LUNDS TEKNISKA HÖGSKOLA
Lunds universitet

Lund University–Lunds Tekniska Högskola

Optimum and Adaptive Signal Processing

HT 2024

LAB 2

Adaptive Line Enhancer

Linnéa Larsson & Mikael Swartling
2013

Department of Biomedical Engineering, Lund University, Sweden

1 Introduction

One main objective within adaptive signal processing is noise suppression, i.e., the detection of an information-bearing signal in noise. This lab is using an adaptive line enhancer (ALE) to do real-time signal processing.

The ALE is a special form of adaptive noise canceler that is designed to suppress the wide-band noise component of the input, while passing the narrow-band signal component with little attenuation. An ALE consists of the interconnection of a delay element and a linear predictor, as is illustrated in the block diagram in Fig. 1. The input signal $d(n)$ is formed of the desired signal $s(n)$ which is periodic, i.e. narrow-banded, and the disturbing noise $v(n)$ which is colored, i.e., wide-banded. The predictor output $\hat{d}(n)$ is subtracted from the input signal $d(n)$ to produce the estimation error $e(n)$. This estimation error is, in turn, used to adaptively control the predictor. The predictor input $u(n)$ equals $d(n - \Delta)$, which is the input signal delayed with Δ samples. The delay Δ has to be chosen such that the noise in the original input signal $d(n)$ and in the delayed predictor input $d(n-\Delta) \dots d(n-\Delta-M)$, where M is the filter length, is uncorrelated, so that it can be suppressed by the linear predictor. This linear predictor is a FIR filter whose tap weights are controlled by the adaptive algorithm, which in this lab will be implemented as least mean square (LMS) algorithm.

The lab consists of three parts. The first part are preparation assignments. Besides of that you read the lab instructions carefully when attending the lab, those assignments are expected to be completed **before** the lab. The second part of the lab is aiming for a better understanding of the LMS and the leaky LMS with the help of Matlab. During the third part of the lab a realtime ALE will be tested under different conditions.

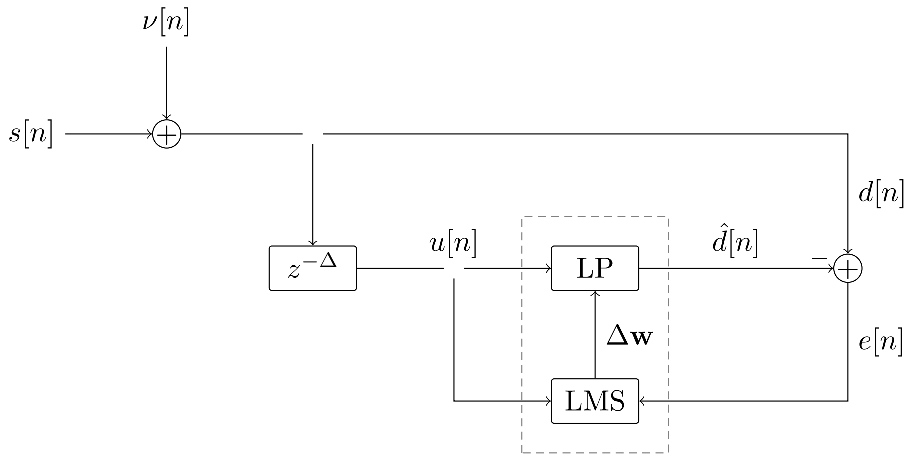

Figure 1: Adaptive Line Enhancer (ALE)

2 Preparation assignments

Preparation assignment 1 In a real environment, the input signal $d(n)$ to an ALE is typically distorted by additional noise and possibly 50 Hz distortions from the power line. Why is it important to remove possible 50 Hz distortions, i.e., what difficulties would they introduce to the ALE?

. . . .

Preparation assignment 2 The choice of the step size μ of the LMS algorithm is critical for its convergence. The algorithm converges when the step size is within the following range:

$$0 \ < \ \mu \ < \ \frac{2}{\lambda_{max}} \ ,$$

where λ_{max} is the largest eigenvalue of the autocorrelation matrix $\mathbf{R}$ of the input signal $u(n)$. Often, the eigenvalues of $\mathbf{R}$ are not known, and an approximation is used instead:

$$0 \ < \ \mu \ < \ \frac{2}{\sum_{k=0}^{M-1} E\{|u(n-k)|^2\}} \ ,$$

where M is the length of the FIR filter used for linear prediction. Explain why this is a reasonable approximation.

. . . .

In this lab, M is chosen to 100. For $u(n)$ being a signal which can take values between -1 and 1 and whose probability distribution is not known, determine the step size μ_{max} that in any case assures a stable LMS.

Preparation assignment 3 As mentioned in preparation assignment 2, the largest eigenvalue of the autocorrelation matrix $\mathbf{R}$ of the input signal $u(n)$ can be used to determine the convergence of the LMS algorithm. Besides of λ_{max}, also the eigenvalue spread $\chi = \frac{\lambda_{max}}{\lambda_{min}}$ plays an important role. It is known that the larger the eigenvalue spread, the slower is the adaption of the LMS algorithm. Explain this statement.

. . . .

Further on, determine the eigenvalue spread $\chi = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}}$ of the correlation matrix R_x (dimension 2×2) of an AR(2)-process whose process generator is given by

$$H(z) = \frac{1}{1 - a_1 z^{-1} + 0.95 z^{-2}}$$

with *(i)* $a_1 = 0.195$, *(ii)* $a_1 = 0.975$, and *(iii)* $a_1 = 1.9114$.

3 Lab assignments

3.1 Preprocessing

As indicated in preparation assignment 1, a filter is needed to suppress low-frequency noise and 50 Hz distortions that were eventually introduced in $d(n)$. Thus, a high-pass filter is applied to

$d(n)$ before proceeding it to the ALE. The coefficients of this filter can be calculated in Matlab with the Filter Design & Analysis Tool.

Assignment 1 Start Matlab and type `fdatool` in order to open the graphical user interface (GUI) of this tool. Design a FIR equiripple high-pass filter with order 50 that reliably suppresses frequencies below 50 Hz and passes through frequencies above 200 Hz at a sampling frequency of 8 kHz. After looking at the magnitude response, select the pole-zero plot in the *Analysis* menu. What indicates that your filter is a FIR and high-pass filter?

3.2 Parameters influencing an adaptive system

This part of the lab deals with the influence of different parameters, like the step size μ or the eigenvalue spread χ, on the adaptation. Furthermore, we shall compare the behaviour of the LMS algorithm and the leaky LMS algorithm.

Assignment 2 We start with studying the influence of μ on the adaptation of a one-step predictor of an AR(2)-process. Run the Matlab script `lms1` and choose the AR(2)-process of preparation assignment 3. Adjust the remaining parameters using `help lms1`. The dashed line shows the minimum mean square error.

Identify the remaining curves.

....

Identify and describe the curves for different μ values.

....

Vary μ and the other parameters.

Conclusion: The larger μ the quicker the adaptation and the larger the excess error.

Assignment 3 Now we will investigate the influence of the eigenvalue spread on the adaptation. Start `lms2`.

Identify and describe the different curves of the AR(2) process.

....

Conclusion: The larger the eigenvalue spread the slower the adaptation.

Assignment 4 We will now shortly look over a variant of the LMS algorithm. The leaky algorithm applies the update equation

$$\mathbf{w}_{n+1} = (1 - \mu\gamma)\mathbf{w}_n + \mu e(n)\mathbf{x}^*(n),$$

where the only difference compared to the LMS algorithm is the so called leakage factor $(1-\mu\gamma)$. The value of the leakage factor is typically close to 1. The error $\xi = E\{e^2(n)\}$ can be written as

$$\xi = \xi_{\min} + (\mathbf{w} - \mathbf{w_o})^H \mathbf{R}_x(\mathbf{w} - \mathbf{w_o}) = \xi_{\min} + (\mathbf{V}^H(\mathbf{w} - \mathbf{w_o}))^H \mathbf{\Lambda} \mathbf{V}^H(\mathbf{w} - \mathbf{w_o})$$

using the eigenvalue decomposition $\mathbf{R}_x = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^H$. If $\mathbf{R}_x$ is singular, i.e., there exists an eigenvalue $\lambda_i = 0$, the error can be $\xi = \xi_{\min}$ although $\mathbf{V}^H(\mathbf{w} - \mathbf{w_o}) \neq 0$, i.e., $\mathbf{w}$ is not equal to the optimum weight vector. This means that $\mathbf{w}$ can "run away" without increasing ξ. The idea of the leaky LMS algorithm is to avoid this problem.

Start `leaky2`. Why do the leaky LMS algorithm's filter coefficients remain constant while the LMS algorithm's coefficients run away?

....

Conclusion: The advantage of the leaky LMS algorithm compared to the LMS algorithm is that it avoids the drift of the weights. The disadvantage of the leaky LMS is its "bias", i.e., $E\{\mathbf{w}(n)\} \nrightarrow \mathbf{w_o}$.

3.3 Application example: Adaptive Line Enhancer

In this part of the lab, we will investigate the performance of a realtime adaptive line enhancer.

Starting the ALE

Start the ALE from Matlab with `ale`. From the application you can select:

- what kind signal to simulate and enhance,
- what to display in the graph panel,
- adjust the ALE and signal parameters.

When the ALE is started, a tonal signal embedded in noise is generated and filtered through the ALE. You can choose to display the impulse or amplitude response of the adaptive filter or the PSD of the signals $d(n)$, $\hat{d}(n)$ or $e(n)$. You can also adjust the ALE parameters such as step size and leakage factor, and signal parameters such as the amount of noise to generate, and the frequency of the tonal signal.

Assignment 5 In the first task, we will begin by filtering one sinus tone in white Gaussian noise. Select *Play sinusoid* and start the ALE. Look at the various signals, the impulse and amplitude response of the filter, and also the PSD of the signals. Is the LMS adapting to the correct frequency? Which signal contains your desired signal, which signal contains the noise?

. . . .

Repeat this assignment with different frequencies. The frequency is changed by changing the *Primary frequency*. Also, change the used step size μ. It may be necessary to restart the adaption by resetting the LMS algorithm.

The delay Δ in the ALE is fixed to 200 samples. Especially in this example, how many samples would theoretically be sufficient for the adaption, and why?

. . . .

Assignment 6 We will now filter signals with alternating frequencies in white Gaussian noise. Select *Play alternating sinusoids* and start the ALE. Again, look at the different signals, look at the impulse and amplitude response, and the PSD of the signals. Check the adaption when the frequency of the sinus tone changes. The generated signals will alternate between *Primary frequency* and *Secondary frequency* every 10 seconds.

Assignment 7 Repeat the assignment 6 with signals of alternating frequencies, but this time we will instead observe the behaviour of the leaky LMS in the ALE. The leaky LMS is selected by simply adjusting the leakage factor to less than 1.

Begin by leaving the leakage factor at 1. Start the ALE and observe the amplitude response of the adaptive filter. The LMS is not able to suppress the previous tonal component from the filter when the frequency changes and the filter contains two pass-bands. Change the frequencies and new pass-bands will appear without suppressing the previous pass-bands. Why is that?

. . . .

Adjust the leakage factor to less than 1, what happens now to the adaptive filter when the frequency changes? Also observe the PSD of the signals, what happens when the leakage factor changes, and why?

. . . .

Assignment 8 The last assignment with sinusoidal tones in white Gaussian noise will be with a chirp, i.e., a swept-frequency cosine. You can choose between a chirp with linear and quadratic instantaneous frequency deviation by selecting *Play linear sweep* or *Play quadratic* *sweep*. The sweep will be from *Primary frequency* to *Secondary frequency* over a time of 20 seconds.

Test different chirps and look at the different output signals and their PSD. Do the LMS or leaky LMS manage to adapt? Try different step sizes and leakage factor and find try to find some parameters where the filter can adapt to the sweeping frequency.

. . . .