnitude response for each filter bank ontop of each other
figure(88)
hold on
plot(w/pi, 20*log10(abs(H_dft)));
plot(w/pi, 20*log10(abs(H_pm)));
plot(w/pi, 20*log10(abs(H_eig)));
title('Magnitude Response of Bank 3');
xlabel('Normalized Frequency (x pi rad/sample)');ylabel('Magnitude (dB)');
legend('dft bank 3','pm bank 3','eig bank 3');
axis([0.05 0.35 -95 15]);
snapnow
```

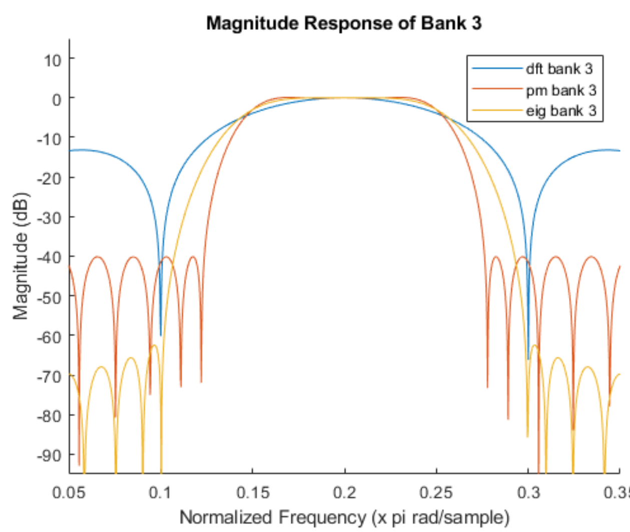












Published with MATLAB® R2017b