[1] Implementing an LMS adaptive equalizer with finite numerical precision:

Consider an M-coefficient, real LMS adaptive equalizer:

\[ W(n+1) = W(n) + \alpha \cdot e(n) \cdot X(n), \]

where: \( X(n) \equiv [x(n-1), \ldots, x(n-M)]^T \); \( e(n) = y(n) - W^T(n) \cdot X(n) \) and \( \alpha \) denotes the positive step-size. In the following it is assumed that the reference input power has been normalized to unity: \( \sigma_x^2 = E[x(n)^2] = 1 \) and that \( y(n) \) is a random sequence of \( \pm 1 \)'s (in practice \( y(n) \) is either a training sequence, in the case of reference-directed equalization, or in the case of decision-directed equalization \( y(n) \) is a quantized (“sliced”) version of \( W^T(n) \cdot X(n) \), i.e., \( y(n) = +1 \), if \( W^T(n) \cdot X(n) \geq 0 \) or \( y(n) = -1 \), if \( W^T(n) \cdot X(n) < 0 \).

(a) (15%) Assuming an average LMS time constant of \( \tau_{\text{LMS}} = 1/(2 \cdot \alpha) \) samples and a desired mean-squared (“slicing”) error of \( \sigma_e^2 = E[e(n)^2] = 10^{-3} \), how many bits \( B \) are required to achieve these specifications assuming the desired (average) LMS filter time constant is 10 msec and that a 1 MHz sampling rate is used? What if the sampling rate is reduced to 10 kHz (with all other parameters remaining the same)?

(b) (20%) In an attempt to reduce \( B \), we can try bit-shifting the LMS weight vector via the operation: \( \hat{W}(n) = 2^B \cdot W(n) \). Derive an update equation for \( \hat{W}(n) \) in terms of \( \alpha \), \( B \), \( e(n) \) and \( X(n) \). Do you think the resulting update equation for \( \hat{W}(n) \) can be implemented with fewer than \( B \) bits?

The number of bits \( B \) must be chosen such that the “freezing” level \( \epsilon_F \) of the LMS mean square error output is 1/10 the desired mean-squared error, \( \sigma_e^2 \). The “freezing” level \( \epsilon_F \) is defined by:

\[ \alpha \cdot \sqrt{\epsilon_F \cdot \sigma_x} = 2^{-B}. \]

[2] (30%) The LMS algorithm as a time invariant filter:
Consider a one-weight, LMS adaptive algorithm:

\[ W_0(n+1) = W_0(n) + \alpha \cdot e(n) \cdot x(n), \]

where: \( e(n) = d(n) - W_0(n) \cdot x(n) \) and \( \alpha \) denotes the positive step-size.

Let \( x(n) = 1.0 \), a constant unit amplitude waveform. Show that for this reference input, \( e(n) \) and \( d(n) \) can be related by a linear, time invariant filter, \( h_0 \), such that: \( e = h_0 \ast d \). Find the Fourier transform of this filter:

\[ H_0(e^{j\omega}) = \sum_{n=-\infty}^{\infty} h_0(n) \cdot e^{-j\omega n}. \]

Categorize this filter as lowpass, bandpass, highpass, etc.

[3] The LMS LPEF algorithm – a computer exercise:

Consider the LPEF noise canceling architecture you addressed in the first project:

\[ s(n) \]
\[ x(n) \]
\[ \text{Unit delay} \]
\[ y(n) \]
\[ LPEF \]
\[ y_0(n) \]
\[ \text{L-tap FIR filter} \]
\[ x(n-1) \]

The LMS adaptive solution to this is:

Compute noise canceller output:

\[ y(n) = x(n) - \sum_{k=0}^{L-1} W_k(n) \cdot x(n - k - 1) \]

Update the prediction coefficients:

\[ W_k(n+1) = W_k(n) + \mu \cdot e(n) \cdot x(n - k - 1), \quad 0 \leq k \leq L - 1 \]

Compute the broadband gain:
\[
G_{\text{LPEF}}(n) = \frac{1}{\left(1 / \psi_{\alpha}(0)\right) \cdot \sum_{l=0}^{L-1} \{|W_{\ell}(n)|^2 + |1 - e^{-j\omega_{\ell}} \sum_{l=0}^{L-1} W_{\ell}(n)e^{-j\omega_{\ell}}|^2\}}
\]

In the above, \( \mu \) is the step size. To avoid instability, we set \( \mu = \alpha / (L \phi_{\alpha}(0)) \) where \( \alpha \approx 0.1 \) First using \( L = 50 \) and the all-zero weight initialization (\( w = \text{zeros}(1,50) \)), plot the broadband gain, \( 10 \log_{10} \{|G_{\text{LPEF}}(n)|\} \), from \( n = 1 \) to \( N = 10000 \) (all other parameters are as defined in the first project – note that you will need to generate a single realization of \( x \) as in part 3 of the first project). Compare this with the ideal gain you found in the first project: \( 10 \log_{10} \{2501\} \approx 34 \text{ dB} \). Second use the all-pass weight initialization (\( w = \text{zeros}(1,50); w(25) = 1; \)) and compare your results with the all-zero initialization. Discuss your findings.