Advanced Digital Signal Processing

Computer Experiment #2

Due Date: Jan 6 (Friday), 13:00

Problem 1. Hum Noise Removal

1. Problem: Power line hum is a sound associated with alternating current at the frequency of the mains electricity. The fundamental frequency of this sound is usually 60 Hz. The sound often has heavy harmonic content above 60 Hz. Because of the presence of mains current in mains-powered audio equipment as well as ubiquitous AC electromagnetic fields from nearby appliances and wiring, the 60 Hz electrical noise can get into audio systems, and is heard as mains hum from their speakers. Mains hum may also be heard coming from powerful electric power grid equipment such as utility transformers, caused by mechanical vibrations induced by the powerful AC current in them. This project is concerned with the removal of this annoying 60 Hz hum noise. Fig. 2 shows the frequency response of a 60 Hz notch filter.

Figure 2. Response of 60-Hz notch filter

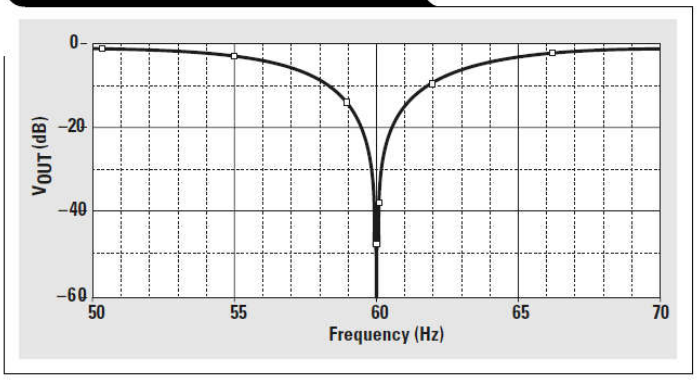
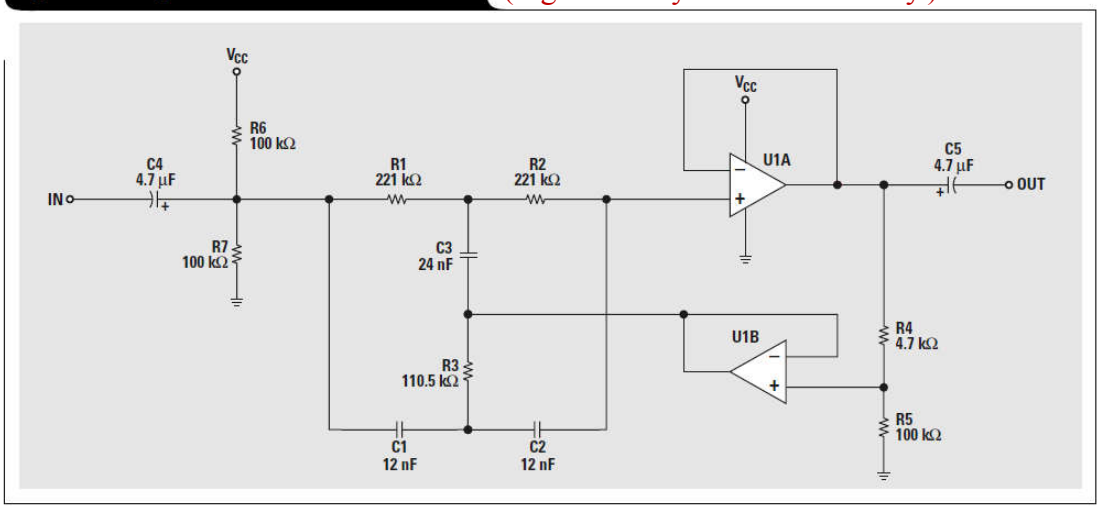


Figure 1. Configuration of a 60-Hz notch filter (Fig. 1 is for your reference only.)



analog filter

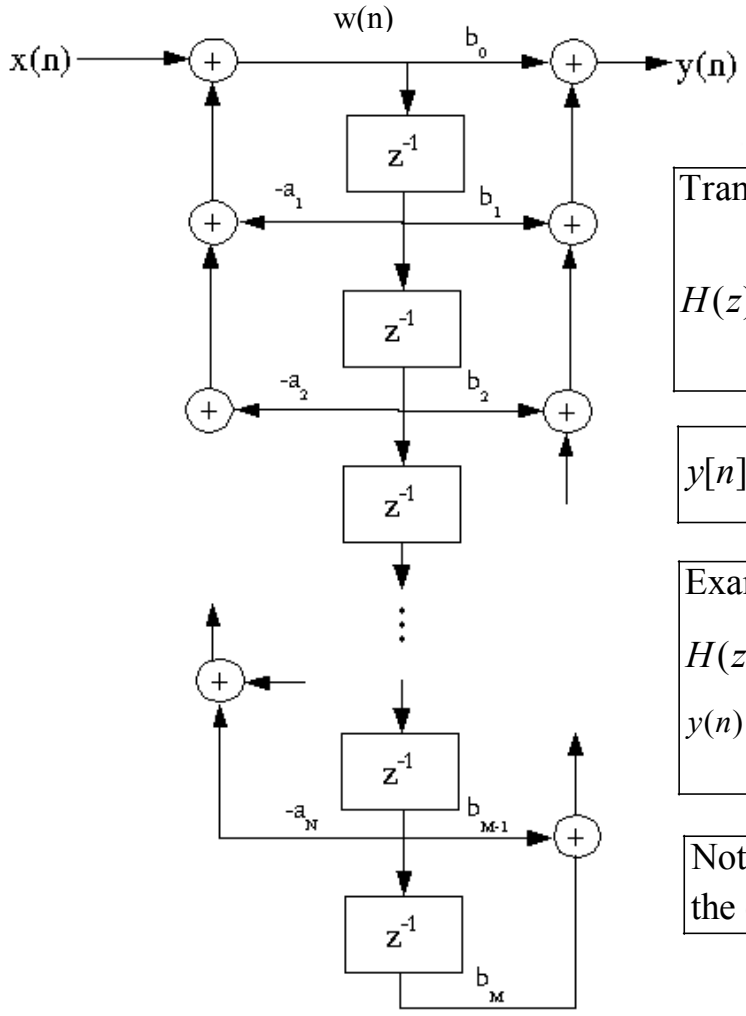
2. Goal: Given a sound file (stored in the .wav format) with hum noise, manage to remove or reduce the 60 Hz hum..

3. Method:

(a) Design a digital IIR notch filter to remove the 60 Hz hum. The notch filter is simply a bandstop filter with a very narrow passband. You may determine your own filter specifications or try the following specs.: (Note that you should modify the specifications if necessary.)

- passband edges (ω_P): 57 Hz and 63 Hz
- stopband edges (ω_S): 59.5 Hz and 60.5 Hz
- passband ripple: 0.5 dB
- stopband attenuation: 40 dB

(b) Implement the IIR filter in the time-domain using the direct form-II structure as shown below.



Transfer Function:

$$H(z)=\frac{\sum_{k=0}^{M} b_k z^{-k}}{1+\sum_{k=1}^{N} a_k z^{-k}}$$

$$y[n]=-\sum_{k=1}^{N} a_k y[n-k]+\sum_{k=0}^{M} b_k x[n-k]$$

Example:

$$H(z)=\frac{4.5+12.5z^{-1}-9.47z^{-2}}{1+3.24z^{-1}-7.6z^{-2}+8.92z^{-3}}$$

$$y(n)=-3.24y(n-1)+7.6y(n-2)-8.92y(n-3)+4.5x(n)+12.5x(n-1)-9.47x(n-2)$$

Note the **negative sign** of the coefficients $a_k, k=1,\cdots,N.$

(c) difference equations for the implementation: (with **zero initial states**)

$$w(n)=-a_1 w(n-1)-a_2 w(n-2)-\cdots-a_N w(n-N)+x(n)$$
$$y(n)=b_0 w(n)+b_1 w(n-1)+\cdots+b_M w(n-M)$$

(d) Listen carefully to the sound files to compare the results before and after the hum noise has been removed.

4. Sample Program: A prototype program in Octave is given here for your reference.

```
clear all;
[xin, fs] = audioread('test_filename.wav');   % fs is the sampling rate
x = xin(:,1);

% Design the IIR Notch Filter for Hum Removal
wp = [57 63]/(fs/2);   % 2*pi* (wp/fs) / pi, normalized passband frequencies
ws = [59.5 60.5]/(fs/2);   % 2*pi* (ws/fs) / pi, normalized stopband frequencies
rp = 0.5;   % passband ripple
rs = 40;   % stopband attenuation
...
Your Own Code
...
% Perform the filtering
...
...
% Do NOT use the built-in function filter() to carry out the filtering process.
% Instead, write your own code to fulfill the task. (i.e., write a new myfilter() function)
out = filter(b, a, x); % b is the coefficient vector for the numerator and
                       % a is the coefficient vector for the denominator
out_scaled = out / max(abs(out)); % amplitude scaling
audiowrite('your_filename.wav', out_scaled, fs);
```

5. Removing Hum Noise with Higher-Order Harmonics: Usually higher-order harmonics also exist in the hum noise, and this phenomenon is due to nonlinearities in the signal chain. Consequently, in addition to the 60 Hz fundamental frequency, we also need to suppress the higher-order harmonic hum components, for instance, 120 Hz, 180 Hz, 240 Hz, etc. In this project, we consider a simpler case that the hum noise only consists of 60 Hz and 120 Hz components in the second test signal. Therefore, to remove these two undesired hum noise components, you need to design two notch filters and put them in cascade, as shown in Fig. 3.

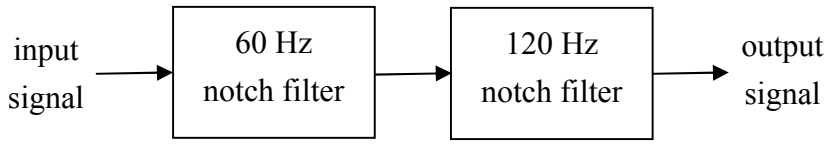


Fig. 3 Filtering process for the second test signal.

6. Key Issues to Be Explored and Discussed:

(1) Do frequency responses of designed filters meet the required specifications?

(2) computational efficiency of the filtering process

(3) BIBO stability of the designed filters

(4) coefficient quantization effects and round-off errors in the filtering process (optional)

7. Report: The following items should be included in your report:

- **tables of all filter coefficients**, including both the numerators and the denominators
- **frequency responses** of the designed filters, with the design specifications verified
- short segments of the filtered waveforms compared to the original waveforms in the time-domain
- Fourier spectra of both the input and the processed signals
- **the filtered signals in the .wav format**

Problem 2. Adaptive Noise Cancellation

Adaptive Noise Cancellation

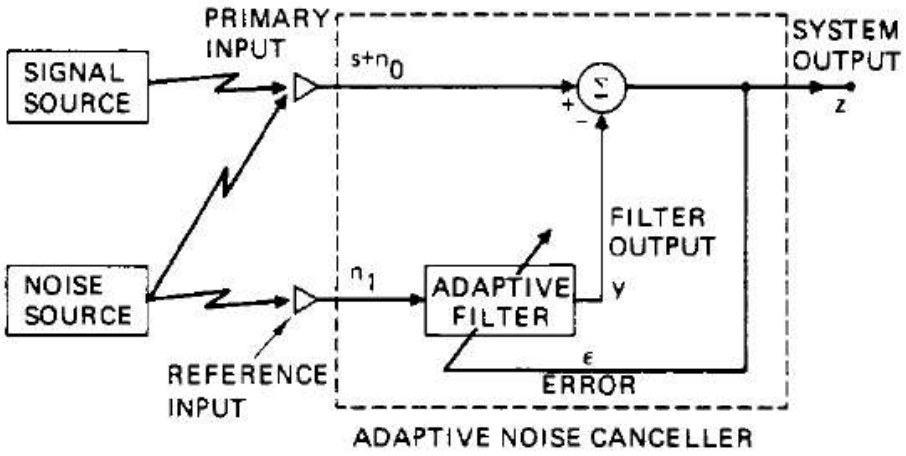


Fig. 1. The adaptive noise cancelling concept.

III. The Concept of Adaptive Noise Cancelling

Fig. 1 shows the basic problem and the adaptive noise cancelling solution to it. A signal s is transmitted over a channel to a sensor that also receives a noise n_0 uncorrelated with the signal. The combined signal and noise $s + n_0$ form the primary input to the canceller. A second sensor receives a noise n_1 uncorrelated with the signal but correlated in some unknown way with the noise n_0. This sensor provides the reference input to the canceller. The noise n_1 is filtered to produce an output y that is as close a replica as possible of n_0. This output is subtracted from the primary input $s + n_0$ to produce the system output $z = s + n_0 - y$.

(from B. Widrow, et al., "Adaptive Noise Cancelling: Principles and Applications, " Proc. IEEE, 1975)

1. Problem: In this project, you learn to apply adaptive filtering technique to reduce noise in the received source signal. When the desired signal and noise occupy fixed and separate frequency bands, conventional linear filters with fixed coefficients are normally used to extract the desired signal. However, there are certain situations that fixed filters are not applicable. For example, there is spectral overlap between the desired signal and the noise, or the frequency band occupied by the noise is unknown or even varies with time. In these circumstances, a better approach is to utilize the adaptive filtering concept, where the filter coefficients are automatically adjusted to achieve the design goal. The mechanisms used to modify the filter coefficients in a suitable way are called *adaptive algorithms*. The well-known LMS algorithm is considered in this project.

2. Goal: Given a recorded sound file (stored in the .wav format) which contains a piece of speech signal contaminated by noise, apply the adaptive noise cancellation scheme to retrieve the desired speech signal.

3. Method: Assume that the *primary input* is $x(n) = d(n) + v_1(n)$, where $d(n)$ is the desired signal and $v_1(n)$ is a noise component to be suppressed. The *reference input* is responsible for picking up the noise source $v_2(n)$. In an audio environment, both the primary and reference signals are obtained through microphones. In the adaptive noise cancellation framework, it is commonly assumed that $v_2(n)$ and $v_1(n)$ are correlated, whereas $v_2(n)$ and $d(n)$ are uncorrelated. As a consequence, the output of the adaptive filter tends to produce a good estimate of $v_1(n)$, i.e., it will generate $\hat{v}_1(n)$. After subtracting $\hat{v}_1(n)$ from $x(n)$, a high-quality estimate of the desired signal $\hat{d}(n)$ can be acquired.

The adaptive filtering algorithm used in this project is the LMS (Least Mean-squares) algorithm which is briefly described here under the adaptive noise cancellation setting. Let the filter coefficients w(n) and the input to the filter $v_2(n)$ be both written in vector form as

$$\mathbf{w}_n = [w_n(0), w_n(1), \cdots, w_n(p)]^T, \quad \text{and}$$
$$\mathbf{v}_2(n) = [v_2(n), v_2(n-1), \cdots, v_2(n-p)]^T,$$

where p is the order of the filter. Then the output of the filter is given by

$$y(n) = \sum_{k=0}^{p} w_n(k) v_2(n-k) = \mathbf{w}_n{}^T \mathbf{v}_2(n)$$

The error between the input source signal and the filter output is expressed as

$$e(n) = x(n) - y(n).$$

Using the above notations, the **LMS algorithm** can now be stated as the following computational procedure:

Parameters: p = filter order, μ = step size

Initialization: $\mathbf{w}_0 = \mathbf{0}$ or user-specified values

Computation: For n = 0, 1, 2, 3, ...

(1) $y(n) = \mathbf{w}_n{}^T \mathbf{v}_2(n)$

(2) $e(n) = x(n) - y(n)$

(3) $\mathbf{w}_{n+1} = \mathbf{w}_n + \mu e(n) \mathbf{v}_2(n)$

Choose p = 10 and $\mu = 0.001 \sim 0.02$ in this project. Note that $v_2(n)$ is the reference signal and $x(n)$ is the primary signal, namely, the received source signal. Fig. 1 shows typical learning and adaptation characteristics of filter coefficients.

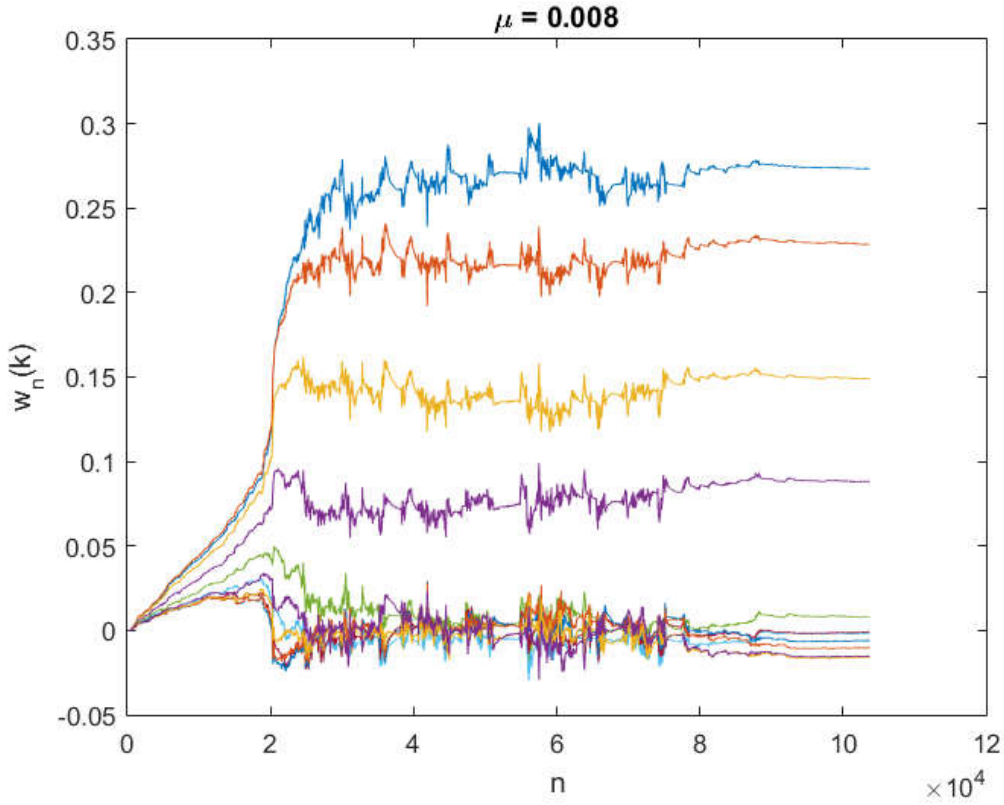


Fig. 1 Adaptation of filter coefficients

4. Another Method (Normalized LMS Algorithm): The normalized LMS algorithm has the merit of determining the step size directly, without knowing the maximum eigenvalue of the autocorrelation matrix of the input signal. The distinction of the normalized LMS algorithm lies in the weight vector update equation, and this modified algorithm can now be listed as the following computational procedure:

Parameters: p = filter order, β = step size, ε = very small positive number

Initialization: $\mathbf{w}_0 = \mathbf{0}$ or user-specified values (e.g., random numbers)

Computation: For n = 0, 1, 2, 3, ...

(1) $y(n) = \mathbf{w}_n{}^T \mathbf{v}_2(n)$

(2) $e(n) = x(n) - y(n)$

(3) $\mathbf{w}_{n+1} = \mathbf{w}_n + \beta\, e(n) \dfrac{\mathbf{v}_2(n)}{\varepsilon + \left\|\mathbf{v}_2(n)\right\|^2}$

5. Key Issues to Be Explored and Discussed:

(1) choice of the step size

(2) choice of initial values for filter coefficients

(3) convergence behavior of filter coefficients

(4) rate of convergence

(5) comparisons among different algorithms

6. Report: The following items should be included in your report:

- curves showing adaptation of filter coefficients for different step sizes
- discussion of the effect of the step size
- comparison of the performance of the two algorithms under exploration
- processed sound files

Note:

The complete report package, including report, programs, and output signals, should be mailed to

dipwork405@gmail.com

prior to the deadline.