

Chapter 4 Least-Mean-Square Algorithm (LMS Algorithm)

1. Search Methods

- The optimum tap-weights of a transversal (FIR) Wiener filter can be obtained by solving the Wiener-Hopf equation provided that the required statistics of the underlying signals are available.
- An alternative way of finding the optimum tap-weights is to use an iterative search algorithm that starts at some arbitrary initial point in the tap-weight vector space and progressively moves towards the optimum point in steps.
- There are many iterative search algorithms derived for minimizing the underlying cost function with the true statistics replaced by their estimate obtained in some manner.
- Gradient-based iterative methods
 - (1) Method of Steepest Descent
 - (2) Newton's Method

2. Method of Steepest Descent

2.1 Consider a transversal Wiener filter shown in Fig. 1.

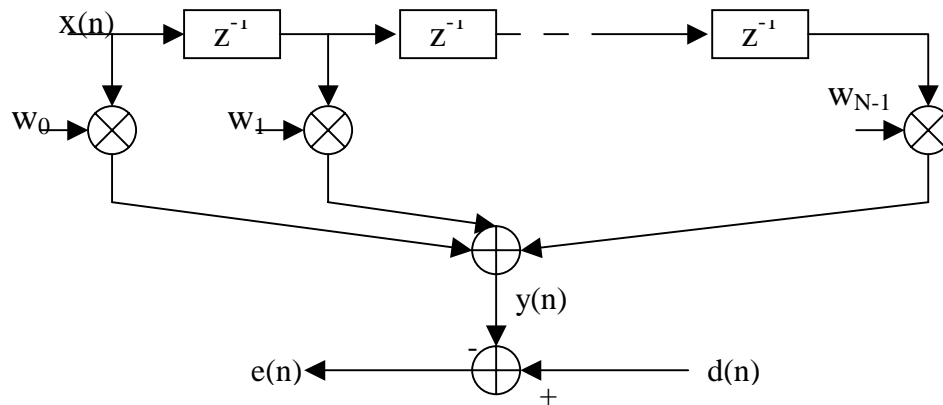


Fig. 1.

- Assuming that all the signals involved are real-valued signals.

- Tap-weight vector $\bar{w} = [w_0 \quad w_1 \quad \dots \quad w_{N-1}]^T$

signal input $\bar{x}(n) = [x(n) \quad x(n-1) \quad \dots \quad x(n-N+1)]^T$

filter output $y(n) = \bar{w}^T \bar{x}(n)$

error signal $e(n) = d(n) - y(n)$

performance function $\xi = E[e^2(n)]$
 $= E[(d^2(n)) - 2\bar{w}^T \bar{p} + \bar{w}^T R \bar{w}] \quad \dots(1)$

where $R = E[\bar{x}(n)\bar{x}^T(n)] \quad \dots$ autocorrelation matrix of the filter input

and $\bar{p} = E[\bar{x}(n)d(n)] \quad \dots$ cross-correlation vector between $\bar{x}(n)$

and $d(n)$

- The performance function ξ is a quadratic function of the filter tap-weight vector \bar{W} . ξ has a single global minimum obtained by solving the Wiener-Hopf equation

$$R\bar{W}_0 = \bar{p} \quad \dots(2)$$

if R and \bar{p} are available.

- Instead of trying to solve equation (2) directly, we employ an iterative search method in which starting with an initial guess for \bar{W}_0 , say $\bar{W}(0)$, a recursive search method that may require many iterations (or steps) to converge to \bar{W}_0 is used.

- The gradient of ξ is given by

$$\nabla \xi = 2R\bar{W} - 2\bar{p} \quad \dots(3)$$

- With an initial guess of \bar{W}_0 at $n=0$ the tap-weight vector at the k -th iteration is denoted as $\bar{W}(k)$.

The following recursive equation may be used to update $\bar{W}(k)$:

$$\bar{W}(k+1) = \bar{W}(k) - \mu \nabla_k \xi \quad \dots(4)$$

where $\mu > 0$ is called the step-size,

$\nabla_k \xi$ denotes the gradient vector $\nabla \xi$ evaluated at the point $\bar{W} = \bar{W}(k)$

- Substituting (3) in (4), we get

$$\bar{W}(k+1) = \bar{W}(k) - 2\mu(R\bar{W}(k) - \bar{p}(k)) \quad \dots(5)$$

- The convergence of $\bar{W}(k)$ to the optimum solution \bar{W}_0 and the convergence speed are dependent on the step-size parameter μ .

2.2 Convergence of the steepest descent method

- Equation(5) can be re-arranged as

$$\bar{W}(k+1) = (1 - 2\mu R)(\bar{W}(k) - \bar{W}_0) \quad \dots(6)$$

Defining the vector $\bar{v}(k)$ as

$$\bar{v}(k) = \bar{W}(k) - \bar{W}_0 \quad \dots(7)$$

Then, we have, from eq.(6),

$$\bar{v}(k+1) = (I - 2\mu R)\bar{v}(k) \quad \dots(8)$$

- The auto-correlation matrix R may be diagonalized by using a unitary similarity decomposition :

$$R = Q\Lambda Q^T \quad \dots(9)$$

where Λ is a diagonal matrix consisting of the eigenvalues $\lambda_0, \lambda_1, \dots, \lambda_{N-1}$ of R and the columns of Q contain the corresponding orthonormal eigenvectors.

Note that $I = QQ^T$ $\dots(10)$

Combining eq.(8),(9)&(10), we get

$$\begin{aligned} \bar{v}(k+1) &= (QQ^T - 2\mu Q\Lambda Q^T)\bar{v}(k) \\ &= Q(I - 2\mu\Lambda)Q^T\bar{v}(k) \quad \dots(11) \end{aligned}$$

Denote that

$$\bar{v}'(k) = Q^T\bar{v}(k) \quad \dots(12)$$

Which transforms $\bar{v}(k)$ to $\bar{v}'(k)$.

With some mathematical manipulations, we obtain

$$\bar{v}'(k+1) = (I - 2\mu\Lambda)\bar{v}'(k) \quad \dots(13)$$

This vector recursive equation may be separated into scalar recursive equations :

$$v_i'(k+1) = (1 - 2\mu\lambda_i)v_i'(k) \quad \dots(14)$$

for $i = 0, 1, 2, \dots, N-1$

where $\bar{v}'(k) = [v_0'(k) \quad v_1'(k) \quad \dots \quad v_{N-1}'(k)]$

From eq.(14), we get

$$v_i'(k) = (1 - 2\mu\lambda_i)^k v_i'(0) \quad \dots(15)$$

for $i = 0, 1, 2, \dots, N-1$

Eq.(15) implies that $\bar{v}'(k)$ can converges to zero if and only if the step-size parameter μ is selected so that

$$|1 - 2\mu\lambda_i| < 1 \quad \text{for } i = 0, 1, 2, \dots, N-1 \quad \dots(16)$$

That is, $0 < \mu < \frac{1}{\lambda_i}$ for all i

or, equivalently, $0 < \mu < \frac{1}{\lambda_{\max}}$... (17)

where λ_{\max} is the maximum of the eigenvalues $\lambda_0 \quad \lambda_1 \quad \dots \quad \lambda_{N-1}$.

3. Newton's method

- The steepest descent algorithm may suffer from slow modes of convergence which arise as a result of the spread in the eigenvalues of R .
- The Newton's method can somehow get rid of the eigenvalue spread.
- Starting from the steepest descent algorithm given in eq.(5) :

$$\bar{W}(k+1) = \bar{W}(k) - 2\mu(R\bar{W}(k) - \bar{p}(k)) \quad \dots(18)$$

Using $\bar{p} = R\bar{W}_0$, eq.(18) becomes

$$\bar{W}(k+1) = \bar{W}(k) - 2\mu R(\bar{W}(k) - \bar{W}_0) \quad \dots(19)$$

- The presence of R in eq.(19) cause the eigenvalue-spread problem in the steepest descent algorithm. Newton's method overcomes this problem by replacing the scalar step-size μ with a matrix step-size given by μR^{-1} . The resulting algorithm is

$$\bar{W}(k+1) = \bar{W}(k) - \mu R^{-1} \nabla_k \xi \quad \dots(20)$$

Substituting $\nabla \xi = 2R\bar{W} - 2\bar{p}$ in eq.(20), we obtain

$$\begin{aligned} \bar{W}(k+1) &= \bar{W}(k) - 2\mu R^{-1}(R\bar{W}(k) - \bar{p}) \\ &= (1 - 2\mu)\bar{W}(k) + 2\mu R^{-1}\bar{p} \\ &= (1 - 2\mu)\bar{W}(K) + 2\mu\bar{W}_0 \quad \dots(21) \end{aligned}$$

And, by subtracting \bar{W}_0 from both sides of eq.(21), we get

$$\bar{W}(k+1) - \bar{W}_0 = (1 - 2\mu)(\bar{W}(k) - \bar{W}_0) \quad \dots(22)$$

Starting with an initial value $\bar{W}(0)$ and iterating eq.(22), we obtain

$$\bar{W}(k) - \bar{W}_0 = (1 - 2\mu)^k (\bar{W}(0) - \bar{W}_0) \quad \dots(23)$$

- In actual implementation of adaptive filters, the exact values of $\nabla_k \xi$ and R^{-1} are not available and have to be estimated.

4. Least-Mean-Square (LMS) Algorithm

- Fig. 2 shows an N -tap transversal adaptive filter.

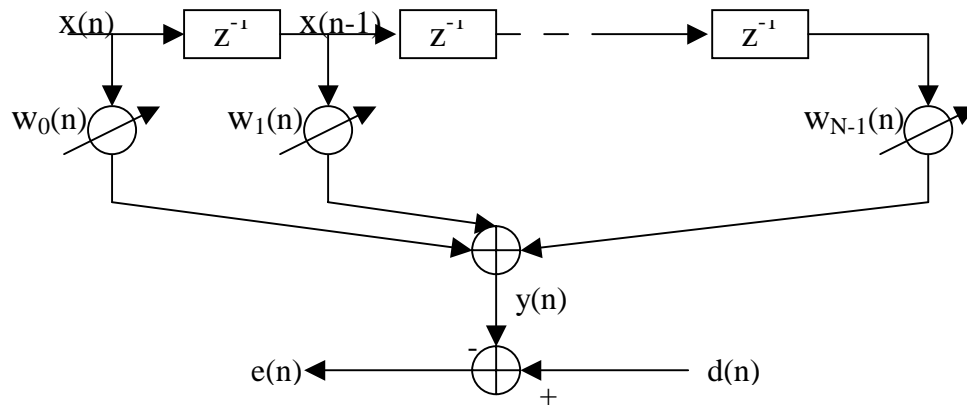


Fig. 2.

$$y(n) = \sum_{i=0}^{N-1} w_i(n)x(n-i) \quad \dots(24)$$

$$e(n) = d(n) - y(n) \quad \dots(25)$$

- We assume that the signals involved are real-valued.
- The LMS algorithm changes (adapts) the filter tap weights so that $e(n)$ is minimized in the mean-square sense. When the processes $x(n)$ & $d(n)$ are jointly stationary, this algorithm converges to a set of tap-weights which, on average, are equal to the Wiener-Hopf solution.
- The LMS algorithm is a practical scheme for realizing Wiener filters, without explicitly solving the Wiener-Hopf equation.
- The conventional LMS algorithm is a stochastic implementation of the steepest descent algorithm. It simply replaces the cost function

$$\xi = E[e^2(n)] \text{ by its instantaneous coarse estimate } \hat{\xi} = e^2(n).$$

- Substituting $\hat{\xi} = e^2(n)$ for ξ in the steepest descent recursion, we obtain

$$\bar{W}(n+1) = \bar{W}(n) - \mu \nabla e^2(n) \quad \dots(26)$$

where $\bar{W}(n) = [w_0(n) \quad w_1(n) \quad \dots \quad w_{N-1}(n)]^T$

$$\nabla = \left[\frac{\partial}{\partial w_0} \quad \frac{\partial}{\partial w_1} \quad \dots \quad \frac{\partial}{\partial w_{N-1}} \right]^T$$

- Note that the i -th element of the gradient vector $\nabla e^2(n)$ is

$$\begin{aligned} \frac{\partial e^2(n)}{\partial w_i} &= 2e(n) \frac{\partial e(n)}{\partial w_i} \\ &= -2e(n) \frac{\partial y(n)}{\partial w_i} \\ &= -2e(n)x(n-i) \quad \dots(27) \end{aligned}$$

Then $\nabla e^2(n) = -2e(n)\bar{x}(n)$

where $\bar{x}(n) = [x(n) \quad x(n-1) \quad \dots \quad x(n-N+1)]^T$

Finally, we obtain

$$\bar{W}(n+1) = \bar{W}(n) + 2\mu e(n)\bar{x}(n) \quad \dots(28)$$

- Eq.(28) is referred to as the LMS recursion.
- Summary of the LMS algorithm

Input :

tap-weight vector, $\bar{W}(n)$

input vector, $\bar{x}(n)$

desired output, $d(n)$

Output :

Filter output, $y(n)$

Tap-weight vector update, $\bar{W}(n+1)$

1. Filtering : $y(n) = \bar{W}^T(n)\bar{x}(n)$

2. Error estimation : $e(n) = d(n) - y(n)$

3. Tap-weight vector adaptation : $\bar{W}(n+1) = \bar{W}(n) + 2\mu e(n)\bar{x}(n)$

• Advantages & disadvantages of LMS algorithm :

(1) Simplicity in implementation

(2) Stable and robust performance against different signal conditions

(3) slow convergence (due to eigenvalue spread)

5. MSE Behavior of the LMS Algorithm

- We are to study