Homework #5
Due We 4/01

You must turn in your code as well as output files. Please generate a report that contains the code and output in a single readable format.

Visit the book website to download companion software, including all the example problems.


1. (KLT 6.4)

Solution
The error function can be found using the correlation properties of white noise \( r_{xx}(k) = \sigma^2 \delta(k) = \delta(k) \) input as

\[
\xi = E[d^2(n)] - 2p^T w + w^T R w
\]

\[
= (b_0^2 + b_1^2 + b_2^2) - 2(b_0w_0 + b_1w_1 + b_2w_2) + (w_0^2 + w_1^2 + w_2^2)
\]

using the the following definitions

\[
R = E[x(n)x^T(n)]
\]

\[
p = E[d(n)x(n)]
\]

\[
R = \begin{bmatrix}
        r_{xx}(0) & r_{xx}(1) & r_{xx}(2) \\
        r_{xx}(1) & r_{xx}(0) & r_{xx}(1) \\
        r_{xx}(2) & r_{xx}(1) & r_{xx}(0)
\end{bmatrix}
\]

\[
p = \begin{bmatrix}
        r_{dx}(0) \\
        r_{dx}(1) \\
        r_{dx}(2)
\end{bmatrix}^T
\]

\[
= \begin{bmatrix}
        1 & 0 & 0 \\
        0 & 1 & 0 \\
        0 & 0 & 1
\end{bmatrix}
\]

\[
= \begin{bmatrix}
        b_0 \\
        b_1 \\
        b_2
\end{bmatrix} = \begin{bmatrix}
        0.2 \\
        0.5 \\
        0.3
\end{bmatrix}^T.
\]

The optimal solution can be found by solving the least squares problem

\[
w^o = R^{-1} p = p = \begin{bmatrix}
        0.2 \\
        0.5 \\
        0.3
\end{bmatrix}^T.
\]

The minimum MSE can be found as

\[
\xi_{min} = E[d^2(n)] - p^T w^o = (b_0^2 + b_1^2 + b_2^2) - p^T w^o = 0.
\]

The results can be verified using Matlab as shown below (Fig. [1]).

2. (KLT 6.6)

Solution
While this problem asks you to compute the correlation using Matlab xcorr.m you should know how to compute this by hand. Note that sample correlation is the same as convolution without and flipping.

(a) \( r_{xy}(k) = [0 \ 0 \ 0 \ 3 \ 7 \ -11 \ 14 \ 13 \ -15 \ 28 \ 6 \ -2 \ 21 \ 12 \ 12 \ 6 \ 4]^T \).

(b) \( r_{xx}(k) = [2 \ 10 \ 12 \ 5 \ -1 \ 19 \ 8 \ 15 \ 26 \ 15 \ 8 \ 19 \ -1 \ 5 \ 12 \ 10 \ 2]^T \).

(c) \( r_{yy}(k) = [6 \ -7 \ 16 \ 2 \ -10 \ 35 \ -10 \ 2 \ 16 \ -7 \ 6]^T \).
3. (KLT 10.3)  
Solution  
The compression rate is defined as  

\[
\text{compression rate} = \frac{\text{uncompressed rate}}{\text{compressed rate}}.
\]

(a) The uncompressed rate for two channels can be computed using 16-bit PCM @ 48 kHz:  

\[
\text{compression rate} = \frac{2 \times 48 \times 16}{128} = \frac{1536}{128} \text{ kpbs} = 12.
\]

(b) The uncompressed rate for two channels can be computed using 16-bit PCM @ 32 kHz:  

\[
\text{compression rate} = \frac{2 \times 32 \times 16}{128} = \frac{1024}{128} = 8.
\]

4. (KLT 10.7)  
Solution  
The input, output, and filter spectrum are provided in Fig. 2. Notice that the input spectrum is constant across the spectrum because it is white noise and that the output spectrum and filter are matched.

Code

```matlab
Fs = 16000; % Sampling frequency
[az1, bz1] = ShelfFilter(1000, -10, 1, Fs, 'L');
[az2, bz2] = PeakFilter(4000, 10, 1, Fs);
[az3, bz3] = ShelfFilter(7000, 10, 1, Fs, 'H');

% Combine 3 cascaded IIR filter
da = cascadc2x2(az1, az2);
db = cascadc2x2(bz1, bz2);
az = cascadc4x2(da, az3);
.bz = cascadc4x2(db, bz3);
```

Figure 1: KLT 6.4
fvtool(bz,az,bz1,az1,bz2,az2,bz3,az3);
title('Magnitude Response (Bass Treble Peak Filters with Fs =16000)');

% replace with white noise
% fid1 = fopen('audioIn.pcm','rb');
% x = fread(fid1,'short');
% fclose(fid1);
% x = randn(Fs*3,1);
% x = x/(max(x)) * 32767;
y = filter(bz,az,x); % IIR Filtering

l=length(x);
out=zeros(l, 2);
out(:,1)=x/32767; % Original audio in left channel
out(:,2)=y/32767; % Equalized audio in right channel
%wavwrite(out, Fs, [odir, '/audioOut.wav']); % Write to output

%disp('play input audio (left channel) vs. output audio (right channel)');
%sound(out, Fs);

% Analyze
h=figure; spectrum(x/32767); grid;
title('Power Spectral Density')
axis([0 1 1e-3 1]);
hline = findobj(gcf, 'type', 'line');
set(hline,'LineWidth',3)
bmSaveFigure(h, 'KLT10_7_pxx');

h=figure; spectrum(y/32767); grid;
title('Power Spectral Density')
axis([0 1 1e-3 1]);
hline = findobj(gcf, 'type', 'line');
set(hline,'LineWidth',3)
bmSaveFigure(h, 'KLT10_7_pyy');

[H,W] = freqz(bz, az, 1024, Fs);
h=figure;
plot(2*W/Fs, 20*log10(abs(H)), 'linewidth', 3);
grid on;
xlabel('Normalized Frequency');
ylabel('Magnitude [dB]');
bmSaveFigure(h, 'KLT10_7_filt');
(a) Input Power Spectrum $P_{x,x}$  
(b) Output Power Spectrum $P_{y,y}$  
(c) Filter Spectrum

Figure 2: KLT 10.7