Revision of Lecture Seventeen

- Previous lecture introduces generic structure of adaptive equalisation
  - Adaptive signal processing/filtering: enabling technology for communications
  - Adaptive equalisation is just a particular example
  - Concepts of cost function and optimisation, adaptive FIR filter
- Communications technology is about "Shannon meets Wiener"
- This lecture looks into optimal FIR filter design known as **Wiener filter** or **minimum mean square error** solution
  - This Wiener design embodies most important ideas of adaptive filtering
  - It is most widely used design principle in communication applications
  - It has important influence on new designs

Wiener Filters

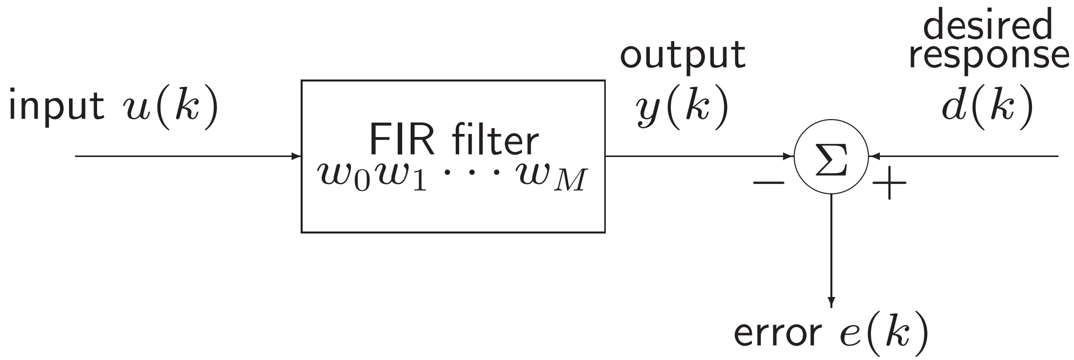
- **Wiener** filter is the optimal FIR filter in the **MMSE** sense

  Define the FIR filter weight vector

$$\mathbf{w} = [w_0 \; w_1 \cdots w_M]^T$$

  and the filter input vector

$$\mathbf{u}(k) = [u(k) \; u(k-1) \cdots u(k-M)]^T$$



  The actual filter output and the error signal are given by

$$y(k) = \sum_{i=0}^{M} w_i^* u(k-i) = \mathbf{w}^H \mathbf{u}(k) \qquad e(k) = d(k) - y(k) = d(k) - \mathbf{w}^H \mathbf{u}(k)$$

- Assuming the desired signal $d(k)$ and the filter input $u(k)$ are wide-sense stationary, the **optimal Wiener solution** $\hat{\mathbf{w}}$ minimises the MSE

$$J(\mathbf{w}) = \mathrm{E}[|e(k)|^2] = \mathrm{E}[e(k)e^*(k)]$$

- Define the desired signal power $\sigma_d^2 = \mathrm{E}[|d(k)|^2]$, the autocorrelations $\gamma(l) = \mathrm{E}[u(k)u^*(k-l)]$ for $0 \le l \le M$, and the crosscorrelations $p(l) = \mathrm{E}[d^*(k)u(k-l)]$ for $0 \le l \le M$

Wiener Filters (continue)

- Since the square error $|e(k)|^2 = d(k)d^*(k) - d(k)\mathbf{u}^H(k)\mathbf{w} - \mathbf{w}^H\mathbf{u}(k)d^*(k) + \mathbf{w}^H\mathbf{u}(k)\mathbf{u}^H(k)\mathbf{w}$,

$$J(\mathbf{w}) = \mathrm{E}[|e(k)|^2] = \sigma_d^2 - \mathbf{p}^H\mathbf{w} - \mathbf{w}^H\mathbf{p} + \mathbf{w}^H\mathbf{R}\mathbf{w}$$

where

$$\mathbf{p} = \begin{bmatrix} p(0) \\ p(1) \\ \vdots \\ p(M) \end{bmatrix} \quad \text{and} \quad \mathbf{R} = \begin{bmatrix} \gamma(0) & \gamma(1) & \cdots & \gamma(M) \\ \gamma^*(1) & \gamma(0) & \cdots & \gamma(M-1) \\ \vdots & \vdots & \ddots & \vdots \\ \gamma^*(M) & \gamma^*(M-1) & \cdots & \gamma(0) \end{bmatrix}$$

- For $\widehat{\mathbf{w}}$ to be a minimum point of $J(\mathbf{w})$:

$$\nabla J(\mathbf{w})|_{\mathbf{w}=\widehat{\mathbf{w}}} = 0 \ \ \text{(necessary)} \qquad \left.\frac{\partial^2 J(\mathbf{w})}{\partial \mathbf{w}^2}\right|_{\mathbf{w}=\widehat{\mathbf{w}}} \text{ is positive definite} \ \ \text{(sufficient)}$$

that is, $-2\mathbf{p} + 2\mathbf{R}\widehat{\mathbf{w}} = 0$ (necessary), and $\mathbf{R}$ is positive definite (sufficient)

- Necessary condition $\longrightarrow$ Wiener-Hopf equations: $\mathbf{R}\widehat{\mathbf{w}} = \mathbf{p}$, which gives the Wiener solution

$$\widehat{\mathbf{w}} = \mathbf{R}^{-1}\mathbf{p}$$

Since this is the only minimum, it is a **global minimum**. Note that the correlation matrix $\mathbf{R}$ is always nonnegative definite. When $\mathbf{R}$ is positive definite, the inverse $\mathbf{R}^{-1}$ exists

Orthogonal Principle and MSE surface

- The Wiener filter error $\widehat{e}(k) = d(k) - \widehat{\mathbf{w}}^H \mathbf{u}(k)$ is **orthogonal** to the filter input vector:
  - $\mathrm{E}[\widehat{e}^*(k)\mathbf{u}(k)] = \mathbf{0}$, and as a consequence, the MMSE filter output $\widehat{y}(k) = \widehat{\mathbf{w}}^H \mathbf{u}(k)$ is orthogonal to its error: $\mathrm{E}[\widehat{e}^*(k)\widehat{y}(k)] = 0$
- The MSE is a bowl-shaped $(2(M+1)+1)$-dimensional surface ($(M+1)+1$ in real case)

$$J(\mathbf{w}) = \sigma_d^2 - \mathbf{p}^H \mathbf{w} - \mathbf{w}^H \mathbf{p} + \mathbf{w}^H \mathbf{R} \mathbf{w}$$

and has a unique minimum at $\mathbf{w} = \widehat{\mathbf{w}}$. Since the MMSE

$$J_{\min} = J(\widehat{\mathbf{w}}) = \sigma_d^2 - \mathbf{p}^H \mathbf{R}^{-1} \mathbf{p} = \sigma_d^2 - \sigma_{\widehat{y}}^2$$

where $\mathrm{E}[|\widehat{y}(k)|^2] = \sigma_{\widehat{y}}^2 = \mathrm{E}[\widehat{\mathbf{w}}^H \mathbf{u}(k)\mathbf{u}^H(k)\widehat{\mathbf{w}}] = \mathbf{p}^H \mathbf{R}^{-1} \mathbf{p}$, the MSE for $\mathbf{w}$ can be written as

$$J(\mathbf{w}) = J_{\min} + (\mathbf{w} - \widehat{\mathbf{w}})^H \mathbf{R} (\mathbf{w} - \widehat{\mathbf{w}})$$

- The **eigenvalues** of $\mathbf{R}$ are the solutions $\lambda_0, \lambda_1, \cdots, \lambda_M$ of $\det(\mathbf{R} - \lambda \mathbf{I}) = 0$, and the **condition number** is the ratio of largest eigenvalue to smallest eigenvalue

$$\chi(\mathbf{R}) = \frac{\lambda_{\max}}{\lambda_{\min}}$$

Eigenvalue Spread

- The ratio $\chi(\mathbf{R})$ is called **eigenvalue spread**, and it determines the **performance** of an adaptive algorithm

  $\chi(\mathbf{R}) \geq 1$: If $\mathbf{R}$ is singular, $\lambda_{\min} = 0$ and $\chi(\mathbf{R}) = \infty$; $\mathbf{R}$ is ill conditioned if $\chi(\mathbf{R})$ is large.
- Example of real channel and modulation with the channel $r(k) = 0.5s(k) + 1.0s(k-1) + n(k)$, the equaliser $y(k) = w_0 r(k) + w_1 r(k-1) + w_2 r(k-2)$, and the desired response $d(k) = s(k-1)$, where $n(k)$ is white Gaussian with zero mean and variance $\sigma_n^2 = 0.25$, and $s(k)$ is BPSK taking value from $\{\pm 1\}$
- The auto-correlation matrix and the cross-correlation vector are:

$$\mathbf{R} = \begin{bmatrix} 1.5 & 0.5 & 0.0 \\ 0.5 & 1.5 & 0.5 \\ 0.0 & 0.5 & 1.5 \end{bmatrix} \qquad \mathbf{p} = \begin{bmatrix} 1.0 \\ 0.5 \\ 0.0 \end{bmatrix}$$

  The eigenvalues and the MMSE error solution are

$$\begin{aligned} \lambda_0 &= 1.5 + \sqrt{0.5} \\ \lambda_1 &= 1.5 \\ \lambda_2 &= 1.5 - \sqrt{0.5} \end{aligned} \qquad \hat{\mathbf{w}} = \begin{bmatrix} 0.6190 \\ 0.1429 \\ -0.0476 \end{bmatrix}$$

  The MMSE is $J_{\min} = 0.3095$, and the eigenvalue spread is $\chi(\mathbf{R}) = 2.7836$

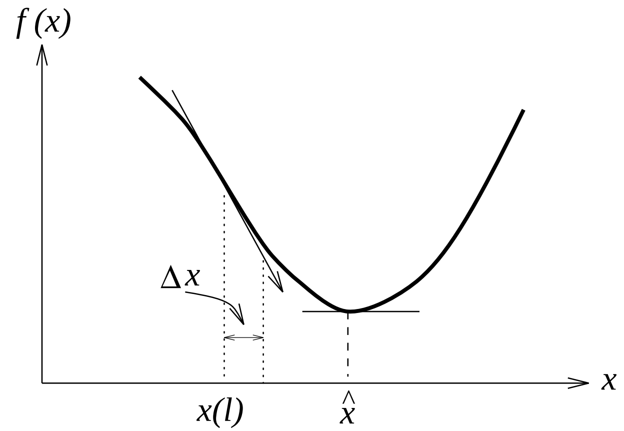
Steepest Descent Algorithm

- There are many reasons for not computing $\mathbf{R}^{-1}$ directly $\longrightarrow$ gradient descent for the MMSE solution
- For function of a scalar variable $f(x)$, noting that negative gradient points "downhill" and starting from an initial guess $x(0)$, we can use:

$$x(l+1) = x(l) + \Delta x(l) = x(l) + \left(-\mu \frac{\partial f}{\partial x}\Big|_{x=x(l)}\right)$$

This iteration loop will leads to $x(l) \longrightarrow \widehat{x}$ at which point

$$\frac{\partial f}{\partial x}\Big|_{x=\widehat{x}} = 0$$



- For the FIR filter $y(k) = \mathbf{w}^H \mathbf{u}(k)$ with $e(k) = d(k) - y(k)$,

$$J(\mathbf{w}) = \sigma_d^2 - \mathbf{p}^H \mathbf{w} - \mathbf{w}^H \mathbf{p} + \mathbf{w}^H \mathbf{R} \mathbf{w} \quad \text{and} \quad \widehat{\mathbf{w}} = \mathbf{R}^{-1} \mathbf{p}$$

- Iteration procedure based on gradient so that $\mathbf{w}(l) \longrightarrow \widehat{\mathbf{w}}$, with **Algorithm**:
  1. Initial value $\mathbf{w}(0)$
  2. $\nabla J(l) = \nabla J(\mathbf{w}(l)) = -2\mathbf{p} + 2\mathbf{R}\mathbf{w}(l)$
  3. $\mathbf{w}(l+1) = \mathbf{w}(l) + \frac{1}{2}\mu(-\nabla J(l)) = \mathbf{w}(l) + \mu(\mathbf{p} - \mathbf{R}\mathbf{w}(l))$
  4. Go back to step 2

Analysis of Steepest Descent Algorithm

- Note that the steepest descent algorithm involves feedback $e(k) \longrightarrow$ stability consideration and the value of μ is critical. Also the underlying system is characterised by the eigenvalue spread
- **Stability analysis**: Necessary and sufficient condition for

$$\lim_{l \to \infty} \mathbf{w}(l) = \widehat{\mathbf{w}}$$

is

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

- **Time constant of the algorithm** τ_a defines how quickly the algorithm converges to a steady-state solution on average. It can be shown that

$$\frac{-1}{\log(|1 - \mu\lambda_{\max}|)} \le \tau_a \le \frac{-1}{\log(|1 - \mu\lambda_{\min}|)}$$

Note

$$\tau_a \approx \frac{1}{\mu\lambda_{\min}}$$

But

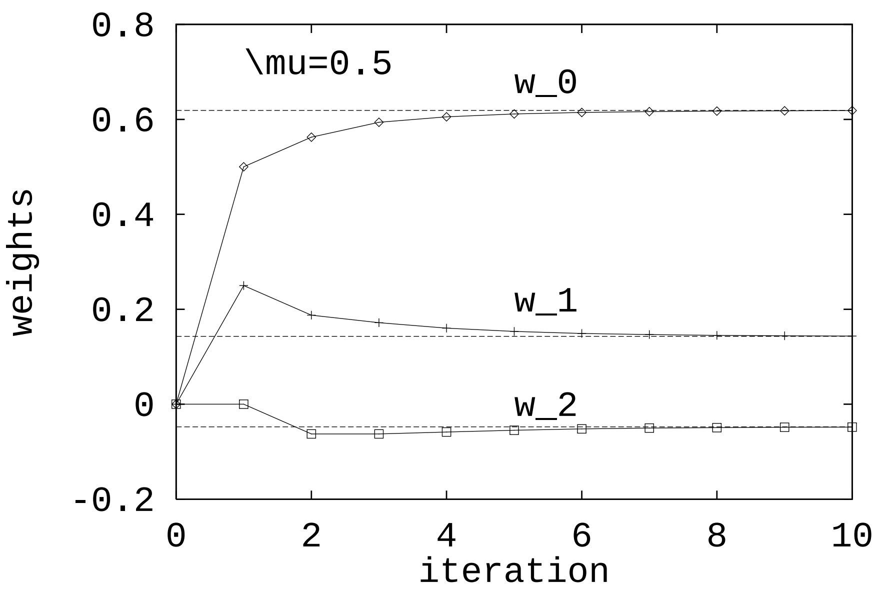
$$\mu \propto \frac{1}{\lambda_{\max}} \Rightarrow \tau_a \propto \frac{\lambda_{\max}}{\lambda_{\min}} = \chi(\mathbf{R})$$

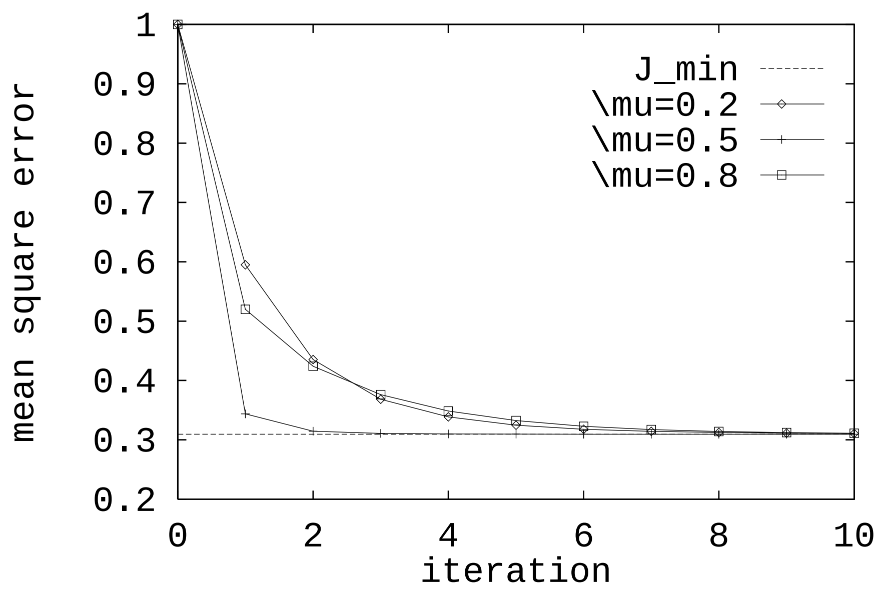
This clearly shows that the eigenvalue spread influences rate of convergence

Example

- Example as in Slide 190: The steepest-descent algorithm is used. The step-size parameter μ should satisfy

$$0 < \mu < \frac{2}{\lambda_{\max}} = \frac{2}{1.5 + \sqrt{0.5}} \quad \text{or} \quad 0 < \mu < 0.9$$





Sample-by-Sample Adaptation

- Recall that the steepest descent algorithm can be used to obtain the **Wiener** (MMSE) solution

  $\Downarrow$ It requires **ensemble averages** $\mathbf{R}$ and $\mathbf{p}$, usually not available. These statistics may be approximated by time-averaging

$$\bar{\gamma}(l) = \frac{1}{N}\sum_{k=1}^{N} u(k)u^*(k-l) \quad \bar{p}(l) = \frac{1}{N}\sum_{k=1}^{N} d^*(k)u(k-l)$$

  $\Downarrow$ But $u(k)$ and $d(k)$ can be nonstationary, and it would be better to update the filter as each new data sample is taken

  $\Downarrow$ Many practical applications require extremely fast computation per sample, as sampling rate can be very fast

- These considerations $\longrightarrow$ a **stochastic gradient-based** method
  - Steepest descent method: $\nabla J(\mathbf{w}(l)) = -2\mathbf{p} + 2\mathbf{R}\mathbf{w}(l)$ with $\mathbf{R} = \mathrm{E}[\mathbf{u}(k)\mathbf{u}^H(k)]$ and $\mathbf{p} = \mathrm{E}[\mathbf{u}(k)d^*(k)]$. All the quantities are deterministic
  - Stochastic gradient-based method: instantaneous "estimates" $\widetilde{\mathbf{R}}(k) = \mathbf{u}(k)\mathbf{u}^H(k)$ and $\widetilde{\mathbf{p}}(k) = \mathbf{u}(k)d^*(k)$ are used to provides gradient of instantaneous squared error $\widetilde{J}(k) = |e(k)|^2$

$$\nabla \widetilde{J}(k) = -2\mathbf{u}(k)d^*(k) + 2\mathbf{u}(k)\mathbf{u}^H(k)\widetilde{\mathbf{w}}(k) = -2\mathbf{u}(k)e^*(k)$$

  - where $e(k) = d(k) - \widetilde{\mathbf{w}}^H(k)\mathbf{u}(k)$. All the quantities are noisy or stochastic

Least Mean Square Algorithm

- This is probably the simplest adaptive algorithm, involving three steps per cycle:
  1. Compute the filter output
  $$y(k) = \widetilde{\mathbf{w}}^H(k)\mathbf{u}(k)$$
  2. Compute the estimation error
  $$e(k) = d(k) - y(k)$$
  3. Update the tap weights
  $$\widetilde{\mathbf{w}}(k+1) = \widetilde{\mathbf{w}}(k) + \frac{1}{2}\mu\nabla\widetilde{J}(k) = \widetilde{\mathbf{w}}(k) + \mu\mathbf{u}(k)e^*(k)$$
- The step size μ must be properly chosen, the mean of $\widetilde{\mathbf{w}}(k)$ is:
  $$\mathbf{E}[\widetilde{\mathbf{w}}(k)]$$
  and the mean square error is:
  $$J(k) = \mathbf{E}[|e(k)|^2] = \mathbf{E}[|d(k) - \widetilde{\mathbf{w}}^H(k)\mathbf{u}(k)|^2]$$
- Note $\tilde{\mathbf{w}}(k)$ is stochastic and we have to talk about convergence in mean and/or mean square error
- Surprisingly, this LMS algorithm actually works, but its convergence analysis is extremely difficult
  - Stochastic gradient descent method is a widely used low-complexity optimisation approach

Analysis in Stationary Environment

- Assuming $u(k)$ and $d(k)$ are jointly wide sense stationary and some other simplified assumptions:
  - **Convergence in mean**:

$$\lim_{k\to\infty} \mathbf{E}[\widetilde{\mathbf{w}}(k)] = \widehat{\mathbf{w}} \quad \text{provided that} \quad 0 < \mu < \frac{2}{\lambda_{\max}}$$

  - **Convergence in mean square**: where $J(\infty)$ $(> J_{\min})$ is finite,

$$\lim_{k\to\infty} J(k) = J(\infty) \quad \text{if and only if} \quad 0 < \mu < \frac{2}{\lambda_{\max}} \quad \text{and} \quad \sum_{i=0}^{M} \frac{\mu\lambda_i}{2(1-\mu\lambda_i)} < 1$$

- Steady state mean square error is given by

$$J(\infty) = \frac{J_{\min}}{1 - \frac{1}{2}\sum_{i=0}^{M} \mu\lambda_i/(1-\mu\lambda_i)}$$

- Excess mean square error is defined as

$$J_{\text{ex}}(\infty) = J(\infty) - J_{\min} = J_{\min} \times \frac{\frac{1}{2}\sum_{i=0}^{M} \mu\lambda_i/(1-\mu\lambda_i)}{1 - \frac{1}{2}\sum_{i=0}^{M} \mu\lambda_i/(1-\mu\lambda_i)}$$

- Misadjustment is defined by

$$\mathcal{M} = \frac{J_{\text{ex}}(\infty)}{J_{\min}} = \frac{\frac{1}{2}\sum_{i=0}^{M} \mu\lambda_i/(1-\mu\lambda_i)}{1 - \frac{1}{2}\sum_{i=0}^{M} \mu\lambda_i/(1-\mu\lambda_i)}$$

Influence of Eigenvalue Spread

- Define the average eigenvalue

$$\lambda_{\text{av}} = \frac{1}{M+1} \sum_{i=0}^{M} \lambda_i$$

and the average time constant of the LMS algorithm

$$\tau_{\text{mse,av}} = \frac{-1}{2\log(|1-\mu\lambda_{\text{av}}|)} \approx \frac{1}{2\mu\lambda_{\text{av}}}$$

- If step size is chosen as $\mu << \frac{1}{\lambda_{\max}}$, condition for convergence in mean square becomes: $0 < \mu < \frac{2}{\sum_{i=0}^{M}\lambda_i}$. With this choice of μ, the misadjustment is approximately by

$$\mathcal{M} \approx \frac{\mu}{2} \sum_{i=0}^{M} \lambda_i = \frac{\mu(M+1)\lambda_{\text{av}}}{2} \approx \frac{M+1}{4\tau_{\text{mse,av}}}$$

- Noting that $\mathcal{M} \propto \mu$ and $\tau_{\text{mse,av}} \propto \frac{1}{\mu}$, a **careful trade off** is required in choosing μ: small μ leads to small $\mathcal{M}$ but large μ leads to fast convergence
- Noting that $\tau_{\text{mse,av}} \approx \frac{1}{\mu\lambda_{\min}} \propto \chi(\mathbf{R})$, the **rate of convergence is determined by the eigenvalue spread**: in general, when $\chi(\mathbf{R})$ is large, the LMS converges slowly
  - LMS works well if $\chi(\mathbf{R})$ is small, and if $\chi(\mathbf{R}) = 1$ its convergence performance is as good as recursive least squares algorithm

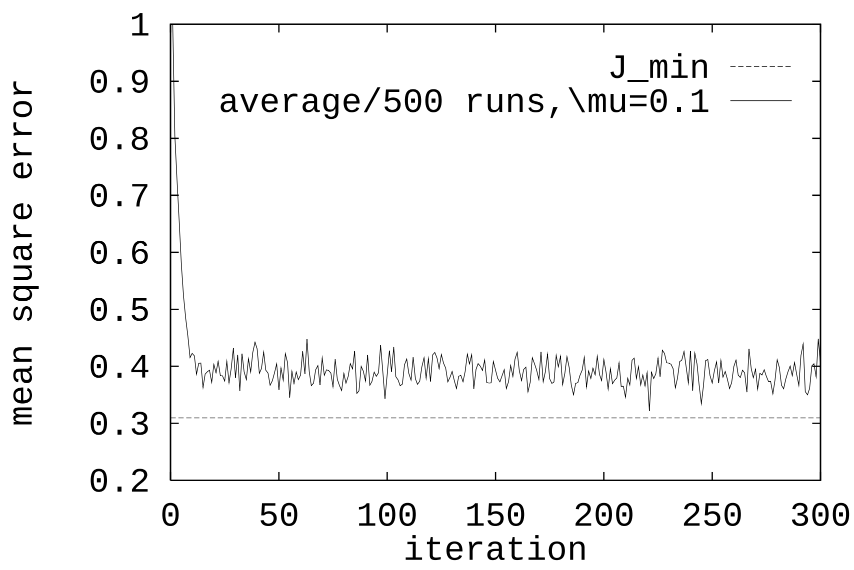
Example

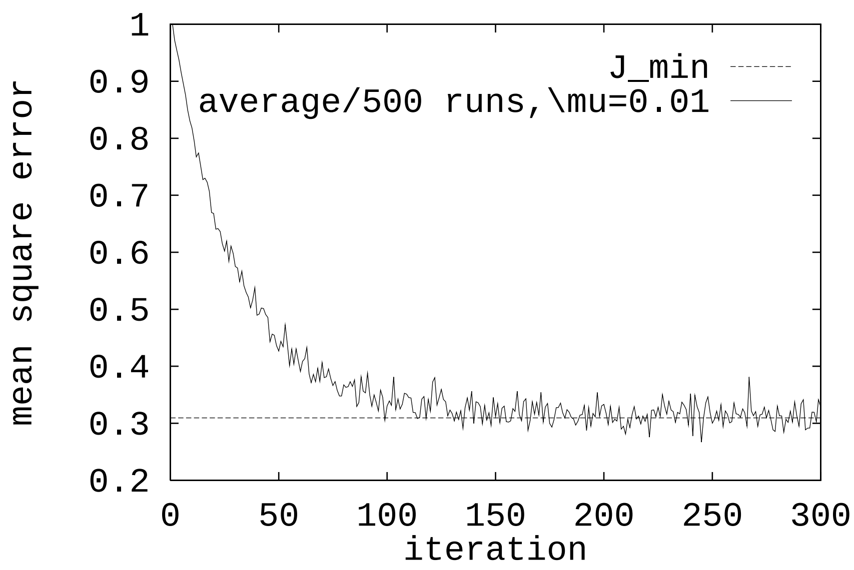
- Example as in Slide 190 but the LMS is used is used. In computer simulation, $\mathrm{E}[\widetilde{\mathbf{w}}(k)]$ and $J(k)$ are approximated using sample averages over 500 different runs

$$\mu << \frac{1}{\lambda_{\max}} \quad \text{and} \quad \mu < \frac{2}{\sum_{i=0}^{M} \lambda_i}$$

$$\Longrightarrow \quad \mu << 0.45 \text{ and } \mu < 0.44$$

μ	Theoretical $J_{\text{ex}}(\infty)$	Simulation $J_{\text{ex}}(\infty)$
0.10	0.1162	0.10
0.05	0.0435	0.04
0.01	0.0073	0.01





Least Squares Estimate

- MMSE estimate is based on ensemble average, while **least squares** estimate relies only on available samples $\{\mathbf{u}(k), d(k)\}_{k=1}^{K}$: From **slide 230**, we can express desired output as

$$d^*(k) = \mathbf{u}^H(k)\mathbf{w} + e^*(k)$$

or

$$\begin{bmatrix} d^*(1) \\ d^*(2) \\ \vdots \\ d^*(K) \end{bmatrix} = \begin{bmatrix} \mathbf{u}^H(1) \\ \mathbf{u}^H(2) \\ \vdots \\ \mathbf{u}^H(K) \end{bmatrix} \mathbf{w} + \begin{bmatrix} e^*(1) \\ e^*(2) \\ \vdots \\ e^*(K) \end{bmatrix} \Rightarrow \mathbf{d}^* = \mathbf{U}\mathbf{w} + \mathbf{e}^*$$

  – From $\mathbf{U}^H\mathbf{d}^* = \mathbf{U}^H\mathbf{U}\mathbf{w} + \mathbf{U}^H\mathbf{e}^*$, since input data are uncorrelated with noise, $\mathbf{U}^H\mathbf{e}^* \approx \mathbf{0}$, LS estimate is given by

$$\widehat{\mathbf{w}}_{\text{LS}} = \left(\mathbf{U}^H\mathbf{U}\right)^{-1}\mathbf{U}^H\mathbf{d}^*$$

- **Recursive least squares** algorithm with forgetting factor λ and initial covariance matrix $\mathbf{P}(0)$

$$\begin{aligned}
\mathbf{k}(k) =& \frac{\lambda^{-1}\mathbf{P}(k-1)\mathbf{u}(k)}{1 + \lambda^{-1}\mathbf{u}^H(k)\mathbf{P}(k-1)\mathbf{u}(k)} \\
e(k) =& d(k) - \widehat{\mathbf{w}}^H(k-1)\mathbf{u}(k) \\
\widehat{\mathbf{w}}(k) =& \widehat{\mathbf{w}}(k-1) + \mathbf{k}(k)e^*(k) \\
\mathbf{P}(k) =& \lambda^{-1}\mathbf{P}(k-1) - \lambda^{-1}\mathbf{k}(k)\mathbf{u}^H(k)\mathbf{P}(k-1)
\end{aligned}$$

Summary

- Wiener (MMSE) solution: $\widehat{\mathbf{w}} = \mathbf{R}^{-1}\mathbf{p}$
  - MSE surface $J(\mathbf{w}) = \sigma_d^2 - \mathbf{p}^H\mathbf{w} - \mathbf{w}^H\mathbf{p} + \mathbf{w}^H\mathbf{R}\mathbf{w} = J_{\min} + (\mathbf{w} - \widehat{\mathbf{w}})^H\mathbf{R}(\mathbf{w} - \widehat{\mathbf{w}})$ is quadratic with the MMSE given by $J_{\min} = \sigma_d^2 - \mathbf{p}^H\mathbf{R}^{-1}\mathbf{p}$
  - Steepest-descent algorithm and convergence analysis
- The LMS algorithm:

$$y(k) = \widetilde{\mathbf{w}}^H(k)\mathbf{u}(k), \quad e(k) = d(k) - y(k), \quad \widetilde{\mathbf{w}}(k+1) = \widetilde{\mathbf{w}}(k) + \mu\mathbf{u}(k)e^*(k)$$

  - Sufficient conditions for stationary convergence of the LMS

$$\mu << \frac{1}{\lambda_{\max}} \text{ and } 0 < \mu < \frac{2}{\sum_{i=0}^{M}\lambda_i}$$

  - Misadjustment and convergence rate of the LMS:

$$\mathcal{M} \approx \frac{\mu}{2}\sum_{i=0}^{M}\lambda_i \Longrightarrow \mathcal{M} \propto \mu \qquad \tau_{\text{mse,av}} \approx \frac{1}{2\mu\lambda_{\text{av}}} \Longrightarrow \text{convergence time} \propto \frac{1}{\mu}$$

  - Effect of eigenvalue spread: the larger eigenvalue spread, the slower convergence rate of LMS
- Least squares estimate and recursive least squares algorithm