Adaptive Filters – Algorithms (Part 1)

Gerhard Schmidt

Christian-Albrechts-Universität zu Kiel
Faculty of Engineering
Electrical Engineering and Information Technology
Digital Signal Processing and System Theory
Contents of the Lecture

Today:

Exercises:
- Topics for the Talks

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)
Adaptive Filters – Talks

Possible Topics

*Suggestions:*

- Hearing aids
- GSM (source) coding
- Localization and tracking
- Active noise control (anti-noise)
- Noise suppression
- Bandwidth extension
- Audio upmix of stereo signals
- Adaptive beamforming
- MPEG audio coding
- Non-linear echo cancellation
- Adaptation of neural networks
- Feedback suppression
- ...

*Your own topic suggestions are welcome...*
Contents

Exercises:
- Topics for the Talks

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)
Motivation

**Why adaptive filters?**

- Signal properties are not known in advance or are time variant.
- System properties are not known in advance or time variant.

**Examples:**

- Speech signals
- Mobile telephone channels
Literature

**Books:**

Two Hook-Ups of Adaptive Filters

Adaptive filter for channel equalization:

Adaptive filter for system identification:
Adaptive filter for cancellation of hybrid echoes:
**Adaptive filter for noise reduction with reference signal:**

**Signal model**

- Signal source
- Noise source
- Transmission path 1
- Transmission path 2
- Noisy signal
- Reference signal
- Adaptive filter
Introductory Remarks

Application Examples – Part 3

*Antenna array:*

\[
\begin{align*}
\text{Adaptive filter 1} & \quad \text{Adaptive filter 2} & \quad \text{Adaptive filter } N \\
\downarrow & \quad \downarrow & \quad \downarrow \\
(+) & \quad (+) & \quad (+) \\
\end{align*}
\]
**Adaptive equalization without reference signal**

**Assumptions:**

\[
\begin{align*}
    d_{est}(n) & \approx d(n) \\
    e_{est}(n) & \approx e(n)
\end{align*}
\]
Introductory Remarks

Generic Setup

- **Adaptive algorithm**
- **Adaptive filter**
- **Desired output signal**
- **Input signal** $x(n)$
- **Output signal** $e(n)$
- **Error signal** $d(n)$
Introductory Remarks

Structure of an Adaptive FIR Filter

\[ x(n) \xrightarrow{\text{z}^{-1}} \hat{h}_0(n) \xrightarrow{+} \hat{h}_1(n) \xrightarrow{+} \hat{h}_2(n) \xrightarrow{+} \hat{h}_3(n) \xrightarrow{+} \hat{h}_4(n) \xrightarrow{+} \ldots \xrightarrow{\text{z}^{-1}} \hat{h}_{N-1}(n) \xrightarrow{+} \hat{d}(n) \]

\[ x(n) = [x(n), x(n-1), x(n-2), \ldots, x(n-N+1)]^T \]

\[ \hat{h}(n) = [\hat{h}_0(n), \hat{h}_1(n), \hat{h}_2(n), \ldots, \hat{h}_{N-1}(n)]^T \]

\[ \hat{d}(n) = \hat{h}^H(n) x(n) = x^T(n) \hat{h}^*(n) \]
Introductory Remarks

Error Measures – Part 1

Mean square (signal) error:

\[ E\left\{ |e(n)|^2 \right\} = E\left\{ |d(n) - \hat{d}(n)|^2 \right\} \]

System distance:

\[ \| h_\Delta (n) \|^2 = \| h - \hat{h}(n) \|^2 \]
\[ = \left[ h(n) - \hat{h}(n) \right]^H \left[ h(n) - \hat{h}(n) \right] \]
Mean Square Error and System Distance

**Relation of the normalized mean square (signal) error power and the system distance:**

\[
\frac{\mathbb{E}\{|e(n)|^2\}}{\mathbb{E}\{|x(n)|^2\}} = \frac{[h(n) - \hat{h}(n)]^H \mathbb{E}\{x(n)x^H(n)\} [h(n) - \hat{h}(n)]}{\mathbb{E}\{|x(n)|^2\}} = h^H_\Delta(n) \mathbb{E}\{x(n)x^H(n)\} h_\Delta(n) \frac{1}{\mathbb{E}\{|x(n)|^2\}}
\]

Let \(x(n)\) be white noise:

\[
\mathbb{E}\{x(n)x^H(n)\} = \text{unit matrix} \times \mathbb{E}\{|x(n)|^2\}
\]

\[
\frac{\mathbb{E}\{|e(n)|^2\}}{\mathbb{E}\{|x(n)|^2\}} = h^H_\Delta(n) h_\Delta(n)
\]
**Introductory Remarks**

**Adaptation**

**Basic principle:**

New = old + correction

**Properties:**

- „Correction“ depends on the input signal $x(n)$ and the error signal $e(n)$.
- Procedures differ by the functions $g(x(n))$ and $f(e(n))$:

$$
\hat{h}(n+1) = \hat{h}(n) + \mu x(n) g(x(n)) f(e(n)).
$$

**Step size**
Error Measures

Three error measures control the adaptation:

- Coefficient error
  \[ h_\Delta(n) = h(n) - \hat{h}(n) \]

- A priori error
  \[ e(n|n) = h^H_\Delta(n) x(n) + b(n) \]

- A posteriori error:
  \[ e(n|n+1) = h^H_\Delta(n+1) x(n) + b(n) \]
Adaptive Filters - Algorithms

Contents

Exercises:
- Topics for the Talks

Adaptive Algorithms:
- Introductory Remarks
- Recursive Least Squares (RLS) Algorithm
- Least Mean Squares Algorithm (LMS Algorithm) – Part 1
- Least Mean Squares Algorithm (LMS Algorithm) – Part 2
- Affine projection Algorithm (AP Algorithm)
Recursive Least Squares (RLS) Algorithm

Algorithmic Properties

Attributes of the RLS algorithm:

- No a priori knowledge of signal statistics is required.
- Optimization criterion is the (weighted) sum of squared errors.
Recursive Least Squares (RLS) Algorithm

**Error Criterion**

\[ E(n) = \sum_{l=0}^{n} \lambda^{n-l} e(l|n) e^*(l|n) \quad \text{with} \quad 0 < \lambda \leq 1 \]

**Signal**

\[ e(l|n) = y(l) - \hat{h}^H(n) x(l) \]

**Filter**

\[ y(n) = d(n) \]

**Forgetting factor**

**Alternative:**

\[ E(n) = \sum_{l=n-L+1}^{n} e(l|n) e^*(l|n) \]

**Filter at time \( n \)**

\[ e(l|n) = y(l) - \hat{h}^H(n) x(l) = y(l) - x^T(l) \hat{h}^*(n) \]

**Signal at time \( l \)**

\[ E(n) = \sum_{l=0}^{n} \lambda^{n-l} [y(l) - \hat{h}^H(n) x(l)] [y^*(l) - x^H(l) \hat{h}(n)] \]
Recursive Least Squares (RLS) Algorithm

Derivation – Part 1

Cost function:

\[ E(n) = \sum_{l=0}^{n} \lambda^{n-l} \left[ y(l) - \hat{h}^H(n) x(l) \right] \left[ y^*(l) - x^H(l) \hat{h}(n) \right] \]

Differentiate with respect to the complex filter coefficients and setting the result to zero:

\[ \sum_{l=0}^{n} \lambda^{n-l} x(l) x^H(l) \hat{h}(n) = \sum_{l=0}^{n} \lambda^{n-l} x(l) y^*(l) \]

Definitions:

\[ \hat{R}_{xx}(n) = \sum_{l=0}^{n} \lambda^{n-l} x(l) x^H(l) \quad \text{... Estimate for the auto correlation matrix} \]

\[ \hat{r}_{xy}(n) = \sum_{l=0}^{n} \lambda^{n-l} x(l) y^*(l) \quad \text{... Estimate for the cross correlation vector} \]
Recursive Least Squares (RLS) Algorithm

Derivation – Part 2

From Simon Haykin, “Adaptive Filter Theory“, Prentice Hall, 2002:

EXAMPLE 3
Consider the real-valued cost function (see Chapter 2)

\[ J(w) = \sigma_d^2 - w^H p - p^H w + w^H R w. \]

Using the results of Examples 1 and 2, we find that the conjugate derivative of \( J \) with respect to the tap-weight vector \( w \) is

\[ \frac{\partial J}{\partial w^*} = -p + R w. \]  \hspace{1cm} (B.11)

Let \( w_o \) be the optimum value of the tap-weight vector \( w \) for which the cost function \( J \) is minimal, or, equivalently, the derivative \( (\partial J/\partial w^*) = 0 \). Then, from Eq. (B.11), we infer that

\[ R w_o = p. \]  \hspace{1cm} (B.12)

This is the matrix form of the Wiener–Hopf equations for a transversal filter operating in a stationary environment, characterized by the correlation matrix \( R \) and cross-correlation vector \( p \).

or: The Matrix Cookbook  [http://matrixcookbook.com]
Recursive Least Squares (RLS) Algorithm

Derivation – Part 3

\[
\sum_{l=0}^{n} \lambda^{n-l} x(l) x^H(l) \hat{h}(n) = \sum_{l=0}^{n} \lambda^{n-l} x(l) y^*(l)
\]

\[
\hat{R}_{xx}(n) = \sum_{l=0}^{n} \lambda^{n-l} x(l) x^H(l)
\]

\[
\hat{r}_{xy}(n) = \sum_{l=0}^{n} \lambda^{n-l} x(l) y^*(l)
\]

Inserting the results leads to:

\[
\hat{R}_{xx}(n) \hat{h}(n) = \hat{r}_{xy}(n)
\]

„Wiener solution“

\[... \text{assuming that the auto correlation matrix is invertible}\]

\[
\hat{h}(n) = \hat{R}_{xx}^{-1}(n) \hat{r}_{xy}(n)
\]
Recursive Least Squares (RLS) Algorithm

Recursion – Part 1

**Recursion of the auto correlation matrix over time:**

\[
\hat{R}_{xx}(n+1) = \sum_{l=0}^{n+1} \lambda^{n+1-l} x(l) x^H(l)
\]

\[
= \lambda \sum_{l=0}^{n} \lambda^{n-l} x(l) x^H(l) + x(n+1) x^H(n+1)
\]

\[
= \lambda \hat{R}_{xx}(n) + x(n+1) x^H(n+1)
\]

**Recursion of the cross correlation vector over time:**

\[
\hat{r}_{xy}(n+1) = \sum_{l=0}^{n+1} \lambda^{n+1-l} x(l) y^*(l)
\]

\[
= \lambda \sum_{l=0}^{n} \lambda^{n-l} x(l) y^*(l) + x(n+1) y^*(n+1)
\]

\[
= \lambda \hat{r}_{xy}(n) + x(n+1) y^*(n+1)
\]
Recursive Least Squares (RLS) Algorithm

Recursion – Part 2

**Recursion for the auto correlation matrix:**

\[
\hat{R}_{xx}(n+1) = \lambda \hat{R}_{xx}(n) + x(n+1) x^H(n+1)
\]

**Matrix Inversion Lemma:**

\[
[ A + u v^H ]^{-1} = A^{-1} - \frac{A^{-1} u v^H A^{-1}}{1 + v^H A^{-1} u}
\]

**Inserting the Lemma in the recursion:**

\[
\hat{R}_{xx}^{-1}(n+1) = \lambda^{-1} \hat{R}_{xx}^{-1}(n) - \frac{\lambda^{-1} \hat{R}_{xx}^{-1}(n) x(n+1) x^H(n+1) \hat{R}_{xx}^{-1}(n) \lambda^{-1}}{1 + \lambda^{-1} x^H(n+1) \hat{R}_{xx}^{-1}(n) x(n+1)}
\]
Recursive Least Squares (RLS) Algorithm

Recursion – Part 3

Recursion for the auto correlation matrix:

\[
\hat{R}_{xx}^{-1}(n + 1) = \lambda^{-1} \hat{R}_{xx}^{-1}(n) - \frac{\lambda^{-1} \hat{R}_{xx}^{-1}(n) x(n + 1) x^H(n + 1) \hat{R}_{xx}^{-1}(n) \lambda^{-1}}{1 + \lambda^{-1} x^H(n + 1) \hat{R}_{xx}^{-1}(n) x(n + 1)}
\]

Definition of a gain vector:

\[
\gamma(n + 1) = \frac{\lambda^{-1} \hat{R}_{xx}^{-1}(n) x(n + 1)}{1 + \lambda^{-1} x^H(n + 1) \hat{R}_{xx}^{-1}(n) x(n + 1)}
\]

Inserting this definition leads to:

\[
\hat{R}_{xx}(n + 1) = \lambda^{-1} \hat{R}_{xx}(n) - \gamma(n + 1) x^H(n + 1) \hat{R}_{xx}(n) \lambda^{-1}
\]
Recursive Least Squares (RLS) Algorithm

Recursion – Part 4

Definition of a gain factor:

\[ \gamma(n+1) = \frac{\lambda^{-1} \hat{R}_{xx}^{-1}(n) x(n+1)}{1 + \lambda^{-1} x^H(n+1) \hat{R}_{xx}^{-1}(n) x(n+1)} \]

Multiplication by the denominator on the right hand side leads to:

\[ \gamma(n+1) \left[ 1 + \lambda^{-1} x^H(n+1) \hat{R}_{xx}^{-1}(n) x(n+1) \right] = \lambda^{-1} \hat{R}_{xx}^{-1}(n) x(n+1) \]

Rewriting leads to:

\[ \gamma(n+1) = \lambda^{-1} \hat{R}_{xx}^{-1}(n) x(n+1) - \lambda^{-1} \gamma(n+1) x^H(n+1) \hat{R}_{xx}^{-1}(n) x(n+1) \]

\[ \gamma(n+1) = \left[ \lambda^{-1} \hat{R}_{xx}^{-1}(n) - \lambda^{-1} \gamma(n+1) x^H(n+1) \hat{R}_{xx}^{-1}(n) \right] x(n+1) \]

\[ \hat{R}_{xx}^{-1}(n+1) \]

\[ \gamma(n+1) = \hat{R}_{xx}^{-1}(n+1) x(n+1) \]
Recursion of the filter coefficient vector:

\[ \hat{h}(n) = \hat{R}_{xx}^{-1}(n) \hat{r}_{xy}(n) \]

Step from \( n \) to \( n+1 \):

\[ \hat{h}(n + 1) = \hat{R}_{xx}^{-1}(n + 1) \hat{r}_{xy}(n + 1) \]

Reducing the right hand side:

\[ \hat{r}_{xy}(n + 1) = \lambda \hat{r}_{xy}(n) + x(n + 1)y^*(n + 1) \]

Inserting the recursion of the cross correlation vector leads to:

\[ \hat{h}(n + 1) = \lambda \hat{R}_{xx}^{-1}(n + 1) \hat{r}_{xy}(n) + \hat{R}_{xx}^{-1}(n + 1) x(n + 1)y^*(n + 1) \]
**Recursive Least Squares (RLS) Algorithm**

**Recursion – Part 6**

*What we have so far:*

\[
\hat{h}(n+1) = \lambda \hat{R}_{xx}^{-1}(n+1) \hat{r}_{xy}(n) + \hat{R}_{xx}^{-1}(n+1) x(n+1) y^*(n+1).
\]

*If we insert the recursive computation of the inverse auto correlation matrix*

\[
\hat{R}_{xx}^{-1}(n+1) = \lambda^{-1} \hat{R}_{xx}^{-1}(n) - \gamma(n+1) x^H(n+1) \hat{R}_{xx}(n) \lambda^{-1},
\]

*we obtain:*

\[
\begin{align*}
\hat{h}(n+1) &= \hat{R}_{xx}^{-1}(n) \hat{r}_{xy}(n) - \gamma(n+1) x^H(n+1) \hat{R}_{xx}^{-1}(n) \hat{r}_{xy}(n) \\
&\quad + \hat{R}_{xx}(n+1) x(n+1) y^*(n+1) \\
&= \hat{h}(n) - \gamma(n+1) x^H(n+1) \hat{h}(n) \\
&\quad + \hat{R}_{xx}^{-1}(n+1) x(n+1) y^*(n+1).
\end{align*}
\]
Recursive Least Squares (RLS) Algorithm

Recursion – Part 7

**What we have so far:**

\[
\hat{h}(n+1) = \hat{h}(n) - \gamma(n+1) x^H(n+1) \hat{h}(n) \\
+ \hat{R}^{-1}_{xx}(n+1) x(n+1) y^*(n+1).
\]

**Inserting** \(\gamma(n+1)\) **according to**

\[
\gamma(n+1) = \hat{R}^{-1}_{xx}(n+1) x(n+1),
\]

**results in**

\[
\hat{h}(n+1) = \hat{h}(n) + \gamma(n+1) \left[ y^*(n+1) - x^H(n+1) \hat{h}(n) \right].
\]

- **Gain factor**
- **Error: old filter with new data**

\[
e(n+1|n) = y(n+1) - \hat{d}(n+1|n) \\
= y(n+1) - x^T(n+1) \hat{h}^*(n)
\]
Adaptation Rule – Part 1

**Inserting previous results:**

\[
\gamma(n+1) = \hat{R}_{xx}^{-1}(n+1) x(n+1).
\]

\[
\hat{h}(n+1) = \hat{h}(n) + \gamma(n+1) \left[ y^*(n+1) - x^H(n+1) \hat{h}(n) \right].
\]

\[
y^*(n+1) - x^H(n+1) \hat{h}(n) = e^*(n+1|n)
\]

**Adaptation rule for the filter coefficients according to the RLS algorithm:**

\[
\hat{h}(n) = \hat{h}(n-1) + \hat{R}_{xx}^{-1}(n) x(n) e^*(n|n-1) \Delta \hat{h}(n-1)
\]
Summary

Computing a preliminary gain vector (complexity prop. $N^2$):

$$\gamma(n+1) = \frac{\lambda^{-1} \hat{R}_{xx}^{-1}(n) x(n+1)}{1 + \lambda^{-1} x^H(n+1) \hat{R}_{xx}^{-1}(n) x(n+1)}$$

Update of the inverse auto correlation matrix (complexity prop. $N^2$):

$$\hat{R}_{xx}^{-1}(n+1) = \lambda^{-1} \hat{R}_{xx}^{-1}(n) - \gamma(n+1) x^H(n+1) \hat{R}_{xx}^{-1}(n) \lambda^{-1}$$

Computing the error signal (complexity prop. $N$):

$$e(n+1|n) = y(n+1) - \hat{h}^H(n) x(n+1)$$

Update of the filter vector (complexity prop. $N$):

$$\hat{h}(n+1) = \hat{h}(n) + \mu \gamma(n+1) e^*(n+1|n)$$

Step size (0 ... 1), will be treated later ...
Adaptive Filters – Algorithms

Contents

Exercises:

- Topics for the Talks

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)
**Optimization criterion:**
- Minimizing the mean square error

**Assumptions:**
- Real, stationary random processes

**Structure:**

![Diagram of adaptive filter and unknown system](image)
Least Mean Square (LMS) Algorithm

Basics – Part 2

Output signal of the adaptive filter:

\[ \hat{d}(n) = \sum_{i=0}^{N-1} \hat{h}_i(n) x(n - i) \]

\[ = \hat{h}^T(n) x(n) = x^T(n) \hat{h}(n) \]

Error signal:

\[ e(n) = d(n) - \hat{d}(n) \]

\[ = d(n) - \hat{h}^T(n) x(n) = d(n) - x^T(n) \hat{h}(n) \]

Mean square error:

\[ \overline{e^2(n)} = E\left\{ [d(n) - \hat{d}(n)]^2 \right\} = E\left\{ [d(n) - \hat{h}^T(n) x(n)]^2 \right\} \]
Least Mean Square (LMS) Algorithm

Basics – Part 3

Mean square error:

$$\overline{e^2(n)} = E\left\{ [d(n) - \hat{d}(n)]^2 \right\} = E\left\{ [d(n) - \hat{h}^T(n) x(n)]^2 \right\}$$

The filter coefficients are adjusted optimally in case of orthogonality:

$$E\left\{ x(n) e_{\text{min}}(n) \right\} = E\left\{ x(n) [d(n) - \hat{h}_{\text{opt}}^T x(n)] \right\} = 0$$

Abbreviations:

$$R_{xx} = E\left\{ x(n) x^T(n) \right\}$$ (auto correlation matrix)

$$r_{xd}(0) = r_{xd} = E\left\{ x(n) d(n) \right\}$$ (cross correlation vector)

Solution (according to Wiener):

$$R_{xx} \hat{h}_{\text{opt}} = r_{xd}$$

$$\hat{h}_{\text{opt}} = R_{xx}^{-1} r_{xd}$$ (assuming that the inverse exists)
Least Mean Square (LMS) Algorithm

Basics – Part 4

**Mean square error:**

\[
e^2(n) = E\left\{ [d(n) - \hat{h}^T(n) x(n)]^2 \right\} = r_{dd}(0) - 2 \hat{h}^T(n) r_{xd} + \hat{h}^T(n) R_{xx} \hat{h}(n)
\]

**Optimal filter vector:**

\[
\hat{h}_{opt} = R_{xx}^{-1} r_{xd}
\]

**Minimum mean square error:**

\[
e_{min}^2(n) = r_{dd}(0) - r_{xd}^T \hat{h}_{opt} = r_{dd}(0) - r_{xd}^T R_{xx}^{-1} r_{xd}
\]
Least Mean Square (LMS) Algorithm

Basics – Part 5

Mean square error:

\[
e^2(n) = r_{dd}(0) - 2 \hat{h}^T(n) r_{xd} + \hat{h}^T(n) R_{xx} \hat{h}(n)
\]

Minimum mean square error:

\[
e^2_{\text{min}}(n) = r_{dd}(0) - \hat{h}_{\text{opt}}^T r_{xd} = r_{dd}(0) - r_{xd}^T R_{xx}^{-1} r_{xd}
\]

Mean square error:

\[
e^2(n) = r_{dd}(0) - r_{xd}^T R_{xx}^{-1} r_{xd} + r_{xd}^T R_{xx}^{-1} r_{xd} - 2 \hat{h}^T(n) r_{xd} + \hat{h}^T(n) R_{xx} \hat{h}(n)
\]

\[
e^2(n) = e^2_{\text{min}}(n) + \left[ \hat{h}(n) - \hat{h}_{\text{opt}} \right]^T R_{xx} \left[ \hat{h}(n) - \hat{h}_{\text{opt}} \right]
\]

Quadratic form \(\Rightarrow\) unique minimum
Least Mean Square (LMS) Algorithm

Derivation – Part 1

**Derivation with respect to the coefficients of the adaptive filter:**

\[
\nabla \hat{h}(n) \overline{e^2(n)} = 2 \mathbb{E} \left\{ e(n) \nabla \hat{h}(n) e(n) \right\} \\
= 2 \mathbb{E} \left\{ e(n) \nabla \hat{h}(n) [d(n) - x^T(n) \hat{h}(n)] \right\} \\
= -2 \mathbb{E} \left\{ e(n) x(n) \right\} \quad \text{... needed later on ...} \\
= -2 \mathbb{E} \left\{ [d(n) - x^T(n) \hat{h}(n)] x(n) \right\} \\
= -2 r_{xd} + 2 R_{xx} \hat{h}(n). 
\]

Inserting \( R_{xx} \hat{h}_{opt} = r_{xd} \), results in:

\[
\frac{1}{2} \nabla \hat{h}(n) \overline{e^2(n)} = R_{xx} \left[ \hat{h}(n) - \hat{h}_{opt} \right]. 
\]
Least Mean Square (LMS) Algorithm

Derivation – Part 2

What we have so far:

\[ \frac{1}{2} \nabla \hat{h}(n) \overline{e^2(n)} = R_{xx} \left[ \hat{h}(n) - \hat{h}_{\text{opt}} \right]. \]

Resolving it to \( \hat{h}_{\text{opt}} \) leads to:

\[ \hat{h}_{\text{opt}} = \hat{h}(n) - \frac{1}{2} R_{xx}^{-1} \nabla \hat{h}(n) \overline{e^2(n)}. \]

With the introduction of a step size \( \mu \), the following adaptation rule can be formulated:

\[ \hat{h}(n+1) = \hat{h}(n) - \mu R_{xx}^{-1} \nabla \hat{h}(n) \overline{e^2(n)}. \]

Method according to Newton
Least Mean Square (LMS) Algorithm

Derivation – Part 3

Method according to Newton:
\[ \hat{h}(n + 1) = \hat{h}(n) - \mu R_{x,x}^{-1} \nabla \hat{h}(n) \overline{e^2(n)} . \]

Method of steepest descent:
\[ \hat{h}(n + 1) = \hat{h}(n) - \mu \nabla \hat{h}(n) \overline{e^2(n)} \]
\[ = \hat{h}(n) + \mu \mathbb{E}\{e(n)x(n)\} . \]

For practical approaches the expectation value is replaced by its instantaneous value. This leads to the so-called least mean square algorithm:
\[ \hat{h}(n + 1) = \hat{h}(n) + \mu e(n)x(n) . \]
Summary and Outlook

This week:

- Topics for the Talks
- Introductory Remarks
- Recursive Least Squares (RLS) Algorithm
- Least Mean Square Algorithm (LMS Algorithm) – Part 1

Next week:

- Least Mean Square Algorithm (LMS Algorithm) – Part 2
- Affine Projection Algorithm (AP Algorithm)