DSP

Chapter-6 : Wiener Filters and the LMS Algorithm

Marc Moonen
Dept. E.E./ESAT-STADIUS, KU Leuven
marc.moonen@esat.kuleuven.be
www.esat.kuleuven.be/stadius/

Part-III : Optimal & Adaptive Filters

Chapter-6 **Wieners Filters & the LMS Algorithm**

- Introduction / General Set-Up
- Applications
- Optimal Filtering: Wiener Filters
- Adaptive Filtering: LMS Algorithm

Chapter-7 **Recursive Least Squares Algorithms**

- Least Squares Estimation
- Recursive Least Squares (RLS)
- Square Root Algorithms
- Fast RLS Algorithms

Introduction / General Set-Up

1. 'Classical' Filter Design

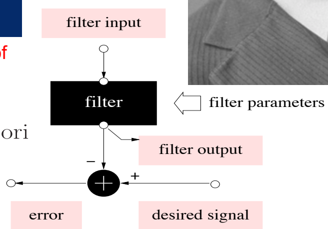
lowpass/bandpass/notch filters/...

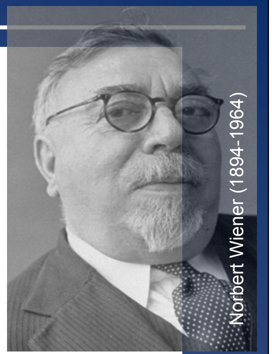
See Part-II

2. 'Optimal' Filter Design

- signals are viewed as realizations of *stochastic processes* (H249-HB78)
- filter optimisation/design in a *statistical sense* based on a priori *statistical information*

$\longrightarrow$ *Wiener filters*

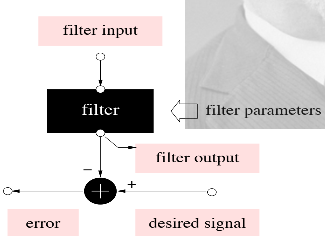


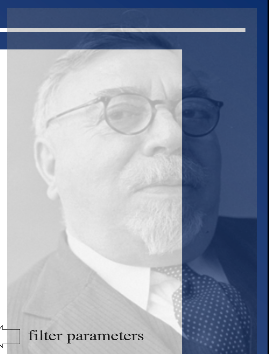


Introduction / General Set-Up

Prototype optimal filtering set-up :

Design filter such that for a given (i.e. 'statistical info available') input signal, filter output signal is 'optimally close' (to be defined) to a given 'desired output signal'.





Introduction / General Set-Up

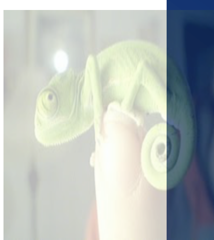
when a priori statistical information is not available :

3. 'Adaptive' Filters

- self-designing
- adaptation algorithm to monitor environment
- properties of adaptive filters :

  convergence/tracking
  numerical stability/accuracy/robustness
  computational complexity
  hardware implementation

Introduction / General Set-Up

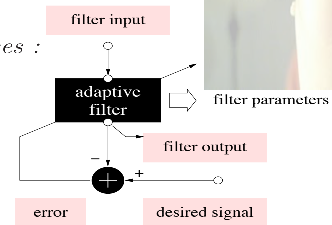


Prototype adaptive filtering set-up :

Basic operation involves 2 processes :

1. *filtering process*
2. *adaptation process*

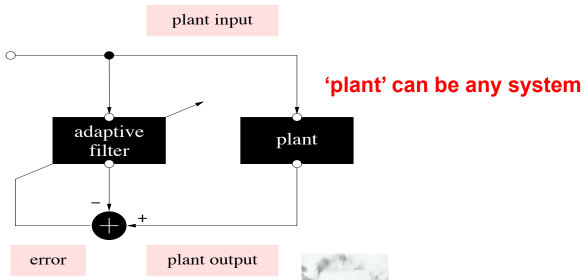
   adjusting filter parameters to
   (time-varying) environment
   adaptation is steered by error signal



- Depending on the application, either the filter parameters, the filter output or the error signal is of interest

Applications

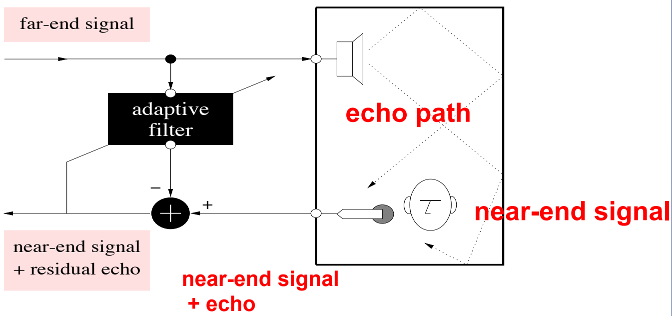
system identification/modeling



Optimal/adaptive filter provides mathematical model for input/output-behavior of the plant

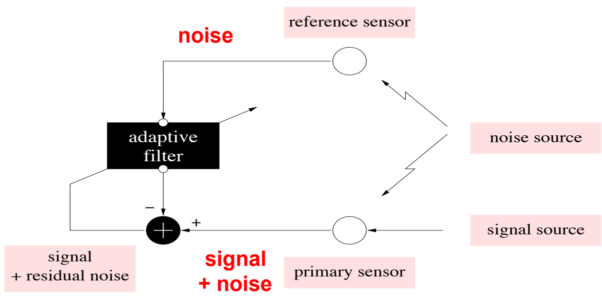
Applications

example : acoustic echo cancellation



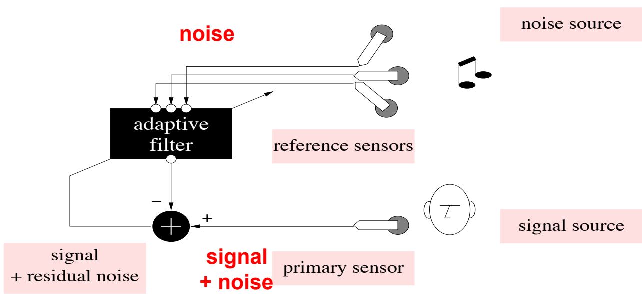
Applications

example : interference cancellation



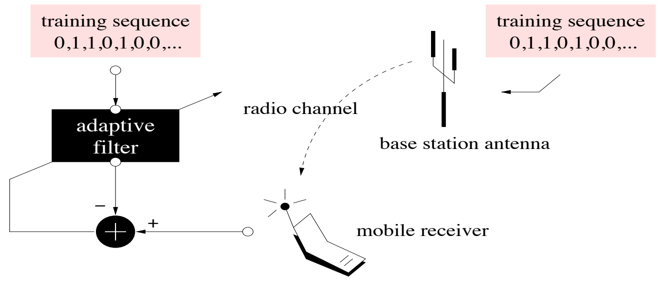
Applications

example : acoustic noise cancellation



Applications

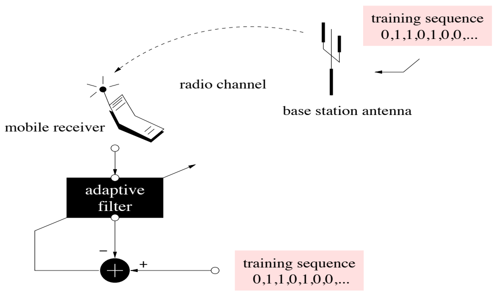
example : channel identification



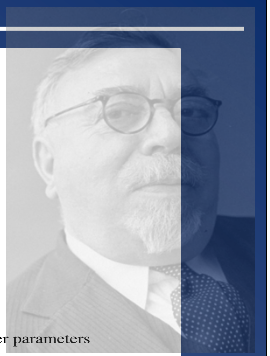
Applications

Inverse modeling :

example : channel equalization (training mode)



Optimal Filtering : Wiener Filters



Prototype optimal filter revisited

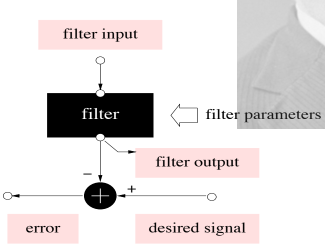
Have to decide on 2 things..

1 *filter structure ?*

→ FIR filters
(=pragmatic choice)

2 *cost function ?*

→ quadratic cost function
(=pragmatic choice)



Optimal Filtering : Wiener Filters

1

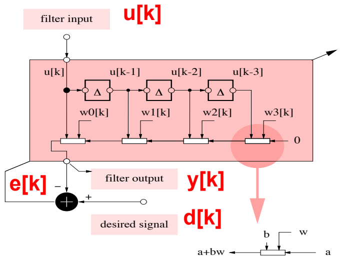
FIR filters (=tapped-delay line filter/'transversal' filter)

$$y_k = \sum_{l=0}^{L} w_l . u_{k-l} = \mathbf{w}^T . \mathbf{u}_k = \mathbf{u}_k^T . \mathbf{w}$$

where

$$\mathbf{w}^T = \begin{bmatrix} w_0 & w_0 & \dots & w_L \end{bmatrix}$$

$$\mathbf{u}_k^T = \begin{bmatrix} u_k & u_{k-1} & \dots & u_{k-L} \end{bmatrix}$$

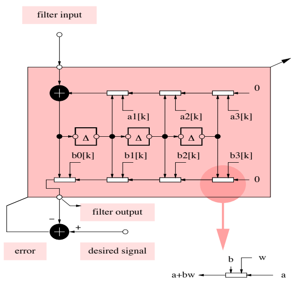


PS: Shorthand notation u_k=u[k], y_k=y[k], d_k=d[k], e_k=e[k],
Filter coefficients ('weights') are w_l (replacing b_l of previous chapters)
For adaptive filters w_l also have a time index $w_l[k]$

Optimal Filtering : Wiener Filters

Note : generalization to *IIR (infinite impulse response)* is non-trivial

- convergence problems
- stability problems

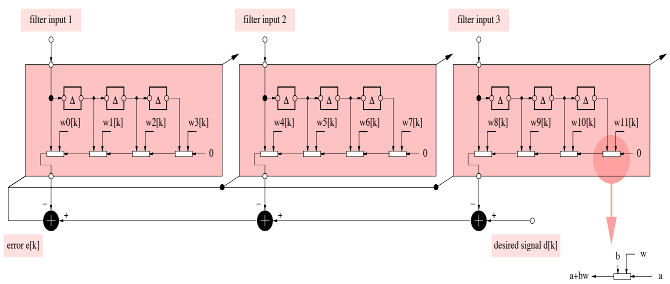


Note : generalization to *non-linear filters* not treated here

Optimal Filtering : Wiener Filters

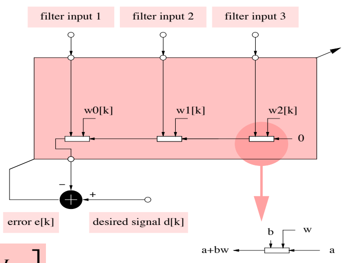
PS: Can generalize FIR filter to 'multi-channel FIR filter'

example: see page 11



Optimal Filtering : Wiener Filters

PS: Special case of 'multi-channel FIR filter' is 'linear combiner'



$$y_k = \mathbf{u}_k^T \mathbf{w}$$

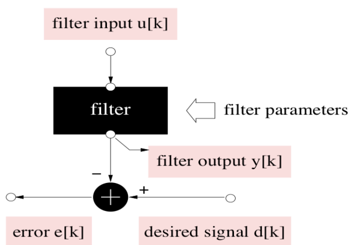
where

$$\mathbf{u}_k^T = \left[\begin{array}{cccc} u_k^0 & u_k^1 & \dots & u_k^L \end{array} \right]$$

FIR filter may then also be viewed as special case of 'linear combiner' where input signals are delayed versions of each other

Optimal Filtering : Wiener Filters

2



Quadratic cost function :

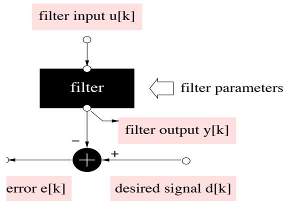
minimum mean-square error (MMSE) criterion

$$J_{MSE}(\mathbf{w}) = \mathrm{E}\left\{e_k^2\right\} = \mathrm{E}\left\{\left(d_k - y_k\right)^2\right\} = \mathrm{E}\left\{\left(d_k - \mathbf{u}_k^T\mathbf{w}\right)^2\right\}$$

$\mathcal{E}\{x\}$ is 'expected value' (mean) of x

Optimal Filtering : Wiener Filters

MMSE cost function can be expanded as…



$$J_{MSE}(\mathbf{w}) = \mathrm{E}\left\{e_k^2\right\}$$
$$= \mathrm{E}\left\{\left(d_k - \mathbf{u}_k^T \mathbf{w}\right)^2\right\}$$
$$= \mathcal{E}\{d_k^2\} + \mathbf{w}^T \underbrace{\mathcal{E}\{\mathbf{u}_k \mathbf{u}_k^T\}}_{\bar{\mathbb{X}}_{uu}} \mathbf{w} - 2\mathbf{w}^T \underbrace{\mathcal{E}\{\mathbf{u}_k d_k\}}_{\bar{\mathbb{X}}_{du}}.$$

$\bar{\mathbb{X}}_{uu}$ = *correlation matrix* $\bar{\mathbb{X}}_{du}$ = *cross-correlation vector*

Optimal Filtering : Wiener Filters

Correlation matrix has a special structure…

for a stationary discrete-time stochastic process $\{u_k\}$:

autocorrelation coefficients : $\bar{x}_{uu}(\delta) = \mathcal{E}\{u_k \cdot u_{k-\delta}\}$

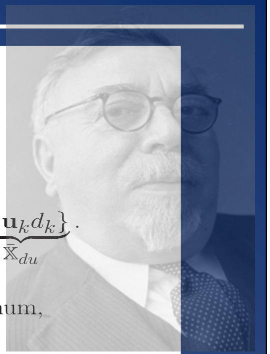
correlation matrix :

with $\mathbf{u}_k^T = \begin{bmatrix} u_k & u_{k-1} & \dots & u_{k-L} \end{bmatrix}$

$$\bar{\mathbb{X}}_{uu} = \mathrm{E}\left\{\mathbf{u}_k.\mathbf{u}_k^T\right\} = \begin{bmatrix} \bar{x}_{uu}(0) & \bar{x}_{uu}(1) & \bar{x}_{uu}(2) & \cdots & \bar{x}_{uu}(L) \\ \bar{x}_{uu}(1) & \bar{x}_{uu}(0) & \bar{x}_{uu}(1) & \cdots & \bar{x}_{uu}(L-1) \\ \bar{x}_{uu}(2) & \bar{x}_{uu}(1) & \bar{x}_{uu}(0) & \cdots & \bar{x}_{uu}(L-2) \\ \vdots & \vdots & \vdots & & \vdots \\ \bar{x}_{uu}(L) & \bar{x}_{uu}(L-1) & \bar{x}_{uu}(L-2) & \cdots & \bar{x}_{uu}(0) \end{bmatrix}$$

i.e. symmetric & Toeplitz & non-negative definite

Optimal Filtering : Wiener Filters



MMSE cost function can be expanded as…(continued)

$$J_{MSE}(\mathbf{w}) = \mathcal{E}\{d_k^2\} + \mathbf{w}^T \underbrace{\mathcal{E}\{\mathbf{u}_k \mathbf{u}_k^T\}}_{\bar{\mathbb{X}}_{uu}} \mathbf{w} - 2\mathbf{w}^T \underbrace{\mathcal{E}\{\mathbf{u}_k d_k\}}_{\bar{\mathbb{X}}_{du}}.$$

cost function is convex, with a (mostly) unique minimum, obtained by setting the gradient equal to zero:

$$0 = [\frac{\partial J_{MSE}(\mathbf{w})}{\partial \mathbf{w}}]_{\mathbf{w}=\mathbf{w}_{WF}} = [2\bar{\mathbb{X}}_{uu}\mathbf{w} - 2\bar{\mathbb{X}}_{du}]_{\mathbf{w}=\mathbf{w}_{WF}}$$

Wiener-Hopf equations :

$\bar{\mathbb{X}}_{uu} \cdot \mathbf{w}_{WF} = \bar{\mathbb{X}}_{du} \quad \rightarrow \quad \mathbf{w}_{WF} = \bar{\mathbb{X}}_{uu}^{-1}\bar{\mathbb{X}}_{du}$ .....simple enough!

This is the 'Wiener Filter' solution

Optimal Filtering : Wiener Filters

How do we solve the Wiener–Hopf equations?

solving linear systems ($L+1$ linear equations in $L+1$ unknowns)

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \cdot \mathbf{w}_{WF} = \begin{bmatrix} 3 \\ 7 \end{bmatrix} \quad \rightarrow \quad \mathbf{w}_{WF} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

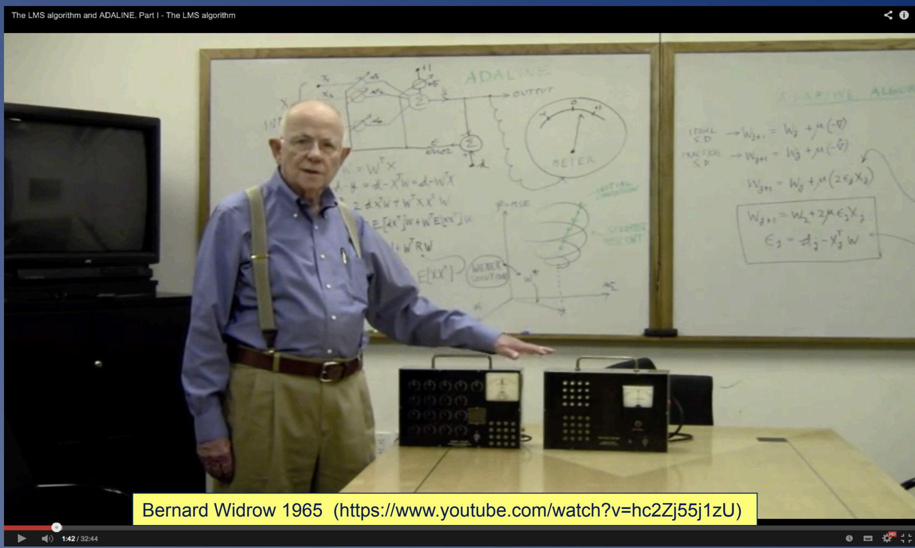
requires $O(L^3)$ arithmetic operations

requires $O(L^2)$ arithmetic operations if $\bar{\mathbb{X}}_{uu}$ is Toeplitz

- Schur algorithm
- Levinson-Durbin algorithm

= used intensively in applications, e.g. in speech codecs, etc. details omitted, see Appendix

Adaptive Filtering: LMS Algorithm



Bernard Widrow 1965 (https://www.youtube.com/watch?v=hc2Zj55j1zU)

Adaptive Filtering: LMS Algorithm

How do we compute the Wiener filter?

1) Cfr supra: By solving Wiener-Hopf equations (L+1 equations in L+1 unknowns)

$$\bar{\mathbb{X}}_{uu} \cdot \mathbf{w}_{WF} = \bar{\mathbb{X}}_{du}$$

2) Can also apply iterative procedure to minimize MMSE criterion, e.g.

Steepest-descent iterations :

$$\begin{aligned} \mathbf{w}(n+1) &= \mathbf{w}(n) + \frac{\mu}{2} \cdot \left[\frac{-\partial J_{MSE}(\mathbf{w})}{\partial \mathbf{w}}\right]_{\mathbf{w}=\mathbf{w}(n)} \\ &= \mathbf{w}(n) + \mu \cdot (\bar{\mathbb{X}}_{du} - \bar{\mathbb{X}}_{uu}\mathbf{w}(n)) \end{aligned}$$

here *n* is iteration index

μ is ‘stepsize’ (to be tuned..)

Adaptive Filtering: LMS Algorithm

Bound on stepsize ?

Steepest-descent iterations :

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu \cdot (\bar{\mathbb{X}}_{du} - \bar{\mathbb{X}}_{uu}\mathbf{w}(n))$$

Stability ?

$$\begin{aligned}[\mathbf{w}(n+1) - \mathbf{w}_{WF}] &= (I - \mu\bar{\mathbb{X}}_{uu}) \cdot [\mathbf{w}(n) - \mathbf{w}_{WF}] \\ &= (I - \mu\bar{\mathbb{X}}_{uu})^{n+1} \cdot [\mathbf{w}(0) - \mathbf{w}_{WF}]\end{aligned}$$

stable iff (λ_i = eigenvalues of $\bar{\mathbb{X}}_{uu}$)

$$-1 < 1 - \mu\lambda_i < 1 \quad \forall i$$

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

➔ large λ_max implies a small stepsize

Adaptive Filtering: LMS Algorithm

Convergence speed?

Transient behavior ?

$$[\mathbf{w}(n+1) - \mathbf{w}_{WF}] = (I - \mu\bar{\mathbb{X}}_{uu})^{n+1} \cdot [\mathbf{w}(0) - \mathbf{w}_{WF}]$$

with (symmetric eigenvalue decomposition)

$$\bar{\mathbb{X}}_{uu} = Q_{uu}\Lambda_{uu}Q_{uu}^T \qquad Q_{uu}^T Q_{uu} = I$$

$$[\mathbf{w}(n+1) - \mathbf{w}_{WF}] = Q_{uu}(I - \mu\Lambda_{uu})^{n+1}Q_{uu}^T \cdot [\mathbf{w}(0) - \mathbf{w}_{WF}]$$

$$Q_{uu}^T[\mathbf{w}(n+1) - \mathbf{w}_{WF}] = \text{diag}\{1 - \mu\lambda_i\}^{n+1} Q_{uu}^T \cdot [\mathbf{w}(0) - \mathbf{w}_{WF}]$$

error vector projected onto eigenvectors — initial error vector projected onto eigenvectors

i.e. $(1 - \mu\lambda_i)^n$ for 'mode' i (=projection on i-th eigenvector)

➔ small λ_i implies slow convergence

➔ λ_min <<λ_max (hence small μ) implies *very* slow convergence

Adaptive Filtering: LMS Algorithm

LMS is derived from WF steepest-descent iterations as follows

Replace *n+1* by *n* for convenience…

$$\mathbf{w}(n) = \mathbf{w}(n-1) + \mu.(E\{\mathbf{u}_k.d_k\} - E\{\mathbf{u}_k.\mathbf{u}_k^T\}.\mathbf{w}(n-1))$$

Then replace iteration index n by time index k

(i.e. perform 1 iteration per sampling interval)

$$\mathbf{w}[k] = \mathbf{w}[k-1] + \mu.(E\{\mathbf{u}_k.d_k\} - E\{\mathbf{u}_k.\mathbf{u}_k^T\}.\mathbf{w}[k-1])$$

Then leave out expectation operators

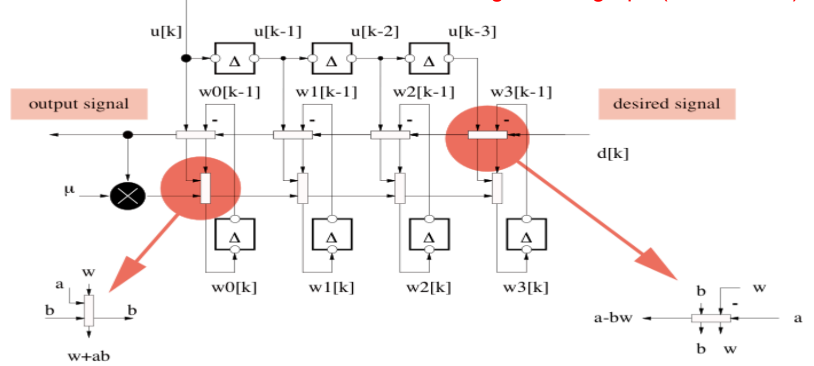
(i.e. replace expected values by instantaneous estimates)

$$\mathbf{w}_{LMS}[k] = \mathbf{w}_{LMS}[k-1] + \mu.\mathbf{u}_k.\underbrace{(d_k - \mathbf{u}_k^T.\mathbf{w}_{LMS}[k-1])}_{\text{'a priori error'}}$$

Adaptive Filtering: LMS Algorithm

$$\mathbf{w}_{LMS}[k] = \mathbf{w}_{LMS}[k-1] + \mu.\mathbf{u}_k.(d_k - \mathbf{u}_k^T.\mathbf{w}_{LMS}[k-1])$$

Simple algorithm, can even draw signal flow graph (=realization)…



Adaptive Filtering: LMS Algorithm

LMS analysis in a nutshell

LMS : stability/covergence ? (proofs/details omitted)

- 'expected behavior'
  = average over ∞ runs
  = steepest-descent behavior

  hence

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

- 'noisy gradients' (next page)

Whenever LMS has reached the WF solution, the **expected** value of $\mathbf{u}_k.(d_k - \mathbf{u}_k^T.\mathbf{w}_{LMS}[k-1])$ (=estimated gradient in update formula) is zero, but the **instantaneous** value is generally non-zero (=noisy), and hence LMS will again move away from the WF solution!

Adaptive Filtering: LMS Algorithm

LMS analysis in a nutshell

- 'noisy gradients' $\rightarrow J_{MSE}(\mathbf{w}[\infty]) > J_{MSE}(\mathbf{w}_{WF})$
  results in **excess MSE** $J_{ex}(\infty)$ and **mismatch** $\mathcal{M}$:

$$J_{MSE}(\mathbf{w}[\infty]) = J_{MSE}(\mathbf{w}_{WF}) + \underbrace{J_{ex}(\mathbf{w}[\infty])}_{\approx J_{MSE}(\mathbf{w}_{WF}) \cdot \underbrace{\frac{\mu}{2}\sum_{i=0}^{L}\lambda_i}_{\mathcal{M}}}$$

PS: FIR case $\sum_{i=0}^{L}\lambda_i = \text{trace}\{\bar{\mathbb{X}}_{uu}\} = L\,\bar{x}_{uu}(0) = L\,\mathcal{E}\{u_k^2\}$

EX: for max 10% excess MSE : $\mu < \frac{0.2}{L.\text{E}\{u_k^2\}}$

means step size has to be much smaller…!

Adaptive Filtering: LMS Algorithm

LMS is an extremely popular algorithm
many LMS-variants have been developed (cheaper/faster/...)...

- *Normalized LMS* (see p.35)

$$\mathbf{w}_{NLMS}[k] = \mathbf{w}_{NLMS}[k-1] + \frac{\bar{\mu}}{\alpha + \mathbf{u}_k^T.\mathbf{u}_k}.\mathbf{u}_k.(d_k - \mathbf{u}_k^T.\mathbf{w}_{NLMS}[k-1])$$

- *Transform domain LMS*
- *Block LMS* : *K is block index,* L_B *is block size*

$$\mathbf{w}_{BLMS}[K] = \mathbf{w}_{BLMS}[K-1] + \frac{\mu}{L}.\sum_{i=1}^{L_B}\mathbf{u}_{(K-1).L_B+i}.(d_{(K-1).L_B+i} - \mathbf{u}_{(K-1).L_B+i}^T.\mathbf{w}_{BLMS}[K-1])$$

- *Frequency domain LMS*
- *Subband (LMS) adaptive filtering*

Adaptive Filtering: LMS Algorithm

normalized LMS (NLMS) = LMS with normalized step size (mostly used in practice)

$$\mathbf{w}_{NLMS}[k] = \mathbf{w}_{NLMS}[k-1] + \frac{\bar{\mu}}{\alpha + \mathbf{u}_k^T.\mathbf{u}_k}.\mathbf{u}_k.(d_k - \mathbf{u}_k^T.\mathbf{w}_{NLMS}[k-1])$$

- NLMS (for $\bar{\mu} = 1$) also solves a specific *optimization problem*:

$$\min_{\mathbf{w}(k)} \tilde{J}(\mathbf{w}[k]) = \alpha.\left\|\mathbf{w}[k] - \mathbf{w}_{NLMS}[k-1]\right\|_2^2 + (d_k - \mathbf{u}_k^T.\mathbf{w}[k])^2$$

- stability/convergence ? :
  convergence if $0 < \bar{\mu} < 2$
  max. 10% excess MSE obtained with $\bar{\mu} < 0.2$