Adaptive Filters – Algorithms (Part 2)

Gerhard Schmidt

Christian-Albrechts-Universität zu Kiel
Faculty of Engineering
Institute of Electrical and Information Engineering
Digital Signal Processing and System Theory

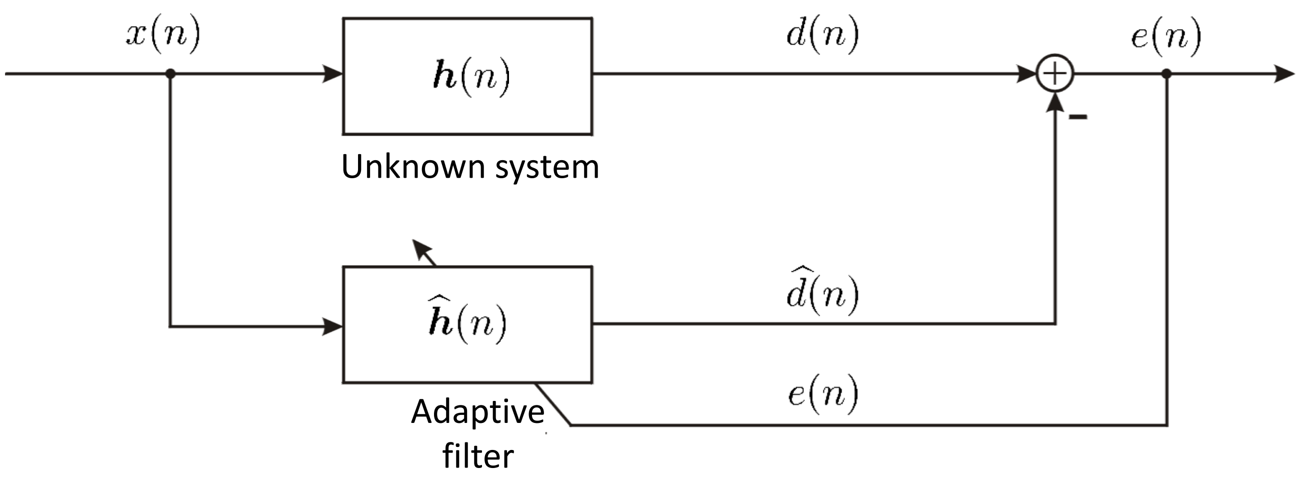
Today

Adaptive Algorithms:

- ❑ Introductory Remarks
- ❑ Recursive Least Squares (RLS) Algorithm
- ❑ Least Mean Square Algorithm (LMS Algorithm) – Part 1
- ❑ Least Mean Square Algorithm (LMS Algorithm) – Part 2
- ❑ Affine Projection Algorithm (AP Algorithm)
- ❑ Fast Affine Projection Algorithm (FAP Algorithm)

Geometrical Explanation of Convergence – Part 1

Structure:



System:

$$\boldsymbol{h} = \left[h_0,\, h_1,\, \ldots,\, h_{N-1}\right]^{\mathrm{T}}$$

System output:

$$d(n) = \sum_{i=0}^{N-1} h_i\, x(n-i) = \boldsymbol{h}^{\mathrm{T}}\, \boldsymbol{x}(n) = \boldsymbol{x}^{\mathrm{T}}(n)\, \boldsymbol{h}$$

Geometrical Explanation of Convergence – Part 2

Error signal:

$$
\begin{aligned}
e(n) &= d(n) - \widehat{d}(n) \\
&= \boldsymbol{h}^{\mathrm{T}}\,\boldsymbol{x}(n) - \widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{x}(n) = \left[\boldsymbol{h} - \widehat{\boldsymbol{h}}(n)\right]^{\mathrm{T}}\boldsymbol{x}(n) \\
&= \boldsymbol{h}_{\Delta}^{\mathrm{T}}(n)\,\boldsymbol{x}(n) \;=\; \boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\Delta}(n)
\end{aligned}
$$

Difference vector:

$$
\boldsymbol{h}_{\Delta}(n) = \boldsymbol{h} - \widehat{\boldsymbol{h}}(n)
$$

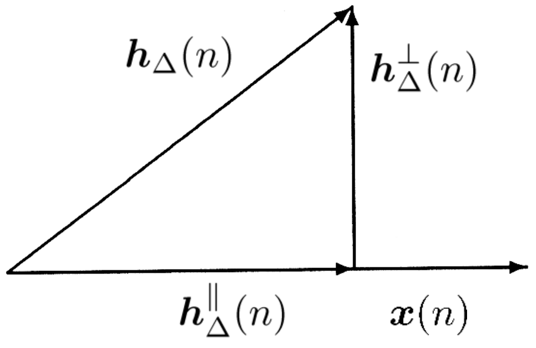
LMS algorithm:

$$
\begin{aligned}
\widehat{\boldsymbol{h}}(n+1) &= \widehat{\boldsymbol{h}}(n) + \mu\, e(n)\,\boldsymbol{x}(n) \\
&= \widehat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\Delta}(n) \\
\underbrace{\widehat{\boldsymbol{h}}(n+1) - \boldsymbol{h}}_{-\boldsymbol{h}_{\Delta}(n+1)} &= \underbrace{\widehat{\boldsymbol{h}}(n) - \boldsymbol{h}}_{-\boldsymbol{h}_{\Delta}(n)} + \mu\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\Delta}(n) \\
\boldsymbol{h}_{\Delta}(n+1) &= \boldsymbol{h}_{\Delta}(n) - \mu\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{h}_{\Delta}(n) \\
&= \left[\mathbf{1} - \mu\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\right]\boldsymbol{h}_{\Delta}(n)
\end{aligned}
$$

Geometrical Explanation of Convergence – Part 3

The vector $\boldsymbol{h}_\Delta(n)$ will be split into ***two components***:

$$\boldsymbol{h}_\Delta(n) \quad = \quad \boldsymbol{h}_\Delta^{||}(n) + \boldsymbol{h}_\Delta^{\perp}(n)$$



It applies to ***parallel components***:

$$\boldsymbol{h}_\Delta^{||}(n) \quad = \quad r(n)\,\boldsymbol{x}(n)$$

With:

$$r(n) \quad = \quad \frac{\boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{h}_\Delta(n)}{\big\|\boldsymbol{x}(n)\big\|^2}$$

$$\big\|\boldsymbol{x}(n)\big\|^2 \quad = \quad \boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{x}(n) \quad = \quad \sum_{l=0}^{N-1} x^2(n-l)$$

$$\boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{h}_\Delta(n) \quad = \quad \boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{h}_\Delta^{||}(n) \quad = \quad \boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{x}(n)\,r(n)$$

Geometrical Explanation of Convergence – Part 4

Contraction of the system error vector:

… result obtained two slides before …

$$\boldsymbol{h}_{\Delta}(n+1) = \Big[\mathbf{1} - \mu\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\Big]\,\boldsymbol{h}_{\Delta}(n)$$

… splitting the system error vector …

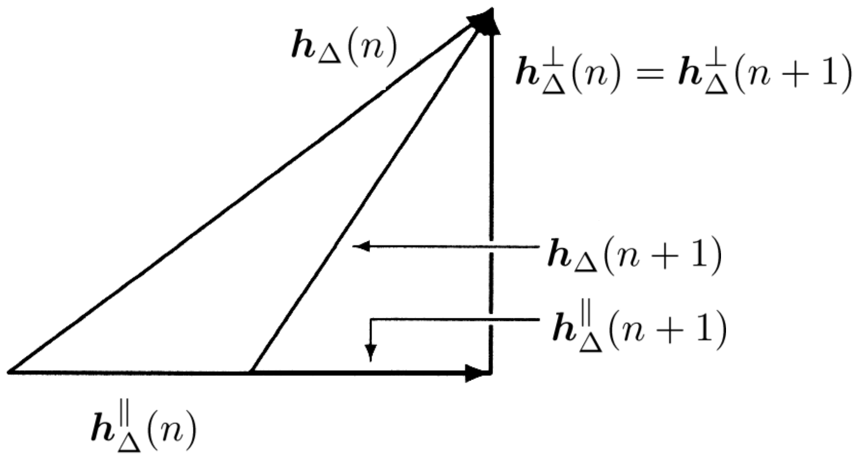
$$= \Big[\mathbf{1} - \mu\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\Big]\,\Big[\boldsymbol{h}_{\Delta}^{\|}(n) + \boldsymbol{h}_{\Delta}^{\perp}(n)\Big]$$

… using $\boldsymbol{h}_{\Delta}^{\|}(n) = r(n)\,\boldsymbol{x}(n)$ and that $\boldsymbol{h}_{\Delta}^{\perp}(n)$ is orthogonal to $\boldsymbol{x}(n)$ …

$$= \Big[1 - \mu\,\big\|\boldsymbol{x}(n)\big\|^2\Big]\,\boldsymbol{h}_{\Delta}^{\|}(n) + \boldsymbol{h}_{\Delta}^{\perp}(n)$$

… this results in …

$$\text{Convergence if } \Big|1 - \mu\,\big\|\boldsymbol{x}(n)\big\|^2\Big| < 1 \text{ and } \boldsymbol{h}_{\Delta}^{\|}(n) \neq 0\,.$$



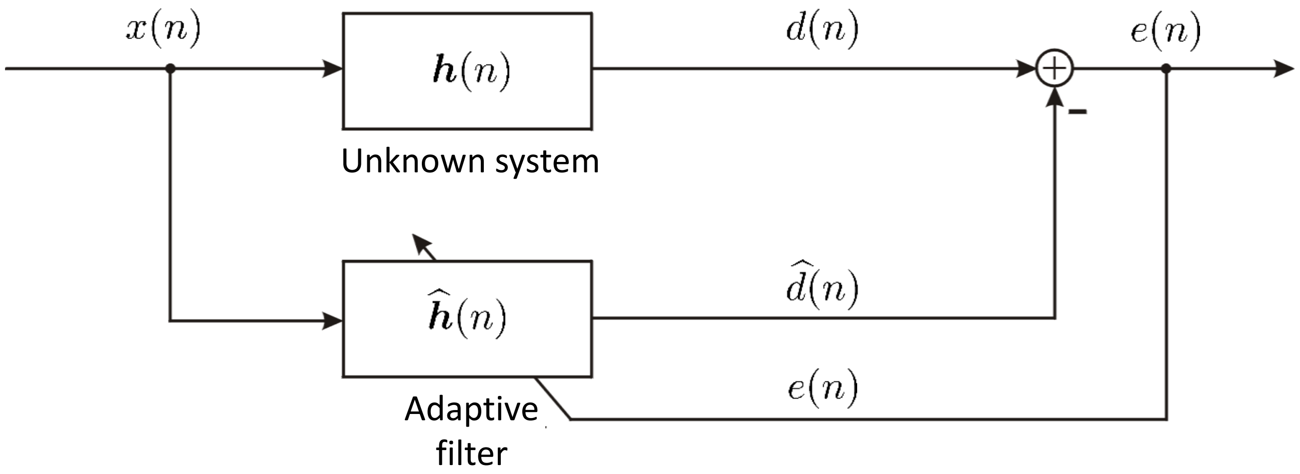
NLMS Algorithm – Part 1

LMS algorithm:

$$\widehat{\boldsymbol{h}}(n+1) \;=\; \widehat{\boldsymbol{h}}(n) + \mu\, e(n)\, \boldsymbol{x}(n)$$

Normalized LMS algorithm:

$$\widehat{\boldsymbol{h}}(n+1) \;=\; \widehat{\boldsymbol{h}}(n) + \mu\, f_{\mathrm{norm}}\big(\boldsymbol{x}(n)\big)\, e(n)\, \boldsymbol{x}(n)$$



NLMS Algorithm – Part 2

Adaption (in general):

$$\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) + \boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n)$$

A priori error:

$$e(n) = e(n|n) = d(n) - \widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{x}(n)$$

A posteriori error:

$$\begin{aligned} e(n|n+1) &= d(n) - \widehat{\boldsymbol{h}}^{\mathrm{T}}(n+1)\,\boldsymbol{x}(n) \\ &= d(n) - \widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{x}(n) - \boldsymbol{\Delta h}^{\mathrm{T}}(n)\,\boldsymbol{x}(n) \\ &= e(n|n) - \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{x}(n) \end{aligned}$$

A ***successful adaptation requires***

$$\big|e(n|n+1)\big| \leq \big|e(n|n)\big|$$

or:

$$\big[e(n|n+1)\big]^2 \leq \big[e(n|n)\big]^2$$

NLMS Algorithm – Part 3

Convergence condition:

$$\big[e(n|n+1)\big]^2 \quad \leq \quad \big[e(n|n)\big]^2$$

Inserting the update equation:

$$\begin{aligned}\big[e(n|n+1)\big]^2 \quad &= \quad \Big[e(n|n) - \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{x}(n)\Big]\,\Big[e(n|n) - \boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n)\Big] \\ &= \quad \big[e(n|n)\big]^2 - e(n|n)\,\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \\ &\quad - \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{x}(n)\,e(n|n) + \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n)\end{aligned}$$

Condition:

$$\begin{aligned} e(n|n)\,\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \quad &+ \quad e(n|n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}\,\boldsymbol{x}(n) \\ &- \quad \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \geq 0\end{aligned}$$

Ansatz:

$$\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \quad = \quad \mu\, e(n|n)\,\frac{\boldsymbol{M}(n)\,\boldsymbol{x}(n)}{\boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{M}(n)\,\boldsymbol{x}(n)}$$

with a real and symmetric matrix $\boldsymbol{M}(n)$

NLMS Algorithm – Part 4

Condition:

$$\begin{aligned} e(n|n)\,\boldsymbol{x}^{\mathrm{T}}(n)\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \quad &+ \quad e(n|n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{x}(n) \\ &- \quad \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \;\geq\; 0 \end{aligned}$$

Ansatz:

$$\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \quad = \quad \mu\, e(n|n)\,\frac{\boldsymbol{M}(n)\,\boldsymbol{x}(n)}{\boldsymbol{x}^{\mathrm{T}}(n)\,\boldsymbol{M}(n)\,\boldsymbol{x}(n)}$$

Step size requirement for the NLMS algorithm (after a few lines …):

$$\mu\;[2-\mu] > 0 \qquad \text{or} \qquad \boxed{0 < \mu < 2}$$

For comparison with LMS algorithm: $0 < \mu < \dfrac{2}{N\,\sigma_x^2}$

NLMS Algorithm – Part 5

Ansatz:

$$\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) = \mu\, e(n|n)\, \frac{\boldsymbol{M}(n)\, \boldsymbol{x}(n)}{\boldsymbol{x}^{\mathrm{T}}(n)\, \boldsymbol{M}(n)\, \boldsymbol{x}(n)}$$

In the most simple case $\boldsymbol{M}(n)$ is a unit matrix:

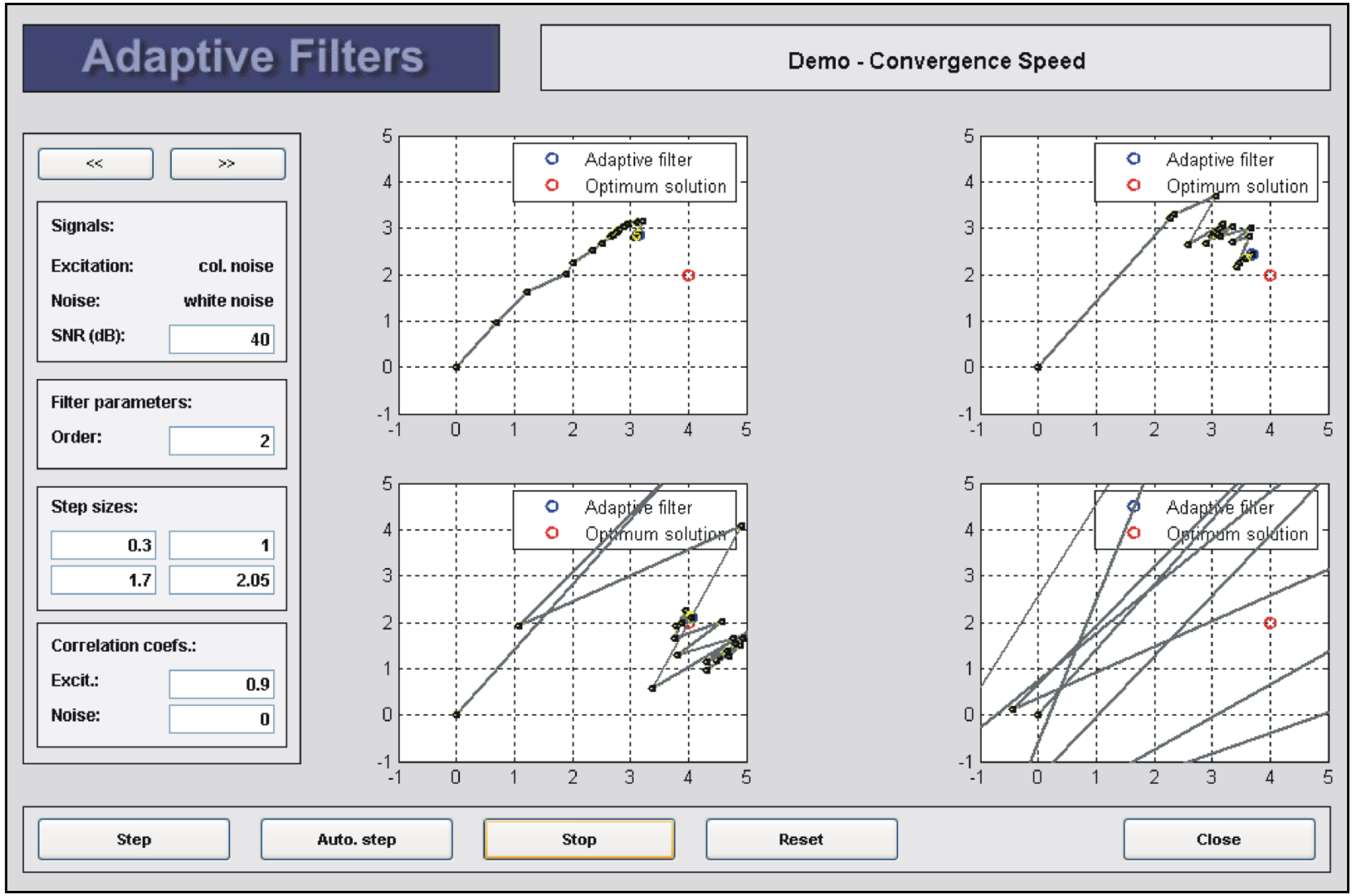
$$\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) = \mu\, e(n)\, \frac{\boldsymbol{x}(n)}{\boldsymbol{x}^{\mathrm{T}}(n)\, \boldsymbol{x}(n)}$$

Adaptation rule for the NLMS algorithm:

$$\boxed{\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) + \frac{\mu}{\boldsymbol{x}^{\mathrm{T}}(n)\, \boldsymbol{x}(n)}\, \boldsymbol{x}(n)\, e(n)}$$

C | A | U
Christian-Albrechts-Universität zu Kiel

Matlab-Demo: Speed of Convergence



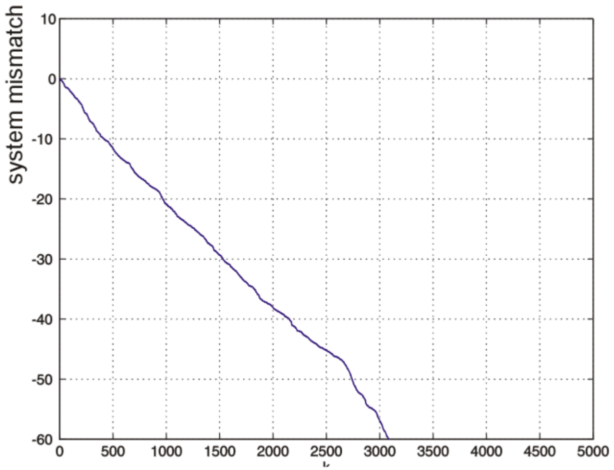
Christian-Albrechts-Universität zu Kiel

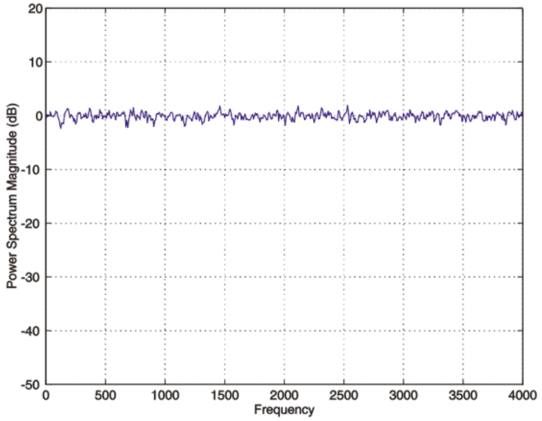
Convergence Examples – Part 1

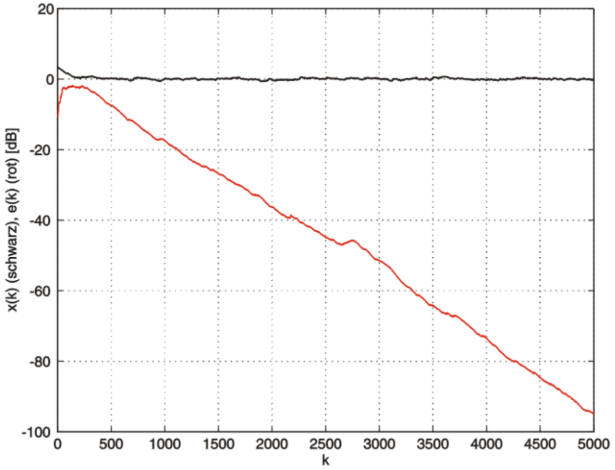
Setup:

White noise:

$$
\begin{aligned}
N &= 256 \\
\mu &= 1
\end{aligned}
$$







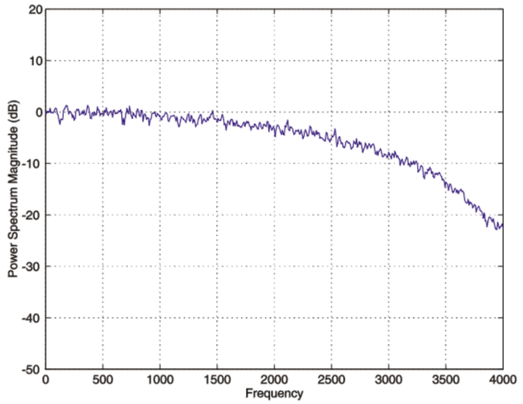
Convergence Examples – Part 2

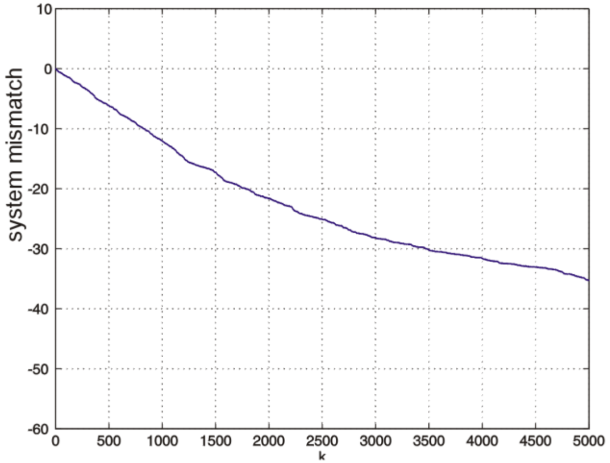
Setup:

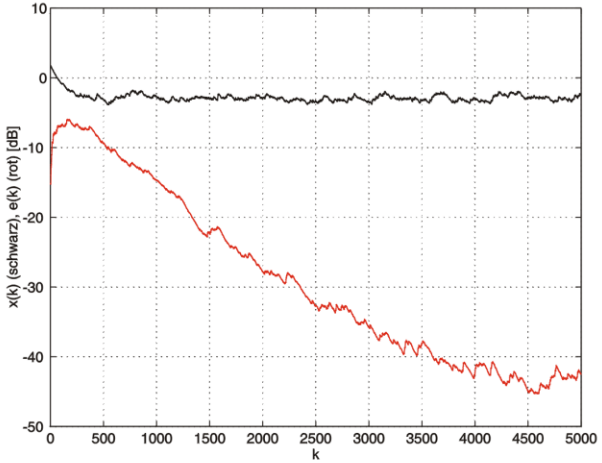
Colored noise:

$$N = 256$$

$$\mu = 1$$



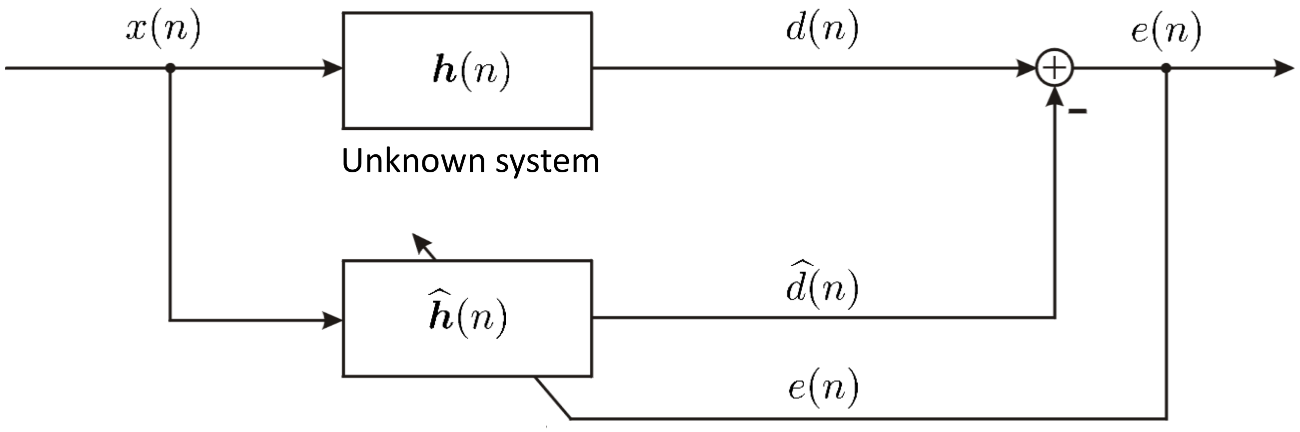




Today

Adaptive Algorithms:

Basics



Signal vector: $\boldsymbol{x}(n) = \Big[x(n),\, x(n-1),\, x(n-2),\, \ldots,\, x(n-N+1)\Big]^{\mathrm{T}}$

Filter vector: $\widehat{\boldsymbol{h}}(n) = \Big[\widehat{h}_0(n),\, \widehat{h}_1(n),\, \widehat{h}_2(n),\, \ldots,\, \widehat{h}_{N-1}(n)\Big]^{\mathrm{T}}$

Filter output: $\widehat{d}(n) = \widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\, \boldsymbol{x}(n) = \boldsymbol{x}^{\mathrm{T}}(n)\, \widehat{\boldsymbol{h}}(n)$

Signal matrix: $\boldsymbol{X}(n) = \Big[\boldsymbol{x}(n),\, \boldsymbol{x}(n-1),\, \ldots,\, \boldsymbol{x}(n-L+1)\Big]$

L describes the order of the procedure

Signal Matrix

Definition of the signal matrix:

$$
\begin{aligned}
\boldsymbol{X}(n) &= \Big[\boldsymbol{x}(n),\, \boldsymbol{x}(n-1),\, ...,\, \boldsymbol{x}(n-L+1)\Big] \\
&= \begin{bmatrix} x(n) & x(n-1) & \dots & x(n-(L-1)) \\ x(n-1) & x(n-2) & \dots & x(n-L) \\ x(n-2) & x(n-3) & \dots & x(n-L-1) \\ \vdots & \vdots & \ddots & \vdots \\ x(n-(N-1)) & x(n-1-(N-1)) & \dots & x(n-(L-1)-(N-1)) \end{bmatrix}_{N\times L} \\
&= \begin{bmatrix} x(n) & x(n-1) & \dots & x(n-(L-1)) \\ x(n-1) & x(n-2) & \dots & x(n-L) \\ x(n-2) & x(n-3) & \dots & x(n-L-1) \\ \vdots & \vdots & \ddots & \vdots \\ x(n-N+1) & x(n-N) & \dots & x(n-L-N+2) \end{bmatrix}_{N\times L}
\end{aligned}
$$

Error Vector – Part 1

Signal matrix:

$$\boldsymbol{X}(n) = \Big[\boldsymbol{x}(n),\, \boldsymbol{x}(n-1),\, \ldots,\, \boldsymbol{x}(n-L+1)\Big]$$

Desired signal vector:

$$\boldsymbol{d}(n) = \Big[d(n),\, d(n-1),\, \ldots,\, d(n-L+1)\Big]^{\mathrm{T}}$$

Filter output vector:

$$\widehat{\boldsymbol{d}}(n) = \Big[\widehat{d}(n),\, \widehat{d}(n-1),\, \ldots,\, \widehat{d}(n-L+1)\Big]^{\mathrm{T}}$$

A priori error vector:

$$\boldsymbol{e}(n|n) = \boldsymbol{d}(n) - \widehat{\boldsymbol{d}}(n) = \boldsymbol{d}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\widehat{\boldsymbol{h}}(n)$$

Adaption rule:

$$\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) + \boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n)$$

A posteriori error vector:

$$\begin{aligned}
\boldsymbol{e}(n|n+1) &= \boldsymbol{d}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\widehat{\boldsymbol{h}}(n+1) \\
&= \boldsymbol{d}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\widehat{\boldsymbol{h}}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \\
&= \boldsymbol{e}(n|n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n)
\end{aligned}$$

Error Vector – Part 2

$$\begin{aligned} \boldsymbol{e}(n|n+1) &= \boldsymbol{d}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\widehat{\boldsymbol{h}}(n+1) \\ &= \boldsymbol{d}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\widehat{\boldsymbol{h}}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \\ &= \boldsymbol{e}(n|n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \end{aligned}$$

Requirement: $\big\|\boldsymbol{e}(n|n+1)\big\| \leq \big\|\boldsymbol{e}(n|n)\big\|$

$$\big\|\boldsymbol{e}(n|n+1)\big\|^2 \leq \big\|\boldsymbol{e}(n|n)\big\|^2$$

$$\begin{aligned} \|\boldsymbol{e}(n|n+1)\|^2 &= \Big[\boldsymbol{e}^{\mathrm{T}}(n|n) - \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{X}(n)\Big]\,\Big[\boldsymbol{e}(n|n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n)\Big] \\ &= \big\|\boldsymbol{e}(n|n)\big\|^2 - \boldsymbol{e}^{\mathrm{T}}(n|n)\,\boldsymbol{X}^{\mathrm{T}}(n)\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \\ &\quad - \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{X}(n)\,\boldsymbol{e}(n|n) + \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{X}(n)\,\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \end{aligned}$$

Requirement:

$$\begin{aligned} \boldsymbol{e}^{\mathrm{T}}(n|n)\,\boldsymbol{X}^{\mathrm{T}}(n)\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \;&+\; \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{X}(n)\,\boldsymbol{e}^{*}(n|n) \\ &-\; \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{X}(n)\,\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \geq 0 \end{aligned}$$

Ansatz

Requirement:

$$\boldsymbol{e}^{\mathrm{T}}(n|n)\,\boldsymbol{X}^{\mathrm{T}}(n)\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \;+\; \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{X}(n)\,\boldsymbol{e}^{*}(n|n) \;-\; \boldsymbol{\Delta}\widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{X}(n)\,\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) \geq 0$$

Ansatz:

$$\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n) = \mu\,\boldsymbol{M}(n)\,\boldsymbol{X}(n)\,\Big[\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{M}(n)\,\boldsymbol{X}(n)\Big]^{-1}\,\boldsymbol{e}(n|n)$$

with the symmetric matrix $\boldsymbol{M}(n) = \boldsymbol{M}^{\mathrm{T}}(n)$

Step-size condition:

$$\mu\,[2-\mu] > 0 \qquad \text{or} \qquad \boxed{0 < \mu < 2}$$

In the most simple case $\boldsymbol{M}(n)$ is the unit matrix:

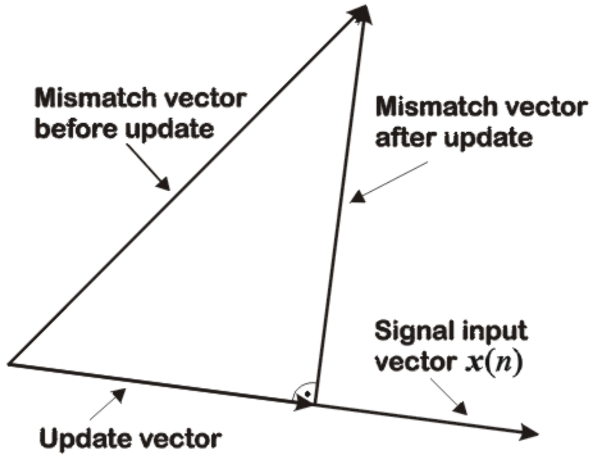
$$\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{X}(n)\,\Big[\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n)\Big]^{-1}\,\boldsymbol{e}(n|n)$$

Geometrical Interpretation

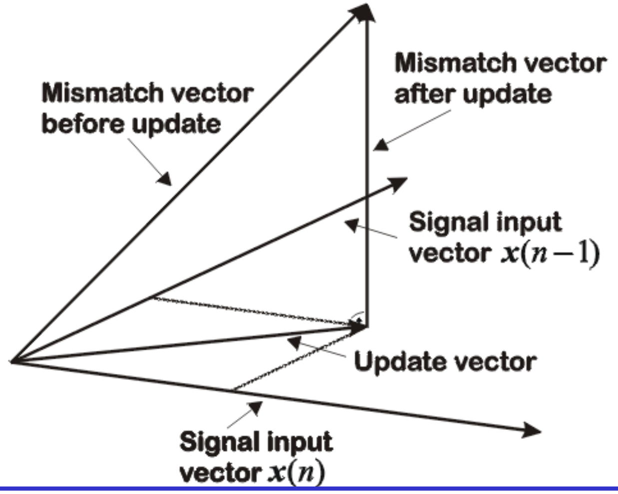
$$\boxed{\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) + \mu\, \boldsymbol{X}(n) \left[\boldsymbol{X}^{\mathrm{T}}(n)\, \boldsymbol{X}(n)\right]^{-1}_{M\times M} \boldsymbol{e}(n|n)}$$

$$\boldsymbol{e}(n|n) = \boldsymbol{d}(n) - \widehat{\boldsymbol{d}}(n) = \boldsymbol{d}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\, \widehat{\boldsymbol{h}}(n)$$

NLMS algorithm



AP algorithm



Regularization

Non-regularised version of the AP algorithm:

$$\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{X}(n)\,\Big[\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n)\Big]^{-1}_{L\times L}\,\boldsymbol{e}(n|n)$$

Regularised version of the AP algorithm:

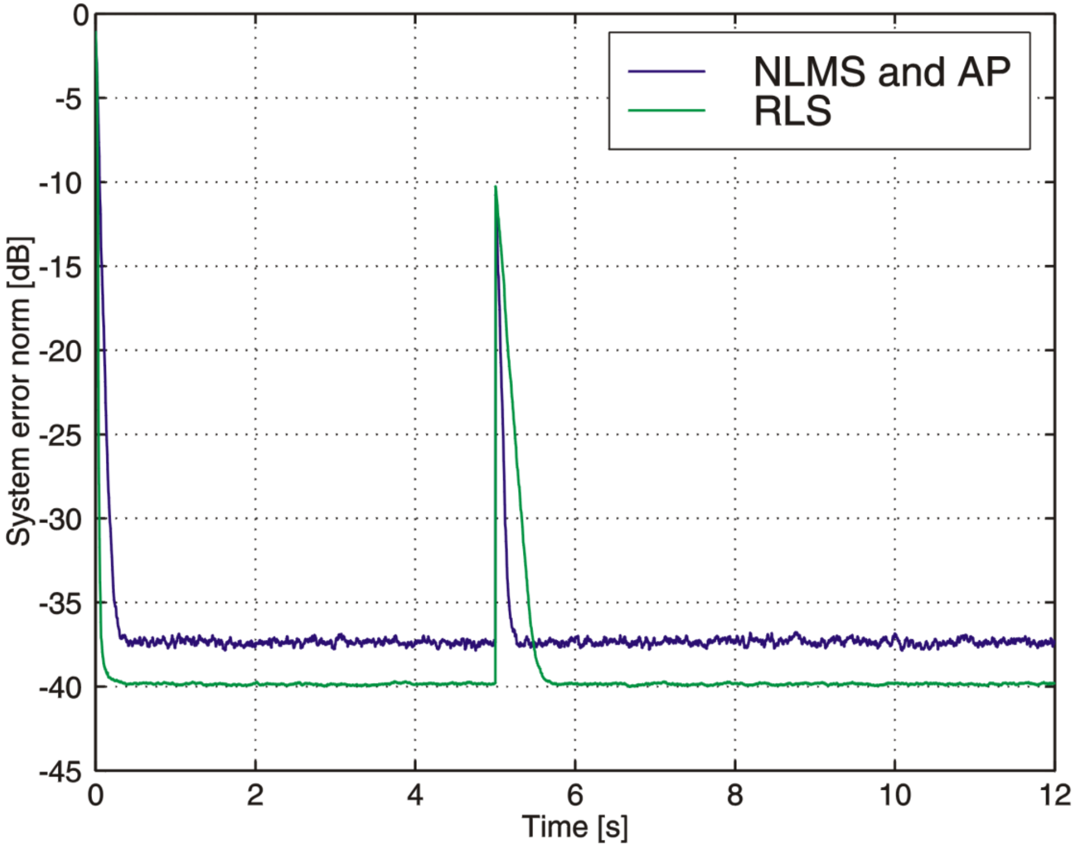
$$\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{X}(n)\,\Big[\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) + \Delta\,\boldsymbol{I}\Big]^{-1}_{L\times L}\,\boldsymbol{e}(n|n)\ ,$$

where Δ is a small positive constant.

Convergence of Different Algorithms – Part 1

White noise:

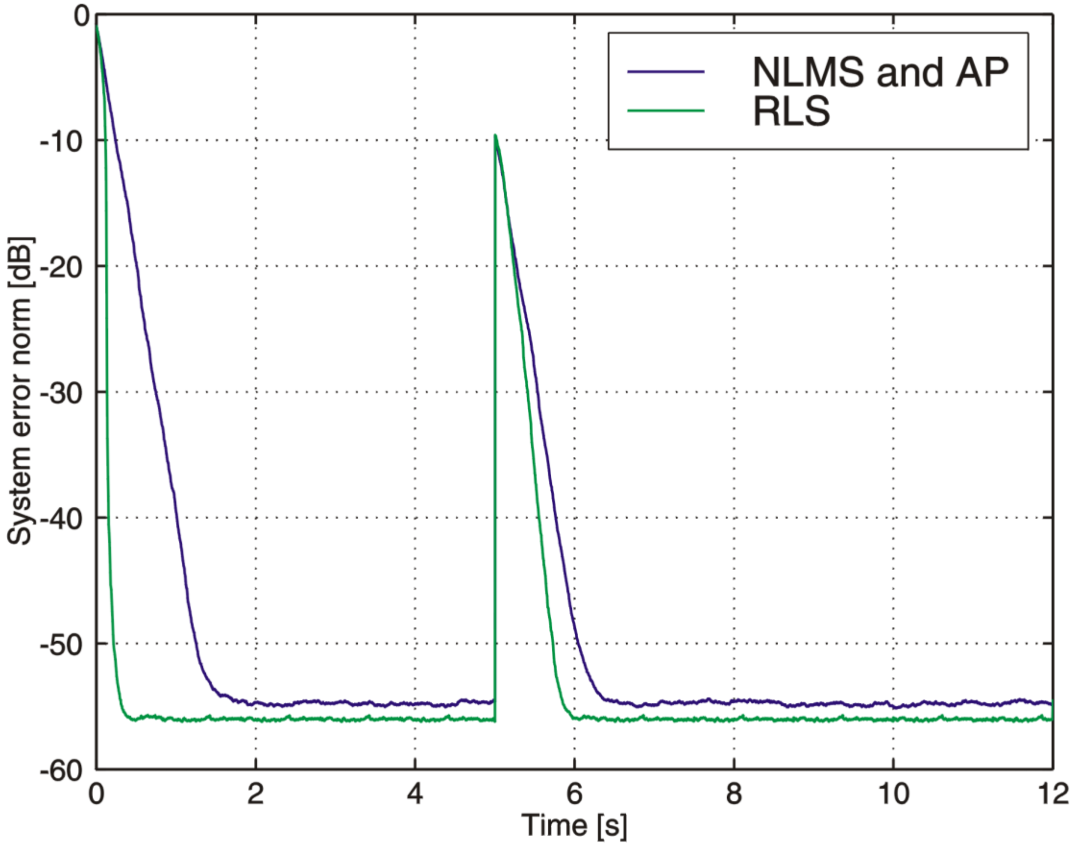
$$
\begin{aligned}
N &= 256 \\
\mu &= 1 \\
\lambda &= 0.999
\end{aligned}
$$



Convergence of Different Algorithms – Part 2

White noise:

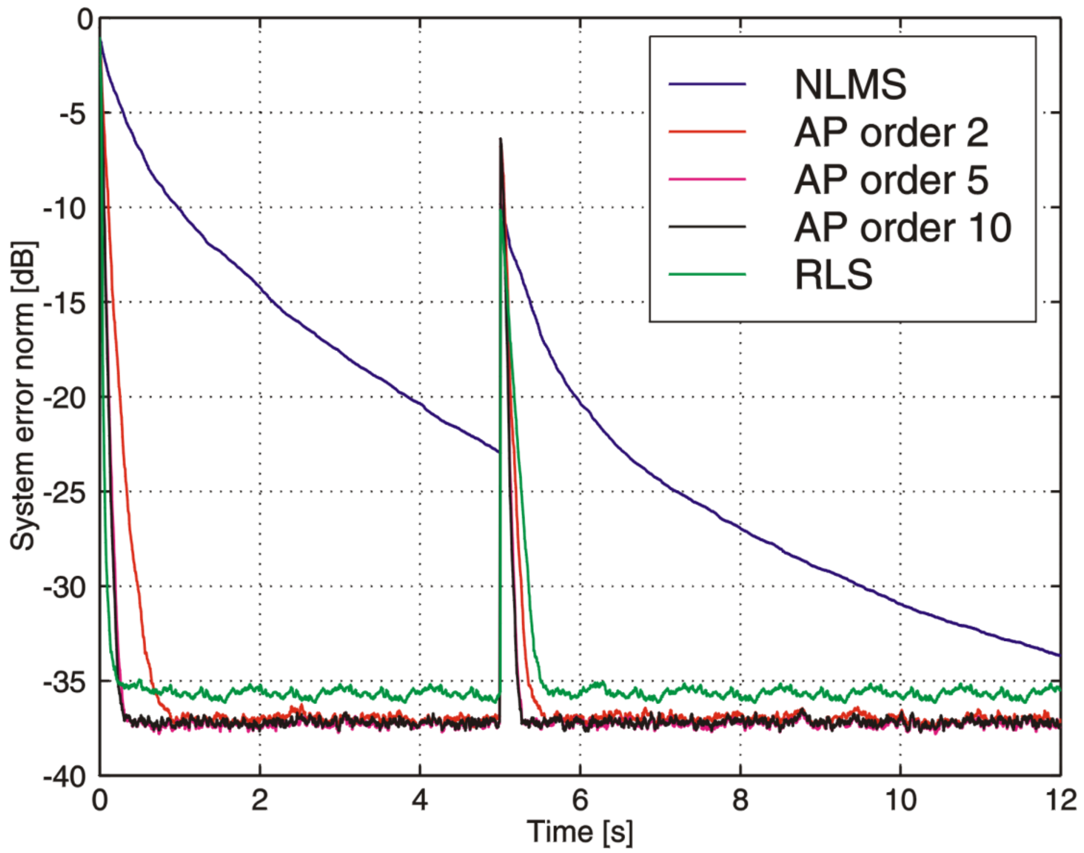
$$\begin{aligned} N &= 1024 \\ \mu &= 1 \\ \lambda &= 0.999 \end{aligned}$$



Convergence of Different Algorithms – Part 3

Colored noise

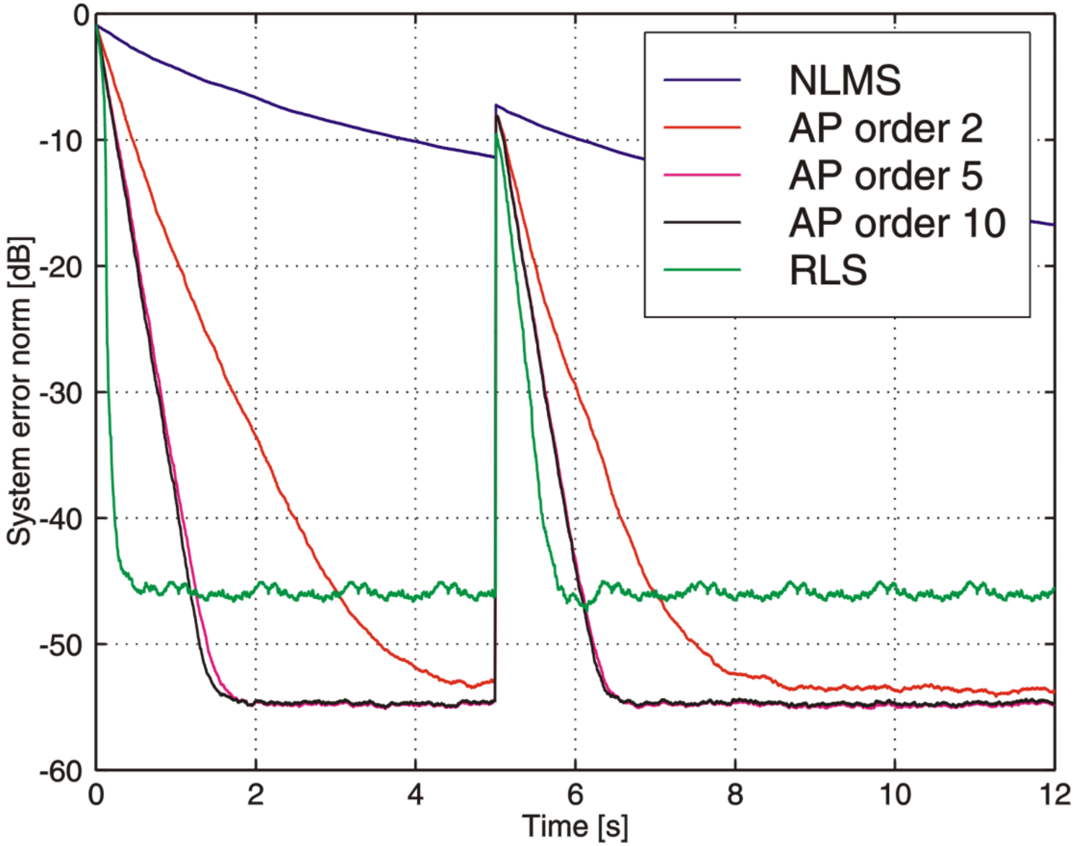
$$N = 256$$
$$\mu = 1$$
$$\lambda = 0.999$$



Convergence of Different Algorithms – Part 4

Colored noise:

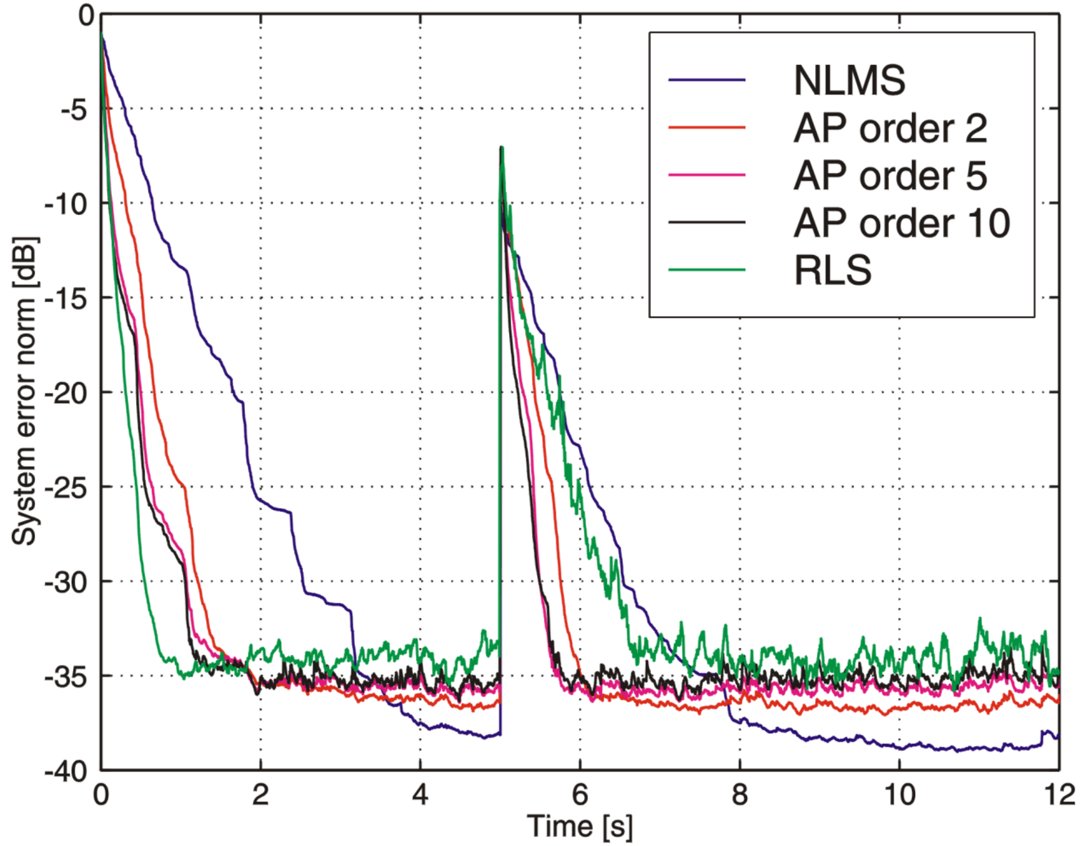
$$\begin{aligned} N &= 1024 \\ \mu &= 1 \\ \lambda &= 0.999 \end{aligned}$$



Convergence of Different Algorithms – Part 5

Speech:

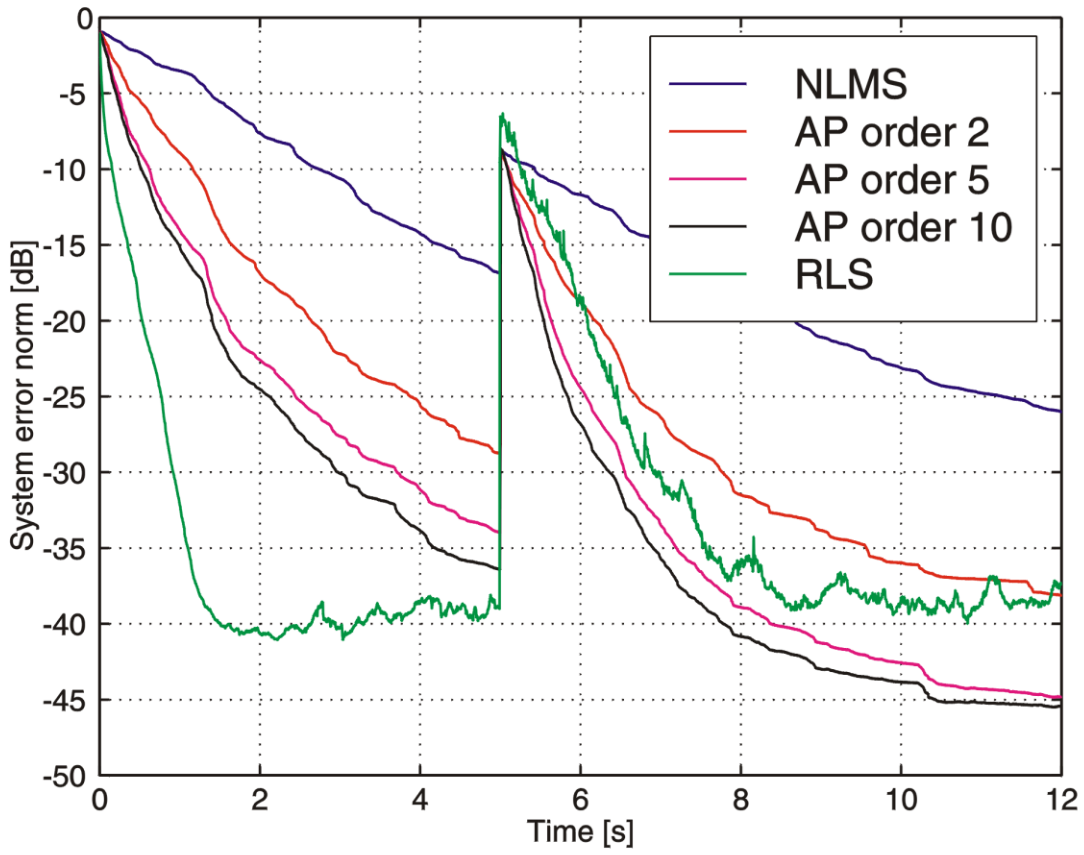
$$N = 256$$
$$\mu = 1$$
$$\lambda = 0.999$$



Convergence of Different Algorithms – Part 6

Speech:

$$
\begin{aligned}
N &= 1024 \\
\mu &= 1 \\
\lambda &= 0.999
\end{aligned}
$$



Summary and Outlook

This week and last week:

- Introductory Remarks
- Recursive Least Squares (RLS) Algorithm
- Least Mean Square Algorithm (LMS Algorithm) – Part 1
- Least Mean Square Algorithm (LMS Algorithm) – Part 2
- Affine Projection Algorithm (AP Algorithm)
- Fast Affine Projection Algorithm (FAP Algorithm)

Next part:

- Control of Adaptive Filters

Contents:

Introduction

Fast version of adaptive algorithms:

- Until now we have mainly focused on ***direct implementations*** of adaptive algorithms.
- Usually, we found that the ***more robust*** (e.g. in terms on independence on the input statistics) an algorithms is, the ***more expensive*** (e.g. in terms of multiplications and additions) it is.

Introduction

Fast version of adaptive algorithms:

- Until new we have mainly focused on ***direct implementations*** of adaptive algorithms.
- Usually, we found that the ***more robust*** (e.g. in terms on independence on the input statistics) an algorithms is, the ***more expensive*** (e.g. in terms of multiplications and additions) it is.
- Now we will focus on so-called fast versions of algorithms.
- These fast versions exist for virtually all algorithms.
- The problem is often, that numerical stability is not easy to achieve.
- We will focus now on a fast version of the fast affine projection algorithm, shorty called FAP.

Introduction

Fast version of adaptive algorithms:

- Invented by Steve(n) L. Grant (AKA Gay) at Bell Labs in 1995.
- A very interesting algorithm, since it combines RLS-like speed for colored signals with LMS-like complexity.
- Steve was (unfortunately, he died a couple of years ago) a very nice guy, and the upcoming slides are dedicated to him: “To Steve, a great, clever and smart researcher with a big friendly heart”.

Steven L. Grant (AKA Gay, together with his wife Maria), picture made at ICASSP Brisbane, 2015 [Photo G. Elko]

Contents:

Steven L. Grant (AKA Gay, together with Peter Eneroth [left], Tomas Gänsler [third] and Jacob Benesty [right]), picture made at ICASSP Seattle, 1998 [Photo Maria Grant]

NLMS versus Affine Projection

Basic NLMS equations:

- Computation of the ***error*** signal

$$e(n) \;=\; y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

- ***Norm*** of the excitation vector

$$\big\|\boldsymbol{x}(n)\big\|^2 \;=\; \big\|\boldsymbol{x}(n-1)\big\|^2 + x^2(n) - x^2(n-N)$$

- ***Normalization*** of the error signal

$$e_{\mathrm{norm}}(n) \;=\; \frac{e(n)}{\|\boldsymbol{x}(n)\|^2 + \Delta}$$

- Coefficient ***update***

$$\hat{\boldsymbol{h}}(n+1) \;=\; \hat{\boldsymbol{h}}(n+1) + \mu\,\boldsymbol{x}(n)\,e_{\mathrm{norm}}(n)$$

NLMS versus Affine Projection

Basic NLMS equations:

- Computation of the ***error*** signal

$$e(n) \;=\; y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

- ***Norm*** of the excitation vector

$$\|\boldsymbol{x}(n)\|^2 \;=\; \|\boldsymbol{x}(n-1)\|^2 + x^2(n) - x^2(n-N)$$

- ***Normalization*** of the error signal

$$e_{\mathrm{norm}}(n) \;=\; \frac{e(n)}{\|\boldsymbol{x}(n)\|^2 + \Delta}$$

- Coefficient ***update***

$$\hat{\boldsymbol{h}}(n+1) \;=\; \hat{\boldsymbol{h}}(n+1) + \mu\,\boldsymbol{x}(n)\,e_{\mathrm{norm}}(n)$$

Computational complexity

N additions,
N multiplications

2 additions,
2 multiplications

1 addition,
1 multiplication,
1 division

N additions,
N multiplications

NLMS versus Affine Projection

Basic NLMS equations:

- Computation of the ***error*** signal

$$e(n) \;=\; y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

- ***Norm*** of the excitation vector

$$\|\boldsymbol{x}(n)\|^2 \;=\; \|\boldsymbol{x}(n-1)\|^2 + x^2(n) - x^2(n-N)$$

- ***Normalization*** of the error signal

$$e_{\mathrm{norm}}(n) \;=\; \frac{e(n)}{\|\boldsymbol{x}(n)\|^2 + \Delta}$$

- Coefficient ***update***

$$\hat{\boldsymbol{h}}(n+1) \;=\; \hat{\boldsymbol{h}}(n+1) + \mu\,\boldsymbol{x}(n)\,e_{\mathrm{norm}}(n)$$

Computational complexity

N additions,
N multiplications

2 additions,
2 multiplications

1 addition,
1 multiplication,
1 division

N additions,
N multiplications

Complexity NLMS:

$2N+3 \approx 2N$ additions,
$2N+3 \approx 2N$ multiplications,
1 division

Example:

$f_{\mathrm{s}} = 48\,\mathrm{kHz}$

$N = 12000$

1.2 billion additions per second,
1.2 billion multiplic. per second,
48.000 divisions per second

Long and Short Excitation Vectors

Excitation vector definitions:

- Conventional excitation vector

$$\boldsymbol{x}(n) = \big[x(n),\, x(n-1),\, ...,\, x(n-L+1),\, x(n-L),\, ...,\, x(n-N+1)\big]^{\mathrm{T}}$$

- Short excitation vector (usually contained in conventional excitation vector)

$$\boldsymbol{x}_{\text{short}}(n) = \big[x(n),\, x(n-1),\, ...,\, x(n-L+1)\big]^{\mathrm{T}}$$

NLMS versus Affine Projection (Continued)

Basic affine projection equations:

- Computation of the ***error*** signal vector

$$\boldsymbol{e}(n) = \boldsymbol{y}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

- ***Normalization*** matrix

$$\begin{aligned}\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) &= \boldsymbol{X}^{\mathrm{T}}(n-1)\,\boldsymbol{X}(n-1)\\ &\quad + \boldsymbol{x}_{\mathrm{short}}^{\mathrm{T}}(n)\,\boldsymbol{x}_{\mathrm{short}}(n)\\ &\quad - \boldsymbol{x}_{\mathrm{short}}^{\mathrm{T}}(n-N)\,\boldsymbol{x}_{\mathrm{short}}(n-N)\end{aligned}$$

- ***Normalization*** of the error vector

$$\boldsymbol{e}_{\mathrm{norm}}(n) = \Big[\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) + \Delta\,\boldsymbol{I}\Big]^{-1}\boldsymbol{e}(n)$$

- Coefficient ***update***

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{X}(n)\,\boldsymbol{e}_{\mathrm{norm}}(n)$$

Basic NLMS equations (for comparison):

- Computation of the ***error*** signal

$$e(n) = y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

- ***Norm*** of the excitation vector

$$\big\|\boldsymbol{x}(n)\big\|^2 = \big\|\boldsymbol{x}(n-1)\big\|^2 + x^2(n) - x^2(n-N)$$

- ***Normalization*** of the error signal

$$e_{\mathrm{norm}}(n) = \frac{e(n)}{\|\boldsymbol{x}(n)\|^2 + \Delta}$$

- Coefficient ***update***

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{x}(n)\,e_{\mathrm{norm}}(n)$$

NLMS versus Affine Projection (Continued)

Basic affine projection equations:

- Computation of the ***error*** signal vector

$$\boldsymbol{e}(n) = \boldsymbol{y}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

- ***Normalization*** matrix

$$\begin{aligned}\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) \;=\;& \boldsymbol{X}^{\mathrm{T}}(n-1)\,\boldsymbol{X}(n-1)\\ &+ \boldsymbol{x}_{\mathrm{short}}^{\mathrm{T}}(n)\,\boldsymbol{x}_{\mathrm{short}}(n)\\ &- \boldsymbol{x}_{\mathrm{short}}^{\mathrm{T}}(n-N)\,\boldsymbol{x}_{\mathrm{short}}(n-N)\end{aligned}$$

- ***Normalization*** of the error vector

$$\boldsymbol{e}_{\mathrm{norm}}(n) = \left[\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) + \Delta\,\boldsymbol{I}\right]^{-1}\boldsymbol{e}(n)$$

- Coefficient ***update***

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{X}(n)\,\boldsymbol{e}_{\mathrm{norm}}(n)$$

Computational complexity

N x L additions,
N x L multiplications

2 x L^2 additions,
2 x L^2 multiplications

L^2 additions,
L^2 multiplications,
1 inversion (L^3 mult.,
L^3 add.)

N x L additions,
N x L multiplications

NLMS versus Affine Projection (Continued)

Basic affine projection equations:

- Computation of the ***error*** signal vector

$$\boldsymbol{e}(n) = \boldsymbol{y}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

- ***Normalization*** matrix

$$\begin{aligned}\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) \;=\; & \boldsymbol{X}^{\mathrm{T}}(n-1)\,\boldsymbol{X}(n-1) \\ & + \boldsymbol{x}_{\mathrm{short}}^{\mathrm{T}}(n)\,\boldsymbol{x}_{\mathrm{short}}(n) \\ & - \boldsymbol{x}_{\mathrm{short}}^{\mathrm{T}}(n-N)\,\boldsymbol{x}_{\mathrm{short}}(n-N)\end{aligned}$$

- ***Normalization*** of the error vector

$$\boldsymbol{e}_{\mathrm{norm}}(n) = \Big[\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) + \Delta\,\boldsymbol{I}\Big]^{-1}\boldsymbol{e}(n)$$

- Coefficient ***update***

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{X}(n)\,\boldsymbol{e}_{\mathrm{norm}}(n)$$

Computational complexity

N x L additions,
N x L multiplications

2 x L^2 additions,
2 x L^2 multiplications

L^2 additions,
L^2 multiplications,
1 inversion (L^3 mult.,
L^3 add.)

N x L additions,
N x L multiplications

Complexity AP (approx.):

$2NL + 3L^2 + L^3 \approx 2NL$ add.,
$2NL + 3L^2 + L^3 \approx 2NL$ mul.

Example:

$f_{\mathrm{s}} = 48\,\mathrm{kHz}$

$N = 12000$

$L = 4$

4.6 billion additions per second,
4.6 billion multiplic. per second

NLMS versus Affine Projection (Continued)

Basic affine projection equations:

- Computation of the ***error*** signal vector

$$\boldsymbol{e}(n) = \boldsymbol{y}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

- ***Normalization*** matrix

$$\begin{aligned}\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) &= \boldsymbol{X}^{\mathrm{T}}(n-1)\,\boldsymbol{X}(n-1)\\ &\quad + \boldsymbol{x}_{\mathrm{short}}^{\mathrm{T}}(n)\,\boldsymbol{x}_{\mathrm{short}}(n)\\ &\quad - \boldsymbol{x}_{\mathrm{short}}^{\mathrm{T}}(n-N)\,\boldsymbol{x}_{\mathrm{short}}(n-N)\end{aligned}$$

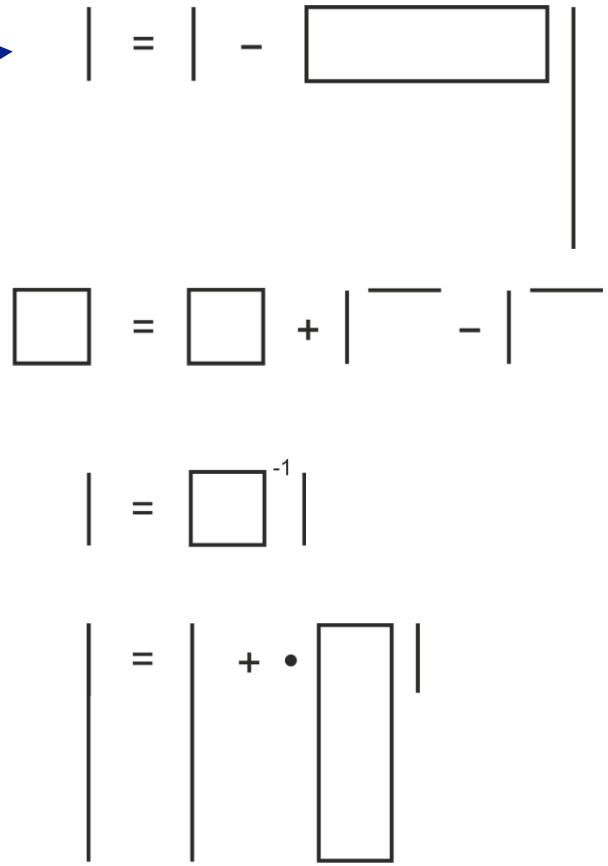
- ***Normalization*** of the error vector

$$\boldsymbol{e}_{\mathrm{norm}}(n) = \left[\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) + \Delta\,\boldsymbol{I}\right]^{-1}\boldsymbol{e}(n)$$

- Coefficient ***update***

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{X}(n)\,\boldsymbol{e}_{\mathrm{norm}}(n)$$

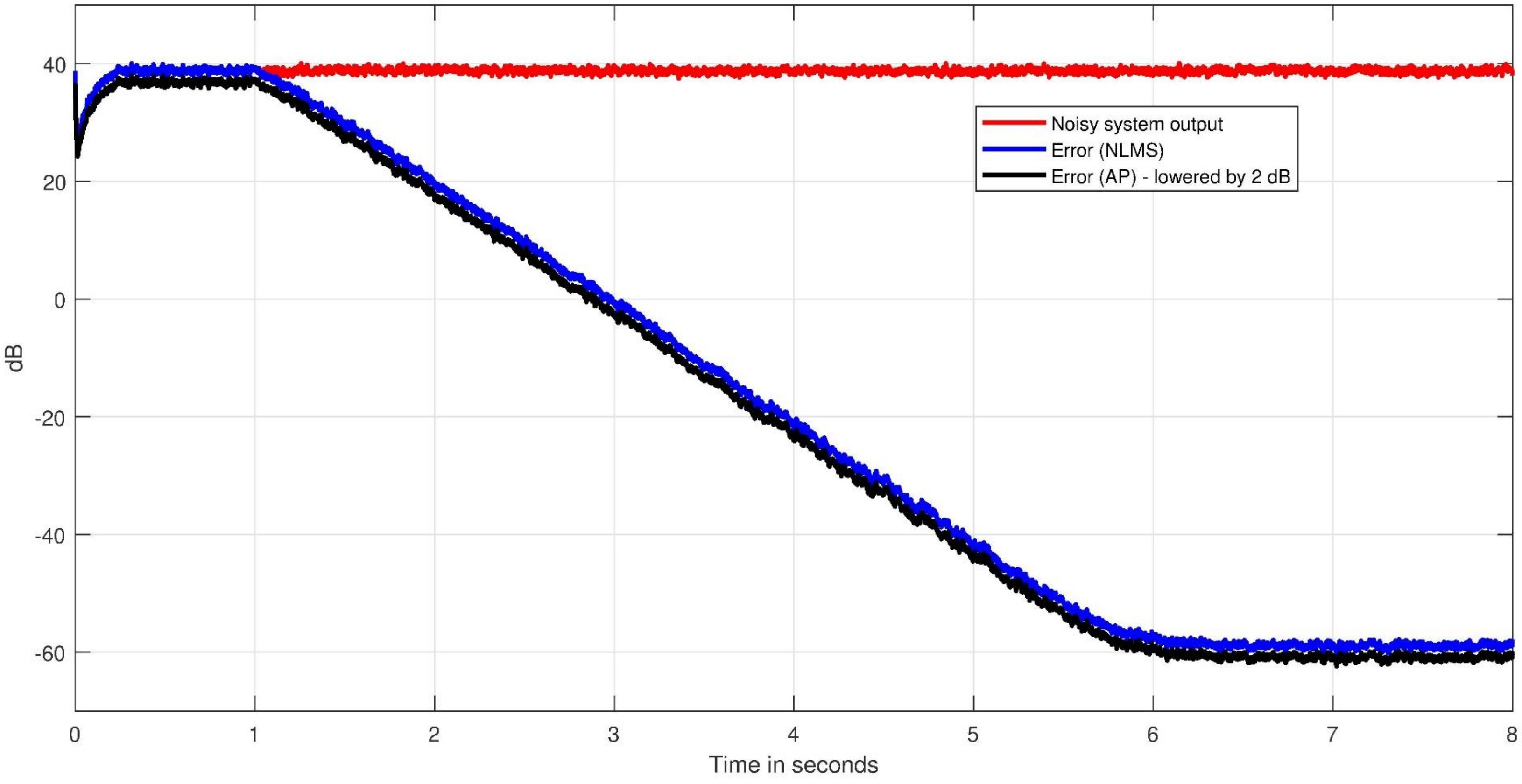
Graphical visualization:



NLMS versus Affine Projection (Continued)

Boundary conditions of the simulation:

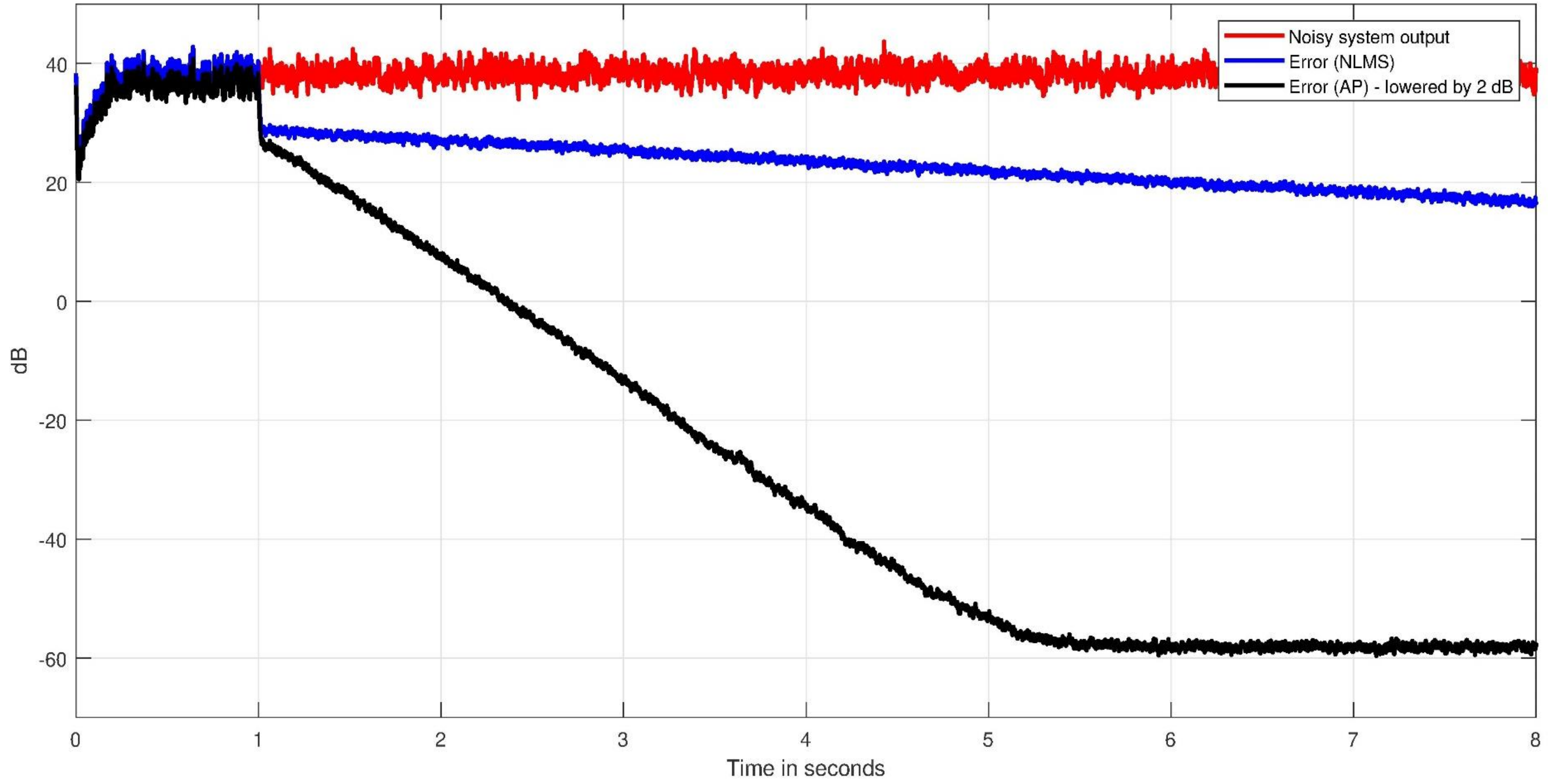
- Excitation: white noise
- Local noise: white noise
- SNR: 60 dB
- Filter length: 12000
- Sample rate: 48 kHz
- Projection order: 4



NLMS versus Affine Projection (Continued)

Boundary conditions of the simulation:

- Excitation: colored noise
- Local noise: white noise
- SNR: 60 dB
- Filter length: 12000
- Sample rate: 48 kHz
- Projection order: 4



Contents:

Steven L. Grant (AKA Gay) while doing pool billiard
[Photo Maria Grant]

From Affine Projection to Fast Affine Projection

Fast computation of the error vector:

- Rearranging the equation for computing the ***error vector***:

$$\boldsymbol{e}(n) = \boldsymbol{e}(n|n) = \boldsymbol{y}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

… splitting the error vector into its first element and the remaining ones …

$$= \begin{bmatrix} y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n) \\ \bar{\boldsymbol{y}}(n-1) - \bar{\boldsymbol{X}}^{\mathrm{T}}(n-1)\,\hat{\boldsymbol{h}}(n) \end{bmatrix}$$

… inserting the definitions of "shortened" vectors and matrices …

$$= \begin{bmatrix} e(n|n) \\ \bar{\boldsymbol{e}}(n-1|n) \end{bmatrix}$$

- ***Quantities with a bar*** indicating the uppermost $L-1$ elements of the corresponding quantities (without the bar):

$$\boldsymbol{e}(n|n) = \begin{bmatrix} e(n|n) \\ \bar{\boldsymbol{e}}(n-1|n) \end{bmatrix}, \qquad \boldsymbol{y}(n) = \begin{bmatrix} y(n) \\ \bar{\boldsymbol{y}}(n-1) \end{bmatrix}, \qquad \boldsymbol{X}(n) = \Big[\boldsymbol{x}(n)\ \bar{\boldsymbol{X}}(n-1)\Big].$$

From Affine Projection to Fast Affine Projection

Fast computation of the error vector – continued:

- Furthermore the ***a posteriori error vector*** can also be rewritten:

$$\boldsymbol{e}(n-1|n) = \boldsymbol{y}(n-1) - \boldsymbol{X}^{\mathrm{T}}(n-1)\,\hat{\boldsymbol{h}}(n)$$

… inserting $\hat{h}(n) = \hat{h}(n-1) + \Delta h(n-1)$ *and using the definition of the error vector …*

$$= \boldsymbol{e}(n-1, n-1) - \boldsymbol{X}^{\mathrm{T}}(n-1)\,\boldsymbol{\Delta}\hat{\boldsymbol{h}}(n-1)$$

… inserting the AP update rule $\Delta\hat{h}(n) = \mu\,X(n-1)\left[X^{\mathrm{T}}(n-1)\,X(n-1) + \Delta\right]^{-1} e(n-1|n-1)$ *…*

$$= \boldsymbol{e}(n-1, n-1) - \mu\,\boldsymbol{X}^{\mathrm{T}}(n-1)\,\boldsymbol{X}(n-1)\left[\boldsymbol{X}^{\mathrm{T}}(n-1)\,\boldsymbol{X}(n-1) + \Delta\,\boldsymbol{I}\right]^{-1} \boldsymbol{e}(n-1, n-1)$$

… assuming a small regularization parameter …

$$\approx (1-\mu)\,\boldsymbol{e}(n-1, n-1).$$

- ***Combining this result and the one from the previous slide*** leads to:

$$\boldsymbol{e}(n|n) \approx \begin{bmatrix} e(n|n) \\ (1-\mu)\,\bar{\boldsymbol{e}}(n-1|n-1) \end{bmatrix}.$$

C | A | U
Christian-Albrechts-Universität zu Kiel

From Affine Projection to Fast Affine Projection

Fast computation of the error vector – continued:

- Comparing both versions shows the complexity reduction:
  - ***Original version***:

$$\boldsymbol{e}(n|n) \;=\; \boldsymbol{y}(n) - \boldsymbol{X}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n).$$

  - ***Approximated*** version:

$$\begin{aligned} e(n|n) &= y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n), \\ \boldsymbol{e}(n|n) &\approx \begin{bmatrix} e(n|n) \\ (1-\mu)\,\bar{\boldsymbol{e}}(n-1|n-1) \end{bmatrix}. \end{aligned}$$

Computational complexity

N x L additions,
N x L multiplications

N additions,
N + L multiplications

Reduction by a factor L!

Fast computation of the coefficient update:

- Rearranging the equation for ***updating*** the coefficient vector in an iterative manner:

$$\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}(0) + \mu \sum_{i=0}^{n} \boldsymbol{X}(n-i)\, \boldsymbol{e}_{\mathrm{norm}}(n-i|n-i)$$

… splitting the excitation signal matrix $\boldsymbol{X}(n) = \Big[\boldsymbol{x}(n),\, \boldsymbol{x}(n-1),\, \dots,\, \boldsymbol{x}(n-L+1)\Big]$
and the normalized error vector $\boldsymbol{e}_{\mathrm{norm}}(n) = \Big[e_{\mathrm{norm},0}(n|n),\, e_{\mathrm{norm},1}(n|n),\, \dots,\, e_{\mathrm{norm},L-1}(n|n)\Big]^{\mathrm{T}}$ *…*

$$= \hat{\boldsymbol{h}}(0) + \mu \sum_{i=0}^{n} \sum_{j=0}^{L-1} \boldsymbol{x}(n-i-j)\, e_{\mathrm{norm},j}(n-i|n-i).$$

From Affine Projection to Fast Affine Projection

Fast computation of the coefficient update – continued:

- Result from the ***last slide***:

$$\hat{\boldsymbol{h}}(n+1) \quad = \quad \hat{\boldsymbol{h}}(0) + \mu \sum_{i=0}^{n} \sum_{j=0}^{L-1} \boldsymbol{x}(n-i-j)\, e_{\text{norm},j}(n-i|n-i).$$

- ***Rearranging*** the individual terms for $n = 7,\ L = 3$ (as an example):

$$\begin{aligned}
\hat{\boldsymbol{h}}(7) \;=\; \hat{\boldsymbol{h}}(0) + \mu \;\Big[\; & \boldsymbol{x}(6)\, e_{\text{norm},0}(6|6) + \boldsymbol{x}(5)\, e_{\text{norm},1}(6|6) + \boldsymbol{x}(4)\, e_{\text{norm},2}(6|6) \\
+\; & \boldsymbol{x}(5)\, e_{\text{norm},0}(5|5) + \boldsymbol{x}(4)\, e_{\text{norm},1}(5|5) \\
+\; & \boldsymbol{x}(4)\, e_{\text{norm},0}(4|4) \\
& \qquad\qquad + \boldsymbol{x}(3)\, e_{\text{norm},2}(5|5) \\
& \qquad + \boldsymbol{x}(3)\, e_{\text{norm},1}(4|4) + \boldsymbol{x}(2)\, e_{\text{norm},2}(4|4) \\
+\; & \boldsymbol{x}(3)\, e_{\text{norm},0}(3|3) + \boldsymbol{x}(2)\, e_{\text{norm},1}(3|3) + \boldsymbol{x}(1)\, e_{\text{norm},2}(3|3) \\
+\; & \boldsymbol{x}(2)\, e_{\text{norm},0}(2|2) + \boldsymbol{x}(1)\, e_{\text{norm},1}(2|2) + \boldsymbol{x}(0)\, e_{\text{norm},2}(2|2) \\
+\; & \boldsymbol{x}(1)\, e_{\text{norm},0}(1|1) + \boldsymbol{x}(0)\, e_{\text{norm},1}(1|1) \\
+\; & \boldsymbol{x}(0)\, e_{\text{norm},0}(0|0) \\
& \qquad\qquad + \boldsymbol{x}(-1)\, e_{\text{norm},2}(1|1) \\
& \qquad \boldsymbol{x}(-1)\, e_{\text{norm},1}(0|0) + \boldsymbol{x}(-2)\, e_{\text{norm},2}(0|0) \;\Big].
\end{aligned}$$

Christian-Albrechts-Universität zu Kiel

From Affine Projection to Fast Affine Projection

Fast computation of the coefficient update – continued:

- Result from the ***last slide***:

$$
\begin{aligned}
\hat{\boldsymbol{h}}(7) \;=\; \hat{\boldsymbol{h}}(0) + \mu \Big[\; & \boldsymbol{x}(6)\, e_{\text{norm},0}(6|6) + \boldsymbol{x}(5)\, e_{\text{norm},1}(6|6) + \boldsymbol{x}(4)\, e_{\text{norm},2}(6|6) \\
+ \; & \boldsymbol{x}(5)\, e_{\text{norm},0}(5|5) + \boldsymbol{x}(4)\, e_{\text{norm},1}(5|5) \\
+ \; & \boldsymbol{x}(4)\, e_{\text{norm},0}(4|4) \\
& \hspace{18em} + \boldsymbol{x}(3)\, e_{\text{norm},2}(5|5) \\
& \hspace{9em} + \boldsymbol{x}(3)\, e_{\text{norm},1}(4|4) + \boldsymbol{x}(2)\, e_{\text{norm},2}(4|4) \\
+ \; & \boldsymbol{x}(3)\, e_{\text{norm},0}(3|3) + \boldsymbol{x}(2)\, e_{\text{norm},1}(3|3) + \boldsymbol{x}(1)\, e_{\text{norm},2}(3|3) \\
+ \; & \boldsymbol{x}(2)\, e_{\text{norm},0}(2|2) + \boldsymbol{x}(1)\, e_{\text{norm},1}(2|2) + \boldsymbol{x}(0)\, e_{\text{norm},2}(2|2) \\
+ \; & \boldsymbol{x}(1)\, e_{\text{norm},0}(1|1) + \boldsymbol{x}(0)\, e_{\text{norm},1}(1|1) \\
+ \; & \boldsymbol{x}(0)\, e_{\text{norm},0}(0|0) \\
& \hspace{18em} + \boldsymbol{x}(-1)\, e_{\text{norm},2}(1|1) \\
& \hspace{9em} \boldsymbol{x}(-1)\, e_{\text{norm},1}(0|0) + \boldsymbol{x}(-2)\, e_{\text{norm},2}(0|0) \;\Big].
\end{aligned}
$$

Fast computation of the coefficient update – continued:

- Result from the ***last slide***:

$$
\begin{aligned}
\hat{\boldsymbol{h}}(7) = \hat{\boldsymbol{h}}(0) + \mu \Big[\; & \boxed{\begin{array}{llll} & \boldsymbol{x}(6)\, e_{\text{norm},0}(6|6) & + \; \boldsymbol{x}(5)\, e_{\text{norm},1}(6|6) & + \; \boldsymbol{x}(4)\, e_{\text{norm},2}(6|6) \\ + & \boldsymbol{x}(5)\, e_{\text{norm},0}(5|5) & + \; \boldsymbol{x}(4)\, e_{\text{norm},1}(5|5) & \\ + & \boldsymbol{x}(4)\, e_{\text{norm},0}(4|4) & & \end{array}} \\
& \boxed{\begin{array}{llll} & & & + \; \boldsymbol{x}(3)\, e_{\text{norm},2}(5|5) \\ & & + \; \boldsymbol{x}(3)\, e_{\text{norm},1}(4|4) & + \; \boldsymbol{x}(2)\, e_{\text{norm},2}(4|4) \\ + & \boldsymbol{x}(3)\, e_{\text{norm},0}(3|3) & + \; \boldsymbol{x}(2)\, e_{\text{norm},1}(3|3) & + \; \boldsymbol{x}(1)\, e_{\text{norm},2}(3|3) \\ + & \boldsymbol{x}(2)\, e_{\text{norm},0}(2|2) & + \; \boldsymbol{x}(1)\, e_{\text{norm},1}(2|2) & + \; \boldsymbol{x}(0)\, e_{\text{norm},2}(2|2) \\ + & \boldsymbol{x}(1)\, e_{\text{norm},0}(1|1) & + \; \boldsymbol{x}(0)\, e_{\text{norm},1}(1|1) & \\ + & \boldsymbol{x}(0)\, e_{\text{norm},0}(0|0) & & \end{array}} \\
& \boxed{\begin{array}{ll} & + \; \boldsymbol{x}(-1)\, e_{\text{norm},2}(1|1) \\ \boldsymbol{x}(-1)\, e_{\text{norm},1}(0|0) & + \; \boldsymbol{x}(-2)\, e_{\text{norm},2}(0|0) \end{array}} \; \Big].
\end{aligned}
$$

We assume that all excitation signal with
A negative index are zero (the excitation "starts" at $n = 0$).

Christian-Albrechts-Universität zu Kiel

From Affine Projection to Fast Affine Projection

Fast computation of the coefficient update – continued:

- Result from the ***last slide***:

$$\begin{aligned}
\hat{\boldsymbol{h}}(7) \;=\; \hat{\boldsymbol{h}}(0) + \mu \Big[\;& \boldsymbol{x}(6)\, e_{\text{norm},0}(6|6) + \boldsymbol{x}(5)\, e_{\text{norm},1}(6|6) + \boldsymbol{x}(4)\, e_{\text{norm},2}(6|6) \\
+\; & \boldsymbol{x}(5)\, e_{\text{norm},0}(5|5) + \boldsymbol{x}(4)\, e_{\text{norm},1}(5|5) \\
+\; & \boldsymbol{x}(4)\, e_{\text{norm},0}(4|4) \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + \boldsymbol{x}(3)\, e_{\text{norm},2}(5|5) \\
& \qquad\qquad\qquad\quad + \boldsymbol{x}(3)\, e_{\text{norm},1}(4|4) + \boldsymbol{x}(2)\, e_{\text{norm},2}(4|4) \\
+\; & \boldsymbol{x}(3)\, e_{\text{norm},0}(3|3) + \boldsymbol{x}(2)\, e_{\text{norm},1}(3|3) + \boldsymbol{x}(1)\, e_{\text{norm},2}(3|3) \\
+\; & \boldsymbol{x}(2)\, e_{\text{norm},0}(2|2) + \boldsymbol{x}(1)\, e_{\text{norm},1}(2|2) + \boldsymbol{x}(0)\, e_{\text{norm},2}(2|2) \\
+\; & \boldsymbol{x}(1)\, e_{\text{norm},0}(1|1) + \boldsymbol{x}(0)\, e_{\text{norm},1}(1|1) \\
+\; & \boldsymbol{x}(0)\, e_{\text{norm},0}(0|0) \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + 0 \\
& \qquad\qquad\qquad\qquad\qquad\quad 0 \qquad + 0 \;\Big].
\end{aligned}$$

We assume that all excitation signal with
A negative index are zero (the excitation “starts” at $n = 0$).

From Affine Projection to Fast Affine Projection

Fast computation of the coefficient update – continued:

- Rewriting the result of the ***last slide***:

$$
\begin{aligned}
\hat{\boldsymbol{h}}(7) \;=\; \hat{\boldsymbol{h}}(0) + \mu \Big[\; & \boldsymbol{x}(6)\, e_{\text{norm},0}(6|6) + \boldsymbol{x}(5)\, e_{\text{norm},1}(6|6) + \boldsymbol{x}(4)\, e_{\text{norm},2}(6|6) \\
+ \; & \boldsymbol{x}(5)\, e_{\text{norm},0}(5|5) + \boldsymbol{x}(4)\, e_{\text{norm},1}(5|5) \\
+ \; & \boldsymbol{x}(4)\, e_{\text{norm},0}(4|4) \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + \boldsymbol{x}(3)\, e_{\text{norm},2}(5|5) \\
& \qquad\qquad\qquad\qquad + \boldsymbol{x}(3)\, e_{\text{norm},1}(4|4) + \boldsymbol{x}(2)\, e_{\text{norm},2}(4|4) \\
+ \; & \boldsymbol{x}(3)\, e_{\text{norm},0}(3|3) + \boldsymbol{x}(2)\, e_{\text{norm},1}(3|3) + \boldsymbol{x}(1)\, e_{\text{norm},2}(3|3) \\
+ \; & \boldsymbol{x}(2)\, e_{\text{norm},0}(2|2) + \boldsymbol{x}(1)\, e_{\text{norm},1}(2|2) + \boldsymbol{x}(0)\, e_{\text{norm},2}(2|2) \\
+ \; & \boldsymbol{x}(1)\, e_{\text{norm},0}(1|1) + \boldsymbol{x}(0)\, e_{\text{norm},1}(1|1) \\
+ \; & \boldsymbol{x}(0)\, e_{\text{norm},0}(0|0) \; \Big].
\end{aligned}
$$

$$
\begin{aligned}
\hat{\boldsymbol{h}}(n+1) \;=\; \hat{\boldsymbol{h}}(0) \; & + \; \mu \sum_{k=0}^{L-1} \boldsymbol{x}(n-k) \sum_{j=0}^{k} e_{\text{norm},j}(n-k+j|n-k+j) \\
& + \; \mu \sum_{k=L}^{n} \boldsymbol{x}(n-k) \sum_{j=0}^{L-1} e_{\text{norm},j}(n-k+j|n-k+j)
\end{aligned}
$$

From Affine Projection to Fast Affine Projection

Fast computation of the coefficient update – continued:

- Rearranging the update equation and inserting abbreviations:

$$\hat{\boldsymbol{h}}(n+1) \;=\; \hat{\boldsymbol{h}}(0) + \mu \sum_{k=0}^{L-1} \boldsymbol{x}(n-k) \sum_{j=0}^{k} e_{\mathrm{norm},j}(n-k+j|n-k+j) + \mu \sum_{k=L}^{n} \boldsymbol{x}(n-k) \sum_{j=0}^{L-1} e_{\mathrm{norm},j}(n-k+j|n-k+j)$$

… exchanging the order of the last two terms…

$$\hat{\boldsymbol{h}}(n+1) \;=\; \hat{\boldsymbol{h}}(0) + \mu \sum_{k=L}^{n} \boldsymbol{x}(n-k) \sum_{j=0}^{L-1} e_{\mathrm{norm},j}(n-k+j|n-k+j) + \mu \sum_{k=0}^{L-1} \boldsymbol{x}(n-k) \sum_{j=0}^{k} e_{\mathrm{norm},j}(n-k+j|n-k+j)$$

… inserting abbreviations for the first two and the last term …

$$\hat{\boldsymbol{h}}(n+1) \;=\; \underbrace{\hat{\boldsymbol{h}}(0) + \mu \sum_{k=L}^{n} \boldsymbol{x}(n-k) \sum_{j=0}^{L-1} e_{\mathrm{norm},j}(n-k+j|n-k+j)}_{\hat{\boldsymbol{h}}_{\mathrm{pre}(n+1)}} + \underbrace{\mu \sum_{k=0}^{L-1} \boldsymbol{x}(n-k) \sum_{j=0}^{k} e_{\mathrm{norm},j}(n-k+j|n-k+j)}_{\boldsymbol{X}(n)\,\boldsymbol{e}_{\mathrm{norm,acc}}(n)}$$

… writing the update equation compactly …

$$\hat{\boldsymbol{h}}(n+1) \;=\; \hat{\boldsymbol{h}}_{\mathrm{pre}}(n+1) + \mu\, \boldsymbol{X}(n)\, \boldsymbol{e}_{\mathrm{norm,acc}}(n)$$

From Affine Projection to Fast Affine Projection

Fast computation of the coefficient update – continued:

- Definition of the accumulated error vector that is used in the update equation:

$$\boldsymbol{e}_{\text{norm,acc}}(n) = \begin{bmatrix} e_{\text{norm},0}(n|n) \\ e_{\text{norm},1}(n|n) + e_{\text{norm},0}(n-1|n-1) \\ e_{\text{norm},2}(n|n) + e_{\text{norm},1}(n-1|n-1) + e_{\text{norm},0}(n-2|n-2) \\ \vdots \\ e_{\text{norm},L-1}(n|n) + e_{\text{norm},L-2}(n-1|n-1) + \ldots + e_{\text{norm},0}(n-L+1|n-L+1) \end{bmatrix}$$

- Exploiting that this vector can be computed/updated recursively:

$$\boldsymbol{e}_{\text{norm,acc}}(n) = \boldsymbol{e}_{\text{norm}}(n|n) + \begin{bmatrix} 0 \\ \bar{\boldsymbol{e}}_{\text{norm,acc}}(n-1) \end{bmatrix}$$

From Affine Projection to Fast Affine Projection

Fast computation of the coefficient update – continued:

- After a small amount of steps you will see, that $\hat{\boldsymbol{h}}(n)$ is not required any more. To see this we start with the definition of the scalar error signal:

$$e(n|n) \quad = \quad y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

- Here we can insert our new findings for the update equation $\hat{\boldsymbol{h}}(n+1) = \hat{\boldsymbol{h}}_{\mathrm{pre}}(n+1) + \mu\,\boldsymbol{X}(n)\,\boldsymbol{e}_{\mathrm{norm,acc}}(n)$:

$$e(n|n) \quad = \quad y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}(n)$$

… inserting the new update equation …

$$= \quad y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\left[\hat{\boldsymbol{h}}_{\mathrm{pre}}(n) + \mu\boldsymbol{X}(n-1)\,\boldsymbol{e}_{\mathrm{norm,acc}}(n-1)\right]$$

… simplification …

$$= \quad y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}_{\mathrm{pre}}(n) - \mu\,\boldsymbol{x}^{\mathrm{T}}(n-1)\,\boldsymbol{X}(n-1)\,\boldsymbol{e}_{\mathrm{norm,acc}}(n-1)$$

From Affine Projection to Fast Affine Projection

Fast computation of the coefficient update – continued:

- Result from the ***last slide***:

$$e(n|n) \;=\; y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}_{\mathrm{pre}}(n) - \mu\,\boldsymbol{x}^{\mathrm{T}}(n-1)\,\boldsymbol{X}(n-1)\,\boldsymbol{e}_{\mathrm{norm,acc}}(n-1)$$

Included in the recursive computation of the norm.

- Here an ***autocorrelation-like vector*** that can be computed recursively can be inserted:

$$\hat{\boldsymbol{s}}_{xx}(n) \;=\; \hat{\boldsymbol{s}}_{xx}(n-1) + x(n)\,\boldsymbol{x}_{\mathrm{short}}(n) - x(n-N)\,\boldsymbol{x}_{\mathrm{short}}(n-N)$$

with the vector $\boldsymbol{x}_{\mathrm{short}}(n)$ being defined as:

$$\boldsymbol{x}_{\mathrm{short}}(n) \;=\; \Big[x(n),\, x(n-1),\, \ldots,\, x(n-L+1)\Big]^{\mathrm{T}}$$

- ***Inserting*** this, we obtain:

$$e(n|n) \;=\; y(n) - \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}_{\mathrm{pre}}(n) - \mu\,\hat{\boldsymbol{s}}_{xx}^{\mathrm{T}}(n-1)\,\boldsymbol{e}_{\mathrm{norm,acc}}(n-1)$$

Now we have only the product of two (short, size L) vectors, instead of the (large, size N) vector-matrix-vector product.

From Affine Projection to Fast Affine Projection

Fast computation of the coefficient update – continued:

- Finally, we look again at the definition of $\hat{\boldsymbol{h}}_{\text{pre}}(n)$:

$$\hat{\boldsymbol{h}}_{\text{pre}}(n+1) = \hat{\boldsymbol{h}}(0) + \mu \sum_{k=L}^{n} \boldsymbol{x}(n-k) \sum_{j=0}^{L-1} e_{\text{norm},j}(n-k+j|n-k+j)$$

… excluding the fist element in the outer sum …

$$= \hat{\boldsymbol{h}}(0) + \mu\, \boldsymbol{x}(n-L) \sum_{j=0}^{L-1} e_{\text{norm},j}(n-L+j|n-L+j) + \underbrace{\mu \sum_{k=L+1}^{n} \boldsymbol{x}(n-k) \sum_{j=0}^{L-1} e_{\text{norm},j}(n-k+j|n-k+j)}_{(1)}$$

- When comparing the term (1) with the first line of the equation above, one finds:

$$\mu \sum_{k=L+1}^{n} \boldsymbol{x}(n-k) \sum_{j=0}^{L-1} e_{\text{norm},j}(n-k+j|n-k+j) = \hat{\boldsymbol{h}}_{\text{pre}}(n) - \hat{\boldsymbol{h}}(0)$$

$e_{\text{norm,acc},L-1}(n-1)$

- Inserting this result leads to:

$$\hat{\boldsymbol{h}}_{\text{pre}}(n+1) = \hat{\boldsymbol{h}}_{\text{pre}}(n) + \mu\, \boldsymbol{x}(n-L) \sum_{j=0}^{L-1} e_{\text{norm},j}(n-L+j|n-L+j)$$

From Affine Projection to Fast Affine Projection

Fast computation of the matrix inversion:

- In the original version of the FAP algorithm, also a fast version of the inversion of an autocorrelation matrix was proposed:

$$\boldsymbol{e}_{\mathrm{norm}}(n) \;=\; \left[\boldsymbol{X}^{\mathrm{T}}(n)\,\boldsymbol{X}(n) + \Delta\,\boldsymbol{I}\right]^{-1} \boldsymbol{e}(n)$$

- However, since we use affine projection algorithms usually only for projection order of 2 … 4, we omit this step over here.

- For Interested students it’s recommended to have a look into the dissertation of Steven L. Grant.

Contents:

Steven L. Grant (AKA Gay)
[Photo Maria Grant]

Final Remarks

Fast affine projection equations:

- ***Filtering***

$$
\begin{aligned}
\hat{\boldsymbol{S}}_{xx}(n) &= \hat{\boldsymbol{S}}_{xx}(n-1) + \boldsymbol{x}_{\text{short}}^{\mathrm{T}}(n)\,\boldsymbol{x}_{\text{short}}(n) - \boldsymbol{x}_{\text{short}}^{\mathrm{T}}(n-N)\,\boldsymbol{x}_{\text{short}}(n-N) \\
\hat{d}(n) &= \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}_{\text{pre}}(n) + \mu\,\hat{\boldsymbol{s}}_{xx}^{\mathrm{T}}(n-1)\,\boldsymbol{e}_{\text{norm,acc}}(n-1)
\end{aligned}
$$

- ***Error signal***

$$
\begin{aligned}
e(n|n) &= y(n) - \hat{d}(n) \\
\boldsymbol{e}(n|n) &= \begin{bmatrix} e(n|n) \\ (1-\mu)\,\bar{\boldsymbol{e}}(n-1|n-1) \end{bmatrix}.
\end{aligned}
$$

- ***Normalization***

$$
\begin{aligned}
\boldsymbol{e}_{\text{norm}}(n|n) &= \left[\hat{\boldsymbol{S}}_{xx}(n) + \Delta\,\boldsymbol{I}\right]^{-1} \boldsymbol{e}(n|n) \\
\boldsymbol{e}_{\text{norm,acc}}(n) &= \boldsymbol{e}_{\text{norm}}(n|n) + \begin{bmatrix} 0 \\ \bar{\boldsymbol{e}}_{\text{norm,acc}}(n-1) \end{bmatrix}
\end{aligned}
$$

- ***Filter update***

$$
\hat{\boldsymbol{h}}_{\text{pre}}(n+1) = \hat{\boldsymbol{h}}_{\text{pre}}(n) + \mu\,\boldsymbol{x}(n-L)\,e_{\text{norm,acc},L-1}(n-1)
$$

Final Remarks

Fast affine projection equations:

- ***Filtering***

$$\hat{\boldsymbol{S}}_{xx}(n) = \hat{\boldsymbol{S}}_{xx}(n-1) + \boldsymbol{x}_{\text{short}}^{\mathrm{T}}(n)\,\boldsymbol{x}_{\text{short}}(n) - \boldsymbol{x}_{\text{short}}^{\mathrm{T}}(n-N)\,\boldsymbol{x}_{\text{short}}(n-N)$$

$$\hat{d}(n) = \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}_{\text{pre}}(n) + \mu\,\hat{\boldsymbol{s}}_{xx}^{\mathrm{T}}(n-1)\,\boldsymbol{e}_{\text{norm,acc}}(n-1)$$

- ***Error signal***

$$e(n|n) = y(n) - \hat{d}(n)$$

$$\boldsymbol{e}(n|n) = \begin{bmatrix} e(n|n) \\ (1-\mu)\,\bar{\boldsymbol{e}}(n-1|n-1) \end{bmatrix}.$$

- ***Normalization***

$$\boldsymbol{e}_{\text{norm}}(n|n) = \left[\hat{\boldsymbol{S}}_{xx}(n) + \Delta\,\boldsymbol{I}\right]^{-1}\boldsymbol{e}(n|n)$$

$$\boldsymbol{e}_{\text{norm,acc}}(n) = \boldsymbol{e}_{\text{norm}}(n|n) + \begin{bmatrix} 0 \\ \bar{\boldsymbol{e}}_{\text{norm,acc}}(n-1) \end{bmatrix}$$

- ***Filter update***

$$\hat{\boldsymbol{h}}_{\text{pre}}(n+1) = \hat{\boldsymbol{h}}_{\text{pre}}(n) + \mu\,\boldsymbol{x}(n-L)\,e_{\text{norm,acc},L-1}(n-1)$$

Complexity AP (approx.):

$2NL + 3L^2 + L^3 \approx 2NL$ add.,
$2NL + 3L^2 + L^3 \approx 2NL$ mul.

Example:

$f_{\mathrm{s}} = 48\,\mathrm{kHz}$

$N = 12000$

$L = 4$

4.6 billion additions per second,
4.6 billion multiplic. per second

Final Remarks

Fast affine projection equations:

- ***Filtering***

$$\hat{\boldsymbol{S}}_{xx}(n) = \hat{\boldsymbol{S}}_{xx}(n-1) + \boldsymbol{x}_{\text{short}}^{\text{T}}(n)\,\boldsymbol{x}_{\text{short}}(n) - \boldsymbol{x}_{\text{short}}^{\text{T}}(n-N)\,\boldsymbol{x}_{\text{short}}(n-N)$$

$$\hat{d}(n) = \boldsymbol{x}^{\text{T}}(n)\,\hat{\boldsymbol{h}}_{\text{pre}}(n) + \mu\,\hat{\boldsymbol{s}}_{xx}^{\text{T}}(n-1)\,\boldsymbol{e}_{\text{norm,acc}}(n-1)$$

- ***Error signal***

$$e(n|n) = y(n) - \hat{d}(n)$$

$$\boldsymbol{e}(n|n) = \begin{bmatrix} e(n|n) \\ (1-\mu)\,\bar{\boldsymbol{e}}(n-1|n-1) \end{bmatrix}.$$

- ***Normalization***

$$\boldsymbol{e}_{\text{norm}}(n|n) = \left[\hat{\boldsymbol{S}}_{xx}(n) + \Delta\,\boldsymbol{I}\right]^{-1}\boldsymbol{e}(n|n)$$

$$\boldsymbol{e}_{\text{norm,acc}}(n) = \boldsymbol{e}_{\text{norm}}(n|n) + \begin{bmatrix} 0 \\ \bar{\boldsymbol{e}}_{\text{norm,acc}}(n-1) \end{bmatrix}$$

- ***Filter update***

$$\hat{\boldsymbol{h}}_{\text{pre}}(n+1) = \hat{\boldsymbol{h}}_{\text{pre}}(n) + \mu\,\boldsymbol{x}(n-L)\,e_{\text{norm,acc},L-1}(n-1)$$

Complexity AP (approx.):

$$2NL + 3L^2 + L^3 \approx 2NL \text{ add.,}$$
$$2NL + 3L^2 + L^3 \approx 2NL \text{ mul.}$$

Computational complexity

2 x L^2 additions,	2 x L^2 multiplications
N+L additions,	N+L multiplications
1 addition,	0 multiplications
L additions,	L multiplications
L^2 additions,	L^2 multiplications
1 inversion (L^3 mult., L^3 add.)	
L additions,	0 multiplications
N additions,	N multiplications

Final Remarks

Fast affine projection equations:

- ***Filtering***

$$\hat{\boldsymbol{S}}_{xx}(n) = \hat{\boldsymbol{S}}_{xx}(n-1) + \boldsymbol{x}_{\text{short}}^{\mathrm{T}}(n)\,\boldsymbol{x}_{\text{short}}(n) - \boldsymbol{x}_{\text{short}}^{\mathrm{T}}(n-N)\,\boldsymbol{x}_{\text{short}}(n-N)$$

$$\hat{d}(n) = \boldsymbol{x}^{\mathrm{T}}(n)\,\hat{\boldsymbol{h}}_{\text{pre}}(n) + \mu\,\hat{\boldsymbol{s}}_{xx}^{\mathrm{T}}(n-1)\,\boldsymbol{e}_{\text{norm,acc}}(n-1)$$

- ***Error signal***

$$e(n|n) = y(n) - \hat{d}(n)$$

$$\boldsymbol{e}(n|n) = \begin{bmatrix} e(n|n) \\ (1-\mu)\,\bar{\boldsymbol{e}}(n-1|n-1) \end{bmatrix}.$$

- ***Normalization***

$$\boldsymbol{e}_{\text{norm}}(n|n) = \left[\hat{\boldsymbol{S}}_{xx}(n) + \Delta\,\boldsymbol{I}\right]^{-1} \boldsymbol{e}(n|n)$$

$$\boldsymbol{e}_{\text{norm,acc}}(n) = \boldsymbol{e}_{\text{norm}}(n|n) + \begin{bmatrix} 0 \\ \bar{\boldsymbol{e}}_{\text{norm,acc}}(n-1) \end{bmatrix}$$

- ***Filter update***

$$\hat{\boldsymbol{h}}_{\text{pre}}(n+1) = \hat{\boldsymbol{h}}_{\text{pre}}(n) + \mu\,\boldsymbol{x}(n-L)\,e_{\text{norm,acc},L-1}(n-1)$$

Complexity AP (approx.):

$$2NL + 3L^2 + L^3 \approx 2NL \text{ add.,}$$
$$2NL + 3L^2 + L^3 \approx 2NL \text{ mul.}$$

Complexity FAP (approx.):

$$2N + 3L + 3L^2 + L^3 \approx 2N \text{ add.,}$$
$$2N + 2L + 3L^2 + L^3 \approx 2N \text{ mul.}$$

Example:

$f_{\mathrm{s}} = 48\,\mathrm{kHz} \quad N = 12000 \quad L = 4$

AP: 4.6 billion ops. per second

FAP: 1.2 billion ops. per second

Summary and Outlook

***This week** and last week:*

- Introductory Remarks
- Recursive Least Squares (RLS) Algorithm
- Least Mean Square Algorithm (LMS Algorithm) – Part 1
- Least Mean Square Algorithm (LMS Algorithm) – Part 2
- Affine Projection Algorithm (AP Algorithm)
- Fast Affine Projection Algorithm (FAP Algorithm)

Next part:

- Control of Adaptive Filters