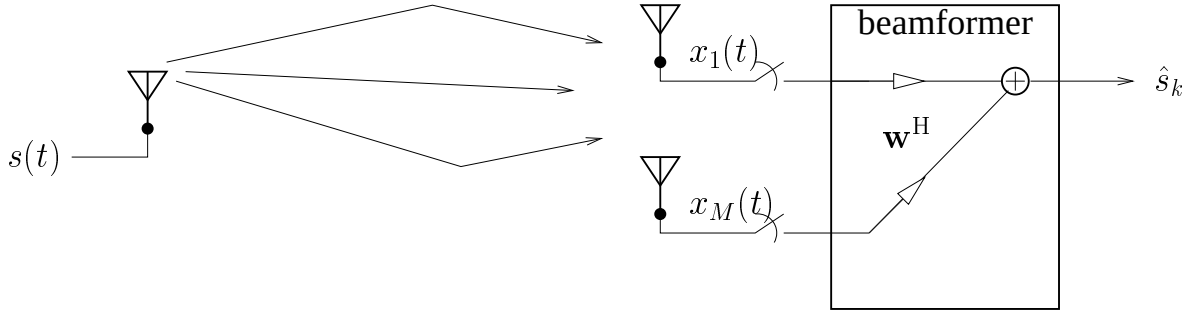
3. ADAPTIVE TECHNIQUES

Outline

1. Wiener filters – revisited
2. Steepest gradient descent algorithm
3. The LMS algorithm
4. Normalized LMS
5. The RLS algorithm
6. Applications

Data model



- We receive 1 signal with noise (plus interference)

$$\mathbf{x}_k = \mathbf{a}s_k + \mathbf{n}_k$$

- The source has unit power: $\mathrm{E}(|s_k|^2) = 1$.
- The received data covariance matrix is $\mathbf{R}_x = \mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})$.
- The correlation between the received data and the symbols is $\mathbf{r}_{xs} = \mathrm{E}(\mathbf{x}_k\bar{s}_k)$.

Objective: construct a receiver weight vector **w** such that

$$y_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k = \hat{s}_k$$

Wiener receiver

- The output error is $e_k = y_k - s_k$ where $y_k = \mathbf{w}^{\mathrm{H}}\mathbf{x}_k$

  The output error cost function is

$$\begin{aligned} J(\mathbf{w}) = \mathrm{E}(|e_k|^2) &= \mathrm{E}(|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k - s_k|^2) \\ &= \mathrm{E}[(\mathbf{w}^{\mathrm{H}}\mathbf{x}_k - s_k)(\mathbf{x}_k^{\mathrm{H}}\mathbf{w} - \bar{s}_k)] \\ &= \mathbf{w}^{\mathrm{H}}\mathbf{R}_x\mathbf{w} - \mathbf{w}^{\mathrm{H}}\mathbf{r}_{xs} - \mathbf{r}_{xs}^{\mathrm{H}}\mathbf{w} + 1\,. \end{aligned}$$

  The gradient of $J(\mathbf{w})$ is

$$\nabla J(\mathbf{w}) = \mathbf{R}_x\mathbf{w} - \mathbf{r}_{xs}$$

- Denote the optimum of $J(\mathbf{w})$ by $\mathbf{w}_0$. At the optimum, $\nabla J(\mathbf{w}_0) = 0$, so that

$$\mathbf{R}_x\mathbf{w}_0 = \mathbf{r}_{xs} \quad \Rightarrow \quad \mathbf{w}_0 = \mathbf{R}_x^{-1}\mathbf{r}_{xs}$$

$$\nabla J(\mathbf{w}) = 0 \quad \Rightarrow \quad \mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\mathbf{w} - \mathbf{x}_k\bar{s}_k) = \mathbf{0} \quad \Rightarrow \quad \mathrm{E}(\mathbf{x}_k\bar{e}_k) = \mathbf{0}$$

  At the optimum, the output error e_k is uncorrelated to the input vector:

  the *orthogonality principle*.

Wiener receiver

- At the optimum $\mathbf{w}_0 = \mathbf{R}_x^{-1}\mathbf{r}_{xs}$, the remaining cost is

$$J(\mathbf{w}_0) \;=\; \mathbf{w}_0^{\mathrm{H}}\mathbf{R}_x\mathbf{w}_0 - \mathbf{w}_0^{\mathrm{H}}\mathbf{r}_{xs} - \mathbf{r}_{xs}^{\mathrm{H}}\mathbf{w}_0 + 1 \;=\; 1 - \mathbf{r}_{xs}^{\mathrm{H}}\mathbf{R}_x^{-1}\mathbf{r}_{xs} \;=:\; J_0$$

  For any other $\mathbf{w}$,

$$J(\mathbf{w}) \;=\; J_0 + (\mathbf{w}-\mathbf{w}_0)^{\mathrm{H}}\mathbf{R}_x(\mathbf{w}-\mathbf{w}_0)$$

  Hence $J(\mathbf{w})$ is a quadratic function of $\mathbf{w}$, and $\mathbf{w}_0$ is really the minimizer.

- With finite data, all expectations are estimated from the available data:

$$\begin{aligned}\hat{\mathbf{R}}_x &= \tfrac{1}{N}\sum_{k=1}^{N}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}} = \tfrac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{H}}\\ \hat{\mathbf{r}}_{xs} &= \tfrac{1}{N}\sum_{k=1}^{N}\mathbf{x}_k\bar{s}_k = \tfrac{1}{N}\mathbf{X}\mathbf{s}^{\mathrm{H}}\end{aligned}$$

- The finite-sample cost function is

$$\hat{J}(\mathbf{w}) = \tfrac{1}{N}\sum_{k=1}^{N}|\mathbf{w}^{\mathrm{H}}\mathbf{x}_k - s_k|^2 = \tfrac{1}{N}\|\mathbf{w}^{\mathrm{H}}\mathbf{X} - \mathbf{s}\|^2$$

  The optimal finite-sample solution is $\hat{\mathbf{w}}_0 = \hat{\mathbf{R}}_x^{-1}\hat{\mathbf{r}}_{xs}$.

Steepest gradient descent algorithm

Optimal Wiener involves $\mathbf{R}_x^{-1}$. To avoid inversion, estimate the optimum *iteratively*.

- **Steepest Gradient descent technique** to find $\min f(x)$:
  - Take initial point $x^{(1)}$ with gradient $\nabla f^{(1)}$
  - For another point $x^{(2)}$ close to $x^{(1)}$, we can write

$$\nabla f^{(1)} \approx \frac{f^{(2)} - f^{(1)}}{x^{(2)} - x^{(1)}} \quad \Rightarrow \quad f^{(2)} \approx f^{(1)} + (x^{(2)} - x^{(1)})\nabla f^{(1)}$$

  - If we choose $x^{(2)} = x^{(1)} - \mu \nabla f^{(1)}$ with μ a small number (the *step size*), then

$$f^{(2)} \approx f^{(1)} - \mu(\nabla f^{(1)})^2 \quad < \quad f^{(1)}$$

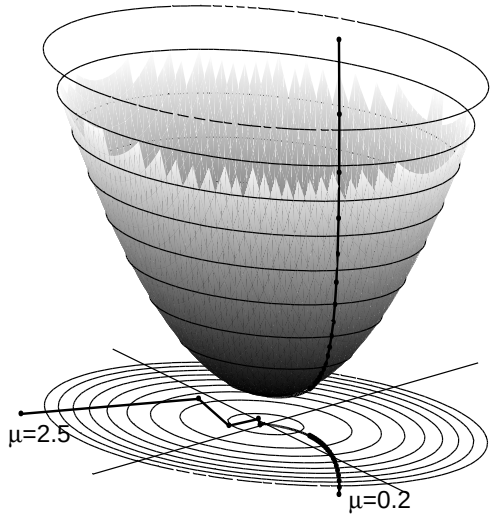
  - At the minimum, $\nabla f^{(1)} = 0$ and $x^{(2)} = x^{(1)}$.

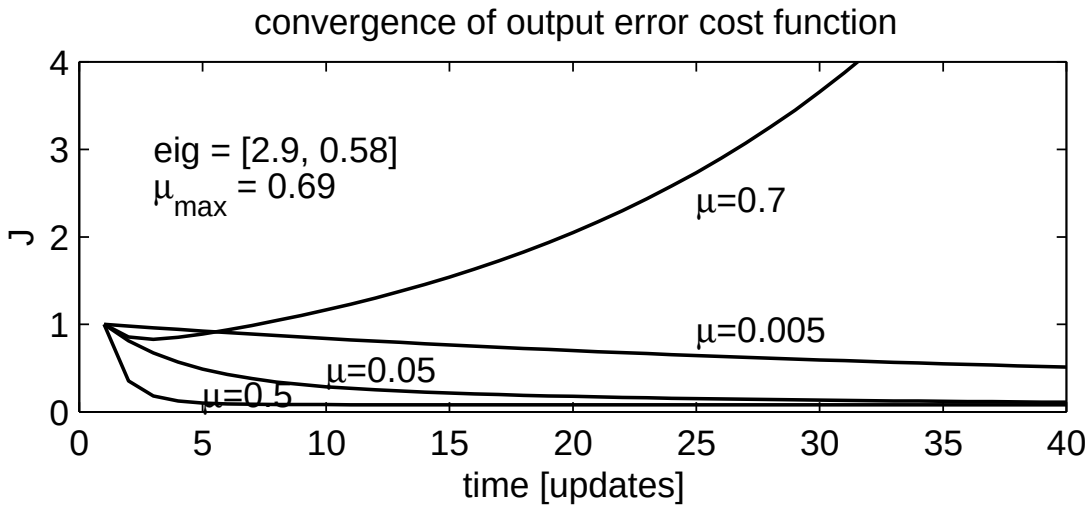
- In our application, we have $J(\mathbf{w})$ with $\nabla J(\mathbf{w}) = \mathbf{R}_x\mathbf{w} - \mathbf{r}_{xs}$:

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \mu(\mathbf{R}_x\mathbf{w}^{(k)} - \mathbf{r}_{xs})$$

Initialized usually by $\mathbf{w}^{(0)} = \mathbf{0}$.

Steepest gradient descent algorithm





Steepest gradient descent algorithm

Stability

- Let $\mathbf{w}_0$ denote the optimum, and define the weight error $\mathbf{c}^{(k)} = \mathbf{w}^{(k)} - \mathbf{w}_0$.

$$\begin{array}{rcll} \mathbf{w}^{(k+1)} & = & \mathbf{w}^{(k)} & - \quad \mu(\mathbf{R}_x\mathbf{w}^{(k)} - \mathbf{r}_{xs}) \\ \mathbf{w}_0 & = & \mathbf{w}_0 & - \quad \mu(\mathbf{R}_x\mathbf{w}_0 - \mathbf{r}_{xs}) \\ \hline \mathbf{c}^{(k+1)} & = & \mathbf{c}^{(k)} & - \quad \mu\mathbf{R}_x\mathbf{c}^{(k)} \end{array}$$

  Hence

$$\mathbf{c}^{(k+1)} = (\mathbf{I} - \mu\mathbf{R}_x)\mathbf{c}^{(k)} = \cdots = (\mathbf{I} - \mu\mathbf{R}_x)^{k+1}\mathbf{c}^{(0)}$$

  The recursion is stable iff $(\mathbf{I} - \mu\mathbf{R}_x)^k$ converges to zero.

- Introduce the eigenvalue decomposition

$$\mathbf{I} - \mu\mathbf{R}_x =: \mathbf{U}\mathbf{\Lambda}_\mu\mathbf{U}^{\mathrm{H}} \qquad \Rightarrow \qquad (\mathbf{I} - \mu\mathbf{R}_x)^k = \mathbf{U}(\mathbf{\Lambda}_\mu)^k\mathbf{U}^{\mathrm{H}}$$

  Change of variables: $\mathbf{v}^{(k)} := \mathbf{U}^{\mathrm{H}}\mathbf{c}^{(k)}$, so that $\mathbf{v}^{(k)} = (\mathbf{\Lambda}_\mu)^k\mathbf{v}^{(0)}$.

- Condition for stability of the recursion:

$$\|\mathbf{c}^{(k)}\| = \|\mathbf{v}^{(k)}\| \quad \rightarrow \quad 0 \qquad \Leftrightarrow \qquad |\lambda_{\mu,i}| < 1 \quad i = 1, \cdots, M$$

Steepest gradient descent algorithm – Stability

- Eigenvalue decomposition of $\mathbf{R}_x$:

$$\mathbf{R}_x := \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{H}}$$

then

$$\mathbf{I}-\mu\mathbf{R}_x = \mathbf{U}\mathbf{U}^{\mathrm{H}}-\mu\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{H}} = \mathbf{U}(\mathbf{I}-\mu\mathbf{\Lambda})\mathbf{U}^{\mathrm{H}}. \qquad \Rightarrow \qquad \mathbf{\Lambda}_\mu = \mathbf{I}-\mu\mathbf{\Lambda}$$

The recursion is stable if and only if

$$|1-\mu\lambda_i|<1, \qquad i=1,\cdots,M \qquad \Leftrightarrow \qquad 0<\mu\lambda_i<2, \qquad i=1,\cdots,M \qquad \Leftrightarrow$$

The steepest gradient descent algorithm is stable if

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

Steepest gradient descent algorithm

Convergence rate

$$\mathbf{v}^{(k)} = (\mathbf{\Lambda}_\mu)^k \mathbf{v}^{(0)} = (\mathbf{I} - \mu\mathbf{\Lambda})^k \mathbf{v}^{(0)}$$

- Each entry of $\mathbf{v}^{(k)}$ converges with a rate determined by $|1 - \mu\lambda_i|$.

  If $1 - \mu\lambda_{\max} > 0$, then the slowest mode is determined by $\lambda_{\min}$.

- Define a time constant τ such that

$$\|\mathbf{v}^{(\tau)}\| = \|\mathbf{v}^{(0)}\|/\mathrm{e} \qquad \Leftrightarrow \qquad (1 - \mu\lambda_{\min})^\tau = \frac{1}{\mathrm{e}}$$

  For sufficiently small μ,

$$\tau = \frac{-1}{\ln(1 - \mu\lambda_{\min})} \approx \frac{1}{\mu\lambda_{\min}}.$$

- If $\mu = \frac{1}{\lambda_{\max}}$, then

$$\tau \approx \frac{\lambda_{\max}}{\lambda_{\min}} =: \mathrm{cond}(\mathbf{R}_x).$$

  If the eigenvalues of $\mathbf{R}_x$ are widely spread then convergence will be slow.

The LMS algorithm

- Until now, $\nabla J(\mathbf{w}) = \mathbf{R}_x\mathbf{w} - \mathbf{r}_{xs}$ with $\mathbf{R}_x$ and vector $\mathbf{r}_{xs}$ perfectly known,

$$\mathbf{R}_x = \mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}), \qquad \mathbf{r}_{xs} = \mathrm{E}(\mathbf{x}_k\bar{s}_k)$$

  These quantities have to be estimated from the data.

- The Least-Mean-Square algorithm (LMS; Widrow 1975) makes simple estimates:

$$\hat{\mathbf{R}}_x = \mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}, \qquad \hat{\mathbf{r}}_{xs} = \mathbf{x}_k\bar{s}_k$$

  The resulting instantaneous gradient estimate is

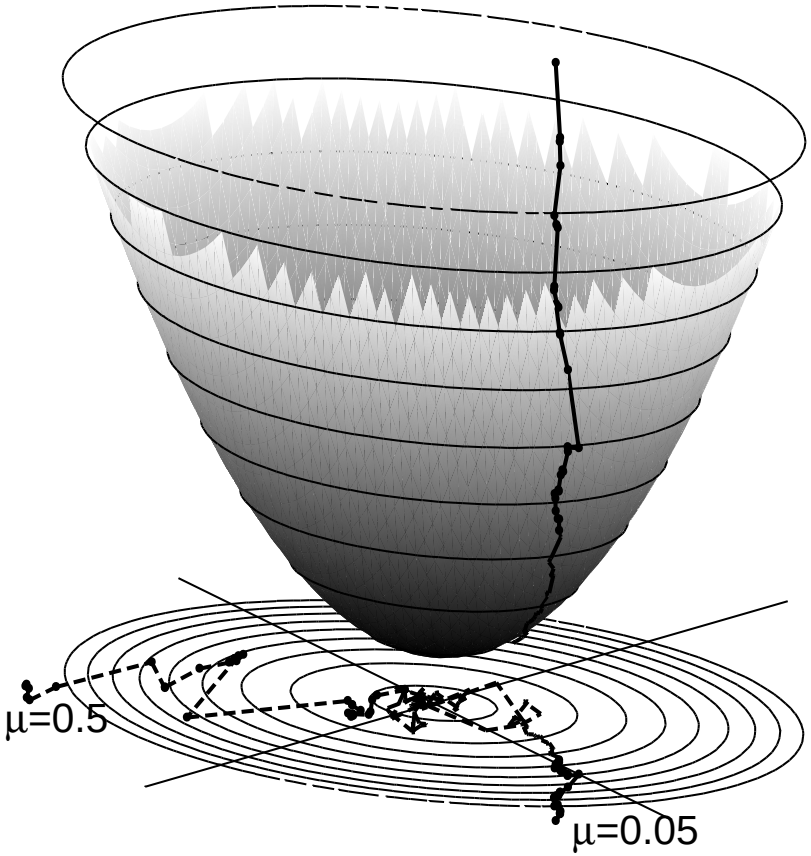
$$\widehat{\nabla J}(\mathbf{w}) = \mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\mathbf{w} - \mathbf{x}_k\bar{s}_k = \mathbf{x}_k(\mathbf{x}_k^{\mathrm{H}}\mathbf{w} - \bar{s}_k) = \mathbf{x}_k\bar{e}_k$$

- **LMS algorithm:**

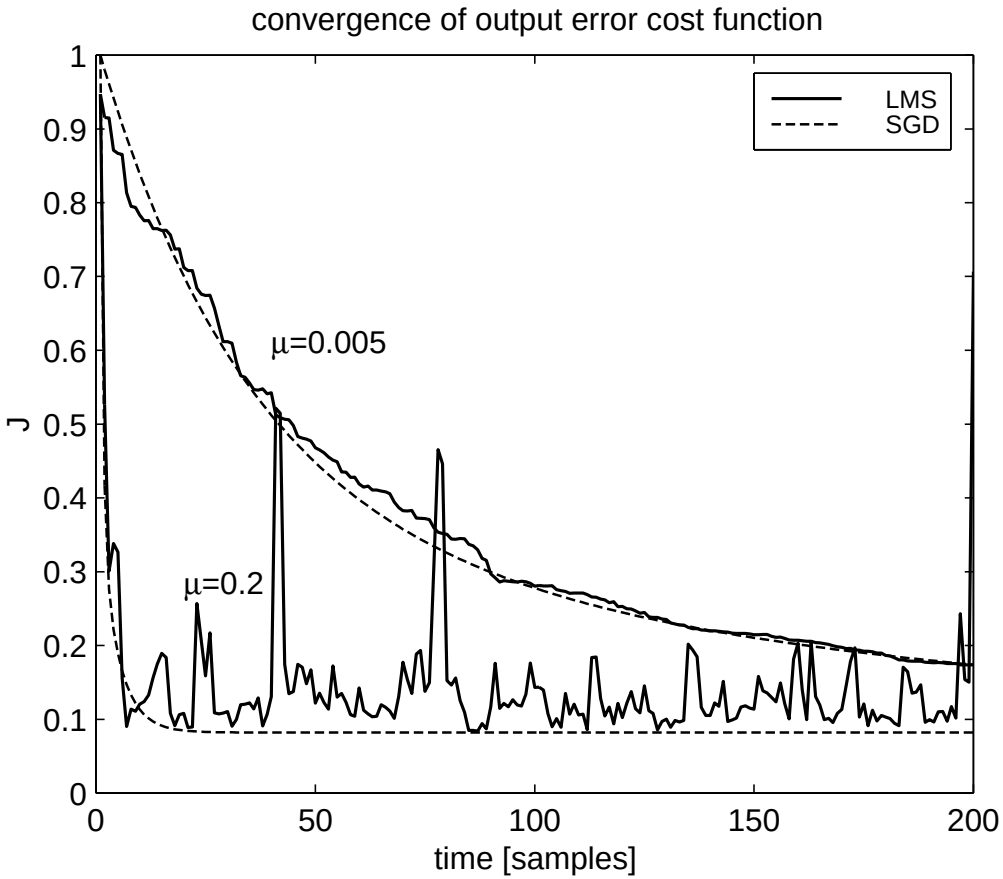
$$\begin{aligned} y_k &:= \hat{\mathbf{w}}^{(k)H}\mathbf{x}_k \\ e_k &:= y_k - s_k \\ \hat{\mathbf{w}}^{(k+1)} &:= \hat{\mathbf{w}}^{(k)} - \mu\mathbf{x}_k\bar{e}_k \end{aligned}$$

  Initialized usually by $\hat{\mathbf{w}}^{(0)} = \mathbf{0}$.

The LMS algorithm



The LMS algorithm



For small μ, the LMS stays close to the Steepest Gradient Descent algorithm.

The LMS algorithm

Convergence in the mean

- Compare the error in the LMS weight vector to the Wiener weight, $\mathbf{w}_0 = \mathbf{R}_x^{-1}\mathbf{r}_{xs}$,

$$\epsilon_k = \mathbf{w}^{(k)} - \mathbf{w}_0$$

- Convergence of $\mathrm{E}(\mathbf{w}^{(k)})$ (the ensemble-averaged weight vector):

$$\begin{array}{rcccccc}
\hat{\mathbf{w}}^{(k+1)} & = & \hat{\mathbf{w}}^{(k)} & - & \mu[\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\hat{\mathbf{w}}^{(k)} & - & \mathbf{x}_k\bar{s}_k] \\
\mathbf{w}_0 & = & \mathbf{w}_0 & - & \mu[\mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})\mathbf{w}_0 & - & \mathrm{E}(\mathbf{x}_k\bar{s}_k)] \\
\hline
\epsilon_{k+1} & = & \epsilon_k & - & \mu[\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\hat{\mathbf{w}}^{(k)} - \mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})\mathbf{w}_0 & - & (\mathbf{x}_k\bar{s}_k - \mathrm{E}(\mathbf{x}_k\bar{s}_k))]
\end{array}$$

$$\begin{aligned}
\mathrm{E}(\epsilon_{k+1}) &= \mathrm{E}(\epsilon_k) - \mu[\mathrm{E}(\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}})\mathrm{E}(\epsilon_k) - \mathbf{0}] \\
&= \mathrm{E}(\epsilon_k) - \mu\mathbf{R}_x\mathrm{E}(\epsilon_k) && \text{(independence)} \\
&= (\mathbf{I} - \mu\mathbf{R}_x)\,\mathrm{E}(\epsilon_k)
\end{aligned}$$

- Average convergence of LMS is the same for SGD. Require $0 < \mu < \frac{2}{\lambda_{\max}}$

The LMS algorithm

Convergence in the mean-square

- Ensemble-average value of the LMS mean-square error:

$$
\begin{aligned}
J_k &:= \mathrm{E}(e_k \bar{e}_k) \\
&= \mathrm{E}([\hat{\mathbf{w}}^{(k)H}\mathbf{x}_k - s_k]^{\mathrm{H}}[\hat{\mathbf{w}}^{(k)H}\mathbf{x}_k - s_k]) \qquad (\hat{\mathbf{w}}^{(k)} = \mathbf{w}_0 + \epsilon_k) \\
&= \mathrm{E}\{[\mathbf{w}_0^{\mathrm{H}}\mathbf{x}_k - s_k]^{\mathrm{H}}[\mathbf{w}_0^{\mathrm{H}}\mathbf{x}_k - s_k] + \epsilon_k^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\epsilon_k + \epsilon_k^{\mathrm{H}}\mathbf{x}_k(\mathbf{w}_0^{\mathrm{H}}\mathbf{x}_k - s_k) + (\mathbf{w}_0^{\mathrm{H}}\mathbf{x}_k - s_k)^{\mathrm{H}}\mathbf{x}_k^{\mathrm{H}}\epsilon_k\} \\
&= \underbrace{J_{\min}}_{\text{Wiener error}} + \underbrace{\mathrm{E}(\epsilon_k^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\epsilon_k)}_{\text{excess error}}
\end{aligned}
$$

$$
\begin{aligned}
J_{ex}(k) &:= \mathrm{E}(\epsilon_k^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\epsilon_k) = \mathrm{E}(\mathrm{tr}[\epsilon_k^{\mathrm{H}}\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\epsilon_k]) = \mathrm{E}(\mathrm{tr}[\mathbf{x}_k\mathbf{x}_k^{\mathrm{H}}\epsilon_k\epsilon_k^{\mathrm{H}}]) \\
&= \mathrm{tr}(\mathbf{R}_x\mathbf{K}_k), \qquad \mathbf{K}_k := \mathrm{E}(\epsilon_k\epsilon_k^{\mathrm{H}})
\end{aligned}
$$

- Can show that if μ satisfies

$$\gamma := \sum_{i=1}^{M} \frac{\mu\lambda_i}{2-\mu\lambda_i} < 1$$

then the mean-squared error of the LMS algorithm converges and

$$J_{ex}(\infty) = J_{\min} \cdot \frac{\gamma}{1-\gamma}$$

The LMS algorithm

- Suppose that $\mu\lambda_i \ll 1$, $i = 1, \cdots, M$, then

$$J_{ex}(\infty) \quad \approx \quad J_{\min}\,\gamma \quad \approx \quad J_{\min}\,\tfrac{1}{2}\mu \sum_{i=1}^{M} \lambda_i$$

  Note: $\sum \lambda_i = \mathrm{E}(\|\mathbf{x}_k\|^2)$: average input power

- In summary, the LMS converges in the mean-square if and only if

$$0 \;<\; \mu \;<\; \frac{2}{\mathrm{E}(\|\mathbf{x}_k\|^2)}$$

  and the MSE is given by

$$J(\infty) \;\approx\; J_{\min}[1 + \tfrac{1}{2}\mu \mathrm{E}(\|\mathbf{x}_k\|^2)]$$

Normalized LMS

- In LMS, the value of μ depends on the *scaling* of the data:

$$\hat{\mathbf{w}}^{(k+1)} = \hat{\mathbf{w}}^{(k)} - \mu \mathbf{x}_k \bar{e}_k\,, \qquad e_k = \mathbf{w}^{(k)H}\mathbf{x}_k - s_k$$

  If $\mathbf{x}_k$ is scaled by α then e_k also scales with α, and μ has to be scaled by α^{-2}.

Normalized LMS

- Scaling-invariant recursion:

$$\hat{\mathbf{w}}^{(k+1)} = \hat{\mathbf{w}}^{(k)} - \frac{\tilde{\mu}}{\|\mathbf{x}_k\|^2}\mathbf{x}_k \bar{e}_k$$

  Effective step size is $\mu_k = \tilde{\mu}/\|\mathbf{x}_k\|^2$: *time-varying step*.

  To avoid division by zero, one often adds a small positive number to $\|\mathbf{x}_k\|^2$.

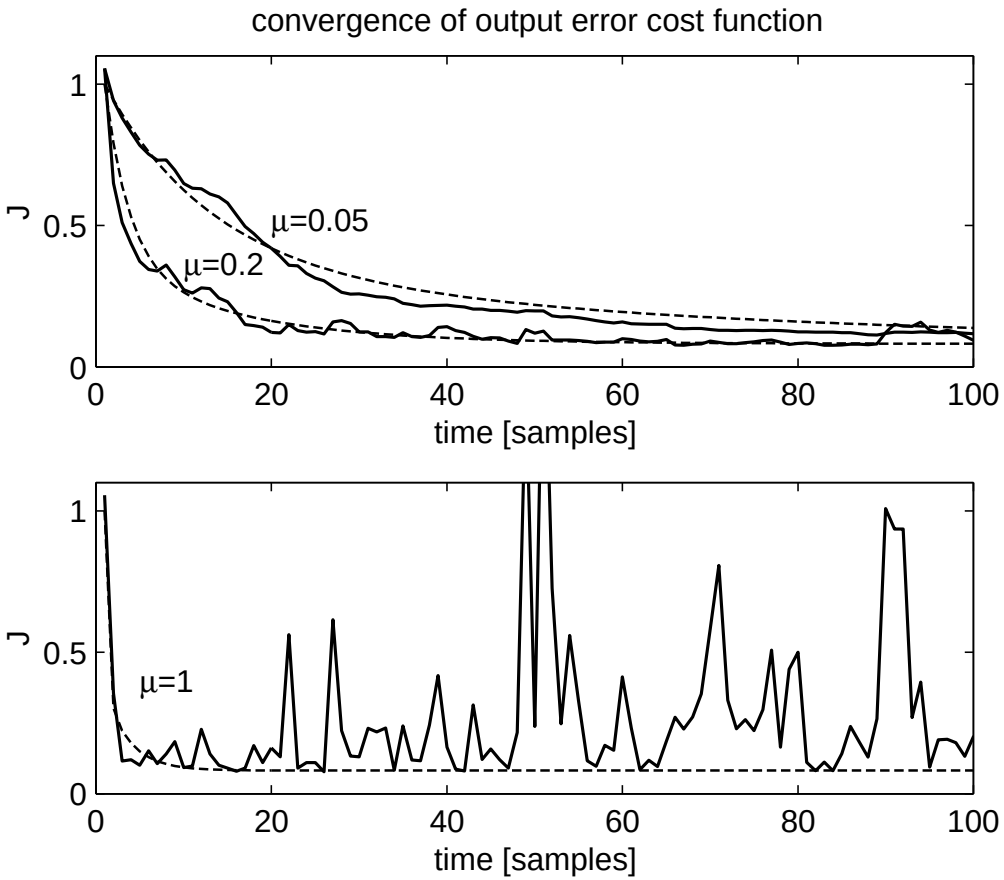
- The NLMS converges in the mean-square if and only if

$$0 < \tilde{\mu} < 2$$

  and the MSE is given by

$$J(\infty) \approx J_{\min}[1 + \tfrac{1}{2}\tilde{\mu}]$$

Normalized LMS



Matrix inversion lemma

$$(\mathbf{A}-\mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1}\mathbf{B})^{-1} = \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}}(\mathbf{C}-\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}})^{-1}\mathbf{B}\mathbf{A}^{-1}.$$

PROOF

$$\begin{bmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{H}} \\ \mathbf{B} & \mathbf{C} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1} \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A}-\mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1}\mathbf{B} & \\ & \mathbf{C} \end{bmatrix} \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{C}^{-1}\mathbf{B} & \mathbf{I} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{B}\mathbf{A}^{-1} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A} & \\ & \mathbf{C}-\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}} \\ 0 & \mathbf{I} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{H}} \\ \mathbf{B} & \mathbf{C} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} & 0 \\ -\mathbf{C}^{-1}\mathbf{B} & \mathbf{I} \end{bmatrix} \begin{bmatrix} (\mathbf{A}-\mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1}\mathbf{B})^{-1} & 0 \\ 0 & \mathbf{C}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & -\mathbf{B}^{\mathrm{H}}\mathbf{C}^{-1} \\ 0 & \mathbf{I} \end{bmatrix}$$

$$\begin{bmatrix} \mathbf{A} & \mathbf{B}^{\mathrm{H}} \\ \mathbf{B} & \mathbf{C} \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{I} & -\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}} \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A}^{-1} & \\ & (\mathbf{C}-\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{I} & 0 \\ -\mathbf{B}\mathbf{A}^{-1} & \mathbf{I} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{A}^{-1}+\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}}(\mathbf{C}-\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^{\mathrm{H}})^{-1}\mathbf{B}\mathbf{A}^{-1} & * \\ * & * \end{bmatrix}$$

The RLS algorithm

- Suppose at time k we know $\quad \mathbf{X}_k := [\mathbf{x}_1, \cdots, \mathbf{x}_k], \quad \mathbf{s}_k := [s_1, \cdots, s_k]$

  The solution to $\min \| \mathbf{w}^{\mathrm{H}} \mathbf{X}_k - \mathbf{s}_k \|^2$ is

$$\hat{\mathbf{w}}^{(k)} = \mathbf{X}_k^{\dagger} \mathbf{s}_k^{\mathrm{H}} = (\mathbf{X}_k \mathbf{X}_k^{\mathrm{H}})^{-1} \mathbf{X}_k \mathbf{s}_k^{\mathrm{H}} =: \mathbf{\Phi}_k^{-1} \theta_k$$

  where

$$\mathbf{\Phi}_k := \mathbf{X}_k \mathbf{X}_k^{\mathrm{H}} = \sum_{i=1}^{k} \mathbf{x}_k \mathbf{x}_k^{\mathrm{H}}, \qquad \theta_k := \mathbf{X}_k \mathbf{s}_k^{\mathrm{H}} = \sum_{i=1}^{k} \mathbf{x}_k \bar{s}_k$$

- Update of $\mathbf{\Phi}_k^{-1}$:

$$\mathbf{\Phi}_{k+1} = \mathbf{\Phi}_k + \mathbf{x}_{k+1} \mathbf{x}_{k+1}^{\mathrm{H}} \quad \Rightarrow \quad \mathbf{\Phi}_{k+1}^{-1} = (\mathbf{\Phi}_k + \mathbf{x}_{k+1} \mathbf{x}_{k+1}^{\mathrm{H}})^{-1} = \mathbf{\Phi}_k^{-1} - \frac{\mathbf{\Phi}_k^{-1} \mathbf{x}_{k+1} \mathbf{x}_{k+1}^{\mathrm{H}} \mathbf{\Phi}_k^{-1}}{1 + \mathbf{x}_{k+1}^{\mathrm{H}} \mathbf{\Phi}_k^{-1} \mathbf{x}_{k+1}}$$

- *Recursive Least Squares* (RLS) algorithm:

$$\begin{aligned} \mathbf{P}_{k+1} &:= \mathbf{P}_k - \frac{\mathbf{P}_k \mathbf{x}_{k+1} \mathbf{x}_{k+1}^{\mathrm{H}} \mathbf{P}_k}{1 + \mathbf{x}_{k+1}^{\mathrm{H}} \mathbf{P}_k \mathbf{x}_{k+1}} \\ \theta_{k+1} &:= \theta_k + \mathbf{x}_{k+1} \bar{s}_{k+1} \\ \hat{\mathbf{w}}^{(k+1)} &:= \mathbf{P}_{k+1} \theta_{k+1} \end{aligned}$$

  Initialization: $\theta_0 = \mathbf{0}$ and $\mathbf{P}_0 = \delta^{-1} \mathbf{I}$, where δ is a very small positive constant.

The RLS algorithm

Finite horizon

For adaptive purposes, we want an effective window of data.

1. *Sliding window:* $\mathbf{\Phi}_k$ and θ_k based on only the last n samples:

$$\mathbf{\Phi}_{k+1} = \mathbf{\Phi}_k + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}} - \mathbf{x}_{k-n}\mathbf{x}_{k-n}^{\mathrm{H}}$$

$$\theta_{k+1} = \theta_k + \mathbf{x}_{k+1}\bar{s}_{k+1} - \mathbf{x}_{k-n}\bar{s}_{k-n}^{\mathrm{H}}$$

   Doubles complexity, and we have to keep n previous data samples in memory.

2. *Exponential window*: scale down $\mathbf{\Phi}_k$ and θ_k by a factor $\lambda \approx 1$:

$$\mathbf{\Phi}_{k+1} = \lambda\mathbf{\Phi}_k + \mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}}$$

$$\theta_{k+1} = \lambda\theta_k + \mathbf{x}_{k+1}\bar{s}_{k+1}$$

   Corresponds to

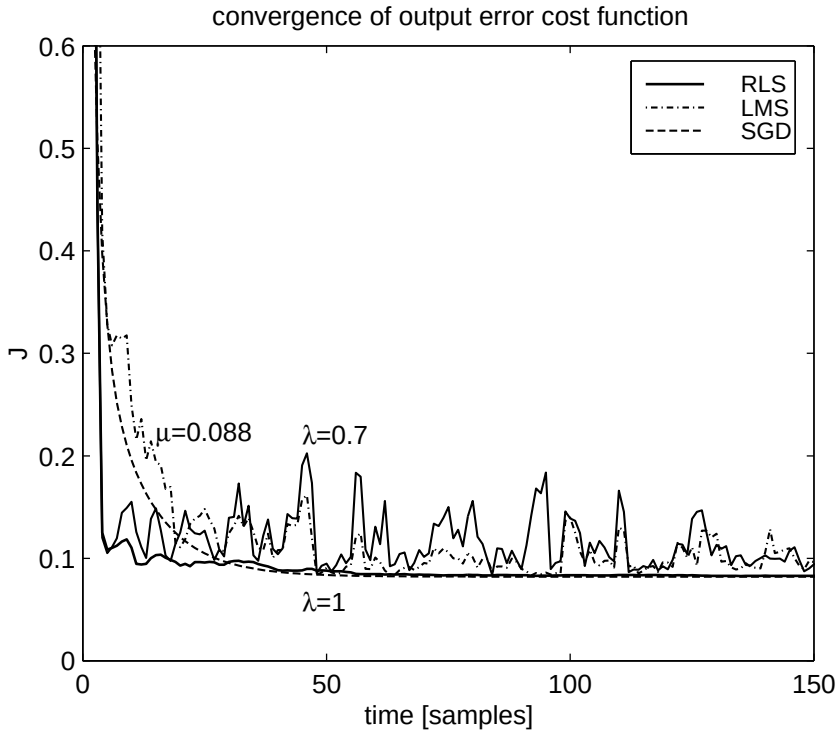
$$\mathbf{X}_{k+1} = [\mathbf{x}_{k+1} \quad \lambda^{1/2}\mathbf{x}_k \quad \lambda\mathbf{x}_{k-1} \quad \lambda^{3/2}\mathbf{x}_{k-2} \quad \cdots]$$

$$\mathbf{s}_{k+1} = [s_{k+1} \quad \lambda^{1/2}s_k \quad \lambda s_{k-1} \quad \lambda^{3/2}s_{k-2} \quad \cdots]$$

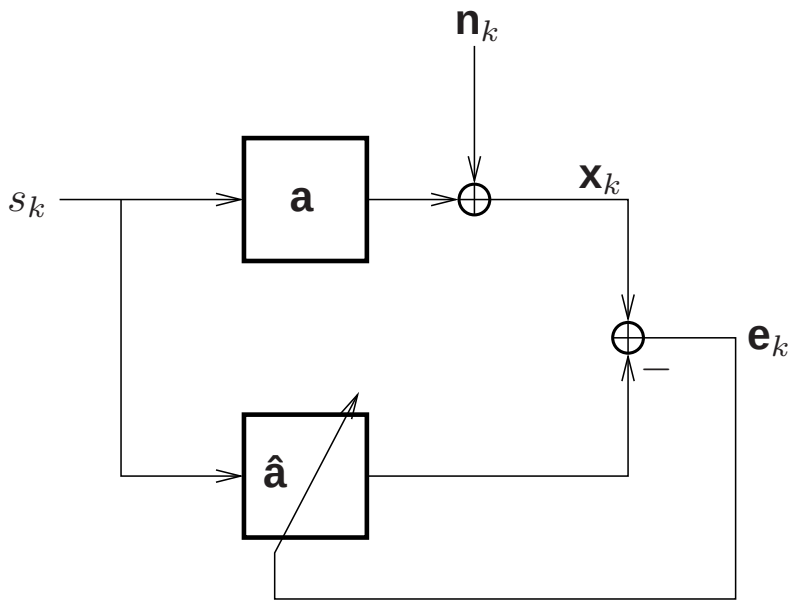
The RLS algorithm

Exponentially weighted RLS algorithm:

$$\begin{aligned}
\mathbf{P}_{k+1} &:= \lambda^{-1}\mathbf{P}_k - \lambda^{-2}\frac{\mathbf{P}_k\mathbf{x}_{k+1}\mathbf{x}_{k+1}^{\mathrm{H}}\mathbf{P}_k}{1+\lambda^{-1}\mathbf{x}_{k+1}^{\mathrm{H}}\mathbf{P}_k\mathbf{x}_{k+1}} \\
\theta_{k+1} &:= \lambda\theta_k + \mathbf{x}_{k+1}\bar{s}_{k+1} \\
\hat{\mathbf{w}}^{(k+1)} &:= \mathbf{P}_{k+1}\theta_{k+1}
\end{aligned}$$



Adaptive model matching



- **Objective:** estimate the channel by minimizing the model error $\mathbf{e}_k = \mathbf{x}_k - \hat{\mathbf{a}} s_k$:

$$\begin{aligned} J(\hat{\mathbf{a}}) &:= \mathrm{E}(\|\mathbf{e}_k\|^2) = \mathrm{E}(\|\mathbf{x}_k - \hat{\mathbf{a}} s_k\|^2) \\ &= \mathrm{E}([\mathbf{x}_k - \hat{\mathbf{a}} s_k]^{\mathrm{H}} [\mathbf{x}_k - \hat{\mathbf{a}} s_k]) \\ &= \mathrm{E}(\|\mathbf{x}_k\|^2) - \hat{\mathbf{a}}^{\mathrm{H}} \mathbf{r}_{xs} - \mathbf{r}_{xs}^{\mathrm{H}} \hat{\mathbf{a}} + \hat{\mathbf{a}}^{\mathrm{H}} r_s \hat{\mathbf{a}}, \qquad r_s = \mathrm{E}(s_k \bar{s}_k) = \mathrm{E}(|s_k|^2) \end{aligned}$$

The gradient is

$$\nabla J(\hat{\mathbf{a}}) = r_s \hat{\mathbf{a}} - \mathbf{r}_{xs} = r_s(\hat{\mathbf{a}} - \mathbf{a})$$

Adaptive model matching

Steepest gradient algorithm

$$\hat{\mathbf{a}}^{(k+1)} = \hat{\mathbf{a}}^{(k)} - \mu \nabla J(\hat{\mathbf{a}}^{(k)}) = \hat{\mathbf{a}}^{(k)} - \mu r_s(\hat{\mathbf{a}}^{(k)} - \mathbf{a}) = \mu r_s \mathbf{a} + (1 - \mu r_s)\hat{\mathbf{a}}^{(k)}$$

Conververgence

$$\begin{aligned} x_k &= b + a x_{k-1} = b + ba + a^2 x_{k-2} = \cdots = b + ba + \cdots + ba^{k-1} + a^k x_0 \\ &= b\frac{1 - a^k}{1 - a} + a^k x_0 \,. \end{aligned}$$

In our case, we obtain similarly

$$\hat{\mathbf{a}}^{(k)} = \mathbf{a}\mu r_s \frac{1 - (1 - \mu r_s)^k}{1 - (1 - \mu r_s)} + (1 - \mu r_s)^k \hat{\mathbf{a}}^{(0)} = \mathbf{a}[1 - (1 - \mu r_s)^k] + (1 - \mu r_s)^k \hat{\mathbf{a}}^{(0)}$$

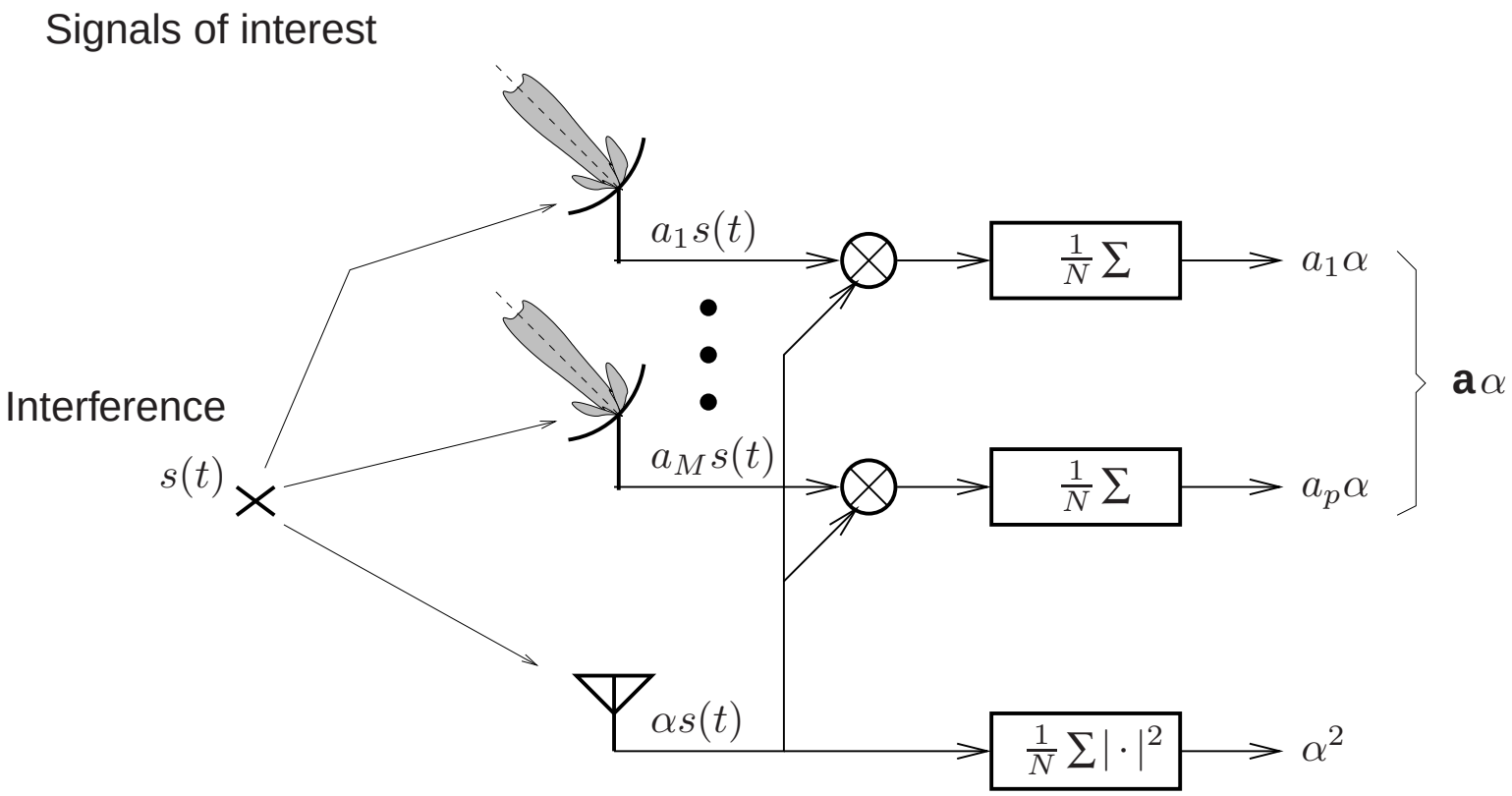
Convergence to **a** if $|1 - \mu r_s| < 1$, i.e., $0 < \mu < \frac{2}{r_s}$.

LMS algorithm

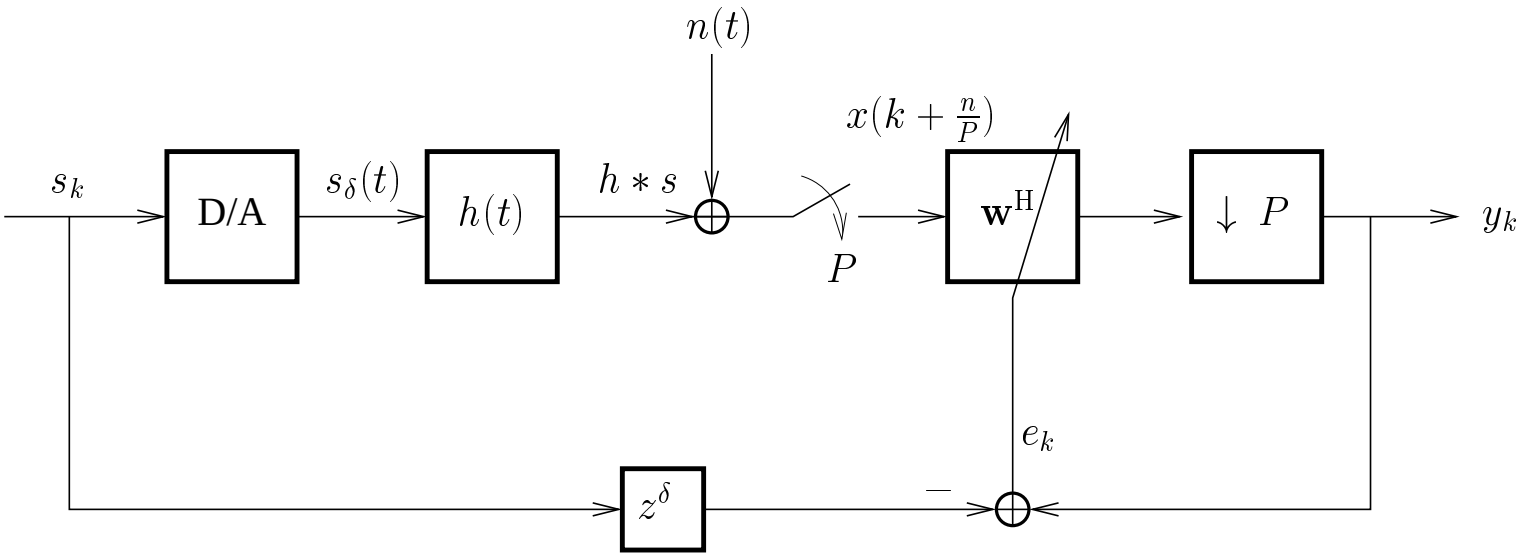
$$\hat{\mathbf{a}}^{(k+1)} = \hat{\mathbf{a}}^{(k)} - \mu(\hat{\mathbf{a}}^{(k)} s_k \bar{s}_k - \mathbf{x}_k \bar{s}_k) = \hat{\mathbf{a}}^{(k)} + \mu \mathbf{e}_k \bar{s}_k$$

Adaptive Model matching

Application to interference cancellation in radio astronomy



Adaptive equalization



$$\mathcal{X} = \begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{N-1} \\ \mathbf{x}_{-1} & \mathbf{x}_0 & \cdots & \mathbf{x}_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_{-m+1} & \mathbf{x}_{-m+2} & \cdots & \mathbf{x}_{N-m} \end{bmatrix}$$

$$:= \begin{bmatrix} \mathbf{H} & & \mathbf{0} \\ & \mathbf{H} & \\ & \ddots & \\ \mathbf{0} & & \mathbf{H} \end{bmatrix} \begin{bmatrix} s_0 & s_1 & \cdots & s_{N-1} \\ s_{-1} & s_0 & \cdots & s_{N-2} \\ \vdots & \vdots & \ddots & \vdots \\ s_{-L-m+1} & s_{-L-m+2} & \cdots & s_{N-L-m} \end{bmatrix} + \mathcal{N} = \mathcal{H}\mathcal{S} + \mathcal{N}$$

Adaptive equalization

- A general linear equalizer is:

$$\mathbf{y} = \mathbf{w}^{\mathrm{H}} \mathcal{X} = \mathbf{w}^{\mathrm{H}} (\mathcal{H}\mathcal{S} + \mathcal{N})$$

- If $\mathcal{H}$ is tall, then there are $L+m-1$ valid ZF equalizers

$$\mathbf{w}^{\mathrm{H}} \mathcal{H} = [1,\, 0,\, 0,\, \cdots,\, 0] \quad \text{or} \quad [0,\, 1,\, 0,\, \cdots,\, 0] \quad \text{or} \quad [0,\, 0,\, 1,\, \cdots,\, 0] \quad \text{or} \quad \cdots$$

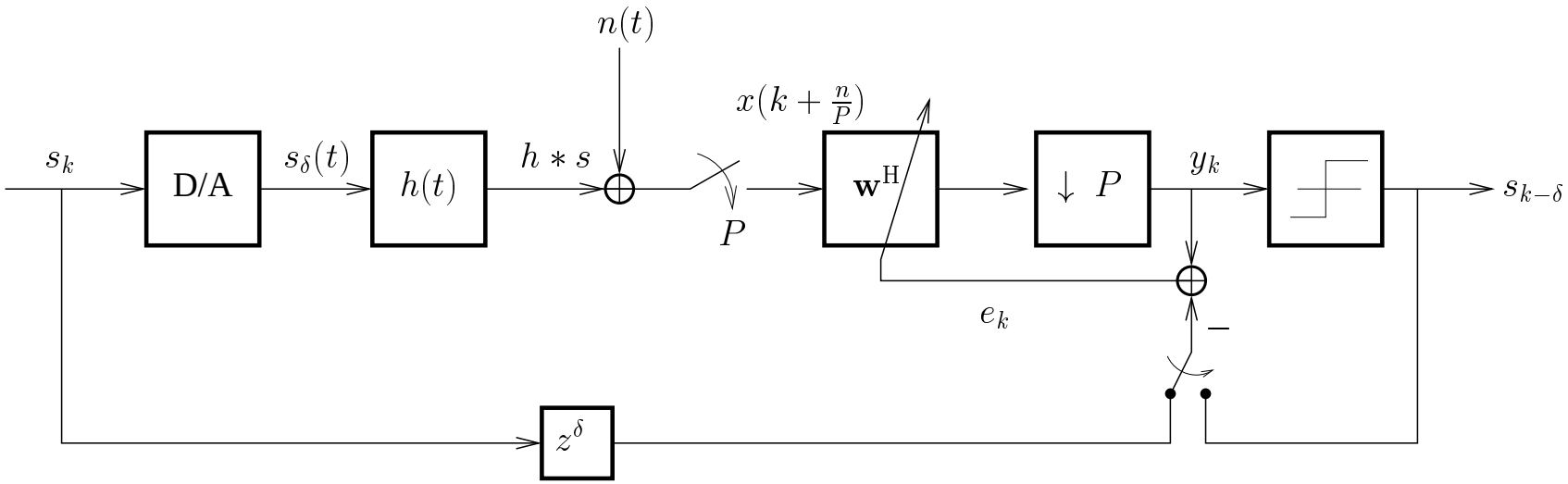
  Desired delay δ: usually the 'center tap': $\delta = \frac{1}{2}(L+m-1)$.

- For an adaptive filter, the reference signal is the original signal at delay δ.

- We can apply LMS, NLMS and RLS algorithms.

Adaptive equalization

Decision directed



- After training, we switch to use estimated symbols as reference signal
- Slowly varying channels can be tracked using decision directed updating

Application

Echo cancellation on telephone channels