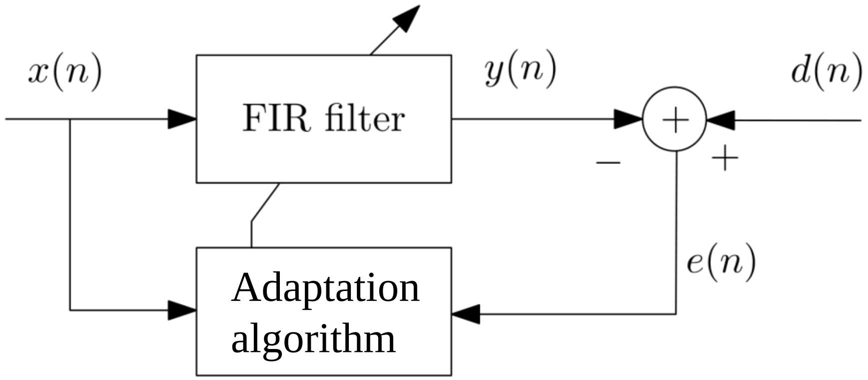
Adaptive filters

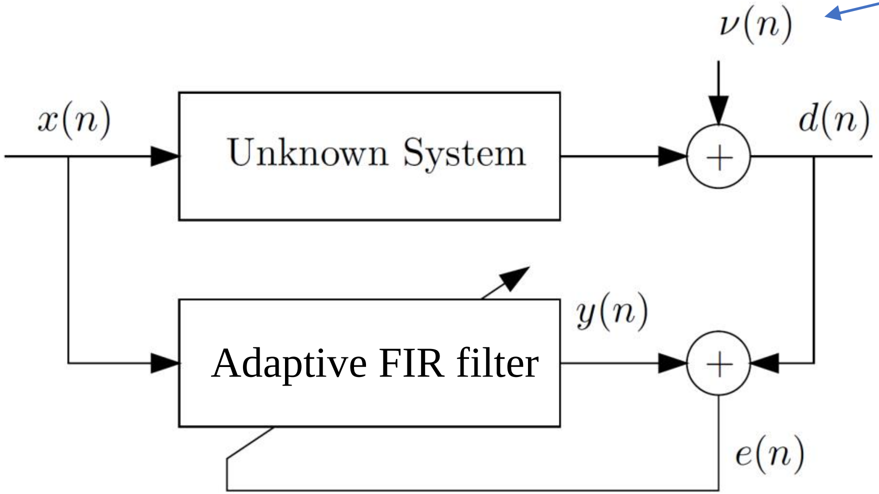
- In adaptive filters the coefficients *vary with time*: they are changed to make the filter comply with an *unknown and possibly time-varying* environment
- The optimization criterion typically is to minimize an *error* signal between the filter output and a *desired* signal: $e(n) = d(n) - y(n)$
- The desired signal $d(n)$ must be *correlated* with the input signal



Note: the *desired* signal is used to generate the error signal, but may not be the actual desired system output

Adaptive filters applications: system identification

- Adaptive control; layered earth modelling; vibration studies in mechanical systems

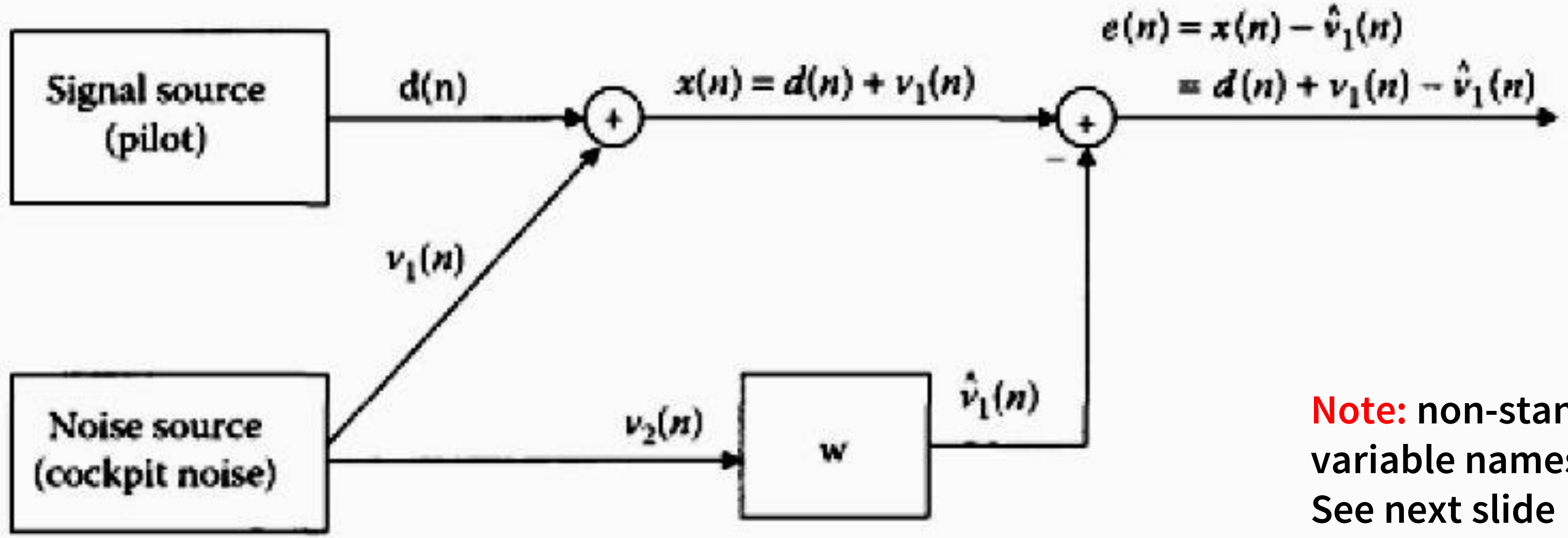


Note: like in the case of inverse systems, the possible presence of some *noise* should be taken into account

Note: the unknown system may be (slowly) time variant

Adaptive filters applications: adaptive noise cancellation (i)

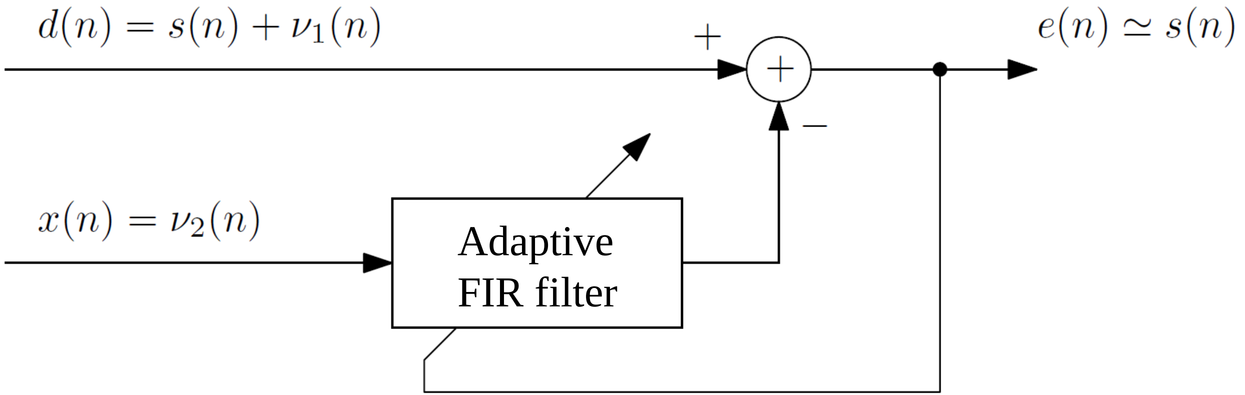
- Cockpit noise cancellation



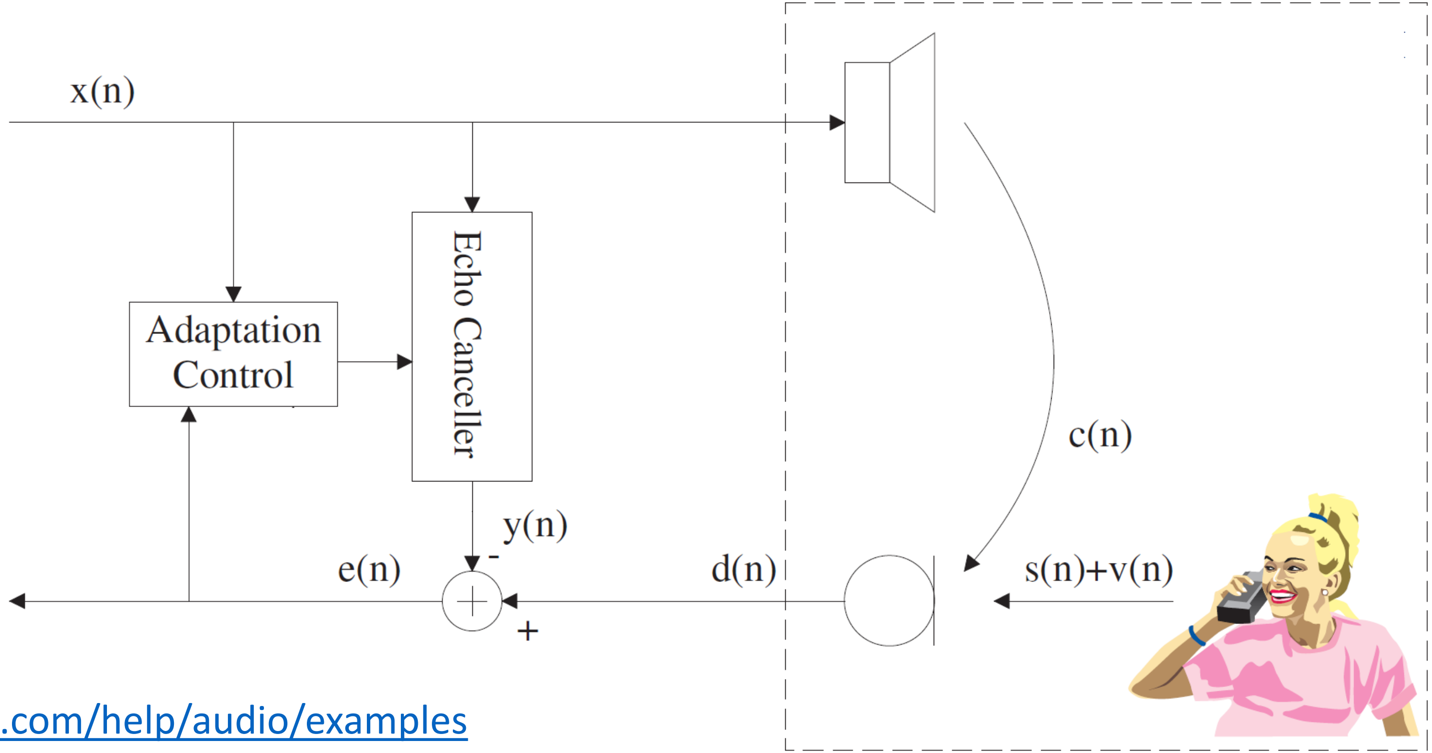
Note: non-standard variable names.
See next slide

Adaptive filters applications: adaptive noise cancellation (ii)

Beware: variable names are different from previous slide

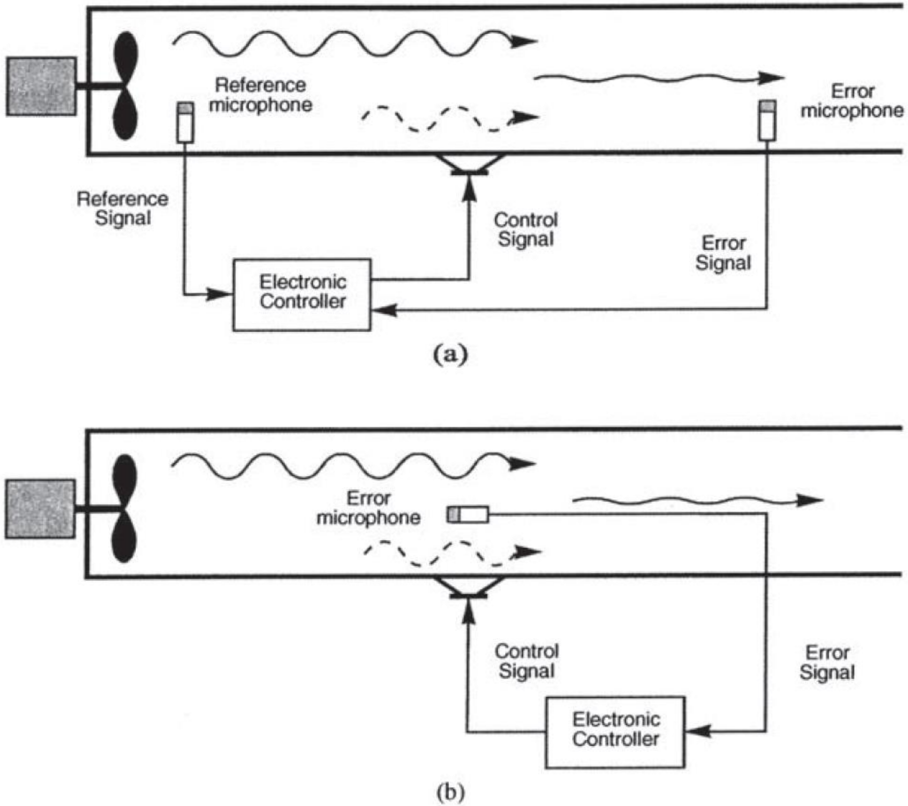


Adaptive filters applications: acoustic echo cancellation

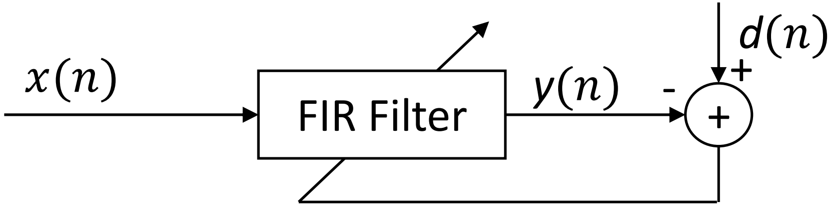


https://it.mathworks.com/help/audio/examples/acoustic-echo-cancellation-aec.html

Adaptive filters applications: active noise control



Wiener's optimal filter



- **It applies to stationary and zero mean signals $x(n)$ and $d(n)$.**
- It is based on the MMSE criterion, i.e. it provides the **static** filter that minimizes

$$J = E[e^2(n)] = E[\left(d(n) - y(n)\right)^2]$$

- Assume the filter be an FIR filter of length N and consider its vector form:

$$\boldsymbol{x}(n) = [x(n), x(n-1), \dots, x(n-N+1)]^T$$

$$\boldsymbol{h} = [h(0), h(1), \dots, h(N-1)]^T$$

$$y(n) = \boldsymbol{x}^T(n)\,\boldsymbol{h} = \boldsymbol{h}^T\,\boldsymbol{x}(n)$$

$$e(n) = d(n) - y(n)$$

Wiener's optimal filter

$$e^2(n) = d^2(n) - 2d(n)y(n) + y^2(n)$$

$$= d^2(n) - 2\boldsymbol{h}^T\boldsymbol{x}(n)d(n) + \boldsymbol{h}^T\boldsymbol{x}(n)\boldsymbol{x}^T(n)\boldsymbol{h}$$

$$E[e^2(n)] = E[d^2(n)] - 2\,\boldsymbol{h}^T E[\boldsymbol{x}(n)d(n)] + \boldsymbol{h}^T E[\boldsymbol{x}(n)\boldsymbol{x}^T(n)]\boldsymbol{h}$$

Note that $\boldsymbol{R}_{xx} = E[\boldsymbol{x}(n)\boldsymbol{x}^T(n)] =$

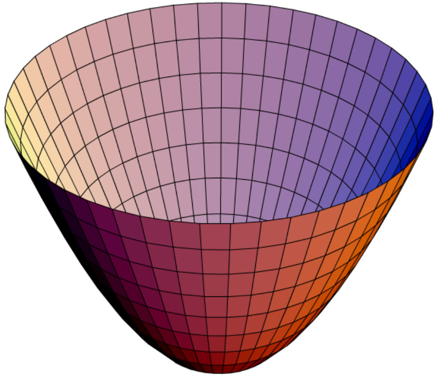
$$E\begin{bmatrix} x(n)x(n) & x(n)x(n-1) & \dots & x(n)x(n-N+1) \\ x(n-1)x(n) & x(n-1)x(n-1) & \dots & x(n-1)x(n-N+1) \\ \vdots & \vdots & & \vdots \\ x(n-N+1)x(n) & x(n-N+1)x(n-1) & \dots & x\hat{\ }2(n-N+1) \end{bmatrix}$$

is the ***input autocorrelation matrix***

Wiener's optimal filter

- $\boldsymbol{R}_{xd} = E[\boldsymbol{x}(n)d(n)] = E[x(n)d(n), \quad x(n-1)d(n), \quad \dots \quad x(n-N+1)d(n)]^T$ is the $(\boldsymbol{x}, d)$ *cross-correlation vector*
- i.e., $\quad E[e^2(n)] = E[d^2(n)] - 2\,\boldsymbol{h}^T\boldsymbol{R}_{xd} + \boldsymbol{h}^T\boldsymbol{R}_{xx}\boldsymbol{h}$
- Assuming $\boldsymbol{R}_{xx}$ and $\boldsymbol{R}_{xd}$ are known, we can minimize $E[e^2(n)]$ with respect to $\boldsymbol{h}$
- Note that $E[e^2(n)]$ is a quadratic function of the coefficients, i.e. it is a paraboloid whose minimum is the solution we seek

  → set the gradient of $E[e^2(n)]$ to zero



Wiener’s optimal filter

- $\nabla E[e^2(n)] = \left[\frac{\partial E[e^2(n)]}{\partial h(0)}, \frac{\partial E[e^2(n)]}{\partial h(1)}, \dots, \frac{\partial E[e^2(n)]}{\partial h(N-1)}\right]^T = 0$
- $\nabla E[e^2(n)] = -2\boldsymbol{R}_{xd} + 2\boldsymbol{R}_{xx}\,\boldsymbol{h}_{\text{opt}} = 0$
- i.e., we have

$$\boxed{\boldsymbol{h}_{\text{opt}} = \boldsymbol{R}_{xx}^{-1}\ \boldsymbol{R}_{xd}}$$

  which is the *Wiener-Hopf equation*
- If we knew the signals statistics we could compute the Wiener optimal filter
- Unfortunately, such statistics are often unknown; moreover, the processes can be nonstationary
- We need to estimate the signal statistics and update the optimal filter while we observe the signal

Least mean square (LMS) adaptive filter

- Sample by sample, it provides an estimate of the optimal filter; for stationary signals it is proved to *converge* to the Wiener filter
- It is the simplest and most diffused adaptive filter
- Remember that $E[e^2(n)]$ is a paraboloid in the coefficients. The optimal filter corresponds to the minimum of the paraboloid
- The LMS algorithm moves the coefficients along the maximum gradient slope in order to reach the minimum after a number of steps
- If the signals are not stationary, the *shape* and the *position* of the paraboloid change. The LMS algorithm tracks the paraboloid minimum, always moving along the direction of maximum slope

Least mean square (LMS) adaptive filter

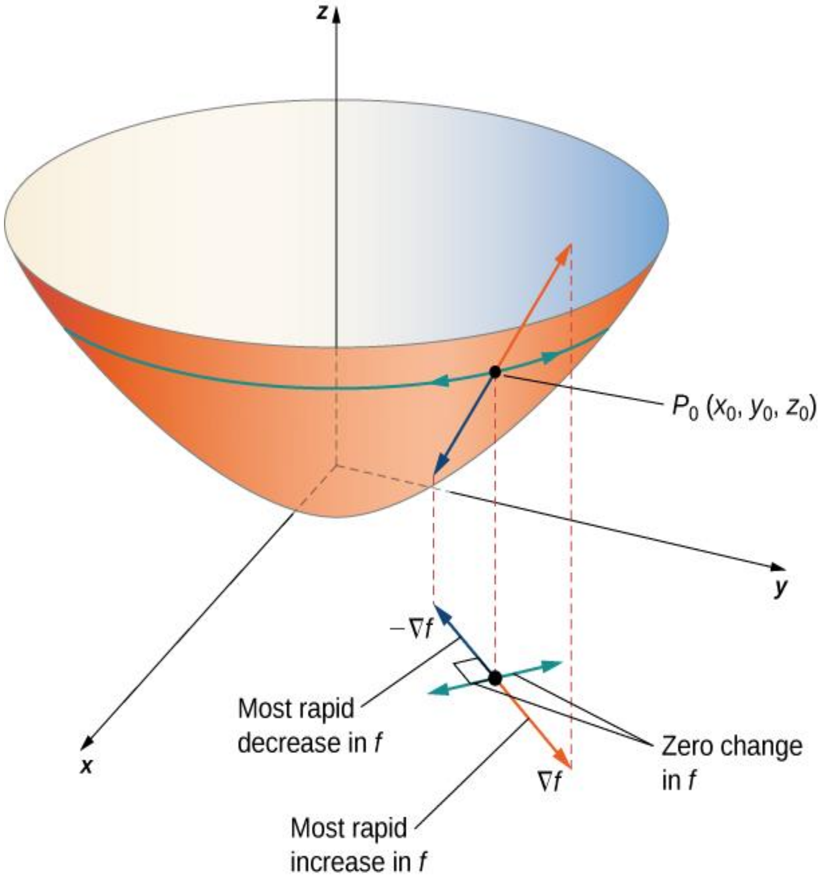
- Coeffs. should move in the gradient's direction, opposite verse:

$$\boldsymbol{h}_{n+1} = \boldsymbol{h}_n - \frac{\mu}{2} \nabla_{\boldsymbol{h}} E[e^2(n)]$$

- Widrow and Hoff proposed a drastic approximation:

$$E[e^2(n)] \simeq e^2(n)$$

- $\nabla_{\boldsymbol{h}} e^2(n) = 2\, e(n)\, \nabla_{\boldsymbol{h}}\, e(n)$
- $\nabla_{\boldsymbol{h}}\, e(n) = \nabla_{\boldsymbol{h}}\big(d(n) - \boldsymbol{h}^T \boldsymbol{x}(n)\big) = -\boldsymbol{x}(n)$



Least mean square (LMS) adaptive filter

*Among the earliest contributions (1960) to **machine learning***

$$\boxed{\begin{aligned} \boldsymbol{h}_{n+1} &= \boldsymbol{h}_n + \mu\, e(n)\, \boldsymbol{x}(n) \\ e(n) &= d(n) - \boldsymbol{h}_n^T\, \boldsymbol{x}(n) \end{aligned}}$$

μ is called *adaptation constant* or *step size*

- **Low computational cost ($2N$ multiplications and $2N$ additions per iteration)**
- **For stationary signals, is proved to converge to the optimal filter provided**

$$0 < \mu < \frac{2}{\lambda_{\max}}\ ,$$ **where $\lambda_{\max}$ is the largest eigenvalue of $\boldsymbol{R}_{xx}$.**

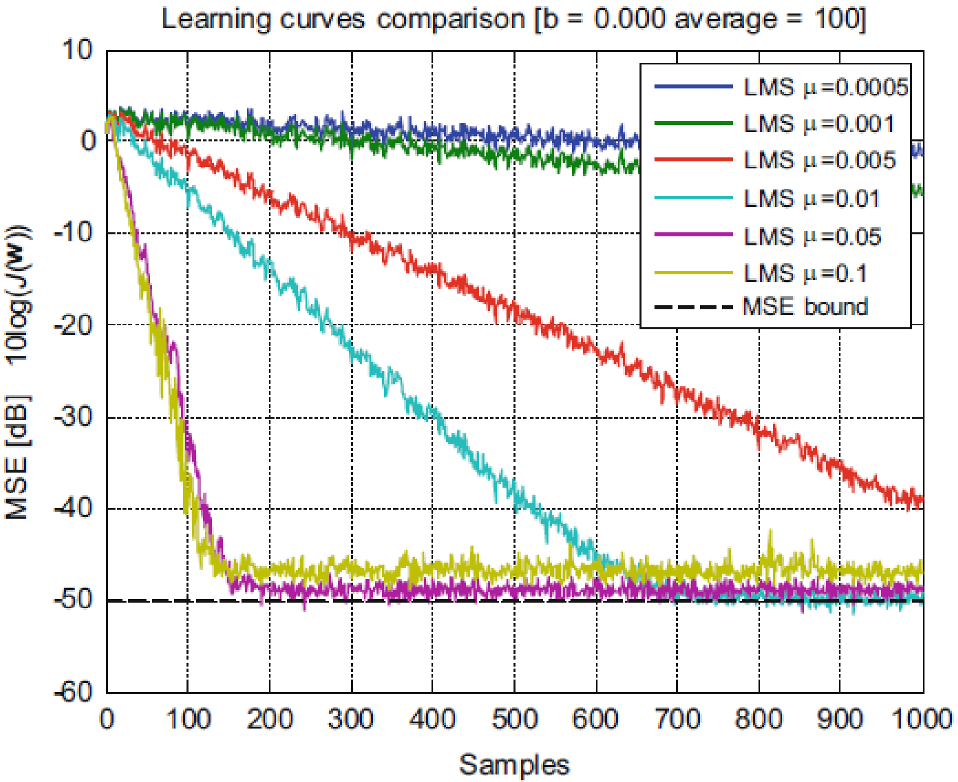
- **Since $\lambda_{\max} \leq \mathrm{Trace}(\boldsymbol{R}_{xx})$, a sufficient condition for convergence is**

$$0 < \mu < \frac{2}{\sum_{i=0}^{N-1} E[x^2(n-i)]}$$

which is easier to approximate

Least mean square (LMS) adaptive filter

- *learning curves* for different step sizes μ (input: *white* noise) →
- Large μ yields fast convergence but large oscillations around the optimal solution
- The lowest MSE bound depends on the ability of the LMS filter to cope with the unknown system (length *N*=?) and on the presence of noise

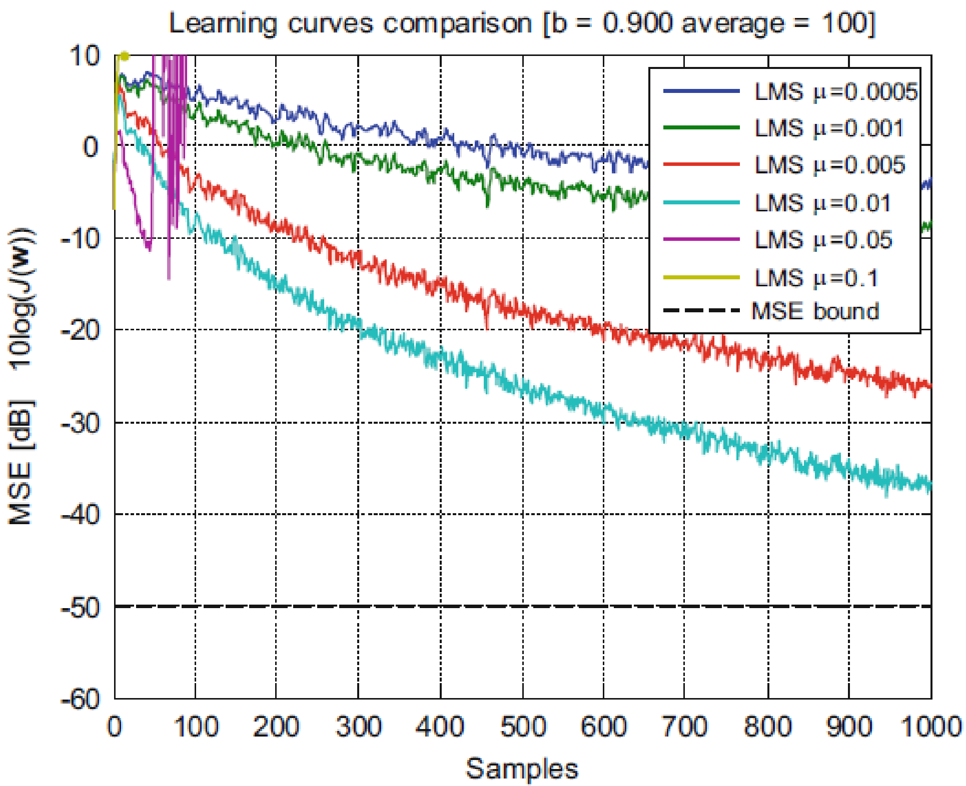


From A. Uncini "Fundamental of adaptive signal processing", Springer, 2015

Least mean square (LMS) adaptive filter

- **Drawback:** **low convergence speed if input signal has high autocorrelation**

- **With a narrowband moving average *colored* input:** →



From A. Uncini "Fundamental of adaptive signal processing", Springer, 2015

Normalized LMS (NLMS) adaptive filter

- **If in the LMS adaptation**

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

the amplitude of $x(n)$ reduces by a factor $\boldsymbol{A}$, so does $e(n)$.

→ both $\Delta\boldsymbol{h}_n = \boldsymbol{h}_{n+1} - \boldsymbol{h}_n$ and the convergence speed reduce by $\boldsymbol{A}^2$

- **It is convenient to modify the adaptation rule:**

$$\boldsymbol{h}_{n+1} = \boldsymbol{h}_n + \frac{\mu}{\boldsymbol{x}^T(n)\, \boldsymbol{x}(n) + \delta}\, e(n)\, \boldsymbol{x}(n)$$

- **Where δ is a small positive constant used to avoid divisions by zero**
- **The convergence speed becomes independent of the input signal amplitude**
- **The algorithm converges for any value of the step size μ such that $0 < \mu < 2$**

Normalized LMS (NLMS) adaptive filter

- **The NLMS algorithm**

$$\boxed{\begin{aligned} e(n) &= d(n) - \boldsymbol{h}_n^T\,\boldsymbol{x}(n) \\ \boldsymbol{h}_{n+1} &= \boldsymbol{h}_n + \frac{\mu}{\boldsymbol{x}^T(n)\,\boldsymbol{x}(n) + \delta}\,e(n)\,\boldsymbol{x}(n) \end{aligned}}$$

- **is the exact solution of the following optimization criterion:**

Minimize the Euclidean norm of the coefficient variation $\Delta\boldsymbol{h}_n = \boldsymbol{h}_{n+1} - \boldsymbol{h}_n$ *imposing the* a posteriori *error* $e_{n+1}(n) = d(n) - \boldsymbol{h}_{n+1}^T\,\boldsymbol{x}(n)$ *to be zero, i.e.,*

$$\min\{\Delta\boldsymbol{h}_n^T\,\Delta\boldsymbol{h}_n\} \quad \text{s.t.} \quad e_{n+1}(n) = d(n) - \boldsymbol{h}_{n+1}^T\,\boldsymbol{x}(n) = 0$$

- **The computational costs of NLMS and LMS are similar (**$2N$ **multiplications and** $2N$ **additions) and also their convergence speed is similar**
- **Like LMS, NLMS has a poor convergence speed with correlated signals**