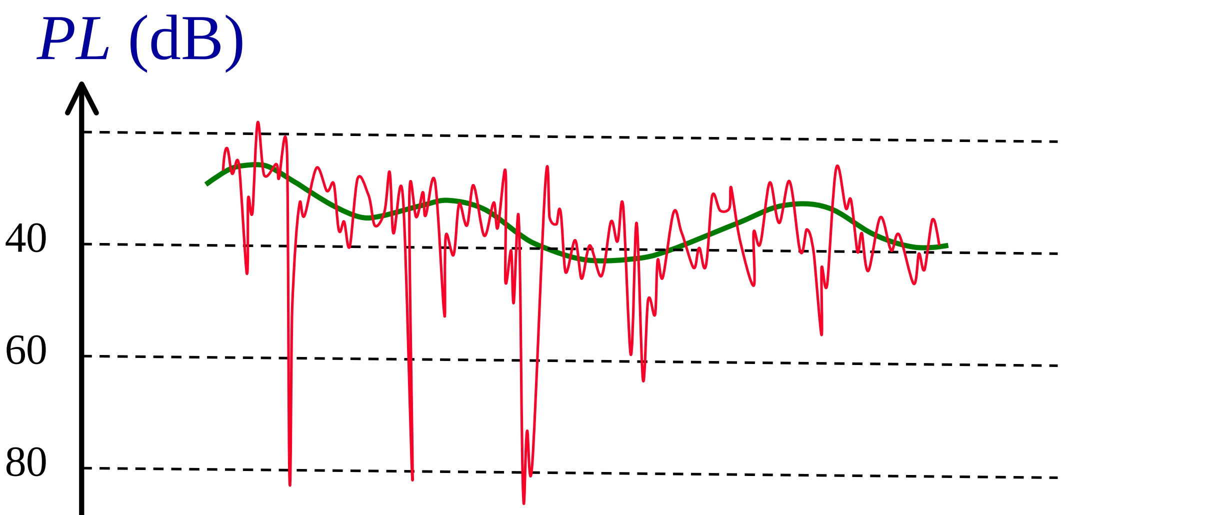
Mobile Radio Communications

Session 5: Equalization, coding & diversity

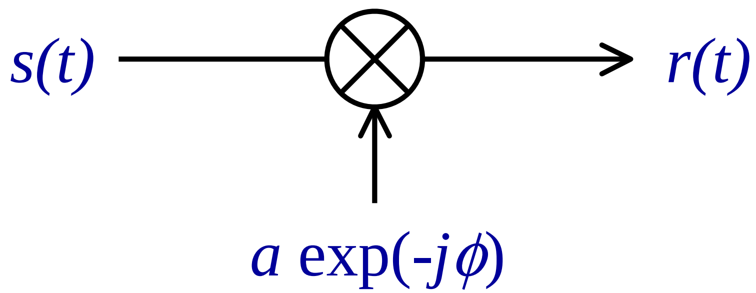
Non-stationary propagation channel



- **slow & fast fading**
- **flat or frequency-selective fading**

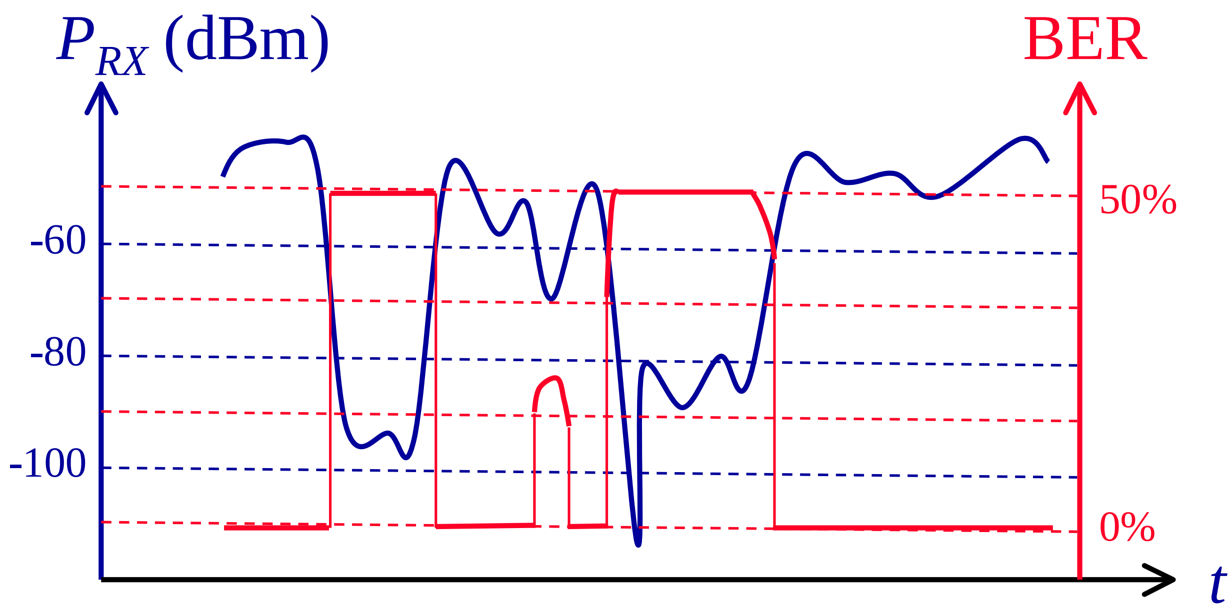
Slow/flat fading channel

- **attenuation/phase shift function of *t*, but constant over T_s**
- **SNR, BER function of *t***



$$r(t) = a(t)\exp(-j\phi(t)) \cdot s(t) + n(t) \qquad 0 \le t \le T_s$$

Non-stationary propagation channel



- **BER is function of E_b/N_0**

Error probability in slow/flat fading

$$P_e = \int_0^{\infty} P_e(t)dt = \int_0^{\infty} P_e(X)p(X)dX$$

- **with** $X = \alpha^2 E_b / N_0$ **or SNR**
- α **is fading gain**
- ***p(X)*** **is probability density of *X***

Fading distribution

- **amplitude α Rayleigh distributed**

$$p(\alpha) = \frac{\pi}{2} \cdot \frac{\alpha}{\overline{\alpha}} \cdot \exp\left(-\frac{\pi}{4} \cdot \frac{\alpha^2}{\overline{\alpha}^2} \right)$$

- **power X exponentially distributed (Chi-square)**

$$p(X) = \frac{1}{\Gamma} \cdot \exp\left(-\frac{X}{\Gamma} \right)$$

$$\Gamma = \overline{\alpha^2}\, E_b / N_0$$

Error probability in slow/flat fading

$$P_e = \frac{1}{\Gamma}\int_0^{\infty} P_e(X)\cdot \exp\left(-\frac{X}{\Gamma}\right)dX$$

$P_e(X)$ depends on modulation scheme

Error probability in slow/flat fading

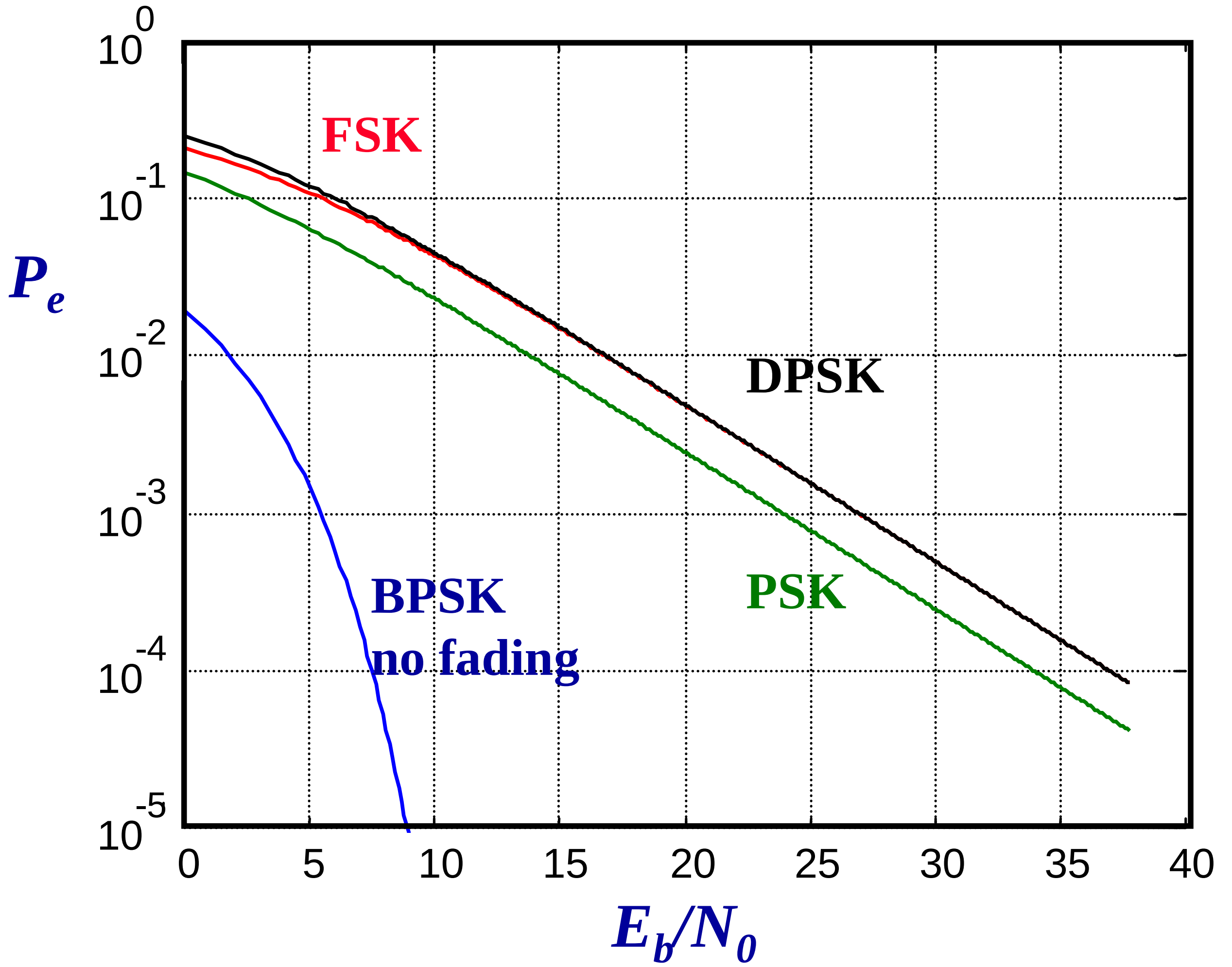
$$\Gamma = \overline{\alpha^2}\, E_b / N_0$$

$$P_{e,PSK} = \frac{1}{2}\left[1 - \sqrt{\frac{1}{1+\Gamma}}\right] \approx \frac{1}{4\Gamma}$$ coherent binary PSK

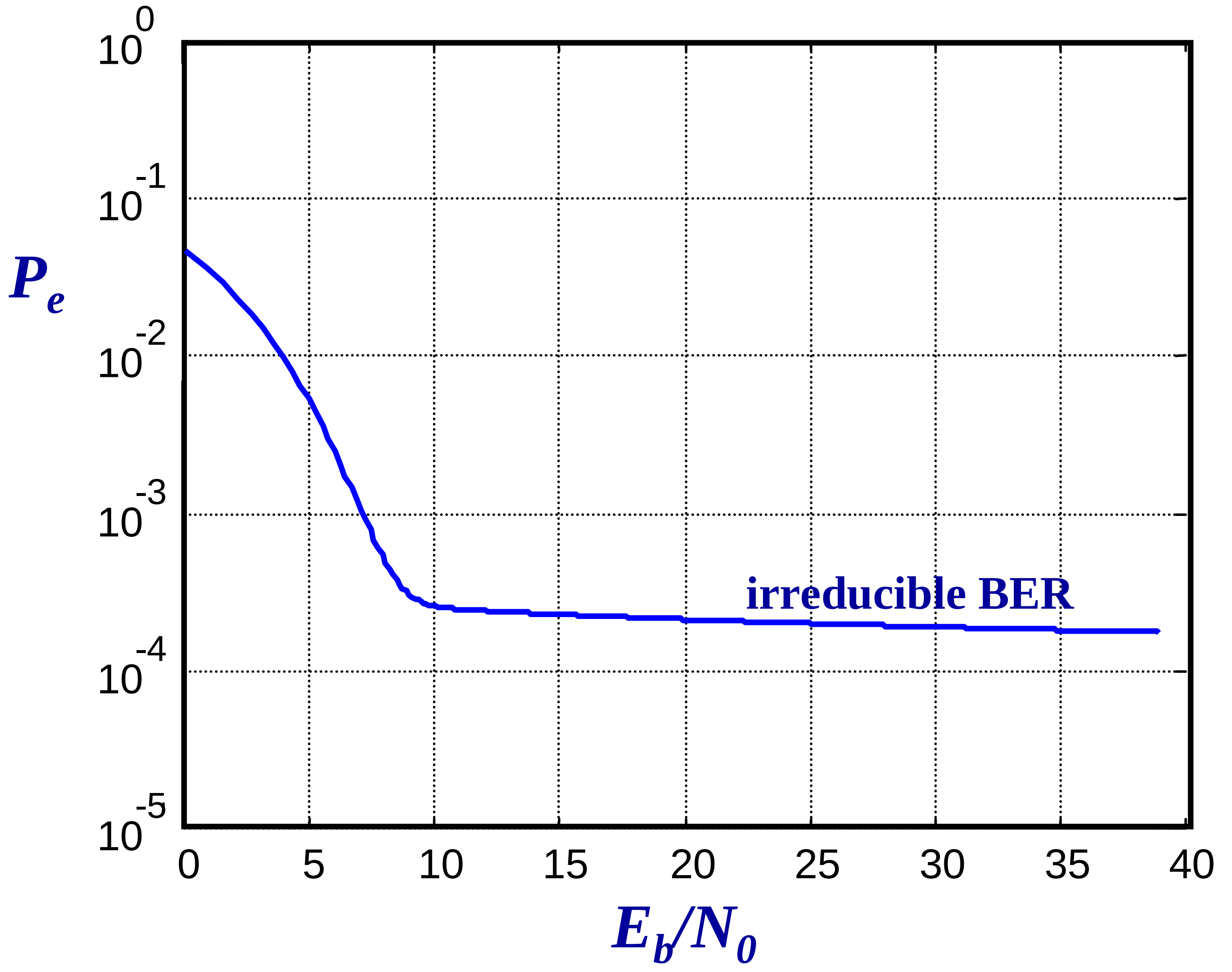
$$P_{e,FSK} = \frac{1}{2}\left[1 - \sqrt{\frac{1}{2+\Gamma}}\right] \approx \frac{1}{2\Gamma}$$ coherent FSK

$$P_{e,DPSK} = \frac{1}{2(1+\Gamma)} = \frac{1}{2\Gamma}$$ differential PSK

Performance in slow/flat fading



Performance in freq. sel. fading



Countermeasures

	flat	frequency selective
slow	**DIVERSITY** **CODING+INTERL.**	**EQUALIZATION**
fast	**DIVERSITY** **CODING+INTERL.**	**DIVERSITY** **CODING+INTERL.**

Equalization

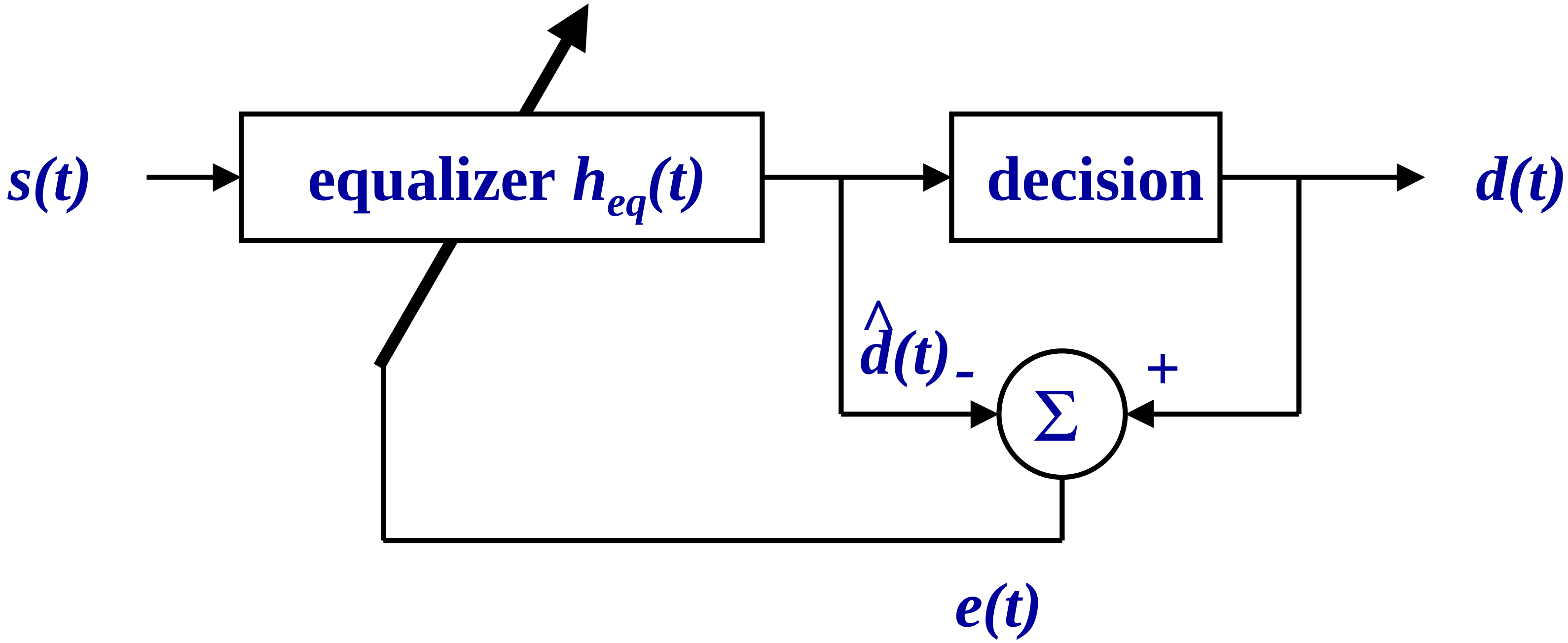
$$s(t) \longrightarrow \boxed{h_{ch}(t)} \longrightarrow \quad r(t)=h_{ch}(t)*s(t)$$

$$s(t) \longrightarrow \boxed{h_{ch}(t)} \longrightarrow r(t) \longrightarrow \boxed{h_{eq}(t)} \longrightarrow \quad x(t)= h_{ch}(t)*\, h_{eq}(t)*s(t)$$

$$x(t)= s(t) \;\Rightarrow\; h_{ch}(t)*\, h_{eq}(t)=\delta(t)$$

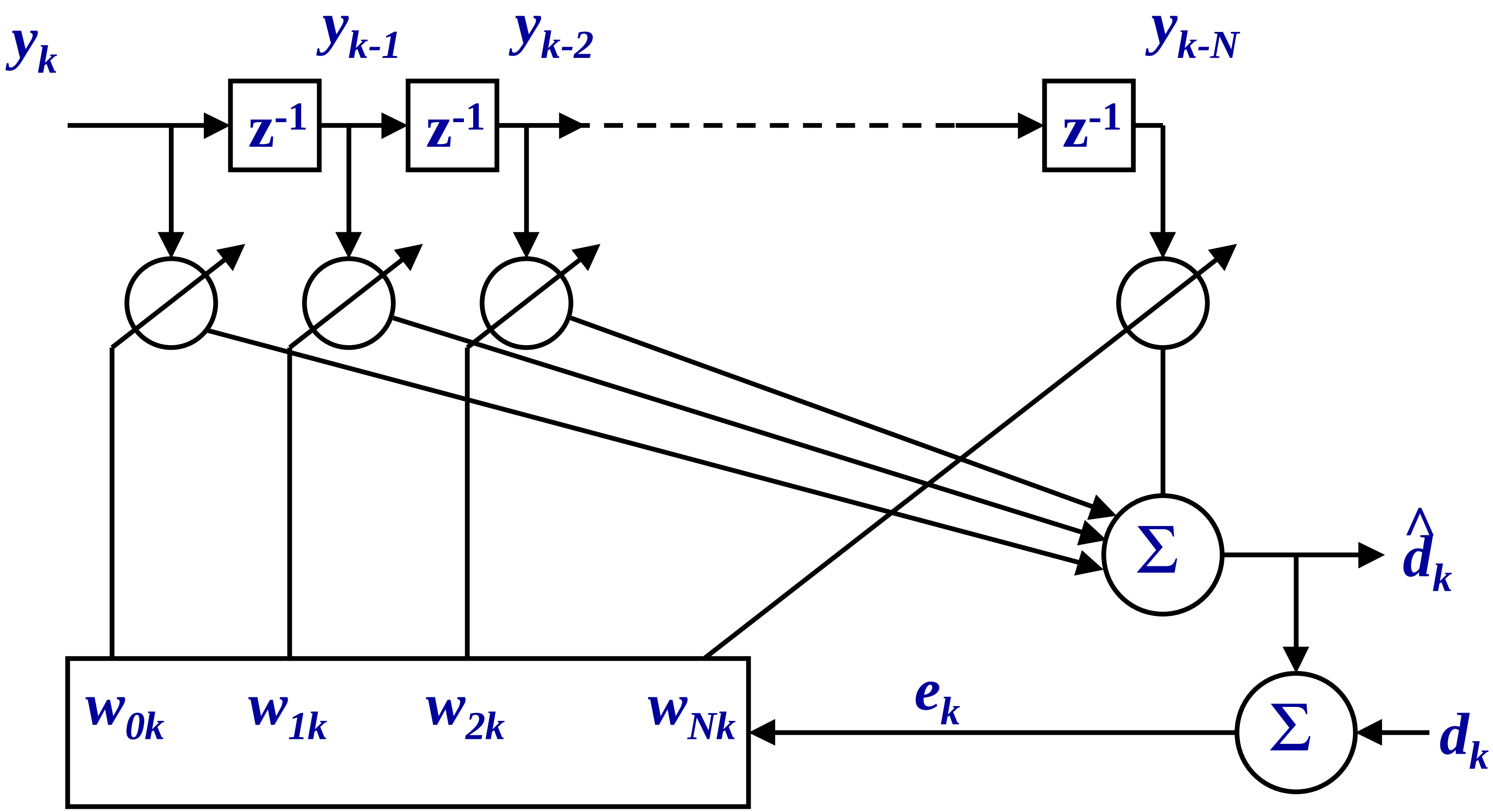
$$H_{eq}(f)= 1/H_{ch}(f)$$

Adaptive equalization



$\hat{d}(t)$	received symbol after equalization
$d(t)$	retrieved symbol
$e(t)$	prediction error

Generic adaptive equalizer



regular update of w_{nk}

Adaptive equalization

- training sequence: exploiting known transmit sequence
- blind equalization: exploiting known modulation
- cost function

  mean square error (MSE)

Adaptive equalization

LMS algorithm

$$\text{MSE:} \quad E\left[|e_k|^2\right] = E\left[x_k^2\right] + \vec{w}^T R \vec{w} + 2\,\vec{p}^T \vec{w}$$

x_k	training sequence symbols
$\vec{w}$	equalizer coefficients
R	input correlation/covariance matrix
$\vec{p}$	cross-correlation vector

Adaptive equalization

$$\vec{p} = E\left[x_k \vec{y}_k\right] = E\left[x_k y_k \quad x_k y_{k-1} \quad \cdots \quad x_k y_{k-N}\right]$$

$$R = E\left[\vec{y}_k \vec{y}_k^T\right] = E\begin{bmatrix} y_k^2 & y_k y_{k-1} & \cdots & y_k y_{k-N} \\ y_{k-1} y_k & y_{k-1}^2 & \cdots & y_{k-1} y_{k-N} \\ \cdots & \cdots & \cdots & \cdots \\ y_{k-N} y_k & y_{k-N} y_{k-1} & \cdots & y_{k-N}^2 \end{bmatrix}$$

MMSE: $\vec{w} = R^{-1}\vec{p}$

$$E\left[\left|e_k\right|^2\right]_{\min} = E\left[x_k^2\right] - \vec{p}^T R^{-1} p$$

Adaptive equalization

RLS algorithm

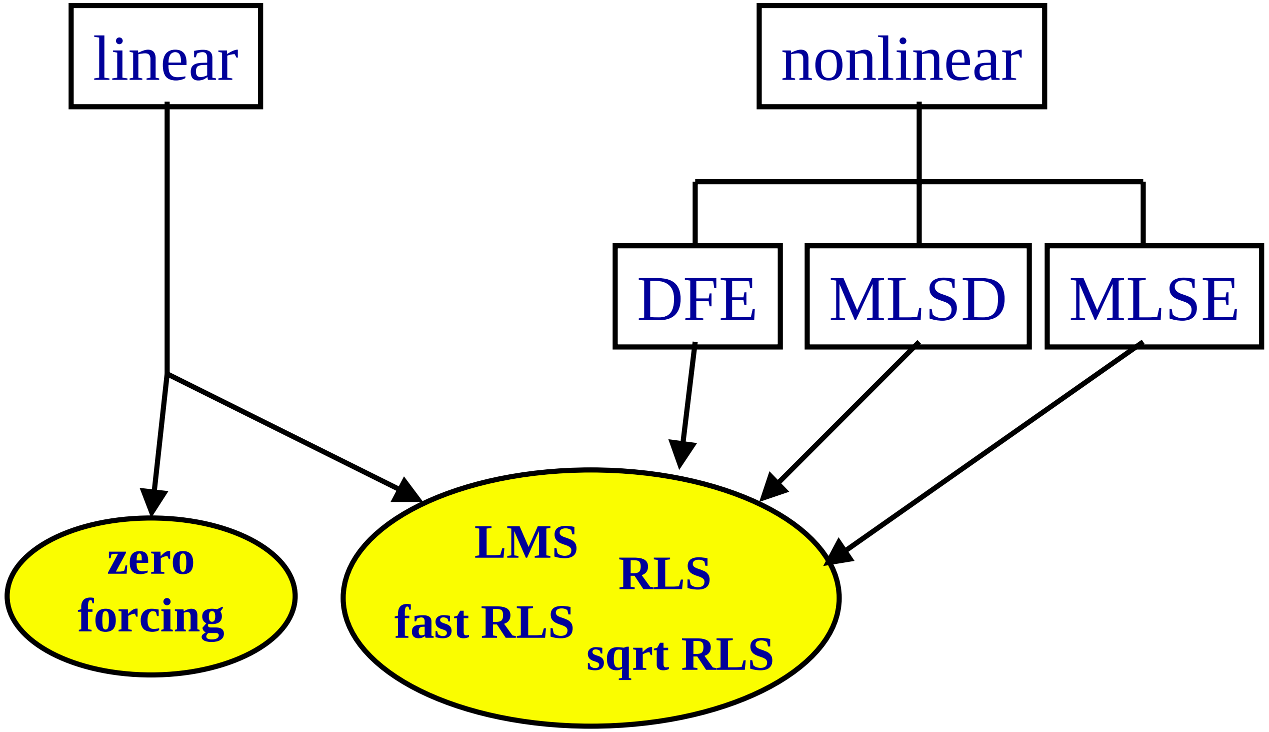
MSE: $$J(n) = \sum_{i=1}^{n} \lambda^{n-i} e(i,n) \cdot e^{*}(i,n)$$

λ weighting factor (λ<1)

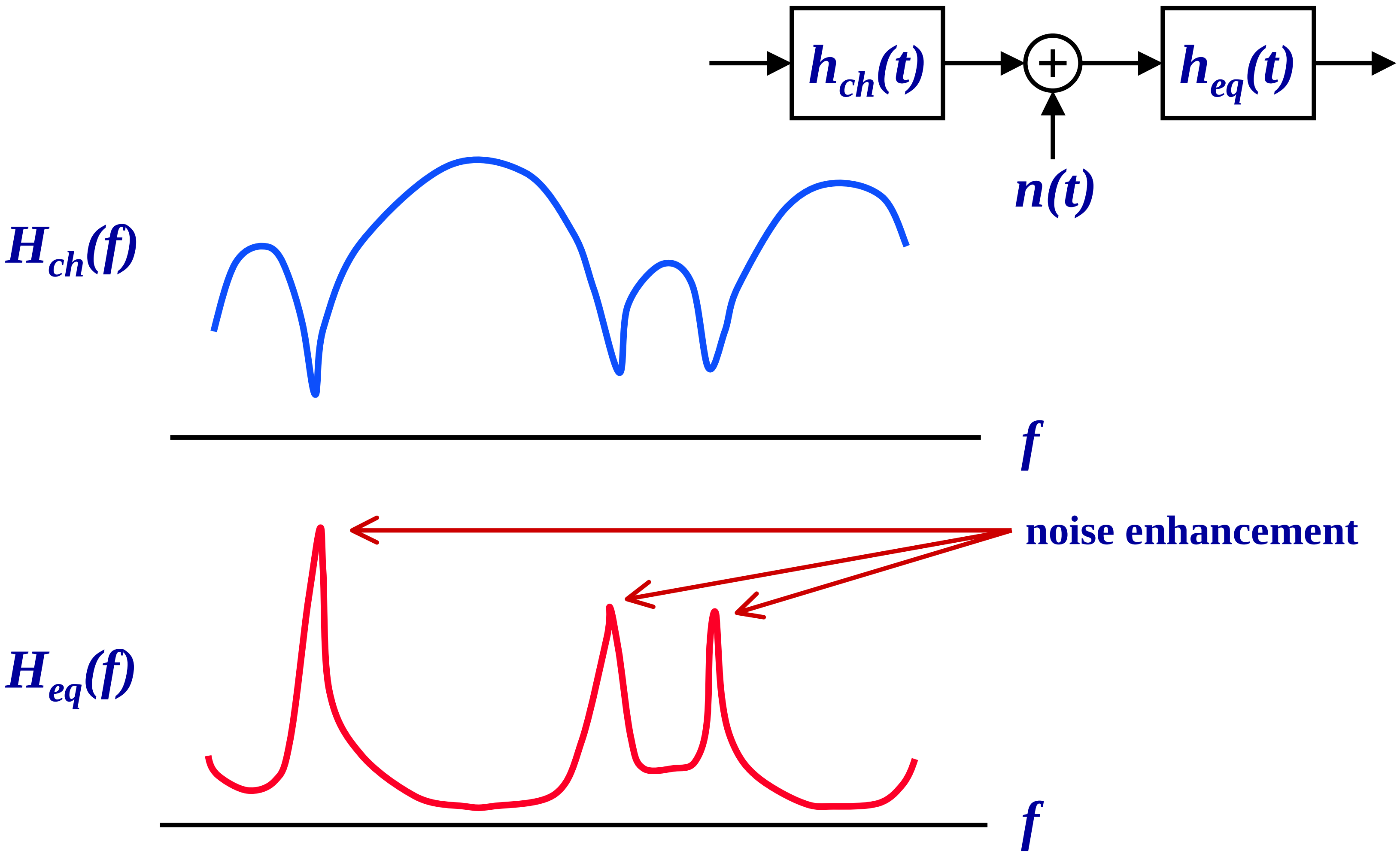
$$e(i,n) = x(i) - \vec{y}_N^{\,T}(i) \cdot \vec{w}_N(n) \qquad 0 \le i \le n$$

Exponential forget; fast convergence.

Equalizer classification



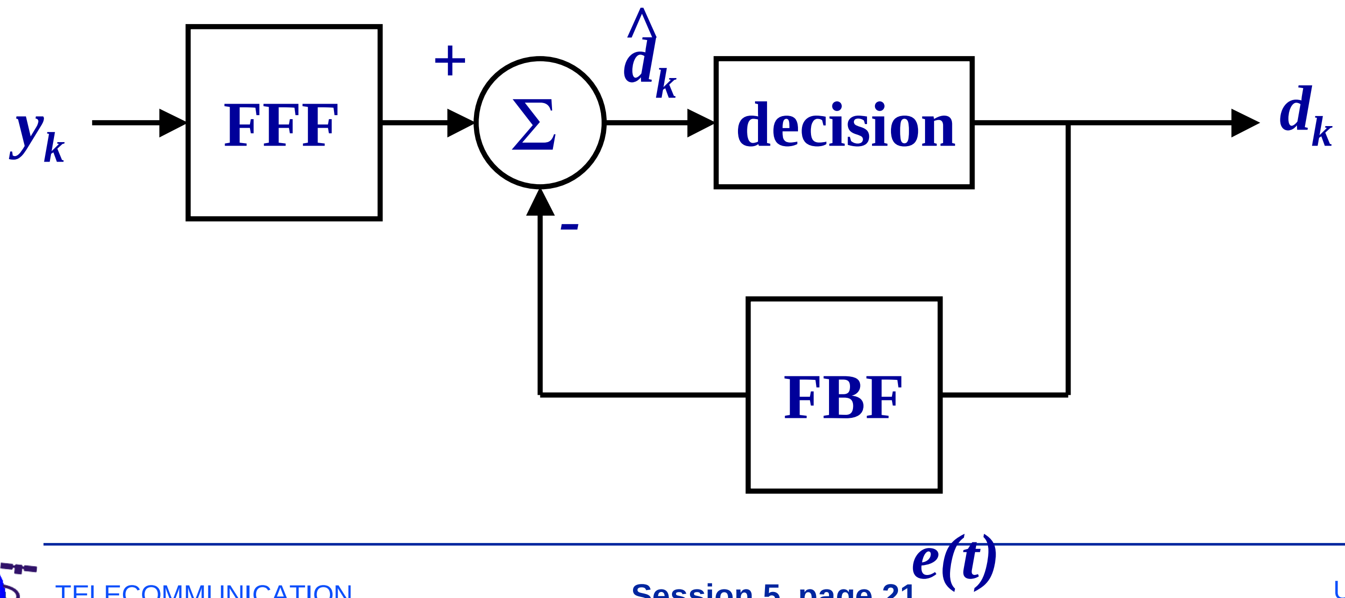
Problems with linear equalizers



Nonlinear equalizers: DFE

Considers symbol by symbols:

- Step 1: decide on symbol
- Step 2: predict ISI
- Step 3: subtract ISI from original signal
- Step 4: goto next symbol



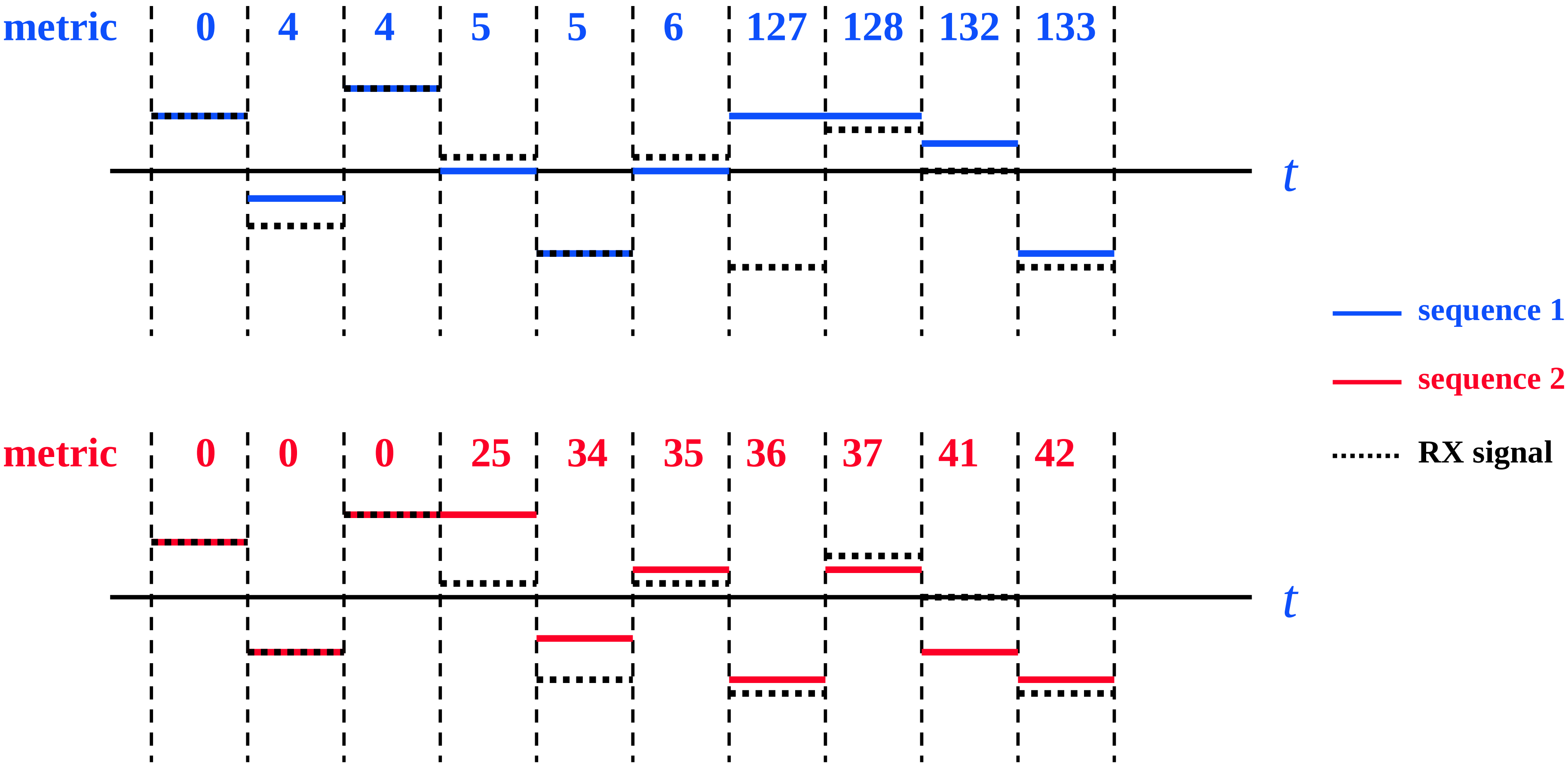
Nonlinear equalizers: MLSE

Considers sequence of symbols:

- Step 1: estimate channel
- Step 2: for all possible symbol combinations, generate receive sequences
- Step 3: compare each generated sequences with actual RX sequence
- Step 4: choose that receive sequence for which accumulated error (metric) is smallest
- Step 5: update channel estimate

Viterbi algorithm

MLSE example



Maximum likelihood estimation

Estimated channel: $\{h_0, h_1, \ldots, h_{L-1}\}$

Received sequence: $\{r_0, r_1, \ldots, r_{L-1}\}$

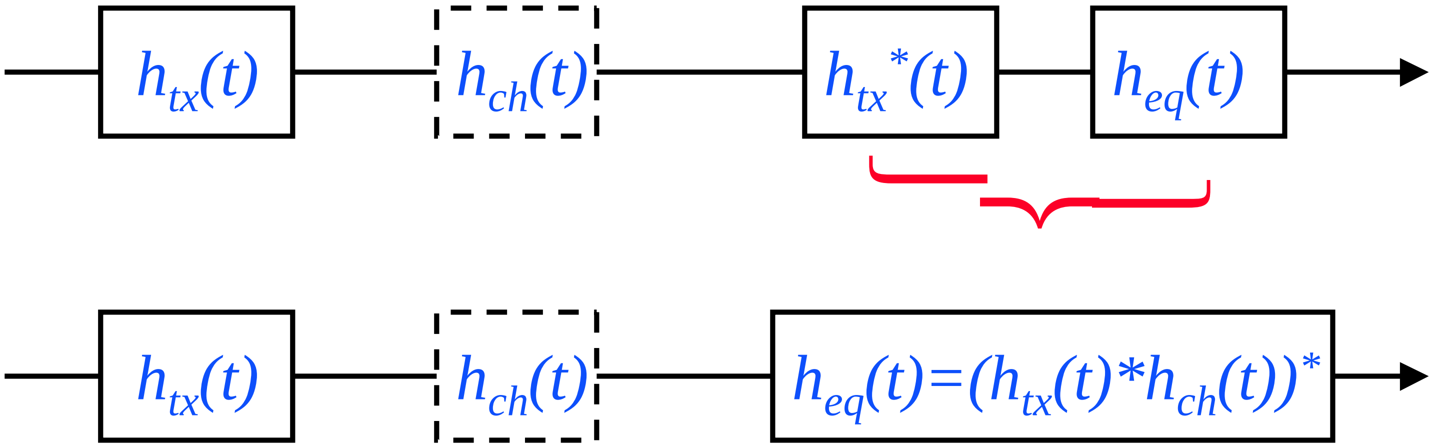
Choose $\{s_0, s_1, \ldots, s_{L-1}\}$ which minimizes $\sum_{i=0}^{L-1} C(h_i s_i, r_i)$

where C is the metric

With M symbols in the alfabet, there are M^L sequence to choose from.

Fractionally spaced equalizers

- **Sampling higher than symbol rate (Nyquist rate)**
- **Combination of matched filter and equalizer**

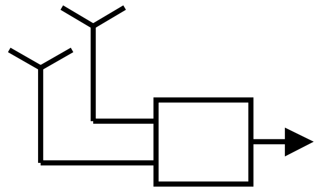


Diversity

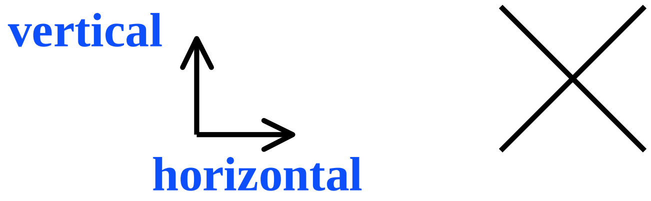
- Apply multiple paths between TX and RX
- Separate paths/branches in
  - space
  - time
  - frequency
- Minimize correlation between paths

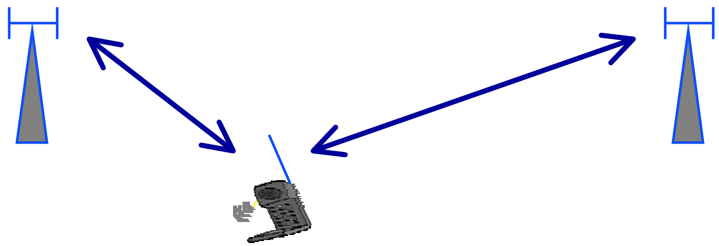
Space diversity

- **Micro-diversity:**
  - **antenna diversity**



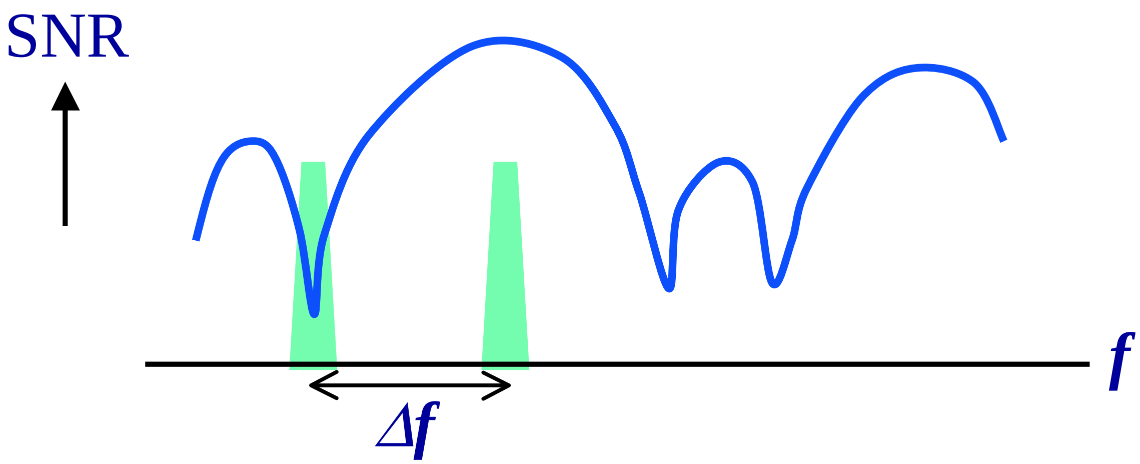
  - **polarization diversity**





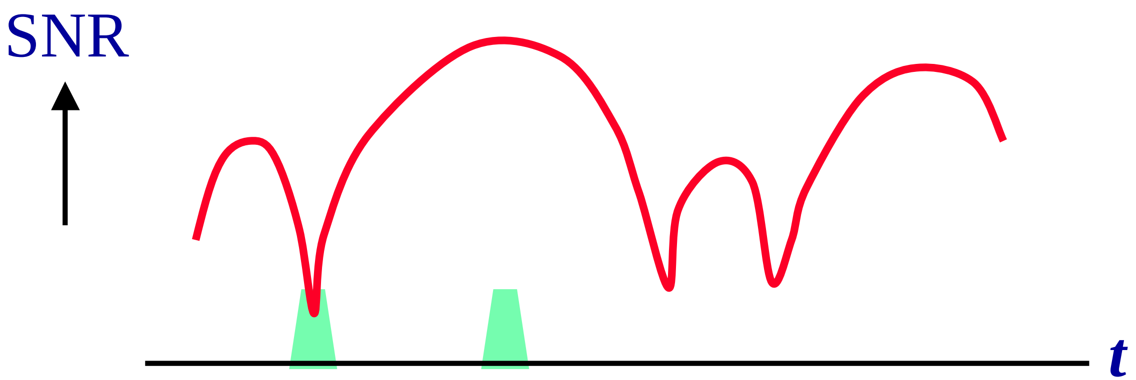
- **Macro-diversity**
  - **base station diversity**

Frequency diversity



- $\Delta f > B_c$
- **Frequency hopping + retransmission**

Time diversity



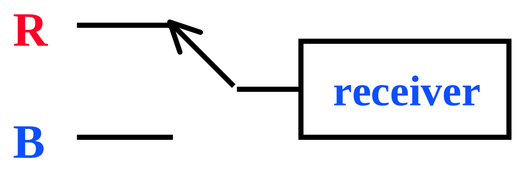
- $\Delta t > T_c$
- **Coding + interleaving**

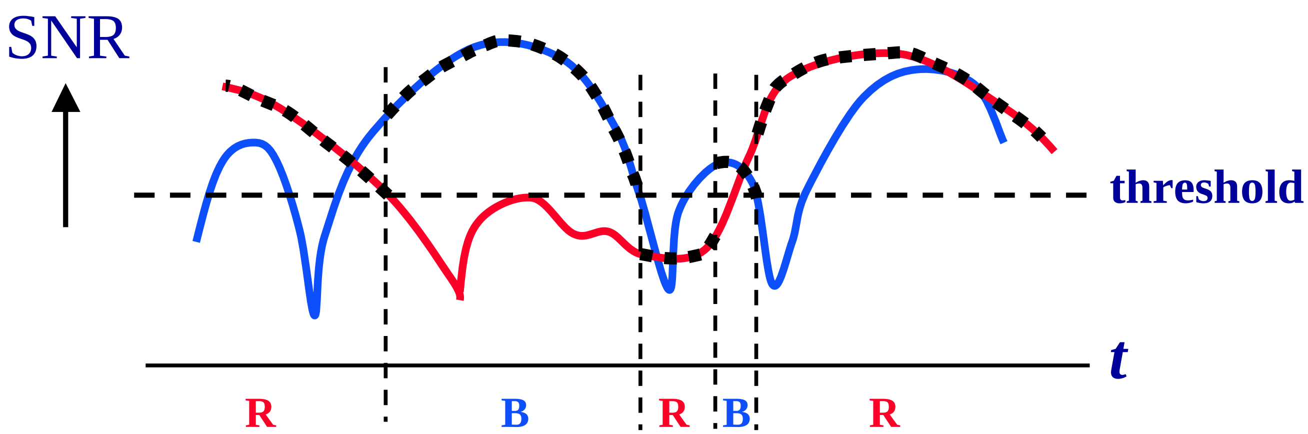
Combining techniques

- **Switch combining**
- **Selection combining**
- **Maximal ratio combining**
- **Equal gain combining**

Switch diversity

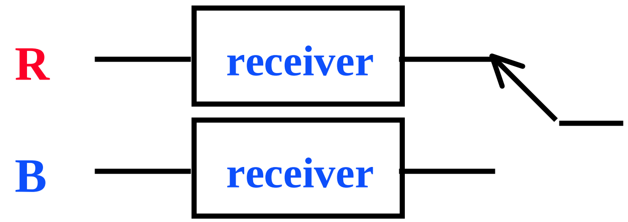
- Remain on one branch until SNR drops below a threshold
- Branches R and B.

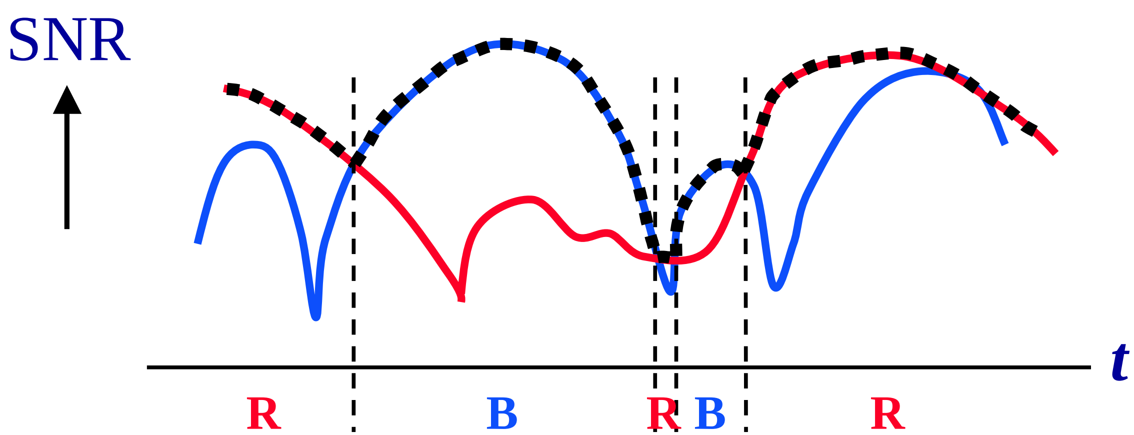




Selection diversity

- Always take the best branch
- Branches R and B.





Selection diversity

Improvement derivation:

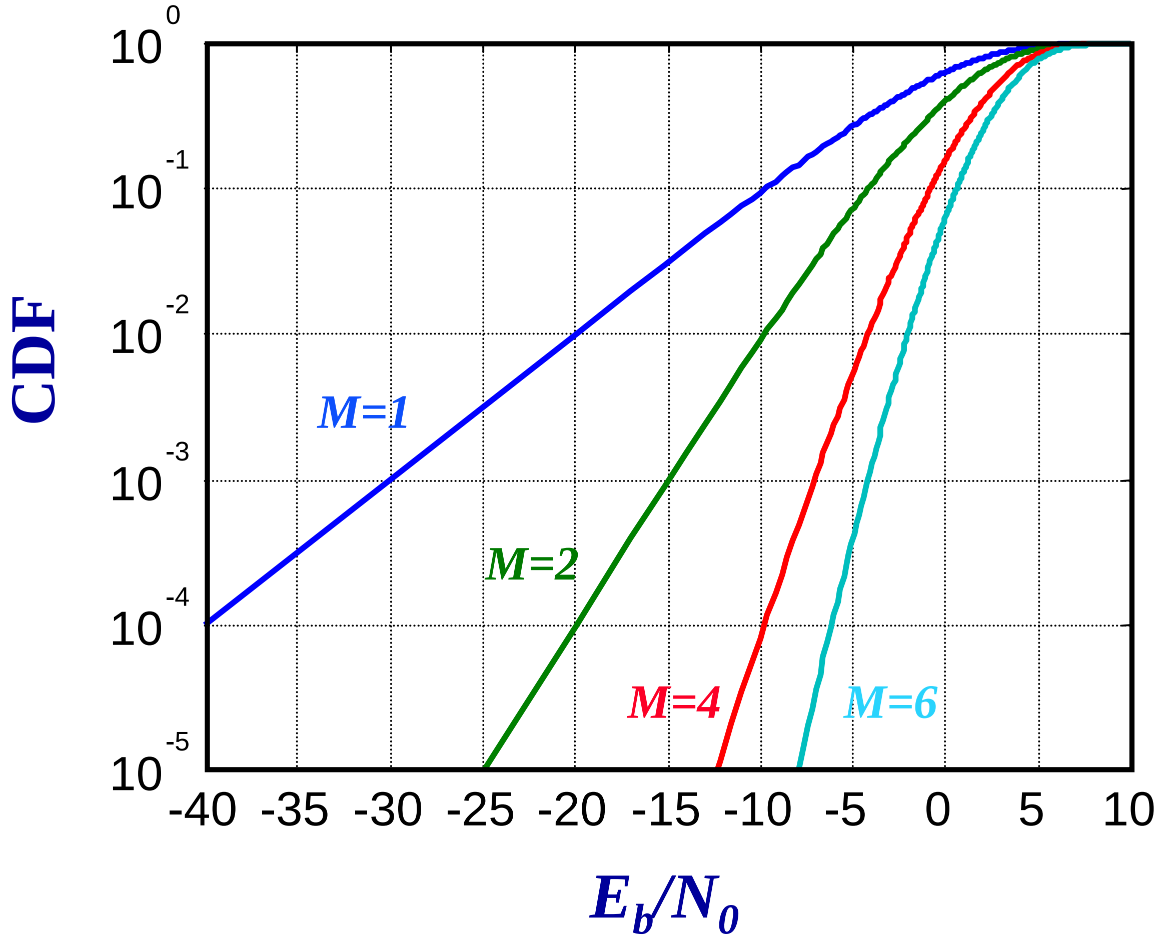
$$SNR = \gamma \qquad \bar{\gamma} = \Gamma = \frac{E_b}{N_0}\alpha^2$$

$$p(\gamma) = \frac{1}{\Gamma} e^{-\gamma/\Gamma}$$

Single branche: $P_r(\gamma \le t) = \int_0^t \frac{1}{\Gamma} e^{-\gamma/\Gamma} d\gamma = 1 - e^{-t/\Gamma}$ **outage**

***M* branches:** $P_r(\gamma_1, \gamma_2, \cdots \gamma_M \le t) = \left(1 - e^{-t/\Gamma}\right)^M$

Selection diversity



Maximum ratio combining

- **Co-phase accumulation of branches**
- **Individually weighted for optimal SNR**

signal $$r_M = \sum_{i=1}^{M} G_i r_i$$

noise $$N_T = \sum_{i=1}^{M} G_i^2 N_i^2$$

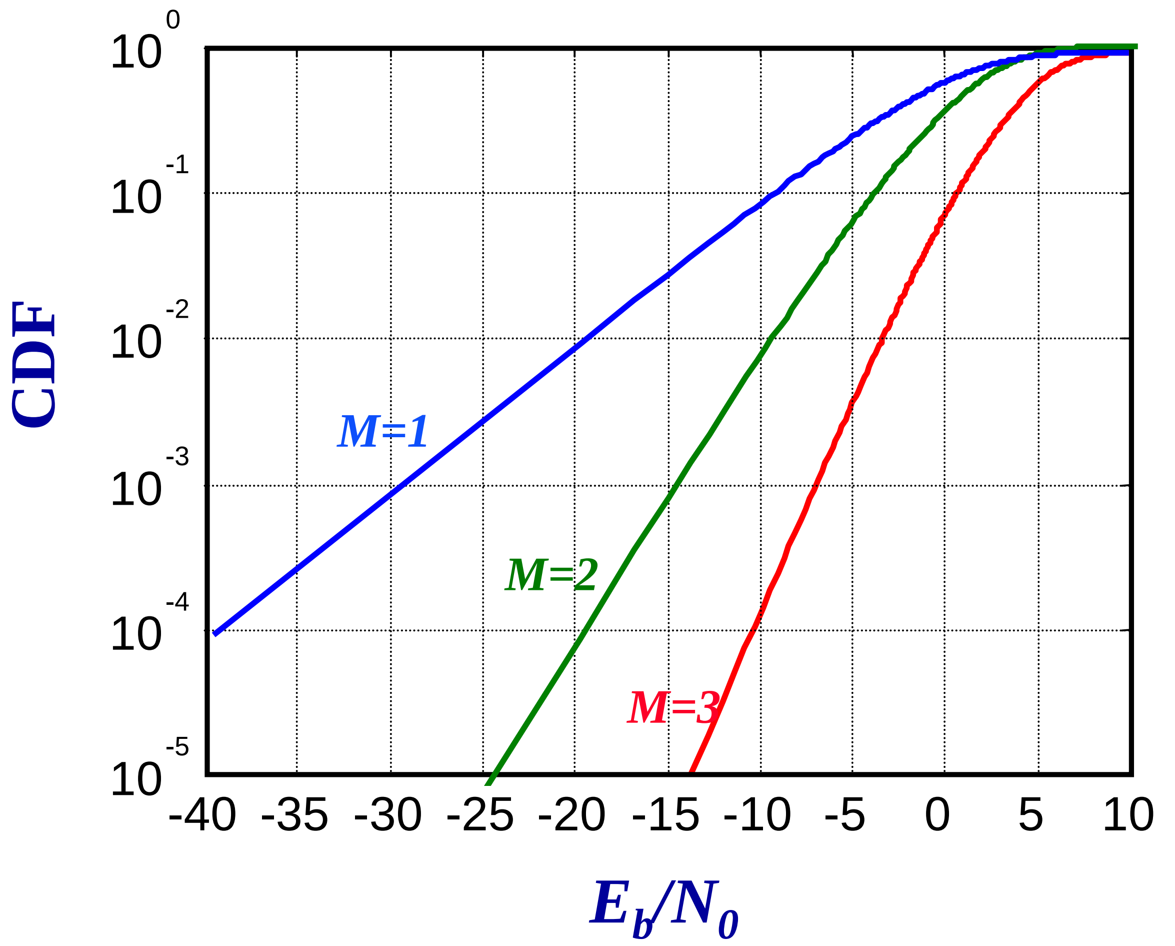
Optimize SNR $$\gamma_M = \frac{r_M^2}{2 N_T} \quad \Rightarrow \quad G_i = \frac{r_i}{N_i}$$

Maximum ratio combining

Total SNR $\gamma_M = \sum_{i=1}^{M} \gamma_i$ **sum of individual SNRs**

Outage $\Pr(\gamma_M \le t) = 1 - e^{-t/M} \sum_{i=1}^{M} \frac{(t/\Gamma)^{i-1}}{(k-1)!}$

Maximal ratio combining



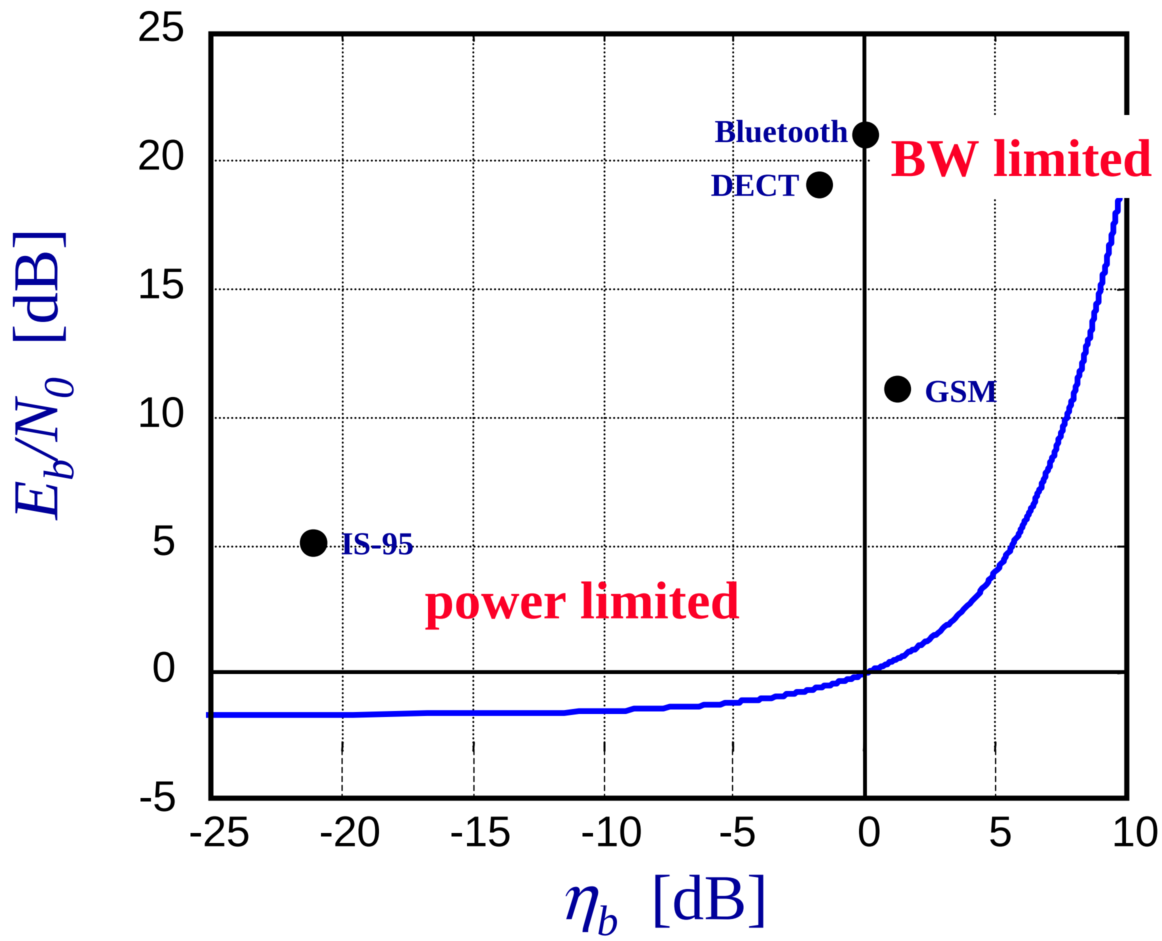
Equal gain combining

- **Co-phase accumulation of branches**
- **Equally weighted with unity gain**
- **Performance between selection and MRC**

signal

$$|r_M| = \sum_{i=1}^{M} |r_i|$$

Power-bandwidth trade-off



Channel coding

- Add controlled redundancy
- Create dependencies between bits
- Correlation between correct and incorrect bits
- The more bits involved, the stronger the code

HELPS ONLY IF CHANNEL CONDITIONS VARY:
“good” bits help “bad” bits

Channel coding

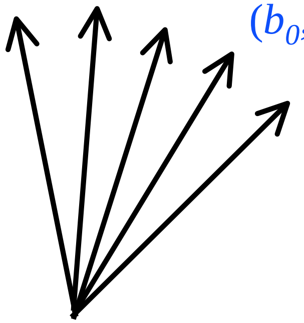
- Make dependencies
- Example: (7,4) code

- 4 user bits → 7 coded bits
- Distance 1 → 3

0000	000
0001	011
0010	110
0011	101
0100	111
0101	100
0110	001
0111	010
1000	101
1001	110
1010	011
1011	000
1100	010
1101	001
1110	100
1111	111

Code space

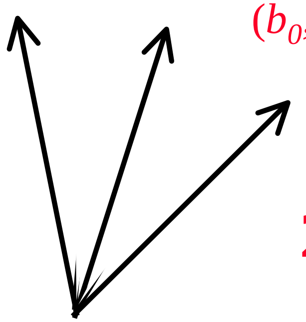
k-dimensional space



$(b_0,b_1,\ldots,b_{k-1})$

2^k vectors

n-dimensional space



$(b_0,b_1,\ldots,b_{k-1})$

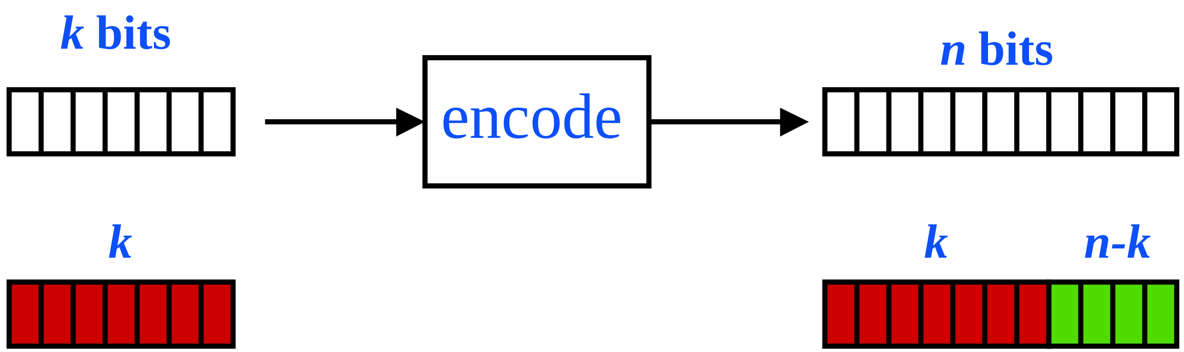
2^k vectors

distance increases

Coding characteristics

k user bits, n coded bits

- n-k "parity" bits
- coding rate $\frac{k}{n}$



systematic

Coding characteristics

<u>linear</u>

If c_i and c_j code words, then $c_i \oplus c_j$ is code word as well.

All-zero code word

<u>cyclic</u>

If c is a code word, then all cyclic shifts all code words as well.

$c_0, c_1, c_2, c_3 \rightarrow$

c_1, c_2, c_3, c_0

c_2, c_3, c_0, c_1

c_3, c_0, c_1, c_2

Coding characteristics

Weight

$$w(c_i) = \sum_{l=0}^{n-1} c_{i,l}$$

Hamming distance

$$d(c_i, c_j) = l \sum_{i=0}^{n-1} \left(c_{i,l} \oplus c_{j,l} \right)$$

Minimum distance

$$d_{\min} = MIN \left\lfloor d(c_i, c_j) \right\rfloor$$

Perfect code

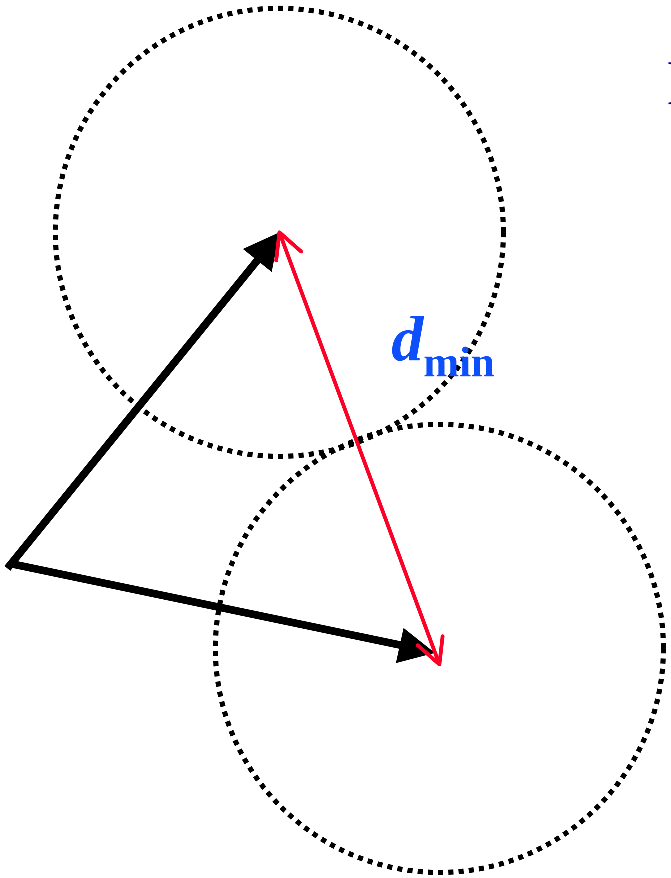
$$d(c_i, c_j) = d_{\min} \quad \forall i, j$$

Example: (7,4) BCH code

0000	000	c_0
0001	011	c_1
0010	110	
0011	101	
0100	111	c_4
0101	100	
0110	001	
0111	010	
1000	101	
1001	110	
1010	011	
1011	000	
1100	010	
1101	001	
1110	100	
1111	111	c_{15}

- Systematic: 4 user bits, 3 parity bits
- Linear: e.g. $c_1+c_4=c_5$
- Non cyclic
- $w(c_1)=3$
- $w(c_{15})=7$
- $d(c_1,c_4)=3$
- $d(c_0,c_{15})=7$
- $d_{min}=3$

Error detection and correction



Detects up to $l = d_{\min} - 1$ errors

Corrects up to $t = \mathrm{int}\left[\frac{d_{\min} - 1}{2}\right]$ errors,

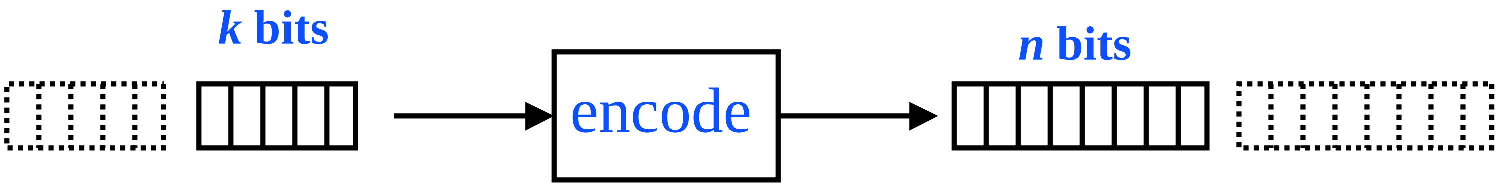
but then detects only $l = \mathrm{int}\left[\frac{d_{\min}}{2}\right]$ errors!

Example: (7,4) BCH code

$d_{min}=3 \Rightarrow t=1, l=1$, or $t=0, l=3$

Block codes

- User bit stream is divided into blocks of *k* bits
- Expand every *k*-bit block to a *n*-bit code word



Block codes

Hamming code

$(n,k) = (2^m\text{-}1, 2^m\text{-}1\text{-}m)$

$n\text{-}k=m$

Hadamard code

$$A_0 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad Ak = \begin{bmatrix} A_{k-1} & A_{k-1} \\ A_{k-1} & -A_{k-1} \end{bmatrix}$$

$d_{\text{min}}=N/2$

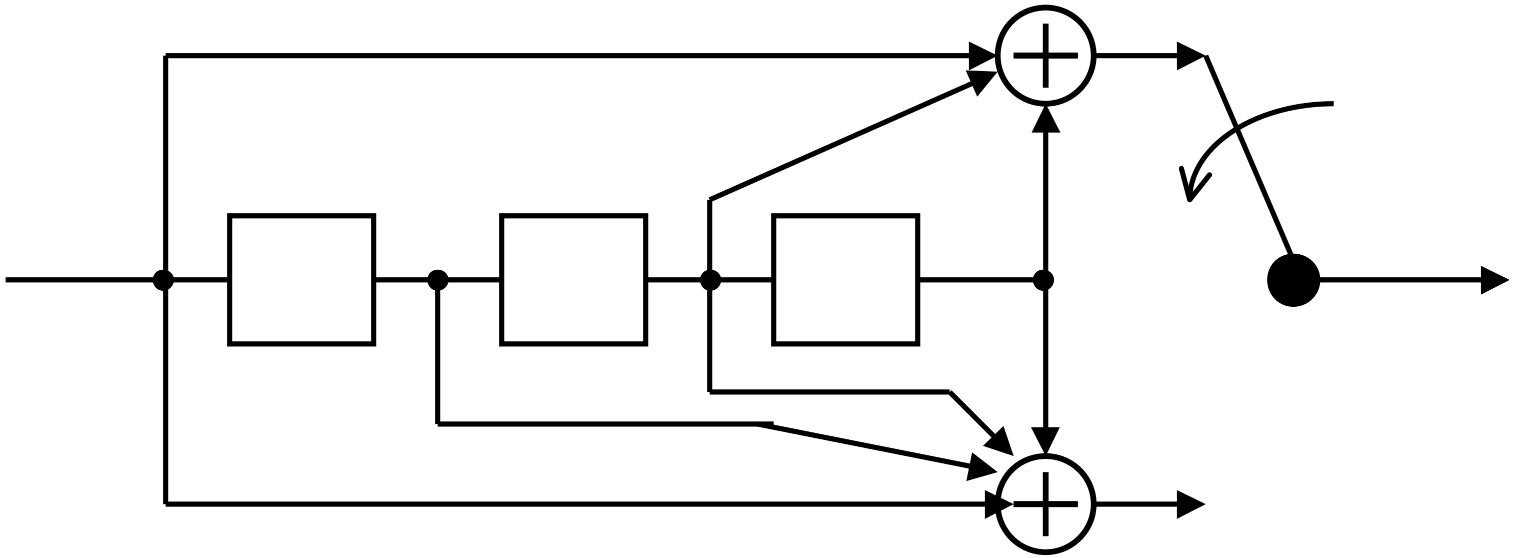
Golay code

perfect (23,12) code

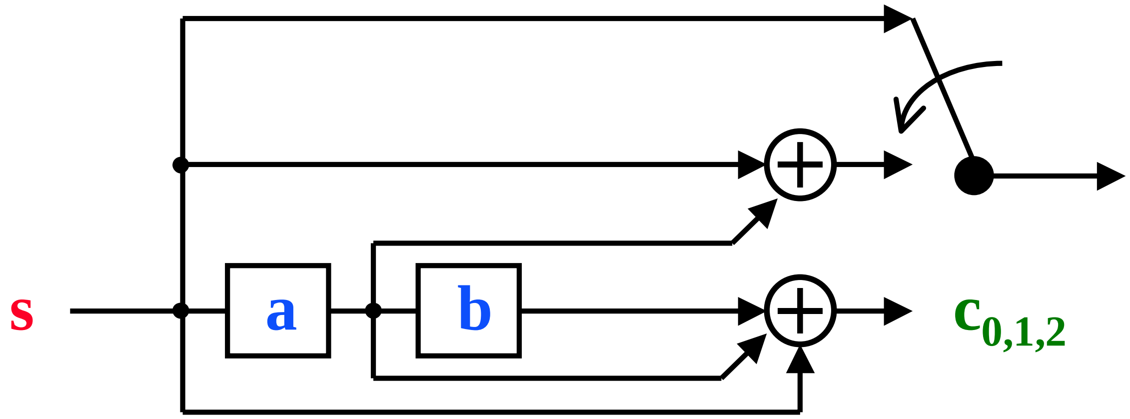
$d_{\text{min}}=7$, $t=3$

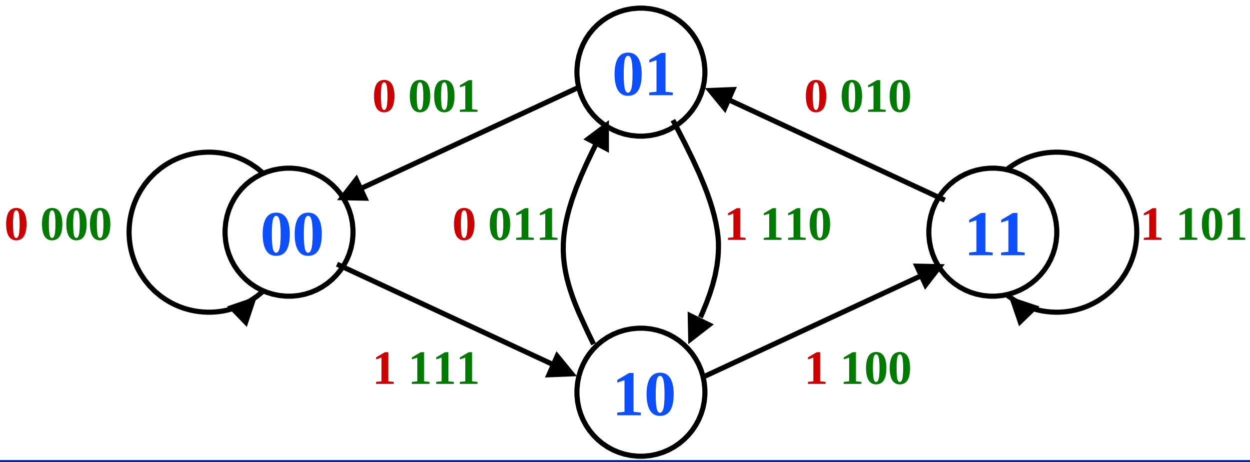
Convolutional codes

- sliding bit streaming process
- (n,k,m) code with m memory elements
- constraint length m+1

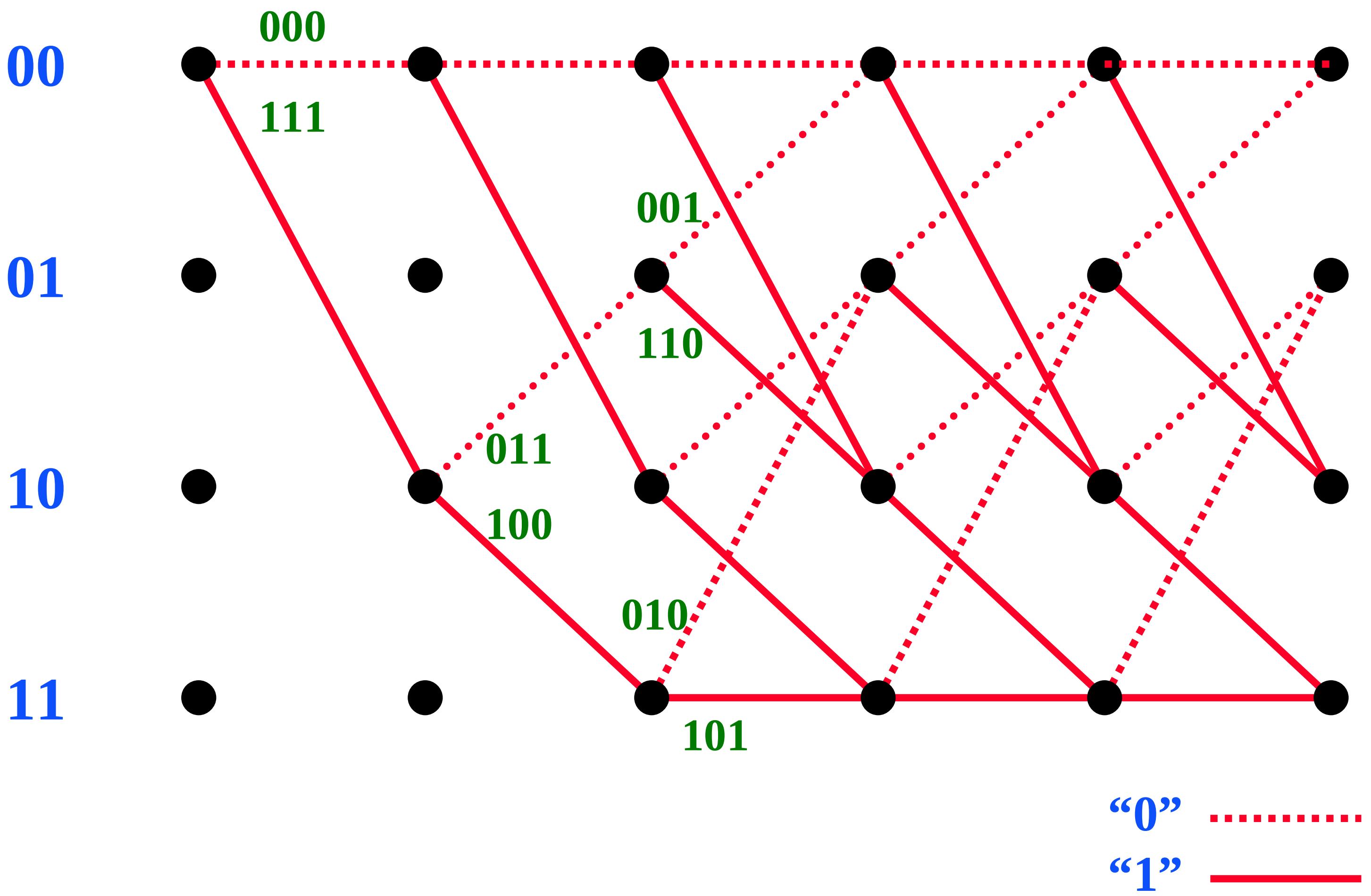


Example convolutional coder



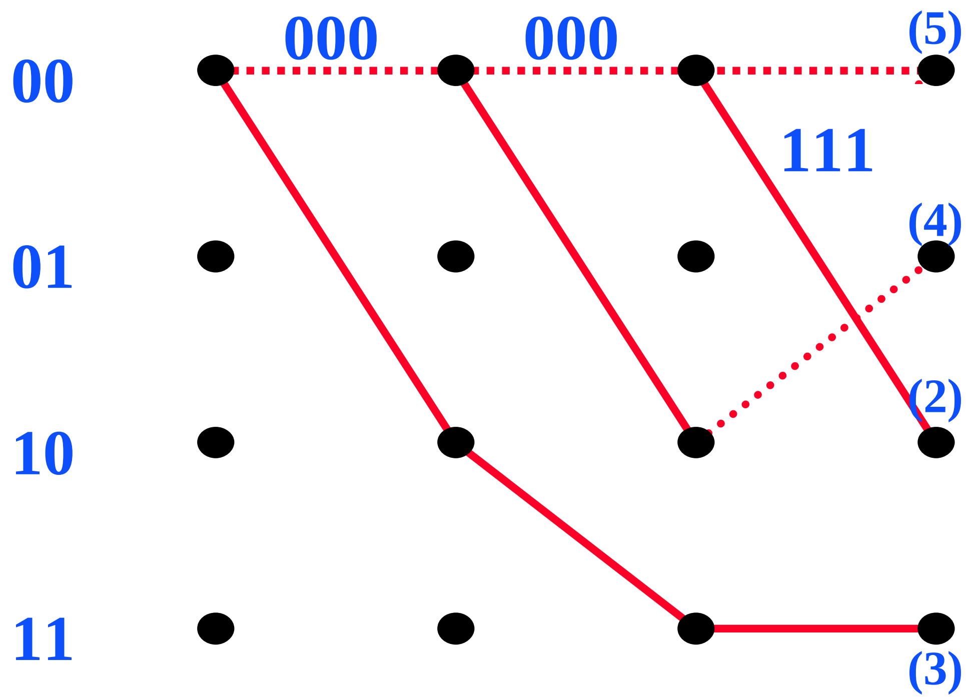


Trellis decoder



Viterbi decoder

- Compare input with possible output sequences metric
- Keep path with minimum accumulated metric

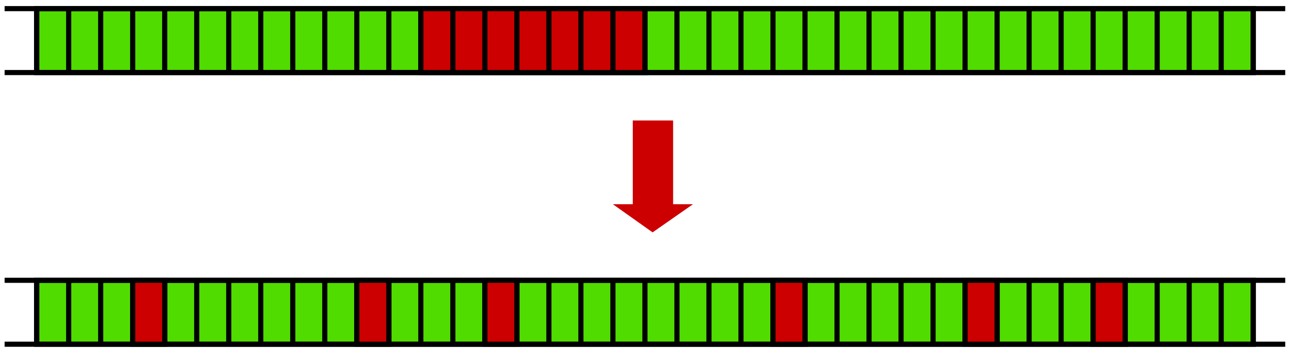


Received: 001 100 111

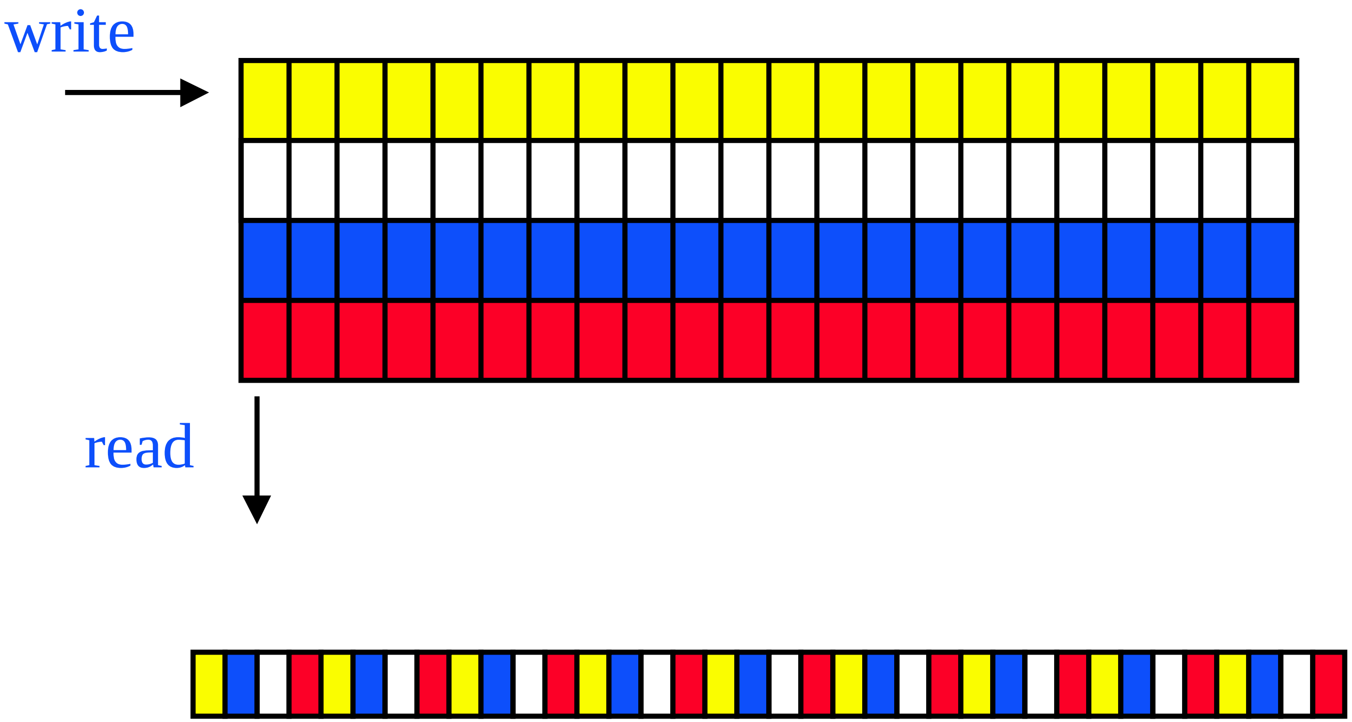
Interleaving

Coding: "good" bits help to correct "bad" bits ⇒ bursts of errors difficult to correct

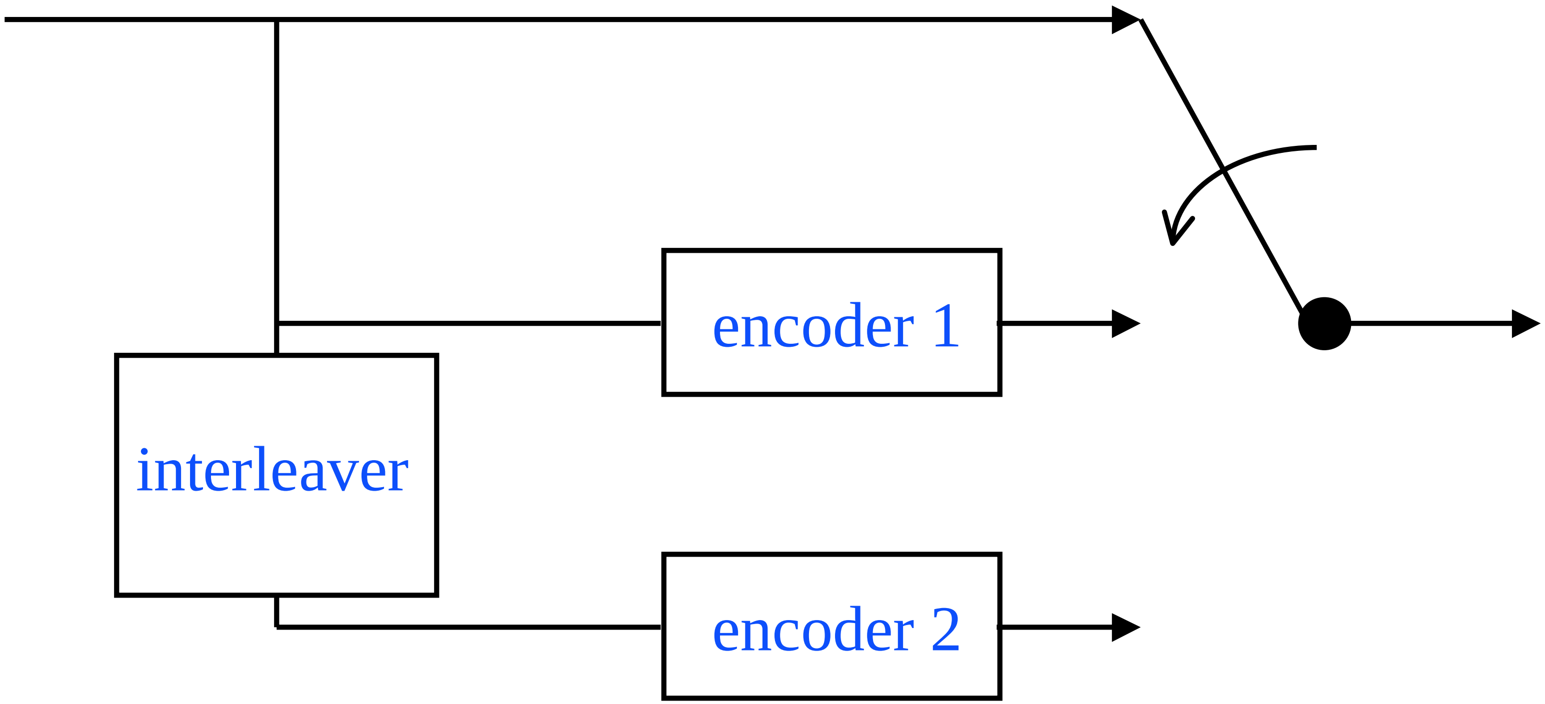
Interleaving: spread "bad" bits so they are surrounded by "good" bits



Block interleaving



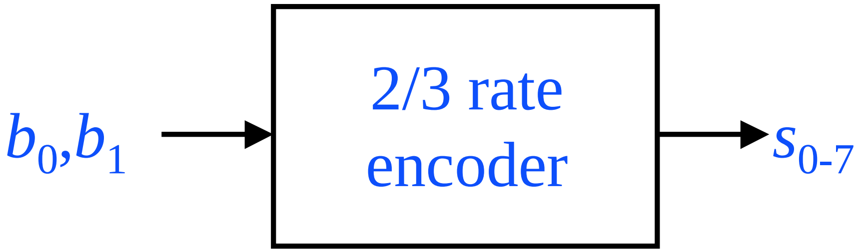
Turbo coding

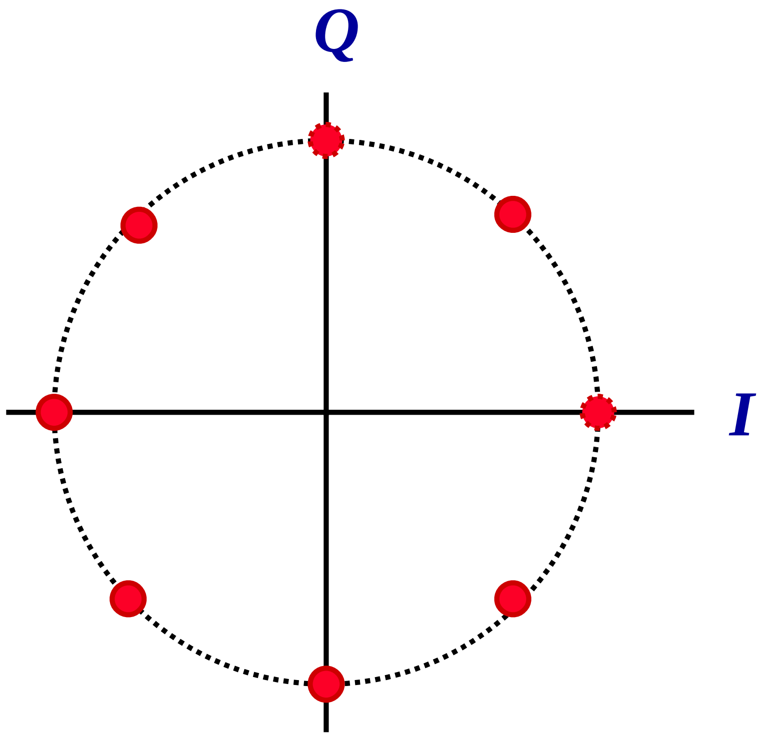


Trellis coded modulation

- Combined coding and modulation
- Increase number of constellation points ⇒ data rate increases
- Use extra capacity for coding
- Coding gain compensates for reduction in symbol distance
- Proper code mapping on symbols
- Based on Euclidean distance rather than on Hamming distance

Trellis coded modulation





FOR NEXT TIME

- **Read:**
  Chapter 5: §5.10
  Chapter 6: §6.11
  Chapter 8: §8.1-8.6, 8.7 (<u>not</u> 8.7.2 and 8.7.3)

- **Solve problems:**
  Chapter 5: 5.30
  Chapter 6: 6.2, 6.4, 6.5, 6.7,