# Revision of Lecture Sixteen

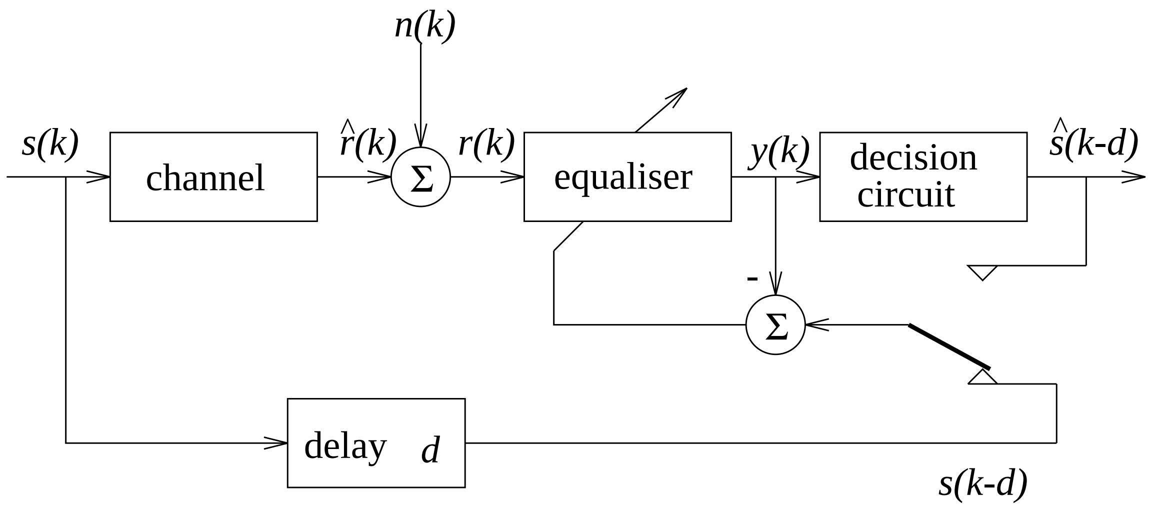
- Previous lecture introduces the **most important** adaptive filter design principle
  - Wiener filter or MMSE solution: design and analysis
  - **Stochastic gradient adaptive** LMS algorithm
- This lecture focuses on particular example of adaptive signal processing ⇒ adaptive equalisation

# Adaptive Equalisation

- Recall the framework of **adaptive equalisation** with two operation modes, training and decision-directed, where the channel model is:

$$r(k) = \sum_{i=0}^{n_C} c_i s(k-i) + n(k)$$

symbols are $N$-QAM $s(k) \in \{s_{i,l} = u_i + ju_l, 1 \le i, l \le \sqrt{N}\}$ with $u_i = 2i - \sqrt{N} - 1$, $u_l = 2l - \sqrt{N} - 1$, and AWGN $n(k)$: $E[|n(k)|^2] = 2\sigma_n^2$



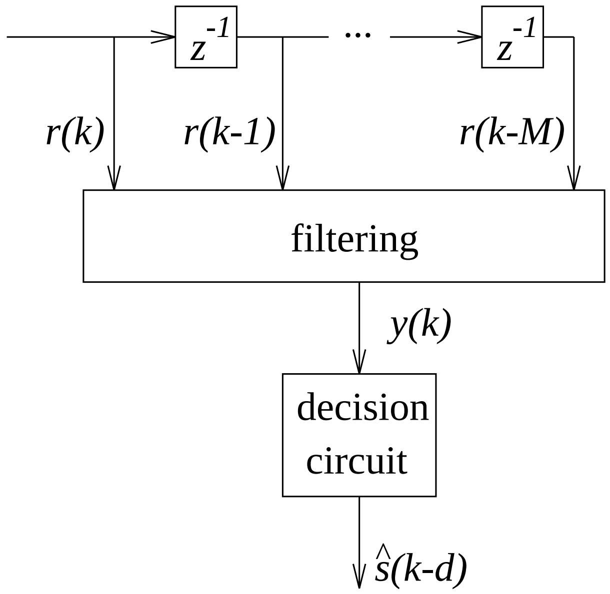

- We will first discuss **symbol-decision** equalisers and follow it by an introduction to the **MLSE**
- An equaliser decision delay $d$ is necessary for coping with non-minimum phase channels
- The zero-forcing equaliser $H_E(z)$ inverses the channel $H_C(z)$: $H_C(z)H_E(z) \approx z^{-d}$
- Solving this gives the linear equaliser's weights. Although this zero-forcing equaliser completely eliminates ISI, it suffers from a serious noise enhancement problem
- The most popular designs are the linear equaliser and decision feedback equaliser based on the **mean square error criterion**

# Linear Transversal Equaliser

- The **linear equaliser** is given by:

$$y(k) = \sum_{i=0}^{M} w_i^* r(k-i) = \mathbf{w}^H \mathbf{r}(k)$$

where $\mathbf{r}(k) = [r(k) \cdots r(k-M)]^T$ and $M$ is the equaliser order




- Typical design is based on mean square error with the **MMSE** solution: $\hat{\mathbf{w}} = \mathbf{R}^{-1}\mathbf{p}$, where $\mathbf{R} = \mathrm{E}[\mathbf{r}(k)\mathbf{r}^H(k)]$, $\mathbf{p} = \mathrm{E}[\mathbf{r}(k)s^*(k-d)]$ and $d$ is decision delay

- Adaptive implementation typically adopts the **LMS**:

$$\tilde{\mathbf{w}}(k+1) = \tilde{\mathbf{w}}(k) + \mu \mathbf{r}(k) e^*(k) \quad \text{with} \quad e(k) = \begin{cases} y(k) - s(k-d), & \text{training} \\ y(k) - \hat{s}(k-d), & \text{decision-directed} \end{cases}$$

# Closed-Form MMSE Solution

- **Equaliser input vector**: $\mathbf{r}(k) = [r(k)\ r(k-1) \cdots r(k-M)]^T = \mathbf{C}\mathbf{s}(k) + \mathbf{n}(k)$, with channel matrix having Toeplitz form

$$\mathbf{C} = \begin{bmatrix} c_0 & c_1 & \cdots & c_{n_c} & 0 & \cdots & 0 \\ 0 & c_0 & c_1 & \cdots & c_{n_c} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & c_0 & c_1 & \cdots & c_{n_c} \end{bmatrix} = [\mathbf{c}_0\ \mathbf{c}_1 \cdots \mathbf{c}_d \cdots \mathbf{c}_{M+n_c}]$$

$\mathbf{s}(k) = [s(k)\ s(k-1) \cdots s(k-M-n_c)]^T$, $\mathbf{n}(k) = [n(k)\ n(k-1) \cdots n(k-M)]^T$

- **Equaliser output**

$$y(k) = \mathbf{w}^H \mathbf{r}(k)$$

and MSE criterion

$$J(\mathbf{w}) = E\left[ |s(k-d) - y(k)|^2 \right]$$

- **MMSE solution** for equaliser weight vector

$$\frac{\partial J}{\partial \mathbf{w}} = 0 \ \rightarrow \hat{\mathbf{w}} = \left( \mathbf{C}\mathbf{C}^H + \frac{2\sigma_n^2}{\sigma_s^2}\mathbf{I} \right)^{-1} \mathbf{c}_d$$

with $2\sigma_n^2 = E[|n(k)|^2]$, $\sigma_s^2 = E[|s(k)|^2]$, $\mathbf{I}$ is $(M + n_c + 1) \times (M + n_c + 1)$ identity matrix

# Choice of Equaliser Length and Decision Delay

- To eliminate ISI, the equaliser

$$H_E(z) = \sum_{i=0}^{M} w_i^* z^{-i}$$

  should be chosen such that $H_C(z)H_E(z) \approx z^{-d}$, but this requires a long equaliser $\rightarrow$ serious noise enhancement, as the noise variance at equaliser output is

$$\mathrm{E}\left[\left|\sum_{i=0}^{M} w_i^* n(k-i)\right|^2\right] = \left(\sum_{i=0}^{M} |w_i|^2\right) 2\sigma_n^2$$

  The larger $M$, the larger noise variance at $y(k)$

- LTE must compromise between eliminating ISI and not enhancing noise too much

- Given $M$, the optimal $d$ in the MSE sense depends on the channel $H_E(z)$

  A simple rule is to choose $d \approx \frac{M}{2}$

# Example

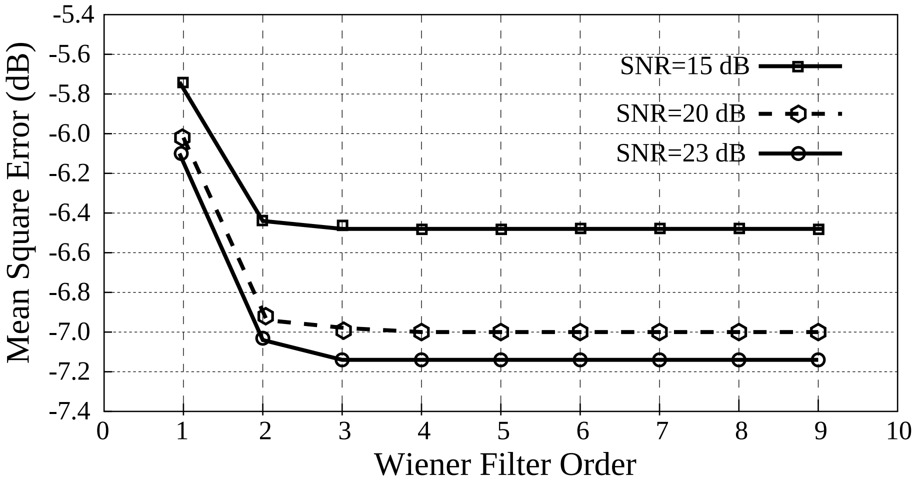
2-ary symbols with the channel $H_C(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}$ and the equaliser delay $d = 1$

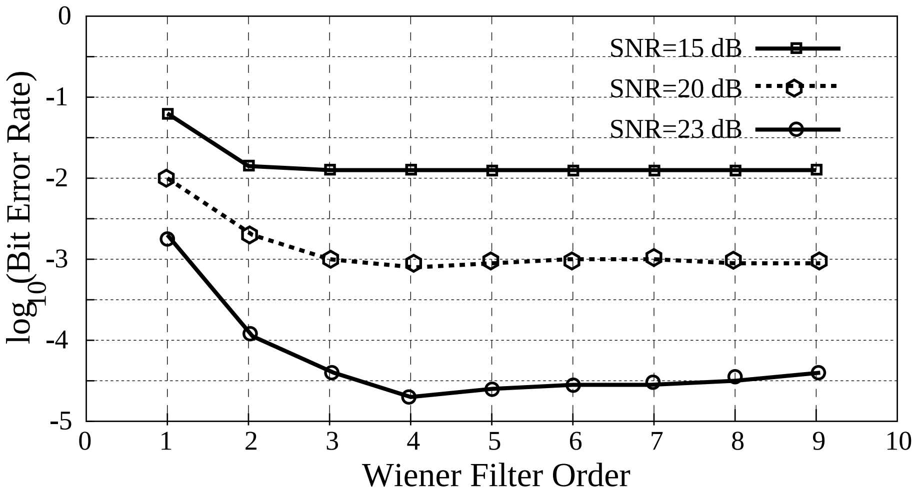
The MSE versus the Wiener filter order $M + 1$ and the BER versus $M + 1$ are shown

The results are better for $d = 2$ but the trends are identical to those shown here

Clearly the noise enhancement severely limits the performance of the LTE

There is no point to increase $M$ beyond certain value, as noise enhancement offsets the benefit



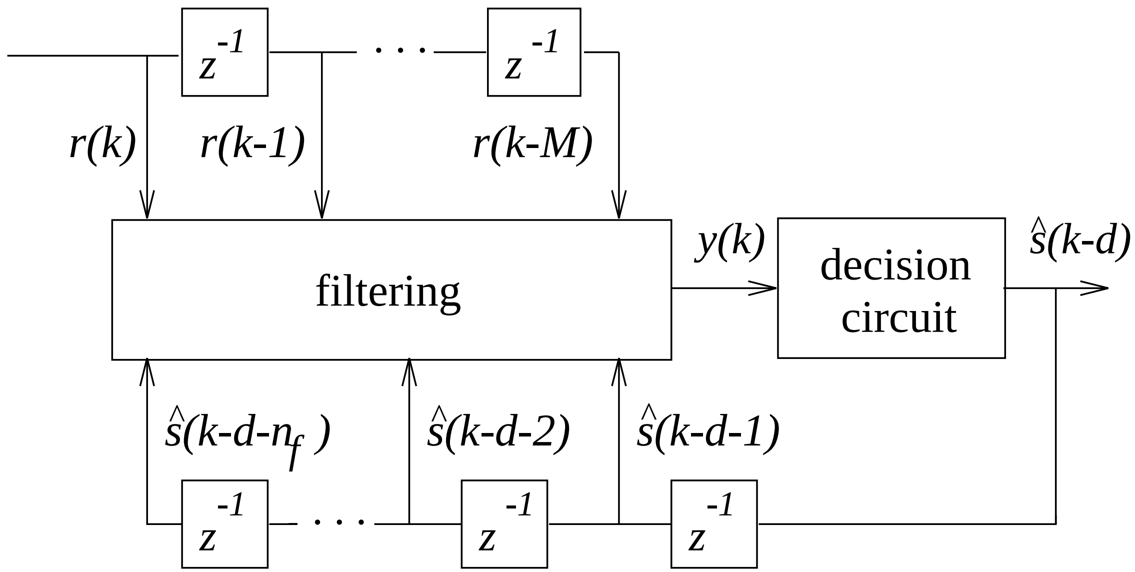
# Decision Feedback Equaliser

- The DFE consists of a feedforward filter and a feedback filter:

$$y(k) = \mathbf{w}^H \mathbf{r}(k) + \mathbf{b}^H \hat{\mathbf{s}}(k-d)$$

$$= \sum_{i=0}^{M} w_i^* r(k-i) + \sum_{i=1}^{n_f} b_i^* \hat{s}(k-d-i)$$

The DFE generally outperforms the LTE in terms MSE and BER




- Assuming equaliser decisions $\hat{\mathbf{s}}(k-d)$ are correct, the feedback filter $\mathbf{b}^H \hat{\mathbf{s}}(k-d)$ **eliminates a large proportion of ISI without enhancing noise** and the feedforward filter $\mathbf{w}^H \mathbf{r}(k)$ takes care the remaining ISI
- **Error propagation**. Occasionally error occurs in symbol detection, i.e. $\hat{s}(k-d) \neq s(k-d)$, it is fed back and will affect subsequent symbol detections $\rightarrow$ further burst errors
- **Choice of structure parameters**. There is an optimal choice of $M$, $n_f$ and $d$ in MMSE sense, which depends on CIR and is difficult to determine

  A simple practical rule: feedforward filter covers entire channel dispersion, i.e. $M = n_c$; decision delay is set to $d = n_c$; and feedback filter order $n_f = n_c$

# MMSE DFE Design

- Define

$$\mathbf{a} = \begin{bmatrix} \mathbf{w} \\ \mathbf{b} \end{bmatrix}, \ \mathbf{u}(k) = \begin{bmatrix} \mathbf{r}(k) \\ \mathbf{s}(k-d) \end{bmatrix} \ \text{and} \ \ y(k) = \mathbf{a}^H \mathbf{u}(k)$$

- The **Wiener solution** is then: $\hat{\mathbf{a}} = \mathbf{R}^{-1}\mathbf{p}$ with $\mathbf{R} = \mathrm{E}[\mathbf{u}(k)\mathbf{u}^H(k)]$ and $\mathbf{p} = \mathrm{E}[\mathbf{u}(k)s^*(k-d)]$

- Adaptive implementation typically adopts the **LMS**:

$$\tilde{\mathbf{a}}(k+1) = \tilde{\mathbf{a}}(k) + \mu \mathbf{u}(k) e^*(k)$$

In training mode:

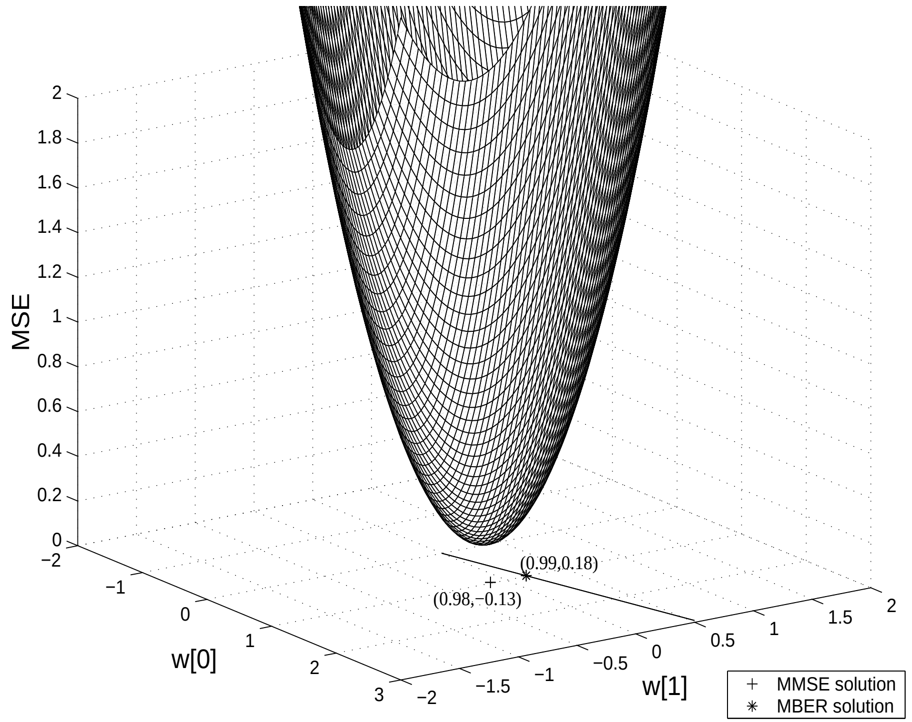
$$\mathbf{u}(k) = \begin{bmatrix} \mathbf{r}(k) \\ \mathbf{s}(k-d) \end{bmatrix}, \quad e(k) = s(k-d) - y(k)$$

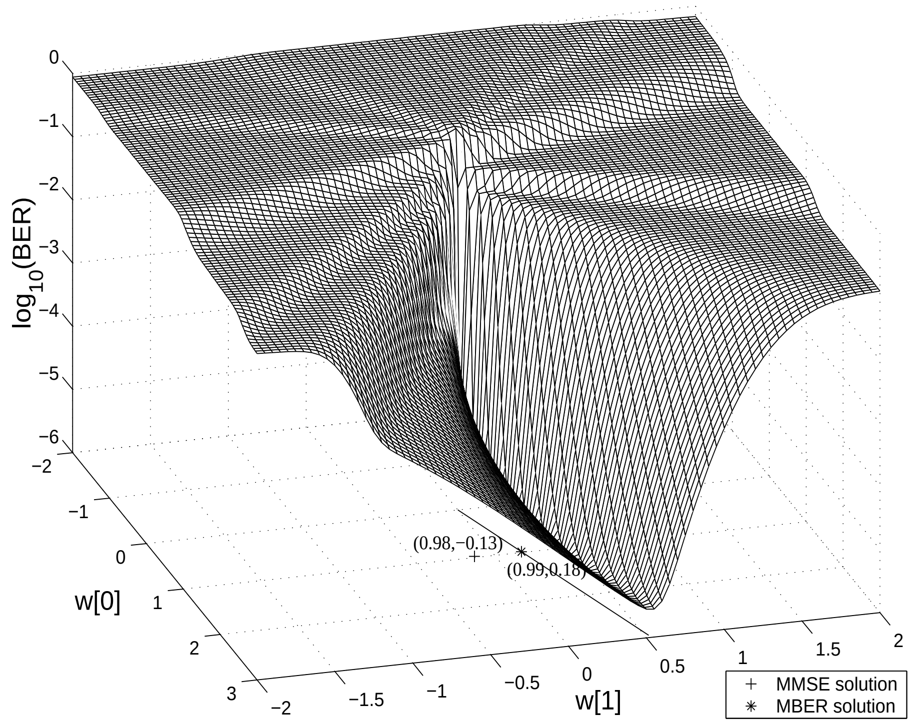
In decision-directed mode:

$$\mathbf{u}(k) = \begin{bmatrix} \mathbf{r}(k) \\ \hat{\mathbf{s}}(k-d) \end{bmatrix}, \quad e(k) = \hat{s}(k-d) - y(k)$$

# Minimum Bit Error Rate Design

- The real goal of equalisation is not the MSE but the **bit error rate** and, for linear equaliser and DFE, the MMSE solution is not necessarily the MBER solution
- The **MMSE design** is typically chosen because it leads to simple and effective adaptive implementation, e.g. the LMS, and it is also rooted in traditional adaptive filtering
- Example: a case of two-tap equaliser for BPSK, where the MMSE solution has $\log_{10}(\text{BER}) = -3.88$ but the MBER solution has $\log_{10}(\text{BER}) = -5.56$



# Equaliser Bit Error Rate

- For simplicity, consider the BPSK linear equaliser, where the decision rule is $\hat{s}(k-d) = \text{sgn}(y(k))$
- Note the received signal $r(k) = c_0 s(k) + \cdots + c_{n_c} s(k-n_c) + n(k) = \bar{r}(k) + n(k)$, or

$$\mathbf{r}(k) = \bar{\mathbf{r}}(k) + \mathbf{n}(k) = \begin{bmatrix} c_0 & c_1 & \cdots & c_{n_c} & 0 & \cdots & 0 \\ 0 & c_0 & c_1 & \cdots & c_{n_c} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & \ddots & 0 \\ 0 & \cdots & 0 & c_0 & c_1 & \cdots & c_{n_c} \end{bmatrix} \mathbf{s}(k) + \mathbf{n}(k) = \mathbf{C}\mathbf{s}(k) + \mathbf{n}(k)$$

  where $\mathbf{s}(k) = [s(k)\ s(k-1) \cdots s(k-M-n_c)]^T$ has $N_s = 2^{M+n_c+1}$ combinations, denoted as $\mathbf{s}_j$, $1 \le j \le N_s$, with the $d$th element of $\mathbf{s}_j$ being $s_j^{(d)}$
- Obviously $\bar{\mathbf{r}}(k)$ can only take values from the finite channel state set:

$$\bar{\mathbf{r}}(k) \in \{\mathbf{r}_j = \mathbf{C}\mathbf{s}_j,\ 1 \le j \le N_s\}$$

- Define the signed decision variable $y_s(k) = \text{sgn}(s(k-d))y(k)$, then

$$y_s(k) = \text{sgn}(s(k-d))\bar{y}(k) + e(k)$$

  where $e(k) = \text{sgn}(s(k-d))\mathbf{w}^T\mathbf{n}(k)$ is Gaussian with variance $\mathbf{w}^T\mathbf{w}\sigma_n^2$, and $\bar{y}(k)$ can only take values from the set: $\bar{y}(k) \in \{y_j = \mathbf{w}^T\mathbf{r}_j,\ 1 \le j \le N_s\}$

# Minimum Bit Error Rate Solution

- The **PDF** of the signed decision variable $y_s(k)$ is a Gaussian mixture

$$p_y(y_s) = \frac{1}{N_s\sqrt{2\pi}\sigma_n\sqrt{\mathbf{w}^T\mathbf{w}}} \sum_{i=1}^{N_s} \exp\left(-\frac{(y_s - \text{sgn}(s_i^{(d)})y_i)^2}{2\sigma_n^2\mathbf{w}^T\mathbf{w}}\right)$$

- The **BER** of the linear equaliser can be shown to be:

$$P_E(\mathbf{w}) = \int_{-\infty}^{0} p_y(y_s)dy_s = \frac{1}{N_s}\sum_{i=1}^{N_s} Q(g_i(\mathbf{w})) \quad \text{with} \quad g_i(\mathbf{w}) = \frac{\text{sgn}(s_i^{(d)})y_i}{\sigma_n\sqrt{\mathbf{w}^T\mathbf{w}}}$$

- The **MBER solution** is defined as

$$\mathbf{w}_{\text{MBER}} = \arg\min_{\mathbf{w}} P_E(\mathbf{w})$$

  Note that the BER is invariant to a positive scaling of $\mathbf{w}$, and there are infinite many $\mathbf{w}_{\text{MBER}}$

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

$$\nabla P_E(\mathbf{w}) = \frac{1}{N_s\sqrt{2\pi}\sigma_n}\left(\frac{\mathbf{w}\mathbf{w}^T - \mathbf{w}^T\mathbf{w}\mathbf{I}}{(\mathbf{w}^T\mathbf{w})^{\frac{3}{2}}}\right)\sum_{j=1}^{N_s}\exp\left(-\frac{y_j^2}{2\sigma_n^2\mathbf{w}^T\mathbf{w}}\right)\text{sgn}(s_j^{(d)})\mathbf{r}_j$$

  The steepest descent algorithm for example can be used to find a $\mathbf{w}_{\text{MBER}}$

# Least Bit Error Rate Algorithm

- The key in deriving the MBER solution is the PDF $p_y(y_s)$ and, since $p_y(y_s)$ is unavailable, using a sample time average, called the Parzen window or kernel density estimate, to estimate $p_y(y_s)$
- Given $\{\mathbf{r}(k), s(k-d)\}_{k=1}^{K}$, a **Parzen window estimate** of $p_y(y_s)$ is

$$\hat{p}_y(y_s) = \frac{1}{K\sqrt{2\pi}\rho_n} \sum_{k=1}^{K} \exp\left(-\frac{(y_s - \mathrm{sgn}(s(k-d))y(k))^2}{2\rho_n^2}\right)$$

- Like in the derivation of the LMS, take to the extreme and use one-sample estimate:

$$\hat{p}_y(y_s;k) = \frac{1}{\sqrt{2\pi}\rho_n} \exp\left(-\frac{(y_s - \mathrm{sgn}(s(k-d))y(k))^2}{2\rho_n^2}\right)$$

- Using the instantaneous or **stochastic gradient**

$$\nabla \hat{P}_E(k) = -\frac{1}{\sqrt{2\pi}\rho_n} \exp\left(-\frac{y^2(k)}{2\rho_n^2}\right) \mathrm{sgn}(s(k-d))\mathbf{r}(k)$$

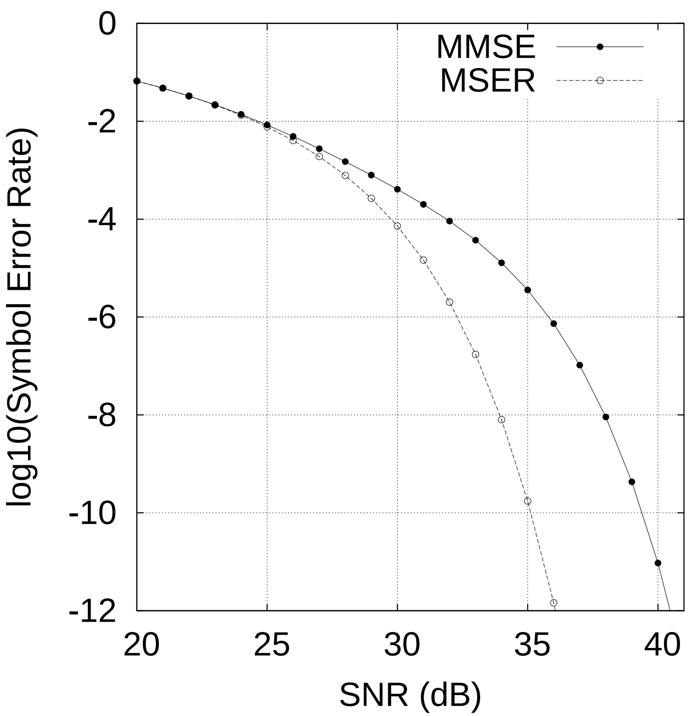
leads to the **LBER algorithm**:

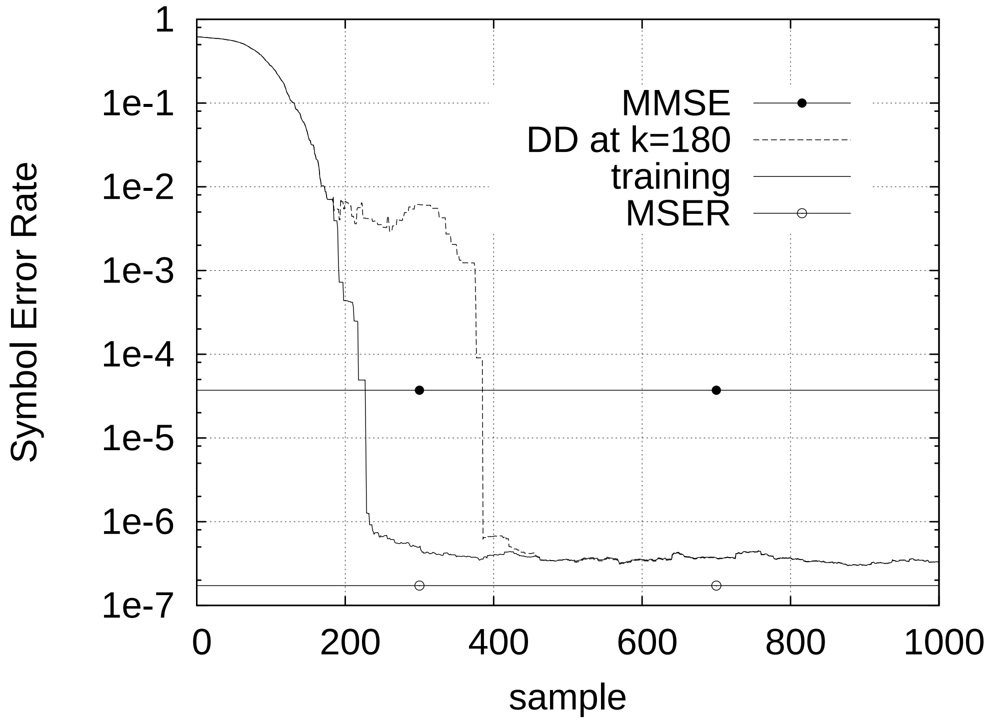
$$\mathbf{w}(k+1) = \mathbf{w}(k) + \mu \frac{\mathrm{sgn}(s(k-d))}{\sqrt{2\pi}\rho_n} \exp\left(-\frac{y^2(k)}{2\rho_n^2}\right) \mathbf{r}(k)$$

where $\mu$ and $\rho_n$ are adaptive gain and width

# Extension to Minimum Symbol Error Rate

- The approach is equally applicable to the decision feedback equaliser
- The approach can be extended to higher-order QAM case: MSER and LSER
- Example: 8-ary with the channel $H_C(z) = 0.3 + 1.0z^{-1} - 0.3z^{-2}$ and DFE



# Summary

- Adaptive equalisation: symbol-decision and sequence-decision, channel model ISI, two adaptive operation modes, and why need decision delay
- Linear transversal equaliser: filter model, compromise between eliminate ISI and enhance noise, design based on MMSE and adaptive implementation using the LMS
- Decision feedback equaliser: filter model, how it overcomes noise enhancement but may suffer from error propagation, design based on MMSE and adaptive implementation using the LMS
- Adaptive minimum bit error equaliser: design based on MBER and adaptive implementation using the LBER, extension to the MSER design and LSER algorithm