EE269
Signal Processing for Machine Learning

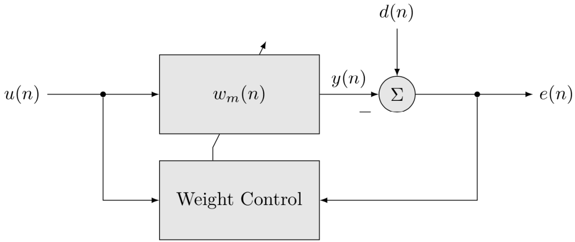
Lecture 14

Instructor : Mert Pilanci

Stanford University

November 23, 2020

Adaptive Filters



$u[n]$ zero mean stationary input signal

w_m length M filter with impulse response $w_0, w_1, ...w_{M-1}$

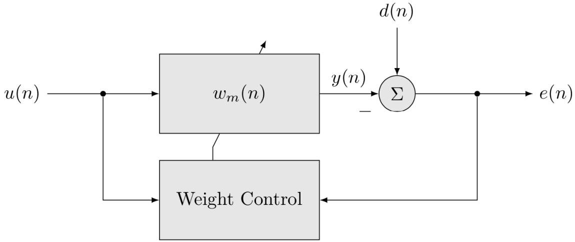
$y[n]$ output signal

$$y[n] = \sum_{m=0}^{M-1} w_m u[n-m]$$

$d[n]$ desired signal

$e[n]$ error signal

Adaptive Filters



$$w = [w_0 \ w_1 \ \ldots \ w_{M-1}]^T$$

$$u_n = [u[n] \ u[n-1] \ \ldots \ u[n-M+1]]^T$$

correlation matrix $R_u \triangleq \mathbb{E}[u_n u_n^T]$

cross-correlation vector $r_{ud} \triangleq \mathbb{E}[u_n d_n]$

Adaptive Filters via Least Squares

- consider a time window of length $K \geq M$
  for $n = n_0, n_0 + 1, ..., n_0 + K - 1$
  output $y[n] = \sum_{m=0}^{M-1} w_m u[n-m]$ in matrix form

$$\begin{bmatrix} y[n_0] \\ y[n_0+1] \\ \vdots \\ y[n_0+K-1] \end{bmatrix} \triangleq \begin{bmatrix} u[n_0] & u[n_0-1] & \dots & u[n_0-M+1] \\ u[n_0+1] & u[n_0] & \dots & u[n_0-M+1] \\ \vdots & \vdots & \vdots & \vdots \\ u[n_0+K-1] & u[n_0] & \dots & u[n_0-M+1] \end{bmatrix} w_m$$

- $y = Aw$
- error vector $e = y - d = Aw - d$
- minimize $||Aw - d||_2^2$ using Least Squares

Wiener-Hopf Equations

- alternative approach: consider minimizing instantaneous error
- optimal filter coefficients $w = \arg\min J(w)$

  error signal $e[n] = y[n] - d[n] = u_n^T w - d$
- $J(w) = \mathbb{E}\, e[n]^2$
- $\mathbb{E}\, e[n]^2 = (u_n^T w - d)(u_n^T w - d)$

Wiener-Hopf Equations

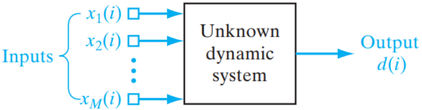
- alternative approach: consider minimizing instantaneous error
- optimal filter coefficients $w = \arg\min J(w)$

  error signal $e[n] = y[n] - d[n] = u_n^T w - d$
- $J(w) = \mathbb{E}\, e[n]^2$
- $\mathbb{E}\, e[n]^2 = (u_n^T w - d)(u_n^T w - d)$

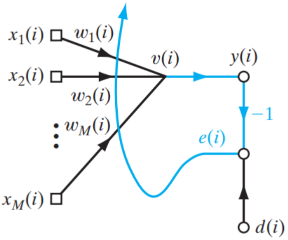
- $\mathbb{E}\, e[n]^2 = \mathbb{E} d[n]^2 + w^T R_u w - 2w^T r_{ud}$
- gradient $\frac{\partial J(w)}{\partial w} = 2R_u w - 2r_{ud}$
- solution $w^* = R_u^{-1} r_{ud}$ Wiener Filter

  (if R_u is invertible)

Least Mean-Square Algorithm



(a)



(b)

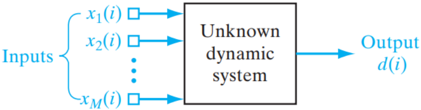
- unknown dynamic system is stimulated by an input vector consisting of the elements $x_1(i), x_2(i), ..., x_M(i)$

$$x(i) = [x_1(i), x_2(i), ...x_M(i)]^T$$

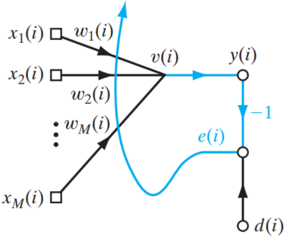
$$
\begin{aligned}
e(n) &= d(n) - [x(1), x(2), \ldots, x(n)]^T w(n) \\
&= d(n) - X(n)w(n)
\end{aligned}
$$

- $d(n)$: $n \times 1$ desired response vector
- $X(n)$: $n \times M$ data matrix

Recap: Least Mean-Square Algorithm



(a)

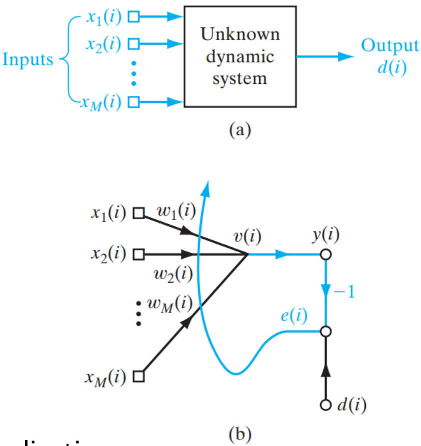


(b)

- 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



different applications:

(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

$\mathbf{x}(n) \triangleq [x_1(n), x_2(n), ..., x_M(n)]$

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

desired signal $d(n)$

error vector:

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

$\mathbf{x}(n) \triangleq [x_1(n), x_2(n), ..., x_M(n)]$

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

- 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:

$$\begin{aligned} \mathbf{w}(n+1) &= w(n) + \eta\mathbf{x}(n)e(n) \\ &= w(n) + \eta\mathbf{x}(n)\Big(d(n) - \mathbf{x}^T(n)\mathbf{w}(n)\Big) \end{aligned}$$

  (stochastic) gradient descent

Training Sample: Input signal vector $= \mathbf{x}(n)$

Desired response $= d(n)$

User-selected parameter: η

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

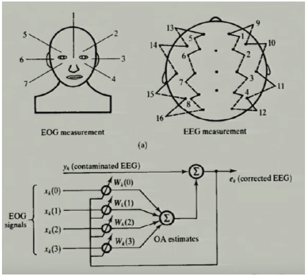
Computation. For $n = 1, 2, \ldots,$ compute

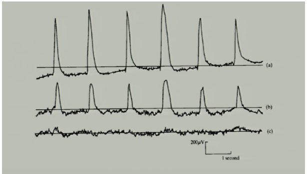
$$e(n) = d(n) - \hat{\mathbf{w}}^T(n)\mathbf{x}(n)$$

$$\hat{\mathbf{w}}(n+1) = \hat{\mathbf{w}}(n) + \eta\mathbf{x}(n)e(n)$$

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.

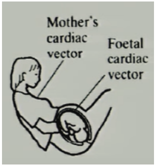


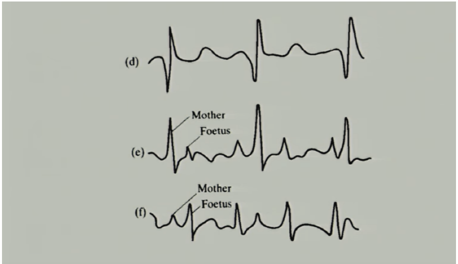


(a) Measured **EOG**, (b) corresponding contaminated **EEG** and (c)

Adaptive filtering applications

- canceling of maternal electrocardiogram (ECG)





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

$$
\begin{aligned}
e(n) &= d(n) - [x(1), x(2), \ldots, x(n)]^T w(n) \\
&= d(n) - X(n)w(n)
\end{aligned}
$$

- $d(n)$: $n \times 1$ desired response vector
- $X(n)$: $n \times M$ data matrix

LMS convergence analysis

- signal correlation matrix
  $R_x = \mathbb{E}\ \mathbf{x}(n)\mathbf{x}^T(n)$
- $w^* \triangleq= R_x^{-1} r_{dx}$ optimal Wiener filter
- $\epsilon(n) = \mathbf{w}^* - \mathbf{w}(n)$

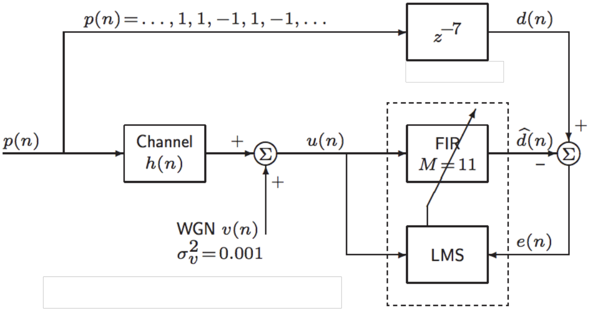
LMS convergence analysis

- signal correlation matrix
  $R_x = \mathbb{E}\ \mathbf{x}(n)\mathbf{x}^T(n)$
- $w^* \triangleq R_x^{-1} r_{dx}$ optimal Wiener filter
- $\epsilon(n) = \mathbf{w}^* - \mathbf{w}(n)$
- The error satisfies the recursion

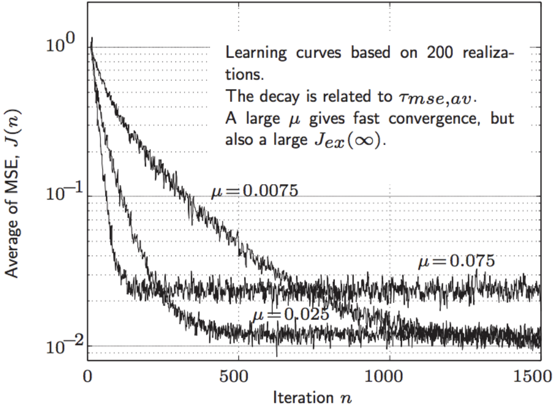
$$\epsilon(n+1) = (I - \eta R_x)\epsilon(n) + \text{zero mean noise}$$

- $E(n)$ cost can be written as
- $E(n) = E_{min} + E_{ex}(\infty) + E_{trans}(n)$
- LMS converges if $0 < \eta < \frac{2}{\lambda_{max}(R_x)}$
- $E_{ex}(\infty) = E_{min} \sum_i \frac{\eta\lambda_i}{2-\eta\lambda_i}$

Adaptive filtering applications: channel equalization



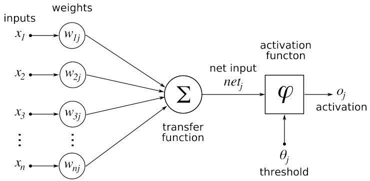
Adaptive filtering applications: channel equalization

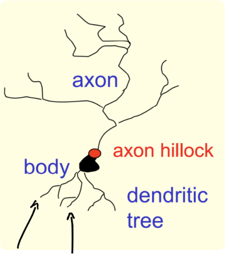


The curves illustrates learning curves for two different μ.

Adaptive filters to neural networks

- Nonlinear models for function approximation





- $w^T x + b \to f(\cdot) = f(w^T x + b)$
- 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