Least Mean-Square Algorithm

- unknown dynamic system is stimulated by an input vector consisting of the elements $x_1(i), x_2(i), ..., x_M(i)$

$$x(i) = [x_1(i), x_2(i), ..., x_M(i)]^T$$
Least Mean-Square Algorithm

(1) The $M$ elements of $x(i)$ originate at different points in space. We view $x(i)$ as a snapshot of data.

(2) The $M$ elements represent the set of present and $(M-1)$ past values of some excitation that are uniformly spaced in time.
input snapshot at discrete time $n$
$x(n) \triangleq [x_1(n), x_2(n), \ldots, x_M(n)]$
output $y(n) = x^T(n)w(n)$
desired signal $d(n)$
error vector:
$e(n) = d(n) - y(n) = d(n) - x^T(n)w(n)$
\[ x(n) \triangleq [x_1(n), x_2(n), \ldots, x_M(n)] \]
\[ e(n) = d(n) - y(n) = d(n) - x^T(n)w(n) \]

- instantaneous cost function \( E(w) \triangleq \frac{1}{2}e^2(n) \)
\[ x(n) \triangleq [x_1(n), x_2(n), \ldots, x_M(n)] \]
\[ e(n) = d(n) - y(n) = d(n) - x^T(n)w(n) \]
- instantaneous cost function \( E(w) \triangleq \frac{1}{2}e^2(n) \)
- differentiate \( E(w) \) with respect to the filter weights \( w \)

\[
\frac{\partial E(w)}{\partial w} = e(n) \frac{e(n)}{\partial w} = e(n) e(n) \]
\( \mathbf{x}(n) \triangleq [x_1(n), x_2(n), \ldots, x_M(n)] \)

\( e(n) = d(n) - y(n) = d(n) - \mathbf{x}^T(n)\mathbf{w}(n) \)

- instantaneous cost function \( E(\mathbf{w}) \triangleq \frac{1}{2} e^2(n) \)

differentiate \( E(\mathbf{w}) \) with respect to the filter weights \( \mathbf{w} \)

\[
\frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = e(n) \frac{e(n)}{\partial \mathbf{w}}
\]

\[
\frac{e(n)}{\partial \mathbf{w}} = -\mathbf{x}(n)
\]

- instantaneous estimate of the gradient

\[ \frac{\partial E(\mathbf{w})}{\partial \mathbf{w}} = -\mathbf{x}(n)e(n) \]

- LMS algorithm:

\[
\mathbf{w}(n + 1) = w(n) + \eta \mathbf{x}(n)e(n)
\]

\[
= w(n) + \eta \mathbf{x}(n)\left(d(n) - \mathbf{x}^T(n)\mathbf{w}(n)\right)
\]

(stochastic) gradient descent
**Training Sample:**
- Input signal vector = $x(n)$
- Desired response = $d(n)$

**User-selected parameter:** $\eta$

**Initialization.** Set $\hat{w}(0) = 0$.

**Computation.** For $n = 1, 2, ..., $ compute

$$e(n) = d(n) - \hat{w}^T(n)x(n)$$

$$\hat{w}(n + 1) = \hat{w}(n) + \eta x(n)e(n)$$
Adaptive filtering applications: EEG denoising

- Electroencephalography (EEG): electrophysiological monitoring method to record electrical activity of the brain.
- Electrooculography (EOG): a technique for measuring the corneo-retinal standing potential that exists between the front and the back of the human eye.
Adaptive filtering applications

- canceling of maternal electrocardiogram (ECG)
Adaptive filtering applications

(d) Mother’s ECG signals (chest leads), (e) contaminated ECG signal (abdominal leads) and (f) noise-reduced ECG signal.
Neural Networks

- Nonlinear models for function approximation

\[ w^T x + b \rightarrow f(\cdot) = f(w^T x + b) \]

- example \( f(u) = \frac{1}{1+e^{-u}} \) gives \( \frac{1}{1+e^{-(w^T x + b)}} \)
Adaline: Adaptive Linear Neuron

- Bernard Widrow and Ted Hoff (1960)
Training Adaline

**LMS algorithm:**

\[
\mathbf{w}(n + 1) = \mathbf{w}(n) + \eta \mathbf{x}(n) \left( d(n) - \mathbf{x}^T(n) \mathbf{w}(n) \right)
\]
Training Adaline

\[ \mathbf{w}(n + 1) = \mathbf{w}(n) + \eta \mathbf{x}(n) \left( d(n) - \mathbf{x}^T(n) \mathbf{w}(n) \right) \]