Lecture

Adaptive Filters

Lecture 5: Adaptation procedures (I): The RLS algorithm

Adaptive algorithms: today and in the next lectures

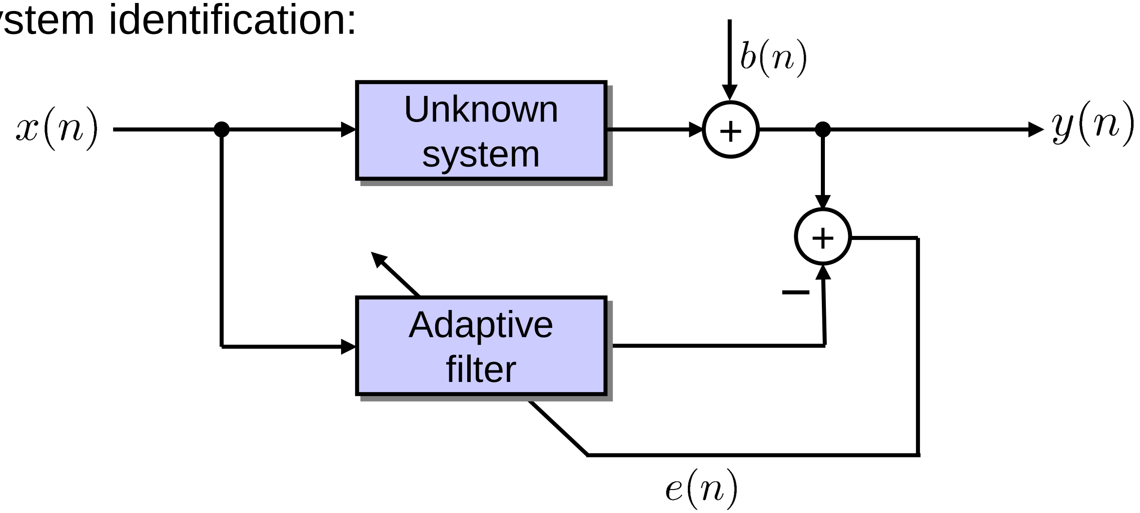
- Introductory remarks
- Recursive Least Squares Algorithm (RLS Algorithm)
- Least Mean Squares Algorithm (LMS Algorithm)
- The filtered-x LMS algorithm
- Affine Projection Algorithm (AP Algorithm)
- Kalman Filter
- Particle Filter

Content

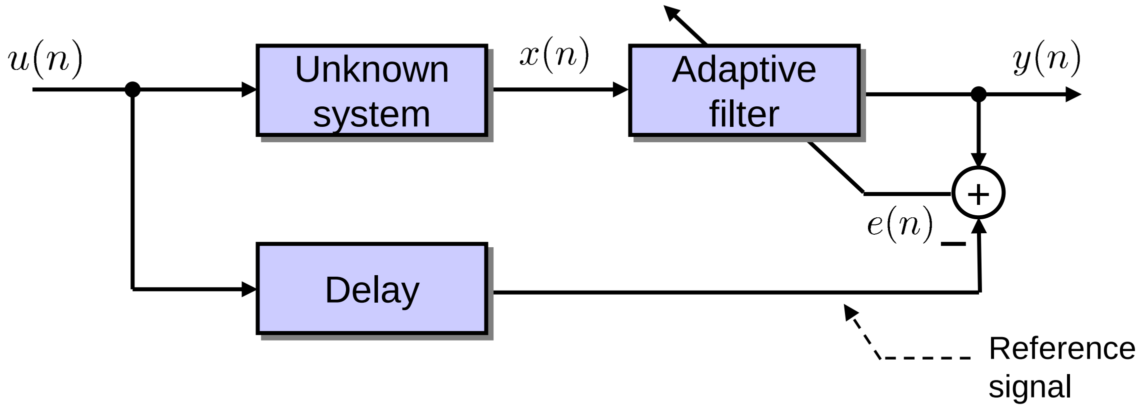
- Repetition of the Wiener filter solution with its main property:
  The identification of a static optimum filter.
- Derivation of the Least Squares (LS) solution based on the minimization of a deterministic error criterion.
- Generalized concept of adaptation rules and error criteria.
- The derivation of the Recursive Least Squares (RLS) adaptation procedure.

Repetition of different applications of adaptive filters

- System identification:

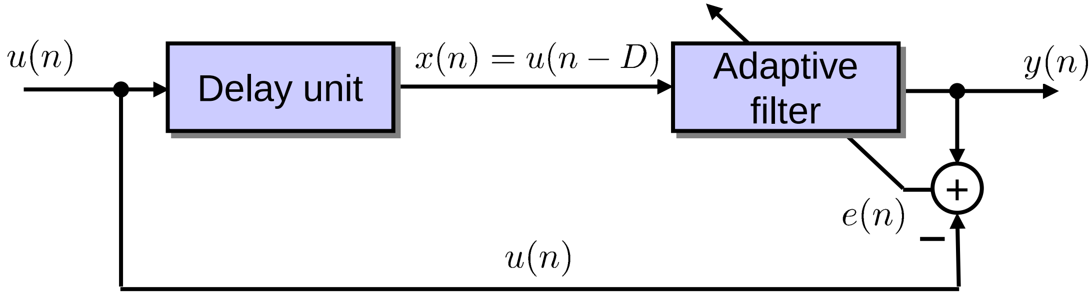


- Inverse modeling:

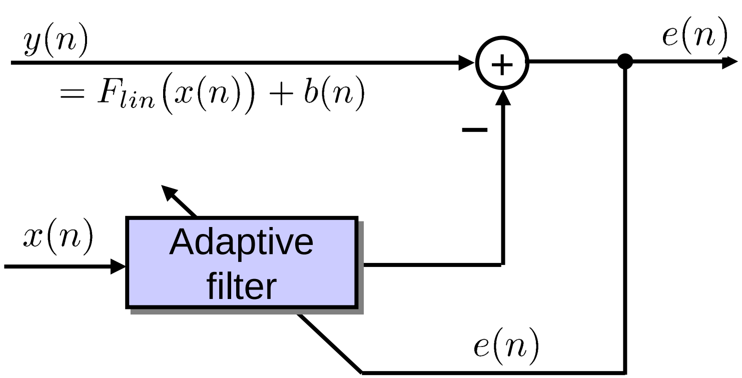


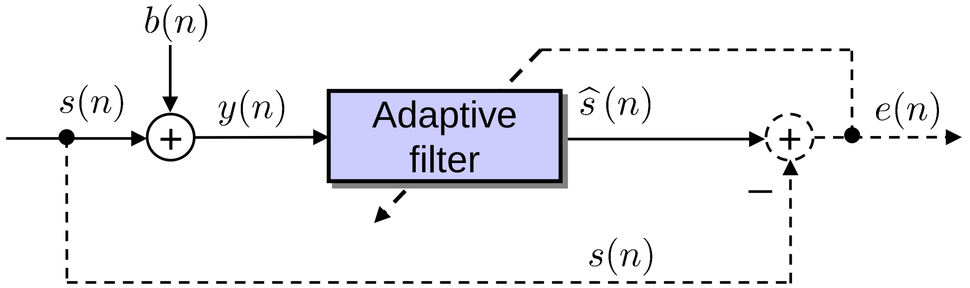
Repetition of different applications of adaptive filters

- Prediction:



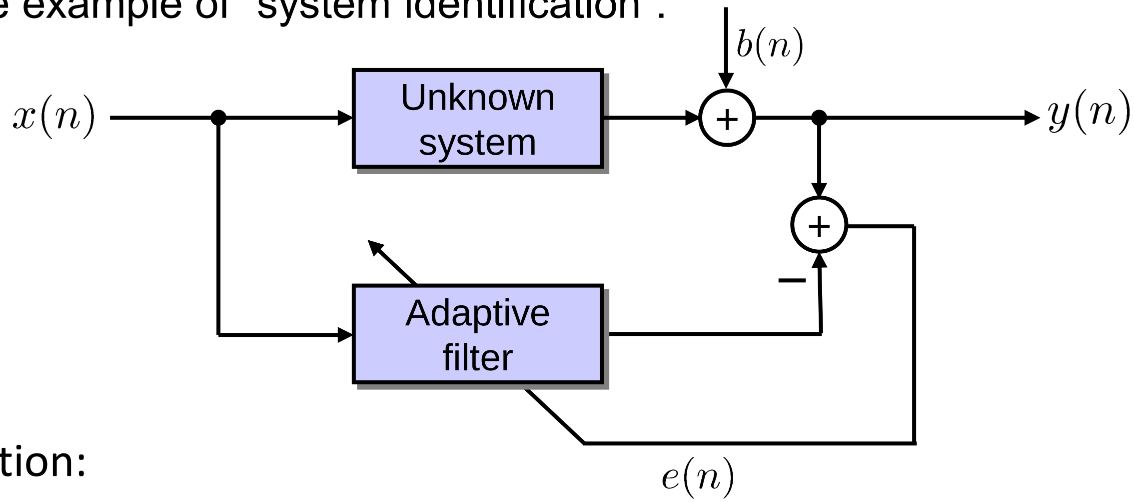
- Noise reduction with / without noise reference:





Repetition of the Wiener filter solution

- Based on the example of “system identification”:



- Wiener solution:

$$\boxed{\boldsymbol{h}_{\text{opt}} = \boldsymbol{R}_{xx}^{-1}\,\boldsymbol{r}_{xy}(0)}$$

Based on the assumption that
- all signals are stationary
- the system to identify is time-invariant

$$\begin{aligned}
\boldsymbol{h}_{\text{opt}} &= \left[h_{\text{opt},0},\, h_{\text{opt},1},\, \ldots,\, h_{\text{opt},N-1}\right]^{\mathrm{T}} \\
\boldsymbol{R}_{xx} &= \begin{bmatrix} r_{xx}(0) & r_{xx}(1) & \ldots & r_{xx}(N-1) \\ r_{xx}(1) & r_{xx}(0) & \ldots & r_{xx}(N-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}(N-1) & r_{xx}(N-2) & \ldots & r_{xx}(0) \end{bmatrix} \\
\boldsymbol{r}_{xy}(k) &= \left[r_{xy}(k),\, r_{xy}(k+1),\, \ldots,\, r_{xy}(k+N-1)\right]^{\mathrm{T}}
\end{aligned}$$

Wiener filter: Calculation of a stationary solution

- The solution was obtained based on the minimization of the mean square error:

$$\mathrm{E}\{e^2(n)\} \underset{\hat{\boldsymbol{h}}=\boldsymbol{h}_{\mathrm{opt}}}{\longrightarrow} \min$$

- Estimation of the correlation values:

$$\hat{r}_{xx}(l) = \begin{cases} \frac{1}{L} \sum\limits_{n=0}^{L-1-l} x(n)\,x(n+l), & \text{for } l \geq 0, \\ \frac{1}{L} \sum\limits_{n=-l}^{L-1} x(n)\,x(n+l), & \text{for } l < 0 \end{cases}$$

- Typically for stationary signals:
  The longer the window length L is the more reliable
  is the estimate.

- Allows to estimate a fixed filter only.

- A block-wise optimal filter calculation is possible by a block-wise estimation of the correlation functions or the power spectral densities, respectively.

- Computationally demanding, esp. due to the ACF-matrix inversion.

Least squares (LS) solution

- The basis for the adaptive "recursive least squares" (RLS) procedure, which we will consider as first adaptive procedure, is the least squares (LS) solution.

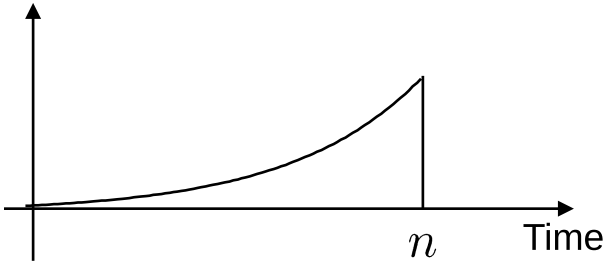
- Here, a **deterministic solution** is calculated based a **deterministic error criterion** which is the squared error, averaged over previous error signal samples:

  - Exponential weighting window:

forgetting factor

$$\mathcal{E}(n) = \sum_{l=0}^{n} \lambda^{n-l} \, |e(l)|^2 \quad \text{with } 0 < \lambda \leq 1$$

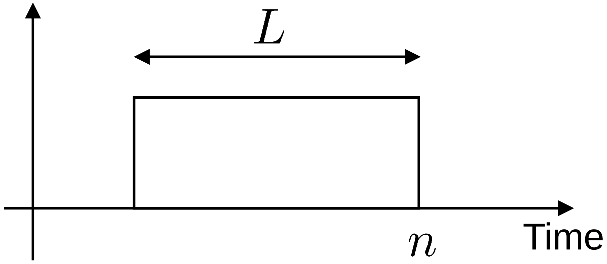
Weighting of the error values:



- Rectangular weighting window (a possible but not further considered alternative):

$$\mathcal{E}(n) = \sum_{l=n-L+1}^{n} |e(l)|^2$$

Weighting of the error values:



Least squares solution

- The error is calculated as follows:

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

- Insertion of the error leads to (for the exponential window):

$$\mathcal{E}(n) = \sum_{l=0}^{n} \lambda^{n-l} \left[y(l) - \widehat{\boldsymbol{h}}^{\mathrm{T}} \boldsymbol{x}(l) \right] \left[y(l) - \boldsymbol{x}^{\mathrm{T}}(l)\, \widehat{\boldsymbol{h}} \right]$$

- Differentiation with respect to the filter coefficients and setting result to zero leads to:

$$\sum_{l=0}^{n} \lambda^{n-l}\, \boldsymbol{x}(l)\, \boldsymbol{x}^{\mathrm{T}}(l)\, \widehat{\boldsymbol{h}} = \sum_{l=0}^{n} \lambda^{n-l}\, \boldsymbol{x}(l)\, y(l)$$

Least squares solution

- Previous matrix-vector equation:

$$\sum_{l=0}^{n} \lambda^{n-l}\, \boldsymbol{x}(l)\, \boldsymbol{x}^{\mathrm{T}}(l)\, \widehat{\boldsymbol{h}} = \sum_{l=0}^{n} \lambda^{n-l}\, \boldsymbol{x}(l)\, y(l)$$

- The differentiation and matrix-vector settings are defined analog to the derivation for the Wiener filter.

- Least squares solution:

$$\widehat{\boldsymbol{R}}_{xx}(n)\, \widehat{\boldsymbol{h}} = \widehat{\boldsymbol{r}}_{xy}(n)$$

$$\boxed{\widehat{\boldsymbol{h}} = \widehat{\boldsymbol{R}}_{xx}^{-1}(n)\, \widehat{\boldsymbol{r}}_{xy}(n)}$$

- Definitions:
  - Estimate for the auto-correlation matrix:

$$\widehat{\boldsymbol{R}}_{xx}(n) = \sum_{l=0}^{n} \lambda^{n-l}\, \boldsymbol{x}(l)\, \boldsymbol{x}^{\mathrm{T}}(l)$$

  - Estimate for the cross-correlation vector:

$$\widehat{\boldsymbol{r}}_{xy}(n) = \sum_{l=0}^{n} \lambda^{n-l}\, \boldsymbol{x}(l)\, y(l)$$

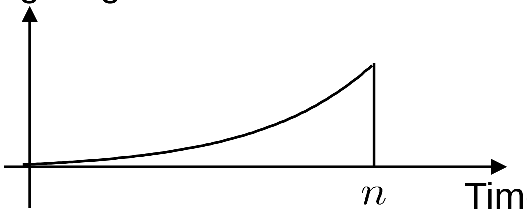
Comments about the forgetting factor

- Review: Exponential weighting window of the error signal:

forgetting factor

$$\mathcal{E}(n) = \sum_{l=0}^{n} \lambda^{n-l} \, |e(l)|^2 \quad \text{with } 0 < \lambda \leq 1$$

Weighting of the error values:



- The least squares solution is based on the following calculations:

$$\widehat{\boldsymbol{h}} = \widehat{\boldsymbol{R}}_{xx}^{-1}(n) \, \widehat{\boldsymbol{r}}_{xy}(n)$$

with:

$$\widehat{\boldsymbol{R}}_{xx}(n) = \sum_{l=0}^{n} \lambda^{n-l} \, \boldsymbol{x}(l) \, \boldsymbol{x}^{\mathrm{T}}(l)$$

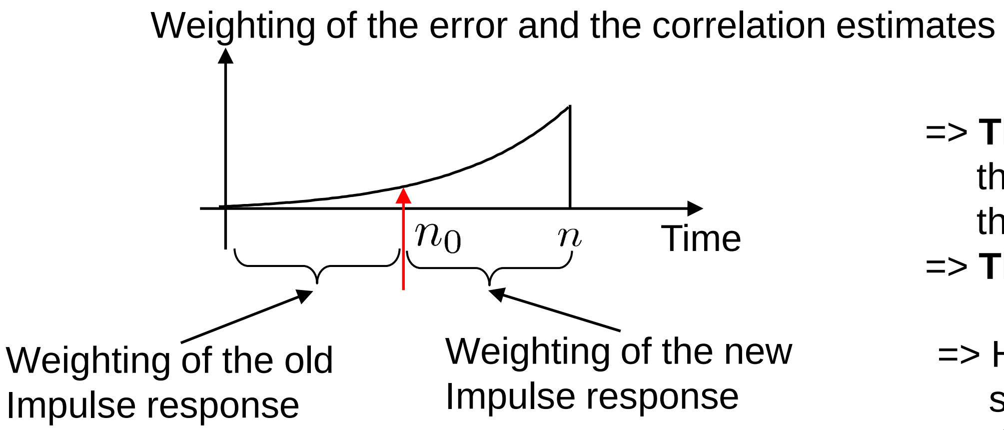
$$\widehat{\boldsymbol{r}}_{xy}(n) = \sum_{l=0}^{n} \lambda^{n-l} \, \boldsymbol{x}(l) \, y(l)$$

- Suppose the linear system which has to be identified is changing over time:
  => past components correspond to the previous impulse response of the linear system and the newer ones to the current impulse response.

Comments about the forgetting factor

- Suppose the linear system which has to be identified is changing over time at time instance n_0 :

  => Past components of the correlation estimates correspond to the previous impulse response of the linear system and the newer ones to the current impulse response.



=> **The smaller** the forgetting factor λ the faster the system can estimate the new impulse response
=> **The faster is the tracking**

=> However, the forgetting factor λ smoothes the estimate and attenuates disturbing noise components
=> **More reliable estimate in stationary situations for a large (close to 1) forgetting factor.**

Least squares solution

- The derivation of the least squares equations can be found here:

  1) Simon Haykin, „Adaptive Filter Theory“, Prentice Hall, 2002:

Section B.2 Examples **797**

EXAMPLE 3

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

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

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

$$\frac{\partial J}{\partial \mathbf{w}^*} = -\mathbf{p} + \mathbf{R}\mathbf{w}. \tag{B.11}$$

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

$$\mathbf{R}\mathbf{w}_o = \mathbf{p}. \tag{B.12}$$

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

  2) The Matrix Cookbook [http://orion.uwaterloo.ca/~hwolkowi/matrixcookbook.pdf]

Least squares: Going the step to a non-static solution

- Analyzing the least squares solution:

$$\widehat{\boldsymbol{h}} = \widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\widehat{\boldsymbol{r}}_{xy}(n)$$

one can see that in general this solution also offers a tracking of varying systems, since the estimates are updated at each time index:

$$\widehat{\boldsymbol{h}}(n) = \widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\widehat{\boldsymbol{r}}_{xy}(n)$$

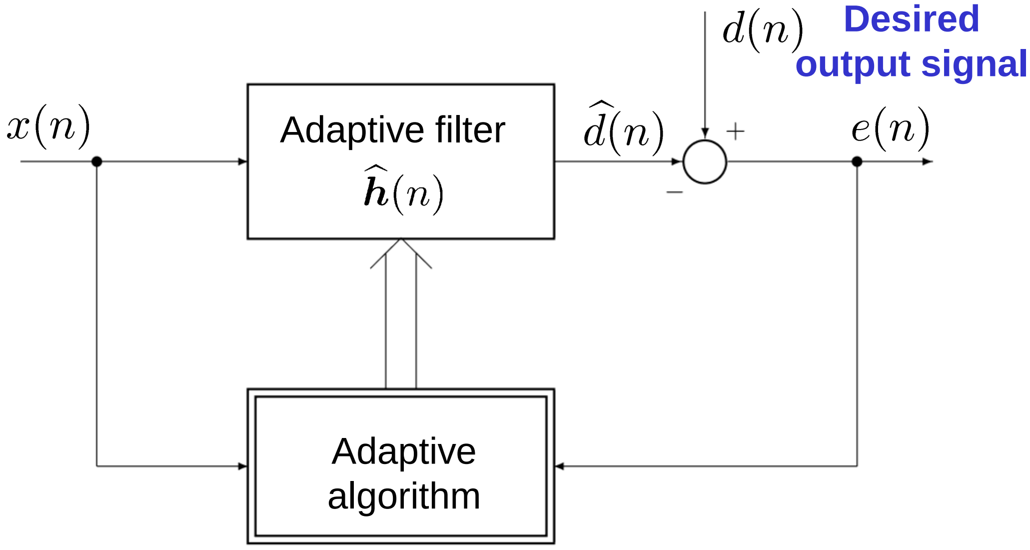
- However, then at each step a computational complex inversion of the auto-correlation matrix would be necessary where the estimate $\widehat{\boldsymbol{R}}_{xx}(n) = \sum_{l=0}^{n} \lambda^{n-l}\,\boldsymbol{x}(l)\,\boldsymbol{x}^{\mathrm{T}}(l)$ has **non**-Toeplitz structure.

- Therefore, recursive methods are developed which allow an update of the previous estimate which requires less computational effort.

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

Adaptive filters - Generals

- Adaptive filters are necessary in varying environments with changing systems to identify:



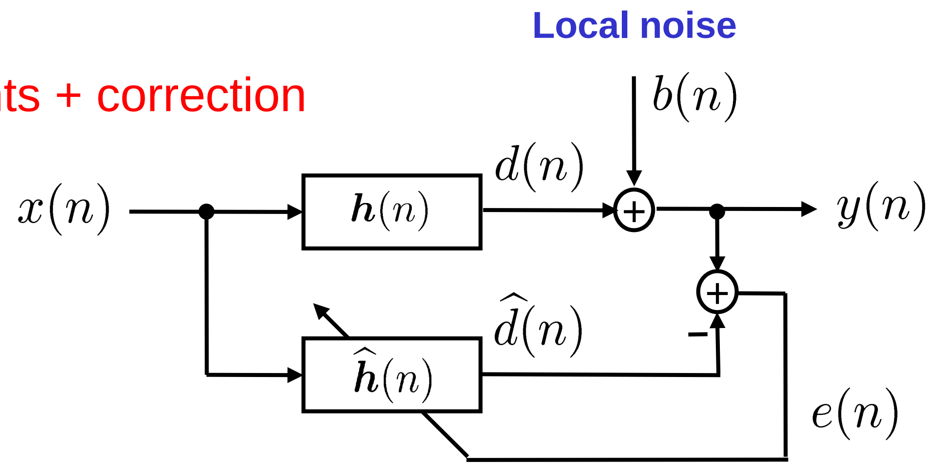
$$
\begin{aligned}
\boldsymbol{x}(n) &= \left[\, x(n),\, x(n-1),\, x(n-2),\, \ldots\,,\, x(n-N+1) \,\right]^{\mathrm{T}} \\
\widehat{\boldsymbol{h}}(n) &= \left[\, \widehat{h}_0(n),\, \widehat{h}_1(n),\, \widehat{h}_2(n),\, \ldots\,,\, \widehat{h}_{N-1}(n) \,\right]^{\mathrm{T}} \\
\widehat{d}(n) &= \widehat{\boldsymbol{h}}(n)^{\mathrm{T}}\, \boldsymbol{x}(n) = \boldsymbol{x}^{\mathrm{T}}(n)\, \widehat{\boldsymbol{h}}(n)
\end{aligned}
$$

Basic adaptation principle

TECHNISCHE UNIVERSITÄT DARMSTADT

- Basic adaptation principle:

  new coefficients = old coefficients + correction



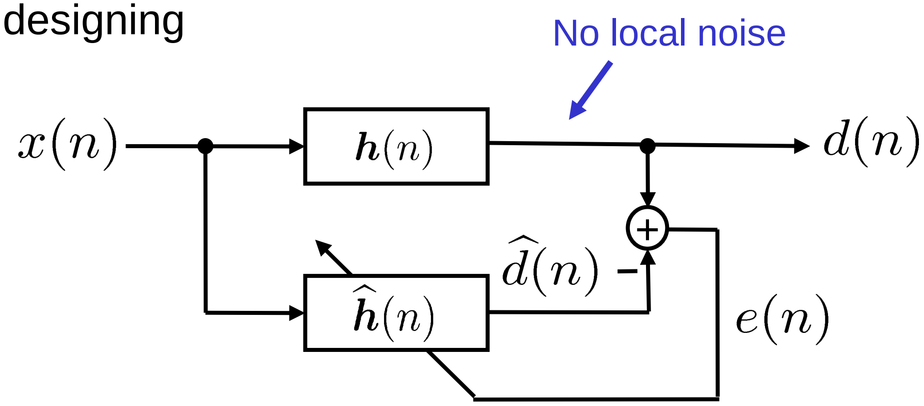
- Properties:
  - „Correction“ depends on the input signal $x(n)$ and the error signal $e(n)$
  - Procedures differ by the functions $g\big(\boldsymbol{x}(n)\big)$ and $f\big(e(n)\big)$.

$$\boxed{\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) + \mu\, \boldsymbol{x}(n)\, g\big(\boldsymbol{x}(n)\big)\, f\big(e(n)\big)}$$

Step size

Error measures

- Error measures are important when analyzing the performance of adaptive filters and designing adaptation control methods



- Mean square (signal) error :

$$\mathrm{E}\left\{\, |e(n)|^2 \,\right\} \quad = \quad \mathrm{E}\left\{\, |d(n) - \widehat{d}(n)|^2 \,\right\}$$

- System distance:

$$\begin{aligned} \|\boldsymbol{h}_\Delta(n)\|^2 \quad &= \quad \|\boldsymbol{h} - \widehat{\boldsymbol{h}}(\boldsymbol{n})\|^2 \\ &= \quad \left[\, \boldsymbol{h}(n) - \widehat{\boldsymbol{h}}(n) \,\right]^{\mathrm{T}} \left[\, \boldsymbol{h}(n) - \widehat{\boldsymbol{h}}(n) \,\right] \end{aligned}$$

Error measures

- Relation of the
  "normalized mean square (signal) error power" and
  "the system distance".

$$\frac{\mathrm{E}\left\{\,|e(n)|^2\,\right\}}{\mathrm{E}\left\{\,|x(n)|^2\,\right\}} = \frac{\left[\,\boldsymbol{h}(n)-\widehat{\boldsymbol{h}}(n)\,\right]^{\mathrm{T}}\mathrm{E}\left\{\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\,\right\}\,\left[\,\boldsymbol{h}(n)-\widehat{\boldsymbol{h}}(n)\,\right]}{\mathrm{E}\left\{\,|x(n)|^2\,\right\}}$$

$$= \frac{\boldsymbol{h}_{\Delta}^{\mathrm{T}}(n)\,\mathrm{E}\left\{\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\,\right\}\,\boldsymbol{h}_{\Delta}(n)}{\mathrm{E}\left\{\,|x(n)|^2\,\right\}}$$

- Let $x(n)$ be white noise:

$$\mathrm{E}\left\{\,\boldsymbol{x}(n)\,\boldsymbol{x}^{\mathrm{T}}(n)\,\right\} = \text{unit matrix} \times \mathrm{E}\left\{\,|x(n)|^2\,\right\}$$

$$\frac{\mathrm{E}\left\{\,|e(n)|^2\,\right\}}{\mathrm{E}\left\{\,|x(n)|^2\,\right\}} = \boldsymbol{h}_{\Delta}^{\mathrm{T}}(n)\,\boldsymbol{h}_{\Delta}(n) = \|\boldsymbol{h}_{\Delta}(n)\|^2$$

=> The *normalized mean square (signal) error* and the *system distance* are identical for white noise as input signal $x(n)$.

Error measures

- Different error measures are considered during the adaptation:
  - System distance vector:

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

  - A posteriori error, i.e., error used for the adaptation:

$$e(n) = e(n|n) = \boldsymbol{h}_\Delta^{\mathrm{T}}(n)\,\boldsymbol{x}(n) + b(n)$$

  - A priori error, error based on the previous coefficients:

$$e(n|n-1) = \boldsymbol{h}_\Delta^{\mathrm{T}}(n-1)\,\boldsymbol{x}(n) + b(n)$$

Data index *Filter index*

Recursive least squares (RLS) algorithm

The Recursive Least Squares (RLS) algorithm

- Analyzing the least squares solution

$$\widehat{\boldsymbol{h}}(n) = \widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\widehat{\boldsymbol{r}}_{xy}(n)$$

a computational efficient solution would be nice where a recursive calculation of the inverse auto-correlation matrix and the cross-correlation vector could be used.

- Definitions:
  - Estimate for the auto-correlation matrix:

$$\widehat{\boldsymbol{R}}_{xx}(n) = \sum_{l=0}^{n} \lambda^{n-l}\,\boldsymbol{x}(l)\,\boldsymbol{x}^{\mathrm{T}}(l)$$

  - Estimate for the cross-correlation vector:

$$\widehat{\boldsymbol{r}}_{xy}(n) = \sum_{l=0}^{n} \lambda^{n-l}\,\boldsymbol{x}(l)\,y(l)$$

Recursion – vector and matrix recursion

- Recursion of the cross-correlation vector over time:

$$\begin{aligned}\widehat{\boldsymbol{r}}_{xy}(n+1) &= \sum_{l=0}^{n+1} \lambda^{n+1-l}\,\boldsymbol{x}(l)\,y(l)\\ &= \lambda \sum_{l=0}^{n} \lambda^{n-l}\,\boldsymbol{x}(l)\,y(l) + \boldsymbol{x}(n+1)\,y(n+1)\end{aligned}$$

$$\boxed{\widehat{\boldsymbol{r}}_{xy}(n+1) = \lambda\,\widehat{\boldsymbol{r}}_{xy}(n) + \boldsymbol{x}(n+1)\,y(n+1)}$$

- Recursion of the auto-correlation matrix over time:

$$\begin{aligned}\widehat{\boldsymbol{R}}_{xx}(n+1) &= \sum_{l=0}^{n+1} \lambda^{n+1-l}\,\boldsymbol{x}(l)\,\boldsymbol{x}^{\mathrm{T}}(l)\\ &= \lambda \sum_{l=0}^{n} \lambda^{n-l}\,\boldsymbol{x}(l)\,\boldsymbol{x}^{\mathrm{T}}(l) + \boldsymbol{x}(n+1)\,\boldsymbol{x}^{\mathrm{T}}(n+1)\end{aligned}$$

$$\boxed{\widehat{\boldsymbol{R}}_{xx}(n+1) = \lambda\,\widehat{\boldsymbol{R}}_{xx}(n) + \boldsymbol{x}(n+1)\,\boldsymbol{x}^{\mathrm{T}}(n+1)}$$

However, a recursion for the inverse is necessary!

Recursion – inverse matrix recursion

TECHNISCHE UNIVERSITÄT DARMSTADT

- Recursion of the auto-correlation matrix over time:

$$\widehat{\boldsymbol{R}}_{xx}(n+1) \quad = \quad \underbrace{\lambda\,\widehat{\boldsymbol{R}}_{xx}(n)}_{\boldsymbol{A}} + \underbrace{\boldsymbol{x}(n+1)}_{\boldsymbol{u}}\,\underbrace{\boldsymbol{x}^{\mathrm{T}}(n+1)}_{\boldsymbol{v}^{\mathrm{T}}}$$

- Using the matrix inversion lemma…

$$\left[\boldsymbol{A}+\boldsymbol{u}\,\boldsymbol{v}^{\mathrm{T}}\right]^{-1} = \boldsymbol{A}^{-1} - \frac{\boldsymbol{A}^{-1}\,\boldsymbol{u}\,\boldsymbol{v}^{\mathrm{T}}\,\boldsymbol{A}^{-1}}{1+\boldsymbol{v}^{\mathrm{T}}\,\boldsymbol{A}^{-1}\,\boldsymbol{u}}$$

- one obtains a recursion for the inverse auto-correlation matrix:

$$\begin{aligned}\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1) \quad &= \quad \lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n) \\ &\quad - \quad \frac{\lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\lambda^{-1}}{1+\lambda^{-1}\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)}\end{aligned}$$

Recursion – inverse matrix recursion

- Repetition of the recursion:

$$\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1) = \lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n) - \frac{\lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\lambda^{-1}}{1+\lambda^{-1}\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)}$$

- One can define a gain vector:

$$\boldsymbol{\gamma}(n+1) = \frac{\lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)}{1+\lambda^{-1}\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)}$$

- And one can simplify the recursion formula:

$$\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1) = \lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n) - \boldsymbol{\gamma}(n+1)\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\lambda^{-1}$$

Recursion – gain factor reformulation

- Definition of the gain vector:

$$\boldsymbol{\gamma}(n+1) = \frac{\lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)}{1+\lambda^{-1}\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)}$$

- Multiplication with the denominator leads to:

$$\boldsymbol{\gamma}(n+1)\left[1+\lambda^{-1}\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)\right] = \lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)$$

- Rewriting (2. term to the right):

$$\begin{aligned}\boldsymbol{\gamma}(n+1) &= \lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)\\ &\quad -\lambda^{-1}\,\boldsymbol{\gamma}(n+1)\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)\end{aligned}$$

$$\boldsymbol{\gamma}(n+1) = \underbrace{\left[\lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)-\lambda^{-1}\,\boldsymbol{\gamma}(n+1)\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\right]}_{\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\ \text{(s. previous page)}}\boldsymbol{x}(n+1)$$

$$\boxed{\boldsymbol{\gamma}(n+1) = \widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\,\boldsymbol{x}(n+1)}$$

Recursion – filter coefficient recursion (I)

- Recursion of the filter coefficients vector:

$$\widehat{\boldsymbol{h}}(n) \quad = \quad \widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\widehat{\boldsymbol{r}}_{xy}(n)$$

- Step from *n* to *n+1*:

$$\widehat{\boldsymbol{h}}(n+1) \quad = \quad \widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\,\widehat{\boldsymbol{r}}_{xy}(n+1)$$

- Replacing the right-hand side by:

$$\widehat{\boldsymbol{r}}_{xy}(n+1) \quad = \quad \lambda\,\widehat{\boldsymbol{r}}_{xy}(n) + \boldsymbol{x}(n+1)\,y(n+1)$$

- one obtains the following recursion for the filter coefficients:

$$\widehat{\boldsymbol{h}}(n+1) \quad = \quad \lambda\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\,\widehat{\boldsymbol{r}}_{xy}(n) + \widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\,\boldsymbol{x}(n+1)\,y(n+1)$$

Recursion – filter coefficient recursion (II)

- So far, we have:

$$\widehat{\boldsymbol{h}}(n+1) = \lambda \boxed{\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)}\, \widehat{\boldsymbol{r}}_{xy}(n) + \widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\, \boldsymbol{x}(n+1)\, y(n+1)$$

- We insert the recursive calculation:

$$\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1) = \lambda^{-1}\, \widehat{\boldsymbol{R}}_{xx}^{-1}(n) - \boldsymbol{\gamma}(n+1)\, \boldsymbol{x}^{\mathrm{T}}(n+1)\, \widehat{\boldsymbol{R}}_{xx}^{-1}(n)\, \lambda^{-1}$$

- and obtain:

$$\begin{aligned}
\widehat{\boldsymbol{h}}(n+1) &= \boxed{\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\, \widehat{\boldsymbol{r}}_{xy}(n)} - \boldsymbol{\gamma}(n+1)\, \boldsymbol{x}^{\mathrm{T}}(n+1)\, \boxed{\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\, \widehat{\boldsymbol{r}}_{xy}(n)} \\
&\qquad + \widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\, \boldsymbol{x}(n+1)\, y(n+1) \\
&= \widehat{\boldsymbol{h}}(n) - \boldsymbol{\gamma}(n+1)\, \boldsymbol{x}^{\mathrm{T}}(n+1)\, \widehat{\boldsymbol{h}}(n) \\
&\qquad + \widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\, \boldsymbol{x}(n+1)\, y(n+1)
\end{aligned}$$

Recursion – filter coefficient recursion (III)

- So far, we have (first formulation with a recursion for $\widehat{\boldsymbol{h}}(n)$):

$$\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) - \boldsymbol{\gamma}(n+1)\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{h}}(n) + \underbrace{\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\,\boldsymbol{x}(n+1)}\,y(n+1)$$

- We insert the gain factor $\gamma(n+1)$:

$$\boldsymbol{\gamma}(n+1) = \widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\,\boldsymbol{x}(n+1)$$

- and obtain:

$$\widehat{\boldsymbol{h}}(n+1) = \widehat{\boldsymbol{h}}(n) + \underbrace{\boldsymbol{\gamma}(n+1)}_{\text{Gain vector}}\underbrace{\left[y(n+1) - \boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{h}}(n)\right]}_{\text{Error: old filter with new data}}$$

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

Recursion – filter coefficient recursion (IV)

- Inserting the previous results…

$$\boldsymbol{\gamma}(n+1) = \widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\,\boldsymbol{x}(n+1)$$

$$\widehat{\boldsymbol{h}}(n+1) \quad = \quad \widehat{\boldsymbol{h}}(n) + \overbrace{\boldsymbol{\gamma}(n+1)}\underbrace{\left[\,y(n+1) - \boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{h}}(n)\,\right]}$$

$$y(n+1) - \boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{h}}(n) = e(n+1|n)$$

- …leads to the following adaptation rule:

$$\widehat{\boldsymbol{h}}(n+1) \quad = \quad \widehat{\boldsymbol{h}}(n) + \underbrace{\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\,\boldsymbol{x}(n+1)\,e(n+1|n)}_{\boldsymbol{\Delta}\widehat{\boldsymbol{h}}(n)}$$

Summary of the Recursive Least Squares Algorithm (RLS)

1) Computation of a gain vector (complexity prop. to N^2):

$$\boldsymbol{\gamma}(n+1) \quad = \quad \frac{\lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)}{1+\lambda^{-1}\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\boldsymbol{x}(n+1)}$$

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

$$\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1) \quad = \quad \lambda^{-1}\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n) - \boldsymbol{\gamma}(n+1)\,\boldsymbol{x}^{\mathrm{T}}(n+1)\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n)\,\lambda^{-1}$$

3) Computation of the error signal (complexity prop. to N):

$$e(n+1|n) \quad = \quad y(n+1) - \widehat{\boldsymbol{h}}^{\mathrm{T}}(n)\,\boldsymbol{x}(n+1)$$

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

Step size (0 … 1)
will be considered later

$$\begin{aligned}\widehat{\boldsymbol{h}}(n+1) \quad &= \quad \widehat{\boldsymbol{h}}(n) + \mu\,\boldsymbol{\gamma}(n+1)\,e(n+1|n) \\ &= \quad \widehat{\boldsymbol{h}}(n) + \mu\,\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\,\boldsymbol{x}(n+1)\,e(n+1|n)\end{aligned}$$

Interpretation of Recursive Least Squares Algorithm (RLS)

- The update term…

$$\widehat{\boldsymbol{h}}(n+1) \quad = \quad \widehat{\boldsymbol{h}}(n) + \mu\, \widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\, \boldsymbol{x}(n+1)\, e(n+1|n)$$

- … has the direction determined by:

$$\widehat{\boldsymbol{R}}_{xx}^{-1}(n+1)\, \boldsymbol{x}(n+1)$$

- Meaning: If the signal is white, the direction of the update goes along the excitation vector $\boldsymbol{x}(n+1)$
- The non-white signals are whitened by multiplying with the inverse ACF matrix. Changes the direction of the vector $\boldsymbol{x}(n+1)$
- This ensures that the direction of the update term changes randomly and not a direction is dominant. This could lead to a decreased adaptation speed.

Interpretation of Recursive Least Squares Algorithm (RLS)

- Remember the error term and the error surfaces:

$$\mathrm{E}\{e^2(n)\} \;=\; E_{\min} + \left(\boldsymbol{h}-\boldsymbol{h}_{\mathrm{opt}}\right)^{\mathrm{T}} \boldsymbol{R}_{xx}\left(\boldsymbol{h}-\boldsymbol{h}_{\mathrm{opt}}\right)$$

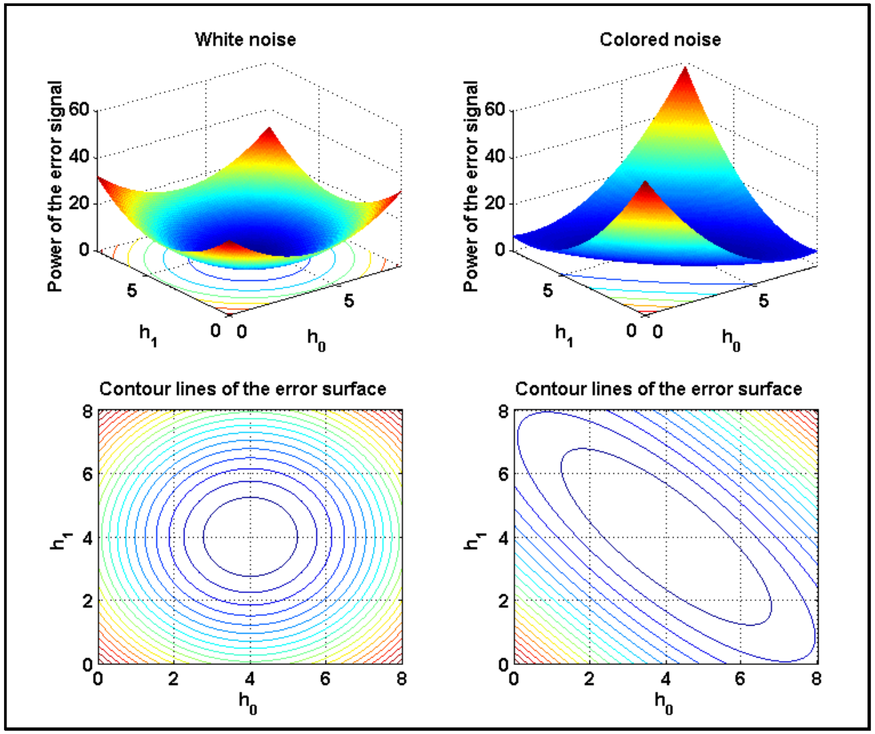
Quadratic equation results
in a unique minimum

- White noise:

$$\boldsymbol{R}_{xx} = \begin{bmatrix} 1.0 & 0 \\ 0 & 1.0 \end{bmatrix}$$

- Colored noise:

$$\boldsymbol{R}_{xx} = \begin{bmatrix} 1.0 & 0.8 \\ 0.8 & 1.0 \end{bmatrix}$$



Summary

This week:

- The Recursive Least Squares (RLS) algorithm.
- First, the properties of the Wiener filter solution have been explained, esp. the calculation of a stationary solution.
- Then the Least Squares (LS) solution has been derived mainly based on a deterministic error criterion.
- The LS solution was the starting point for the derivation of a recursive update formula: The RLS algorithm.

Next week:

- Adaptive filters: Least Mean Squares (LMS) algorithm