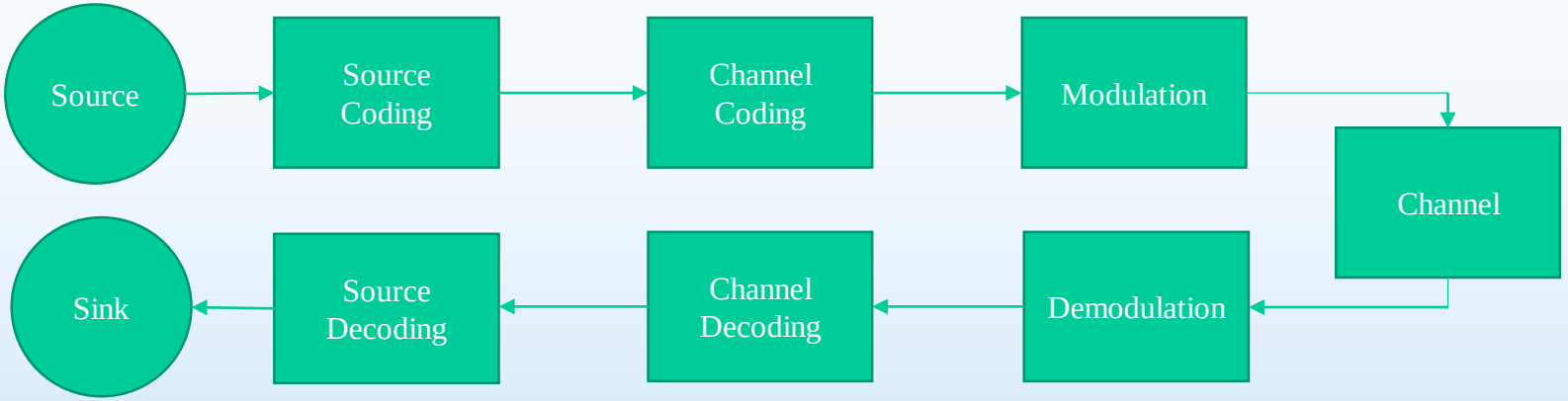
Digital Communication Techniques

Block diagram digital connection



- Source coding: compression.
- Channel coding: Redundancy is added in a systematic way, for protection against errors (coding rate: R_c (information bits/coded bits)).
- Modulation: Operation by which a continuous time signal is associated with the symbols (numbers) emitted by the source, after compression and channel coding.

Digital signal

- Message: discrete succession of numbers the values of which are chosen in set of dimension $M=2^b$. Each value represents b bits.

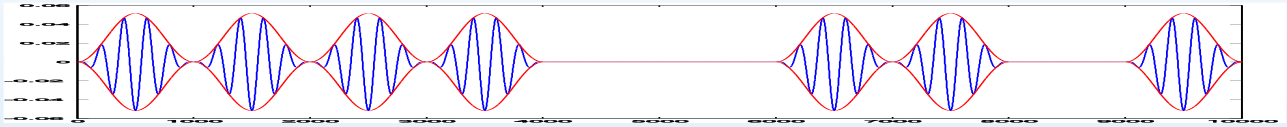
- Modulation: operation that associates each of the possible values with a waveform. A continuous time signal, continuous amplitude is obtained, used to transfer the numeric information.

- A modified signal reaches the receiver (delay, attenuation, distortion, noise, interference) and can be misinterpreted by the receiver (detection/decoding error).

- The performance of a digital transmission system is measured in terms of error rate or, more in general, by error statistic.

Digital Modulator

- It associates to the sequence of numbers to transmit a succession of selected waveforms in a finite dimensional set $M=2^b$, being b a integer. Each waveform is associated with a group of b consecutive bits (*mapping*).



- Modulation operates baseband, if the produced signal has Fourier transform in baseband, or it is defined passband otherwise.

- Parameters:
  - T [s]: Signaling (or symbol) interval, (waveform emission period).
  - $1/T$ [baud] o [symbol/s]: transmission rate in symbols per second.
  - W [Hz]: bandwidth. It is proportional to $1/T$.

A mathematical model

- Consider a set of N orthonormal functions, $\psi_1(t), \psi_2(t), \ldots, \psi_N(t)$, .
$$\langle \psi_i \psi_j \rangle = \int_{<T>} \psi_i(t)\psi_j(t)\,\mathrm{d}t = \delta_{ij}$$
- A generic waveform can be expressed as a linear combination of the orthonormal functions $\Psi_i(t)$, $x(t) = \sum_{i=1}^{N} x_i \psi_i(t)$, being $x_i = \langle x(t)\psi_i(t) \rangle$.
- A generic waveform, $x(t)$, is, therefore, represented by a vector $\mathbf{x}=(x_1, x_2, \ldots, x_N)$, whose components can be used to determine some quantities of interest.
  - Norm: $\|x(t)\|^2 = \sum_{i=1}^{N} |x_i|^2$.
  - Scalar product: $\langle x(t)y(t) \rangle = \sum_{i=1}^{N} x_i y_i^*$
  - Distance: $d^2(x,y) = \sum_{i=1}^{N} |x_i - y_i|^2$.
- It is $N \leq M$.

  Passband Amplitude modulation: $N=1$, $\psi(t) = g(t)\cos(2\pi f_c t)\sqrt{2/E_g}$

  Phase, amplitude and phase modulation: $N=2$, $\psi_1(t) = g(t)\cos(2\pi f_c t)\sqrt{2/E_g}$
  $\psi_2(t) = g(t)\sin(2\pi f_c t)\sqrt{2/E_g}$

Linear Passband Modulations

- Amplitude modulation: N=1. $\psi_1(t) = \sqrt{\frac{2}{E_g}} g(t)\cos(2\pi f_c t)$

  Complex envelope: $\tilde{x}(t) = \sum_{n=-\infty}^{\infty} a_n g(t - nT), \quad a_n \in \{2m - 1 - M\}, m = 1,\ldots, M = 2^b$

- Phase, amplitude and phase modulations: N=2.

$$\psi_1(t) = \sqrt{\frac{2}{E_g}} g(t)\cos(2\pi f_c t) \qquad \psi_2(t) = \sqrt{\frac{2}{E_g}} g(t)\sin(2\pi f_c t)$$

Complex envelope: $\tilde{x}(t) = \sum_{n=-\infty}^{\infty} \alpha_n e^{j\theta_n} g(t - nT),$

PSK ($M \geq 4$): $\alpha_n = \alpha = \text{constant}, \quad \theta_n \in \left\{\frac{2\pi}{M} m + \frac{\pi}{M}\right\}, m = 0,\ldots, M - 1$

QAM ($M=2^{2b}$): $a_n = \alpha_n \cos(\vartheta_n) \in \{2m - 1 - \sqrt{M}\}, m = 1,\ldots, \sqrt{M} = 2^b$

$b_n = \alpha_n \sin(\vartheta_n) \in \{2m - 1 - \sqrt{M}\}, m = 1,\ldots, \sqrt{M} = 2^b$

Power spectrum

- Linear modulation

  Complex envelope: $v(t)=\sum_{n=-\infty}^{+\infty} a_n g(t-nT,)$ (*)

  where the complex coefficients a_n (information) are random variables belonging to a stationary process, characterized by the mean value $\mu_a = E[a_n]$ and the autocorrelation function $R_a(m-n)=E\left[a_n^* a_m\right]$.

- Autocorrelation function of the complex envelope (cyclostationary process of period T):

$$R_v(t,t+\tau)=E\left[\sum_{n=-\infty}^{+\infty} a_n^* g^*(t-nT)\sum_{m=-\infty}^{+\infty} a_m g(t-mT+\tau)\right]=\sum_n\sum_m R_a(m-n)g^*(t-nT)g(t-mT+\tau)$$

- Wiener-Khintchine theorem: the power spectrum is given by the Fourier transform of the average autocorrelation function $S_v(f)=\mathrm{F}\{\overline{R_v}(\tau)\}$. We have $\overline{R_v}(\tau)=\frac{1}{T}\int_{-T/2}^{T/2} R_v(t,t+\tau)dt=\frac{1}{T}\sum_{m=-\infty}^{\infty} R_a(m)R_g(\tau-mT)$ being $R_g(t)=g(t)\otimes g(-t)$.

(*) The actual signal is $s(t)=\mathrm{Re}\left\{v(t)e^{j2\pi f_c t}\right\}$, being f_c the carrier frequency.

Power Spectrum

- A useful equality : $\int_{-\infty}^{\infty}\sum_n \delta(t-nT)e^{-j2\pi ft}\mathrm{d}t = \sum_n e^{-j2\pi fnT} = \frac{1}{T}\sum_n \delta\left(f-\frac{n}{T}\right)$
- Power spectrum (general expression):

$$S_v(f) = \frac{|G(f)|^2}{T}\left(\sigma_a^2 + 2\,\mathrm{Re}\left[\sum_{m=1}^{\infty}\left(R_a(m)-|\mu_a|^2\right)e^{-j2\pi fmT}\right]\right) + \frac{|\mu_a|^2}{T^2}\sum_{m=-\infty}^{\infty}\left|G\left(\frac{m}{T}\right)\right|^2\delta\left(f-\frac{m}{T}\right)$$

- Independent data, with non-zero mean value:

$$R_a(m) = \begin{cases} |\mu_a|^2 + \sigma_a^2 & m=0 \\ |\mu_a|^2 & m\neq 0 \end{cases}$$ being $\sigma_a^2 = E\left[|a_m|^2\right] - |\mu_a|^2$ the data variance.

- $$S_v(f) = \frac{|G(f)|^2}{T}\sigma_a^2 + \frac{|\mu_a|^2}{T^2}\sum_{m=-\infty}^{\infty}\left|G\left(\frac{m}{T}\right)\right|^2\delta\left(f-\frac{m}{T}\right)$$

- Independent data, with zero mean value

$$R_a(m) = \begin{cases} \sigma_a^2 & m=0 \\ 0 & m\neq 0 \end{cases}$$

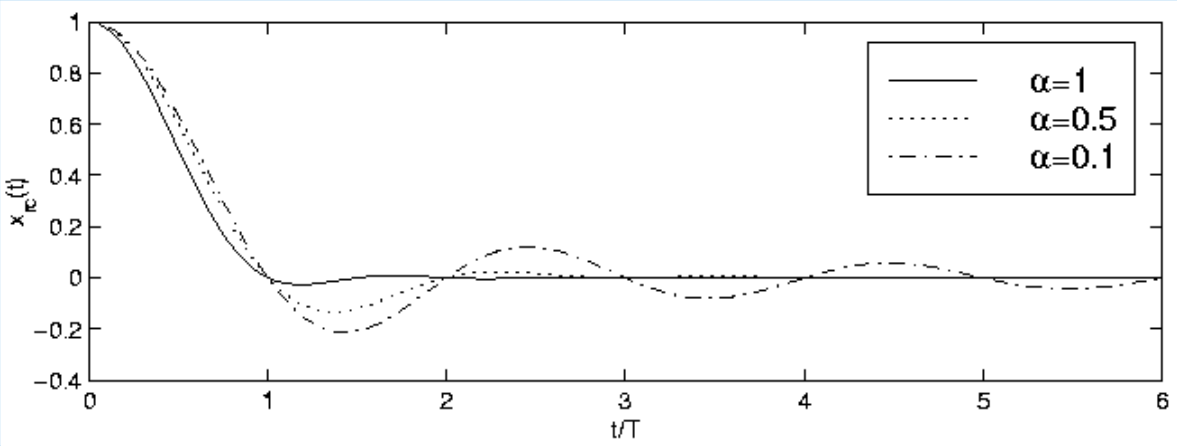
$$S_v(f) = \frac{|G(f)|^2}{T}\sigma_a^2$$

Raised Cosine

- The Raised Cosine pulse allows one to obtain a strictly limited bandwidth.

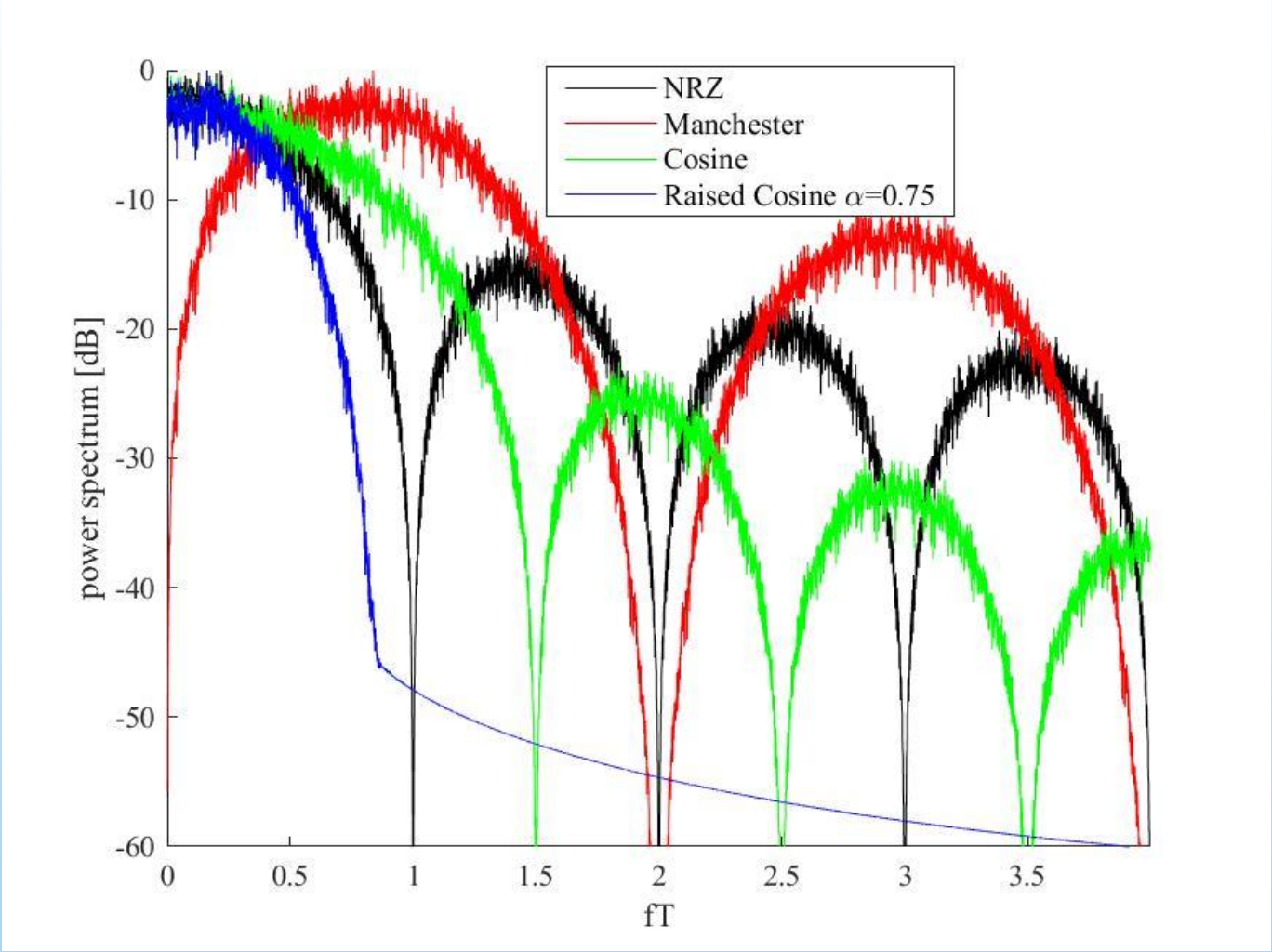
$$H_{RC}(f)=\begin{cases} T & |f|\le(1-\alpha)/2T \\ \dfrac{T}{2}\left(1+\cos\left(\pi\dfrac{T}{\alpha}\left(|f|-\dfrac{1-\alpha}{2T}\right)\right)\right) & (1-\alpha)/2T\le|f|\le(1+\alpha)/2T \\ 0 & |f|\ge(1+\alpha)/2T \end{cases}$$

- As α increases, the bandwidth increases, but also the speed with which the impulse response is damped increases (and therefore the problems arising from imperfect synchronism decrease). It is commonly adopted $0.5\le\alpha\le1$.



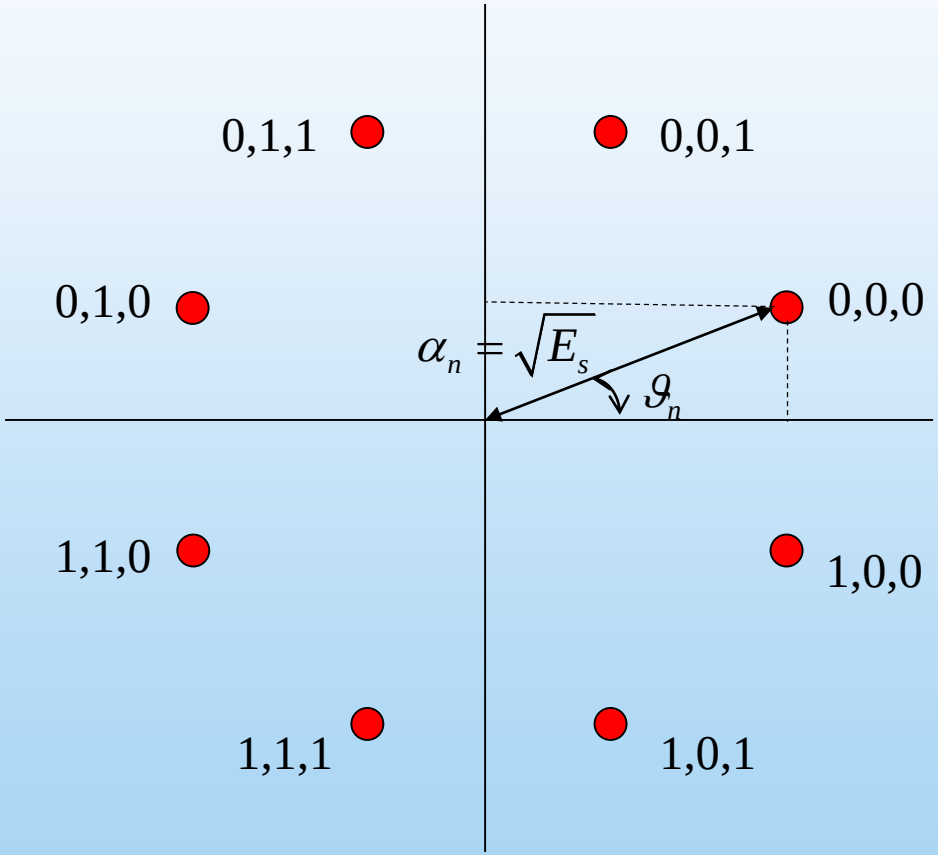
$$h_{RC}(t)=\frac{\sin(\pi t/T)}{\pi t/T}\frac{\cos(\alpha\pi t/T)}{1-(2\pi t/T)^2}$$

Power spectra



Phase Shift Keying example (M-PSK)

- Scattering diagram (8-PSK, with Gray *mapping*)



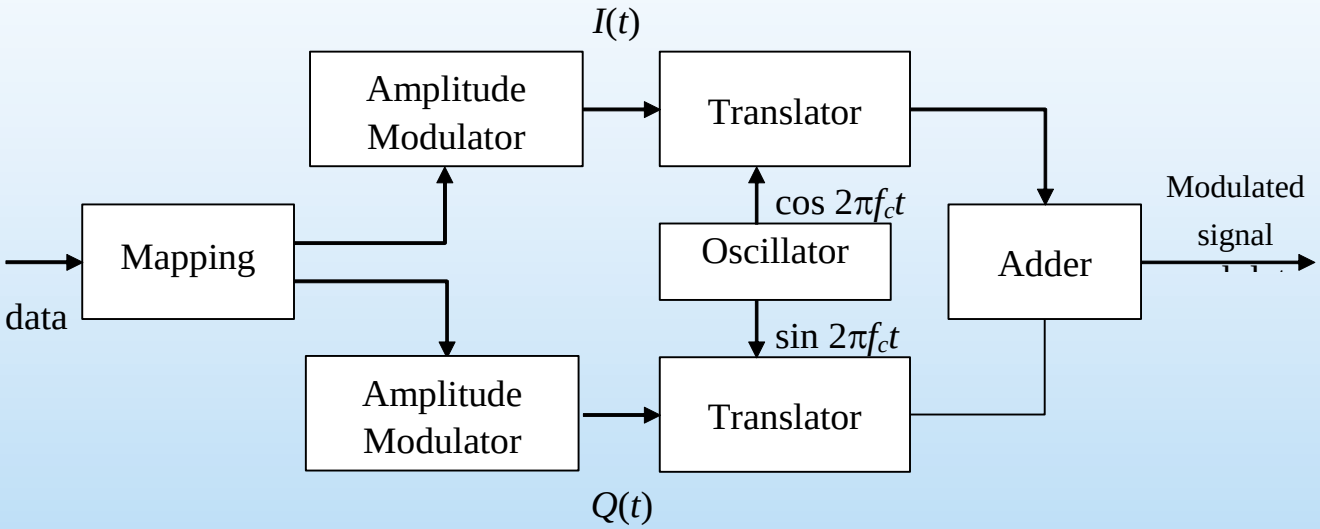
Gray mapping

Bit					Ordine
0	0	0	0	0	0
0	0	0	0	1	1
0	0	0	1	1	2
0	0	0	1	0	3
0	0	1	1	0	4
0	0	1	1	1	5
0	0	1	0	1	6
0	0	1	0	0	7
0	1	1	0	0	8
0	1	1	0	1	9
0	1	1	1	1	10
0	1	1	1	0	11
0	1	0	1	0	12
0	1	0	1	1	13
0	1	0	0	1	14
0	1	0	0	0	15

					Ordine
1	1	0	0	0	16
1	1	0	0	1	17
1	1	0	1	1	18
1	1	0	1	0	19
1	1	1	1	0	20
1	1	1	1	1	21
1	1	1	0	1	22
1	1	1	0	0	23
1	0	1	0	0	24
1	0	1	0	1	25
1	0	1	1	1	26
1	0	1	1	0	27
1	0	0	1	0	28
1	0	0	1	1	29
1	0	0	0	1	30
1	0	0	0	0	31

Phase, Amplitude-Phase modulator

- Modulator PSK, QAM



- The phase component, $I(t)$, and the quadrature components, $Q(t)$, are baseband, amplitude modulated signals.
- f_c is the carrier frequency (specified by the standard).
- Complex envelope: $v(t) = I(t) + jQ(t) = \sum_{n=-\infty}^{+\infty} a_n g(t - nT)$

Correlator (coherent) demodulator

- Determines the received vector components.
  Assume to transmit, in a generic signaling interval of duration T the waveform $s_m(t)$, and to receive $r(t)= s_m(t)+ n(t)$, being $n(t)$ the receiver noise (Additive White Gaussain AWGN model), with power N_0W watts. The correlator evaluates:

$$r_n = \langle r(t)\psi_n(t)\rangle = \int_0^T s_m(t)\psi_n(t)dt + \int_0^T n(t)\psi_n(t)dt = s_{mn} + n_n,\ n = 1,\ldots,N$$

- The noise vector components, n_n, are Gaussian, independent, zero mean random variables, having variance $N_0/2$.

$$E[n_k n_h] = E\left[\int_0^T n(t)\psi_k(t)dt \int_0^T n(t)\psi_h(t)dt\right] = \int_0^T \int_0^T E[n(t)n(\tau)]\psi_k(t)\psi_h(\tau)dtd\tau = \frac{N_0}{2}\delta_{hk}$$

- Therefore, the received vector components, r_n, are Gaussian, independent, random variables with mean s_{mn} (corresponding to the transmitted symbol vector components), and variance $N_0/2$

$$f(\mathbf{r}|\mathbf{s}_m) = \frac{1}{(\pi N_0)^{N/2}} \exp\left(-\frac{1}{N_0}\sum_{n=1}^{N}(r_n - s_{mn})^2\right)$$

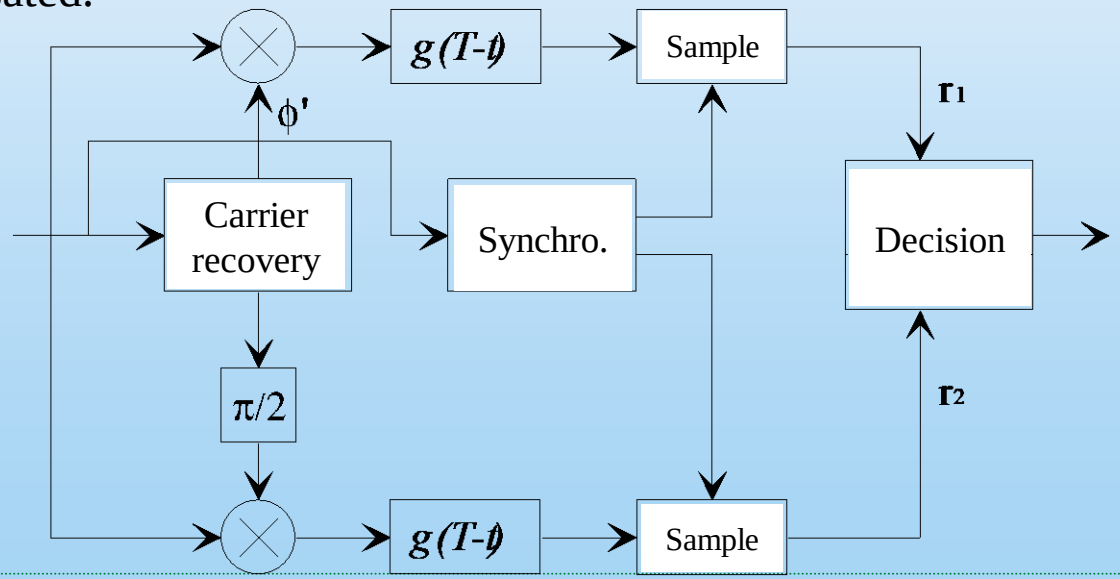
Matched filter demodulator

- Coherent matched filter demodulator

The channel introduces delay, t_d. Neglecting the effects of the distortion introduced by the channel and of the noise introduced by the receiver we have

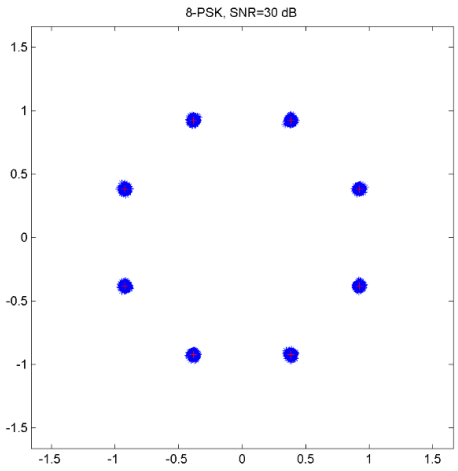
$$x_R(t) = x_T(t - t_d) = \mathrm{Re}\left\{ \sum_{n=-\infty}^{\infty} \alpha_n e^{j\theta n} g(t - t_d - nT) e^{j2\pi f_c (t - t_d)} \right\},$$

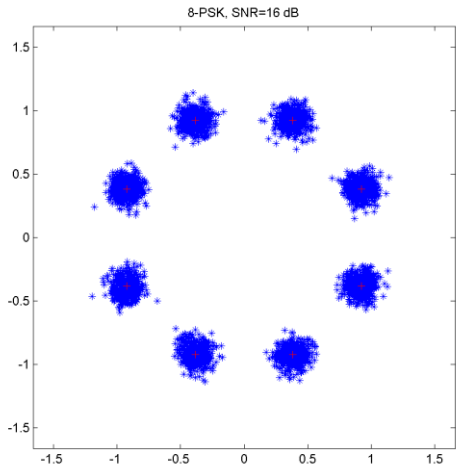
therefore the delay introduces a phase shift $\phi = 2\pi f_c t_d$ which must be compensated.



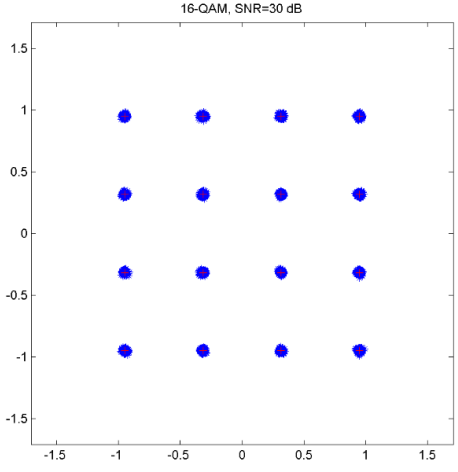
Received scattering diagrams

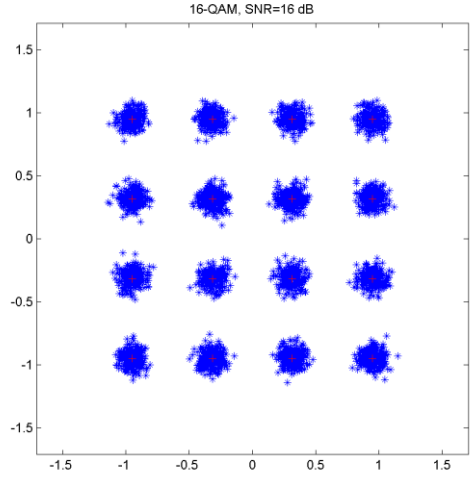
- 8 PSK





- 16 QAM





Hard Detection Theory

- The N-dimensional vector space to which $\mathbf{r}$ belongs, it is divided into M disjointed regions, A_m (Voronoi regions).

- Correct decision probability:

$$p_c = \sum_m p(\mathbf{r} \in A_m | \mathbf{s}_m) p(\mathbf{s}_m) = \sum_m p(\mathbf{s}_m) \int_{\mathbf{r} \in A_m} f(\mathbf{r}|\mathbf{s}_m) d\mathbf{r}$$

- MAP criterion: $\mathbf{r} \in A_m : m = \arg\max_j \left(f(\mathbf{r}|\mathbf{s}_j) p(\mathbf{s}_j) \right)$

- Binary choice $r \in A_0 : f(r|s_0)p(s_0) > f(r|s_1)p(s_1)$, equivalent to :
(the relationship between the conditional probabilities is called the likelihood ratio).

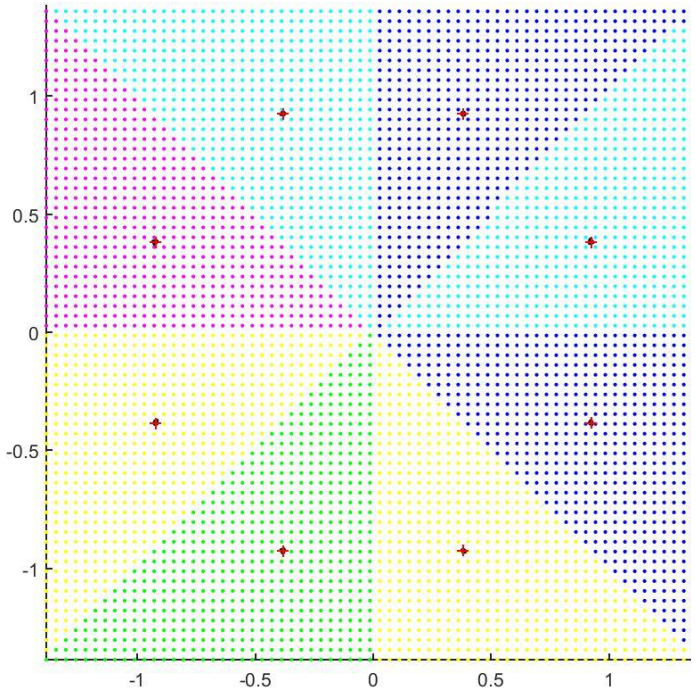
$$r \in A_0 : \frac{f(r|s_1)}{f(r|s_0)} > \frac{p(s_0)}{p(s_1)}$$

- If $p(s_0)=p(s_1)$, or if the two probabilities are not known, the ML (Maximum Likelihood or Minimum Distance) criterion is adopted:

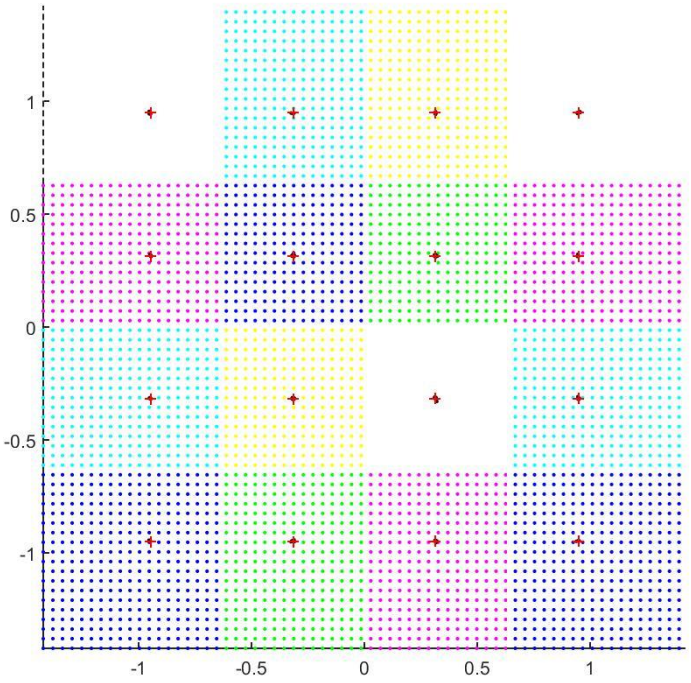
$$r \in A_0 : \frac{f(r|s_0)}{f(r|s_1)} > 1$$

Minimum distance detection regions

- 8-PSK



16 QAM

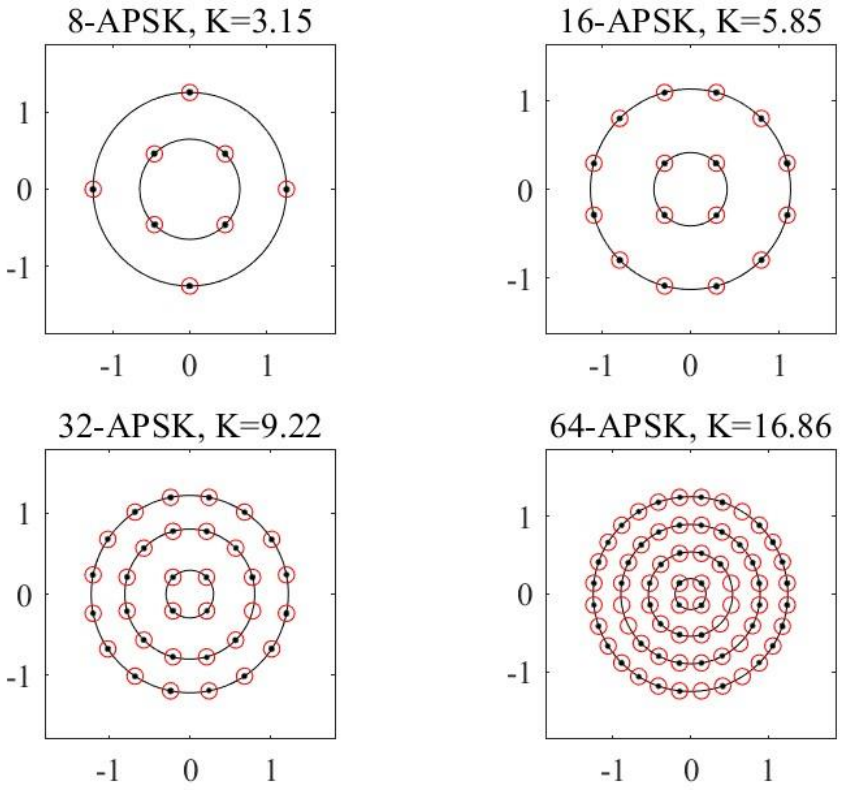


Hard detection performance (AWGN)

- Binary antipodal modulation (BPSK): $P_e = Q\left(\sqrt{2R_c \frac{E_b}{N_0}}\right) = \frac{1}{2}\operatorname{erfc}\left(\sqrt{R_c \frac{E_b}{N_0}}\right)$
- M-ASK: $p_E = \frac{2(M-1)}{M} Q\left(\sqrt{\frac{6R_c \log_2 M}{M^2-1} \frac{E_b}{N_0}}\right)$ $W_{\min} = f_N = \frac{1}{2T}$
- M-QAM: $p_{E_{M\cdot\text{QAM}}} = 1 - p_{C_{M\cdot\text{QAM}}} \approx 2 p_{E_{\sqrt{M}-\text{ASK}}} < 4Q\left(\sqrt{\frac{3R_c \log_2 M}{M-1} \frac{E_b}{N_0}}\right)$
- M-PSK $P_{E_{M-\text{PSK}}} \approx 2Q\left(\sqrt{2R_c \log_2 M \cdot E_b/N_0} \sin(\pi/M)\right)$ $W_{\min} = \frac{1}{T},\ r_{\max} = \log_2 M = b$
- M-FSK: $p_E \le (M-1)Q\left(\sqrt{\frac{E_s}{N_0}}\right) = (M-1)Q\left(\sqrt{R_c \frac{E_b}{N_0} \log_2 M}\right)$ $W_{\min} = \frac{M}{2T}, r_{\max} = \frac{b}{2^{b-1}}$

M-APSK

- Define R_i the radium of the i-th circle (square root of energy) and δ_{ii} the distance between two adjacent points on the same energy level.
- Constellation examples



4-12 APSK

$$d_{\min} = \delta_{11} = \delta_{22} = R_1\sqrt{2}$$

$$R_2 = R_1 / \left(\sqrt{2}\sin(\pi/12)\right)$$

$$E_s = KR_1^2$$

$$K = \frac{1}{4}\left(1 + \frac{3}{2\sin^2(\pi/12)}\right) \approx 5.85$$

$$p_E \approx 2Q\left(\sqrt{\frac{4R_c}{K}\frac{E_b}{N_0}}\right)$$

16-QAM comparison

2 energy levels instead of 3;
the required average energy
is 0.68 dB higher.

A comparison

- E_b/N_0 and E_{av}/N_0 required for obtaining a symbol error rate $P_e=10^{-6}$, as a function of the number of bits b, for different modulations with relevant energy levels /observe that, for PSK we have a single energy level). For all the modulation in the table $r_{max}=b$.

b	PSK		QAM			APSK		
	E_b/N_0	E_{av}/N_0	E_b/N_0	E_{av}/N_0	Levels	E_b/N_0	E_{av}/N_0	Levels
1	10.53	10.53						
2	10.78	13.78	10.78	13.79	1			
3	14.36	19.13						
4	18.96	24.98	15.00	21.02	3	15.68	21.70	2
5	23.97	30.96				17.65	24.64	3
6	29.19	36.97	19.47	27.25	6	20.28	28.06	4

Information theory

An information source

- Consider a source *X* that emits a value *x*, chosen within a given alphabet, with probability *p*(*x*).

- In the following, unless otherwise specified, only discrete sources are considered, which emit sequences of digits, binary or non-binary. Time-continuous sources can be made discrete over time by sampling, with negligible degradation if the sampling rate is chosen appropriately.

- Continuous sources in the amplitudes obtained by sampling analog sources are usually discretized by quantization. This is an operation that introduces an error of an easily predictable extent

Information

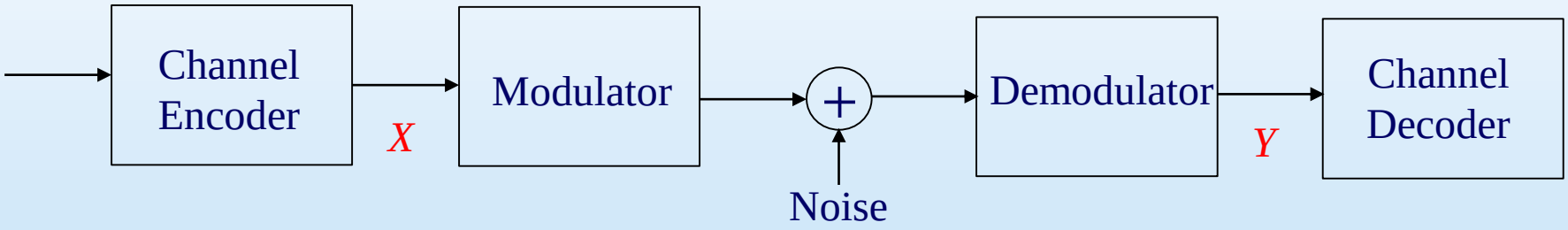
- (Self)information is a function of probability, $f(p)$, which satisfies the following requirements:
  - $p_A < p_B \Rightarrow f(p_A) > f(p_B)$: it is a decreasing function as the probability increases;
  - $f(0) = \infty$: the impossible event, if it occurs, provides infinite information;
  - $f(1) = 0$: the certain event, if it occurs, provides zero information;
  - given two events, A and B, statistically independent, is

$$f(p_A p_B) = f(p_A) + f(p_B)$$

- A function that satisfies the mentioned requirements is $f(p) = -c \log_a p$; the constants *c* and *a* were chosen so as to have $f(1/2) = 1$, (the unity of the information corresponds to the occurrence of one of two equally probable events). Therefore, a measure of information is expressed by the function $I(p) = -\log_2 p$; the unit of measurement is called "binary unit" or "bit".

Digital communication

Additive White Gaussian Noise Channel



Transmitted Information bits

Encoded bits

Transmitted signal

Received signal

Received samples

Received Information bits

The values of X belong to a (possibly) finite dimension alphabet $|X|$

Hard detection: the values of Y belong to finite dimension alphabet $|Y|=|X|$.

Soft detection: $|Y|>|X|$. $|Y|$ may be infinite (we assume that Y is a real number).

Entropy

- Given an experiment X, with M possible outcomes, the average information provided regarding the occurrence of one of the M possible events is called entropy. Therefore the entropy is given by the relation (it is a measure of uncertainty).

$$H(X) = E\left[I\left(p(x)\right)\right] = -\sum_{i=1}^{M} p_i \log_2 p_i \geq 0$$

- Joint and conditional entropy are defined similarly (logarithms are always in base 2).

$$H(X,Y) = E\left[-\log_2 p(x,y)\right] = -\sum_{x,y} p(x,y)\log_2 p(x,y)$$

$$H(X\,|\,Y) = E\left[-\log_2 p(x\,|\,y)\right] = -\sum_{x,y} p(x,y)\log_2 p(x\,|\,y) = -\sum_{y} p(y)\sum_{x} p(x|y)\log_2 p(x|y)$$

- Relationship between entropy, joint entropy and conditional entropy

$$H(X,Y) = H(X) + H(Y|X)$$

(the uncertainty associated with the pair X, Y is equal to the uncertainty associated with X added to the uncertainty associated with Y, when the outcome of X is known. If X and Y are independent $H(Y|X) = H(Y)$

Mutual Information

- A measure of the information shared between X and Y is called mutual information:

$$I(X;Y) = E\left[\log_2 \frac{p(x,y)}{p(x)\,p(y)}\right] \geq 0$$

- It results $I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$ (the mutual information between X and Y is equal to the uncertainty associated with X from which is subtracted the uncertainty associated with X when the outcome of Y is known; conditioning reduces uncertainty).

- If X and Y are independent, $I(X;Y) = 0$

Continuous channel

- Being the entropy infinite for a continuous random variable, for evaluating the mutual information define the differential entropy which, for the output is:

$$h(Y) = -\int f_Y(y)\log\left(f_Y(y)\right)dx = -E\left[\log\left(f_Y(y)\right)\right]$$

- Assume that Y is a zero mean random variable with variance σ_Y^2 . Name G a zero mean Gaussian variable with the same variance and $g(y)$ PDF. We have

$$-\int_y f(y)\log_2 g(y)\mathrm{d}y = \int_y f(y)\left(\log_2\sqrt{2\pi\sigma_y^2} + \frac{y^2}{2\sigma_Y^2}\log_2 e\right)\mathrm{d}y$$

$$= \log_2\sqrt{2\pi\sigma_y^2} + \frac{1}{2}\log_2 e = \frac{1}{2}\log_2\left(2\pi e\sigma_Y^2\right)$$

- Assuming $f(y)=g(y)$ we obtain the differential entropy of a Gaussian variable

$$g(Y) = \frac{1}{2}\log_2\left(2\pi e\sigma_Y^2\right)$$

Continuous channel

- Observe that

$$h(Y)-\frac{1}{2}\log_2\left(2\pi e\sigma_Y^2\right)=\int_y f(y)\log_2\frac{g(y)}{f(y)}\mathrm{d}y\le\log_2 e\int_y f(y)\left(1-\frac{g(y)}{f(y)}\right)\mathrm{d}y=0$$

- Equality holds if and only if $f(y)=g(y)$, i.e. in the Gaussian case.
- Once the variance σ_Y^2 is fixed, the entropy $h(Y)$ is therefore maximum in the case of Gaussian probability density.
- In this case, since $Y=X+N$, where N has Gaussian density, X is also a Gaussian random variable.
- The density $f(x)$ of the input which maximizes the mutual information and allows obtaining the channel capacity is therefore Gaussian.

Some useful channels

- Gaussian channel (or Gaussian additive channel): the output is the sum of the input and a Gaussian random variable with zero mean value and variance σ_N^2. It is the typical model of a transmission system in the presence of additive white Gaussian noise, for example with binary input, and soft detection.
- BSC: Binary Symmetric Channel: binary alphabet {0,1} both at input and output and error probability P(1|0)=P(0|1)=ε independent of the input. It is the typical model of a binary transmission system with hard detection.
- Binary Erasure Channel (BEC): binary alphabet at input and ternary at output {0,1,E} where E indicates complete uncertainty; P(E|0)=P(E|1)=*e*; P(0|0)=P(1|1)=1-*e*. Note that in this simple model it is assumed that the zeros and ones received are always correct.

Channel Capacity

- Consider memoryless channels. The statistical description of the transmission channel requires knowledge of:
  - input alphabet X (for example binary: x_i=0,1);
  - output alphabet Y (for example a real number y, or a pair of real numbers, or a decided bit y_i=0.1);
  - transition probability between input and output: probability density $f(y|x_i)$ in the continuous case, or probability $p(y_j|x_i)$ in the discrete case.
- To evaluate the behavior of the channel it is also useful to know:
  - the probability $p(x_i)$ of each entry (and if there was memory also the joint probabilities of subsequent entries).

Channel Capacity

- Channel capacity (information bits/channel use): $C = \max_{p(x)} I(X;Y)$

- Discrete input / Discrete output:

$$C = \max_{p(x)} H(X) - H(X|Y) = \max_{p(x)} H(Y) - H(Y|X)$$

- Discrete input / continuous output:

$$C = \max_{p(x)} H(X) - H(X|Y) = \max_{p(x)} h(Y) - h(Y|X)$$

- Continuous input / continuous output:

$$C = \max_{f(x)} h(X) - h(X|Y) = \max_{f(x)} h(Y) - h(Y|X)$$

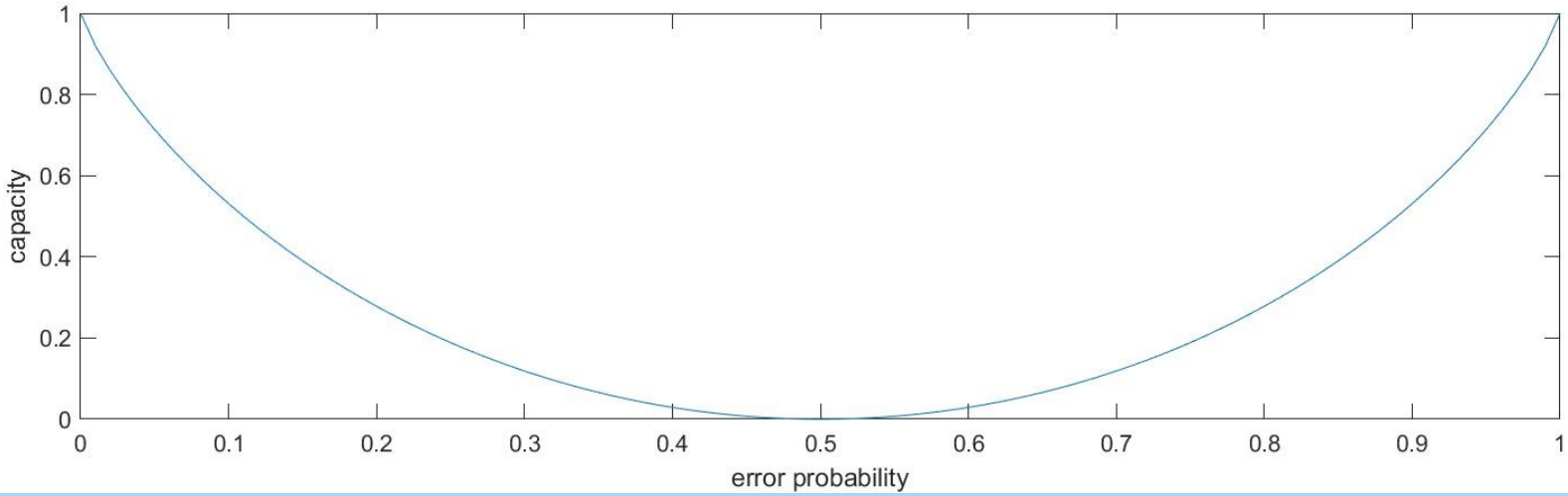
Shannon's theorem on capacity

- Channel capacity, C, limits the maximum rate, R (in information bits per channel use), at which information can be reliably transmitted over the channel. Consider an encoded source that emits information at a rate R.
- Negative statement: if $R>C$, considered a sufficiently long sequence of bits of length N, $M = 2^{NR}$ equiprobable messages are obtained. Whichever way the associated waveforms are chosen, the probability of error tends towards 1.
- Positive statement: if $R<C$, considered a sufficiently high sequence of bits of length N, it is possible to identify a set of waveforms that allows obtaining an arbitrarily small error probability.
- Alternative formulation: If $R<C$, given ε, there exists a code of sufficiently long length for which $p_e < \varepsilon$, p_e being the error probability conditioned on the transmission of the i-th message.

Hard capacity: symmetric binary channel

- Binary discrete input (symbols '0_T' e '1_T', with probability p and 1-p).
- Discrete binary output ('0_R' e '1_R') (hard detection).
- Error probability: ε=P('0_R'| '1_T')=P('1_R'| '0_T'). $C = 1 + \varepsilon \log_2 \varepsilon + (1-\varepsilon)\log_2 (1-\varepsilon)$

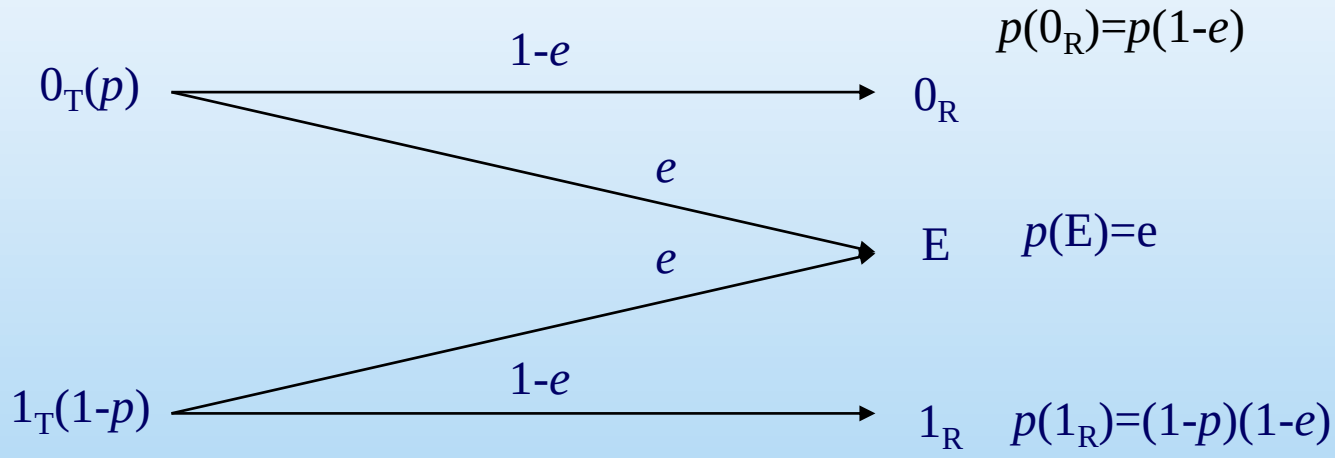
Antipodal binary modulation: $\varepsilon = Q\left(\sqrt{2\frac{E_s}{N_0}}\right)$ being $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty \exp\left(-\frac{u^2}{2}\right) du$



The Erasure Channel

- Binary discrete input (symbols '0_T' e '1_T', with probability p and 1-p).

- An input value may be received correctly or it may be erased with probability e.



- Capacity:

$$C = 1-e \ \frac{\text{information bits}}{\text{channel use}}$$

Additive White Gaussian Noise channel

- Consider $Y=X+N$, with N Gaussian r.v., independent from X, with zero mean and of variance σ^2_N.

$$I(X;Y)=h(Y)-h(X+N|X)=h(Y)-h(N)$$

$$C=\max_{f(x)} I(X;Y)=\frac{1}{2}\log_2\left(2\pi e\sigma_Y^2\right)-\frac{1}{2}\log_2\left(2\pi e\sigma_N^2\right) \qquad \sigma_Y^2=\sigma_X^2+\sigma_N^2$$

$$C=\frac{1}{2}\log_2\left(1+\frac{\sigma_x^2}{\sigma_z^2}\right)=\frac{1}{2}\log_2\left(1+\frac{S}{N}\right)=\frac{1}{2}\log_2\left(1+\frac{2E_s}{N_0}\right) \quad \frac{\text{information bits}}{\text{channel use}}$$

AWGN Shannon bound

- Shannon Theorem: for a *N*-dimensional channel, for a reliable transmission, the coded modulation rate (for every modulation) is bounded as follows:

$$R < C = \frac{N}{2}\log_2\left(1+\frac{2(E_s/N)}{N_0}\right)\ \frac{\text{information bits}}{\text{channel use}}$$

- If follows that the source rate, $R_s=R/T$, is bounded as follows:

$$R_s < \frac{C}{T} = \frac{N}{2T}\log_2\left(1+\frac{2T}{N}\frac{RE_b}{TN_0}\right) = W\log_2\left(1+\frac{1}{W}R_s\frac{E_b}{N_0}\right)\frac{\text{information bits}}{\text{second}}$$

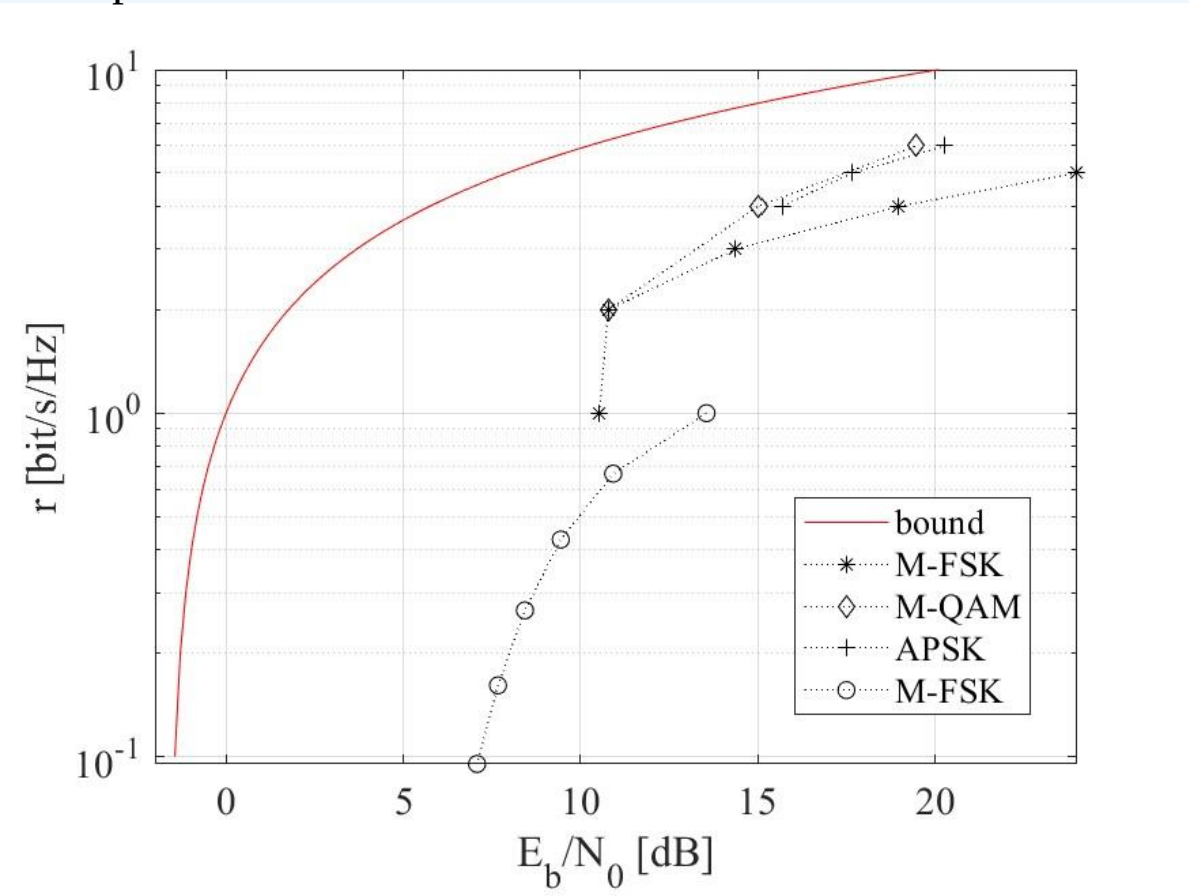
- The spectral efficiency is bounded as follows

$$r_{\max} = \frac{R_{s_{\max}}}{W} = \log_2\left(1+r_{\max}\frac{E_b}{N_0}\right)\frac{\text{information bits}}{\text{second}\cdot\text{Hz}}$$

- From which we obtain $\left(\frac{E_b}{N_0}\right)_{\min} = \frac{2^r-1}{r}$ (Shannon bound)

Comparison with Shannon bound

- E_b/N_0 required for obtaining a symbol error rate $P_e=10^{-6}$. With hard detection without channel coding there is a large gap between the actual and the achievable performance.



Soft detection

- Square root energy signal space: $f\left(\mathbf{r}\middle|\mathbf{s}_j\right)=\frac{1}{\left(\pi N_0\right)^{N/2}}\exp\left(-\frac{1}{N_0}\sum_{h=1}^{N}\left(r_h-s_{jh}\right)^2\right)$
-
- Change of variable: $y_h=\frac{r_h}{\sqrt{N_0}},\qquad x_{jh}=\frac{s_{jh}}{\sqrt{N_0}}$
- Square root SNR signal space: $f\left(\mathbf{y}\middle|\mathbf{x}_j\right)=\frac{1}{\left(\pi\right)^{N/2}}\exp\left(-\sum_{h=1}^{N}\left(y_h-x_{jh}\right)^2\right)$
- Log-Likelihood Ratio antipodal binary modulation: (to be used as the input of the channel decoder) $\log\frac{f\left(y\middle|x_0\right)}{f\left(y\middle|x_1\right)}=4y\sqrt{\frac{E_s}{N_0}}$

Soft demodulation

- Pragmatic approach
- Consider a one(two)-dimensional modulation with b bits, being $M=2^b$.
- Let $\mathbf{r}$ be the value of the observed vector. The likelihood ratio of the i-th bit, with $i=1,\ldots,b$ is given by:

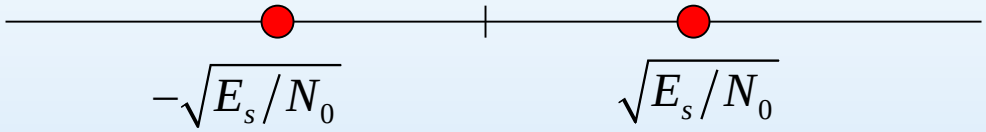
$$\Lambda(x_i) = \log\left(\frac{P[\mathbf{r}|x_i=1]}{P[\mathbf{r}|x_i=0]}\right).$$

- Applying Bayes, in the hypothesis of equiprobability of the sequences of b bits present on the channel, let $\mathbf{s}_{j,1}$ be the j-th sequence with i-th bit equal to 1, and $\mathbf{s}_{j,0}$ the one with i-th bit equal to 0, we obtain:

$$\Lambda(x_i) = \log\left(\frac{\sum_{j=1}^{2^{b-1}} \exp\left(-\frac{\|\mathbf{r}-\mathbf{s}_{j,1}\|^2}{2\sigma^2}\right)}{\sum_{j=1}^{2^{b-1}} \exp\left(-\frac{\|\mathbf{r}-\mathbf{s}_{j,0}\|^2}{2\sigma^2}\right)}\right).$$

Antipodal modulation soft capacity

- Consider now a non-quantized output, and the Square root SNR signal space.



$$f(y) = \frac{1}{2\sqrt{\pi}} \exp\left(-\left(y - \sqrt{E_s/N_0}\right)^2\right) + \frac{1}{2\sqrt{\pi}} \exp\left(-\left(y + \sqrt{E_s/N_0}\right)^2\right)$$

$$= \frac{1}{\sqrt{\pi}} \exp\left(-y^2 - E_s/N_0\right) \cosh\left(2y\sqrt{E_s/N_0}\right).$$

$$f(y|x) = \frac{1}{\sqrt{\pi}} \exp\left(-y^2\right)$$

$$C = -\int_{-\infty}^{\infty} f(y) \log_2 f(y) \, dy - \frac{1}{2}\log_2(\pi e) \quad \frac{\text{information bits}}{\text{channel use}}$$

BPSK-QPSK soft capacity

- BPSK

$$C = 1 - \exp\left(-a_1 \left(\frac{E_s}{N_0} \right)^{a_2} + a_3 \right) \quad \frac{\text{bit infor.}}{\text{simbolo}}$$

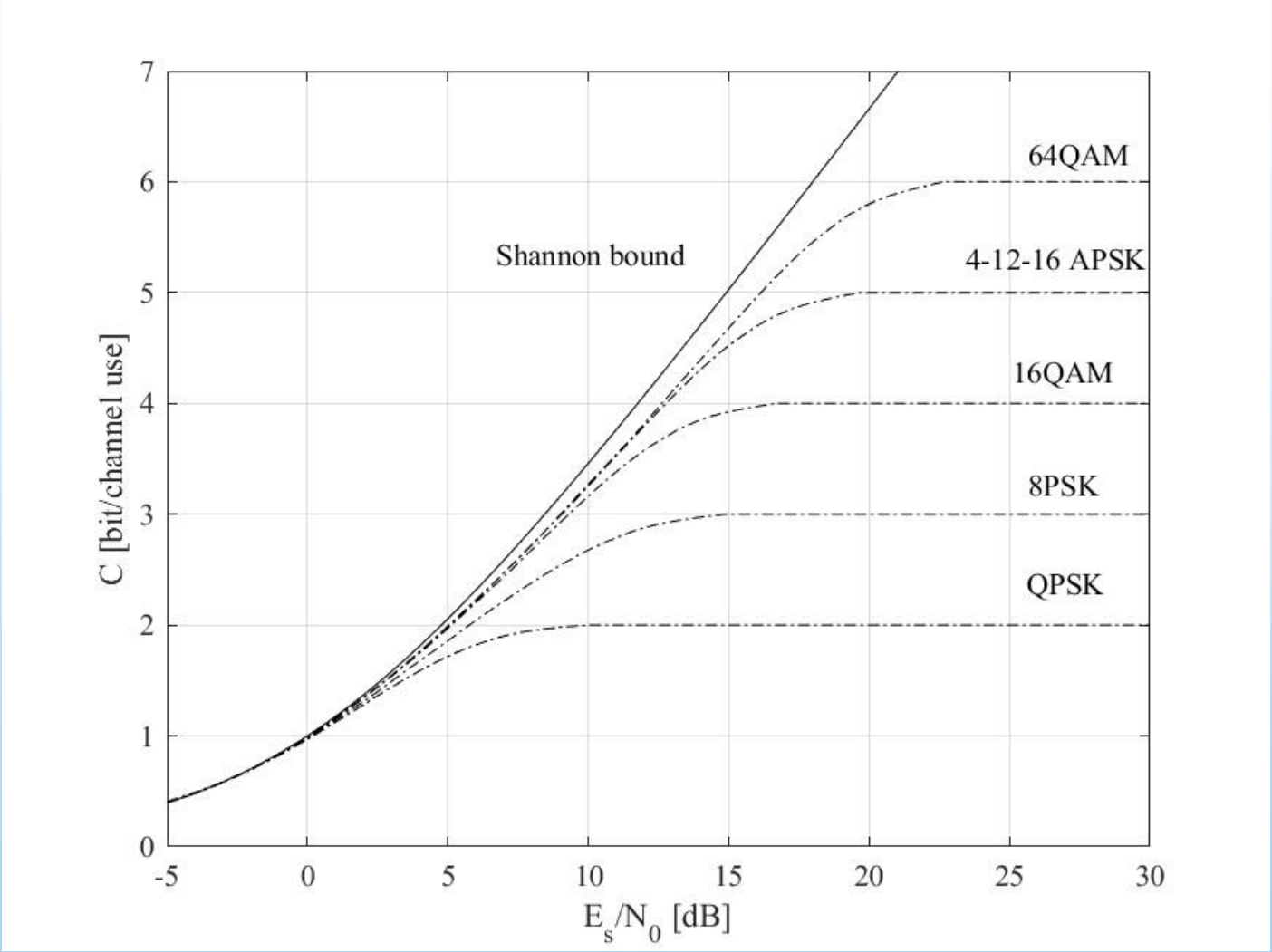
$$a_1 \approx 1.286,\ a_2 \approx 0.931,\ a_3 \approx 0.010$$

- QPSK

$$C = 2\left[1 - \exp\left(-a_1 \left(\frac{1}{2} \frac{E_s}{N_0} \right)^{a_2} + a_3 \right) \right] \quad \frac{\text{bit infor.}}{\text{simbolo}}$$

F. Babich, A. Soranzo, and F. Vatta, "Useful mathematical tools for capacity approaching codes design," IEEE Commun. Lett., vol. 21, no. 9, pp. 1949–1952, Sep. 2017.

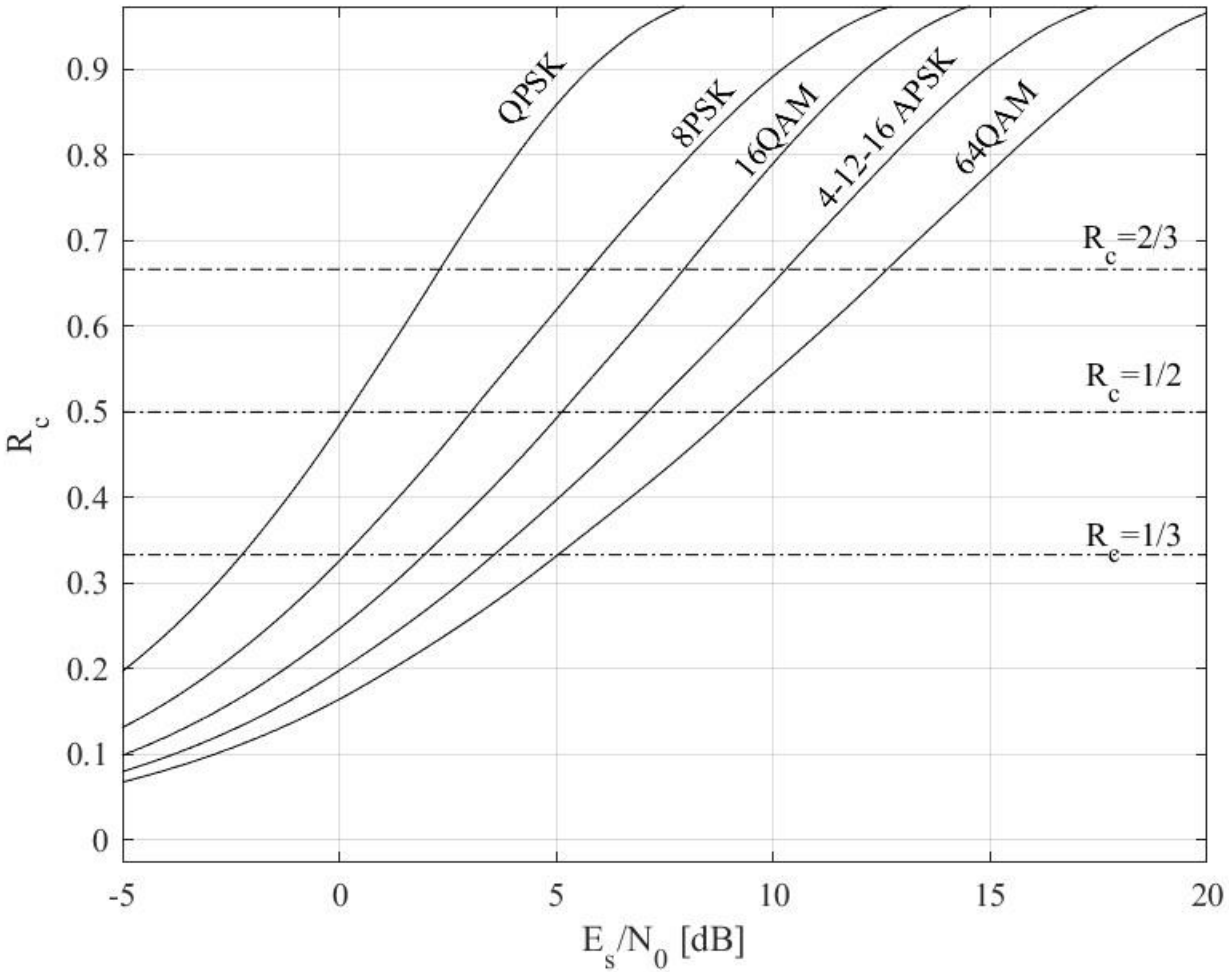
Capacity of bidimensional modulations



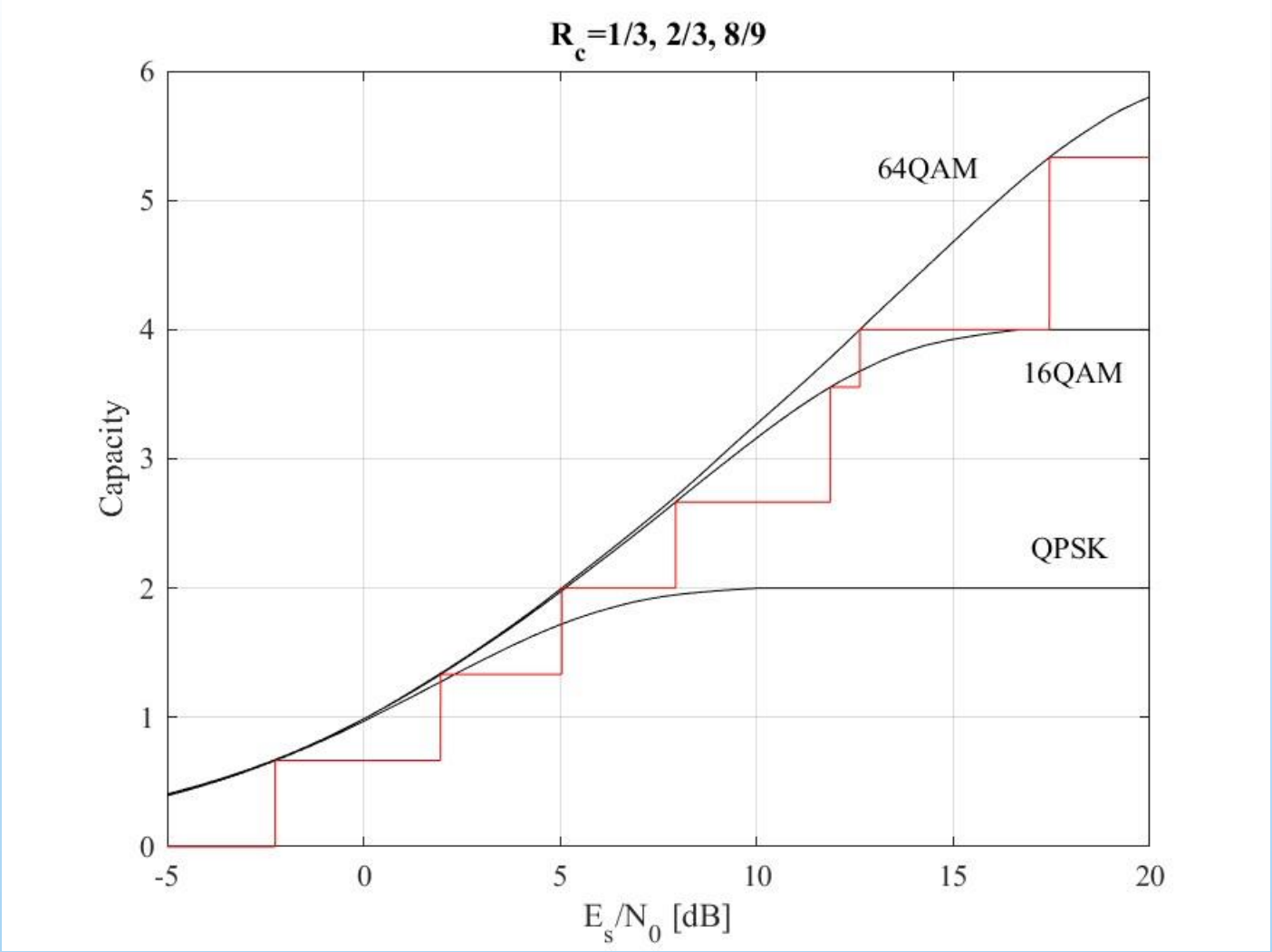
Channel encoding

- Channel encoding introduces redundancy to protect the digital signal from errors.
- It may be used to detect the presence of errors in a message (associated with retransmission algorithms, Automatic Repeat reQuest – ARQ).
- It may be also be used to prevent the occurrence of errors (Forward Error Correction - FEC).
- Encoding: k compressed source bits are associated with n bits for transmission on the channel, with $n>k$; redundancy is thus systematically introduced that can be used in reception to detect and/or prevent errors.
- Code rate: $R_c=k/n<1$. Let E_b be the energy used to transmit a bit of information. Therefore, the energy used to transmit a bit on the channel is equal to $E_b R_c$.
- From the bound on $R=R_c b=R_c \log_2(M)$, we may determine a bound on R_c.

Determining required Code Rate

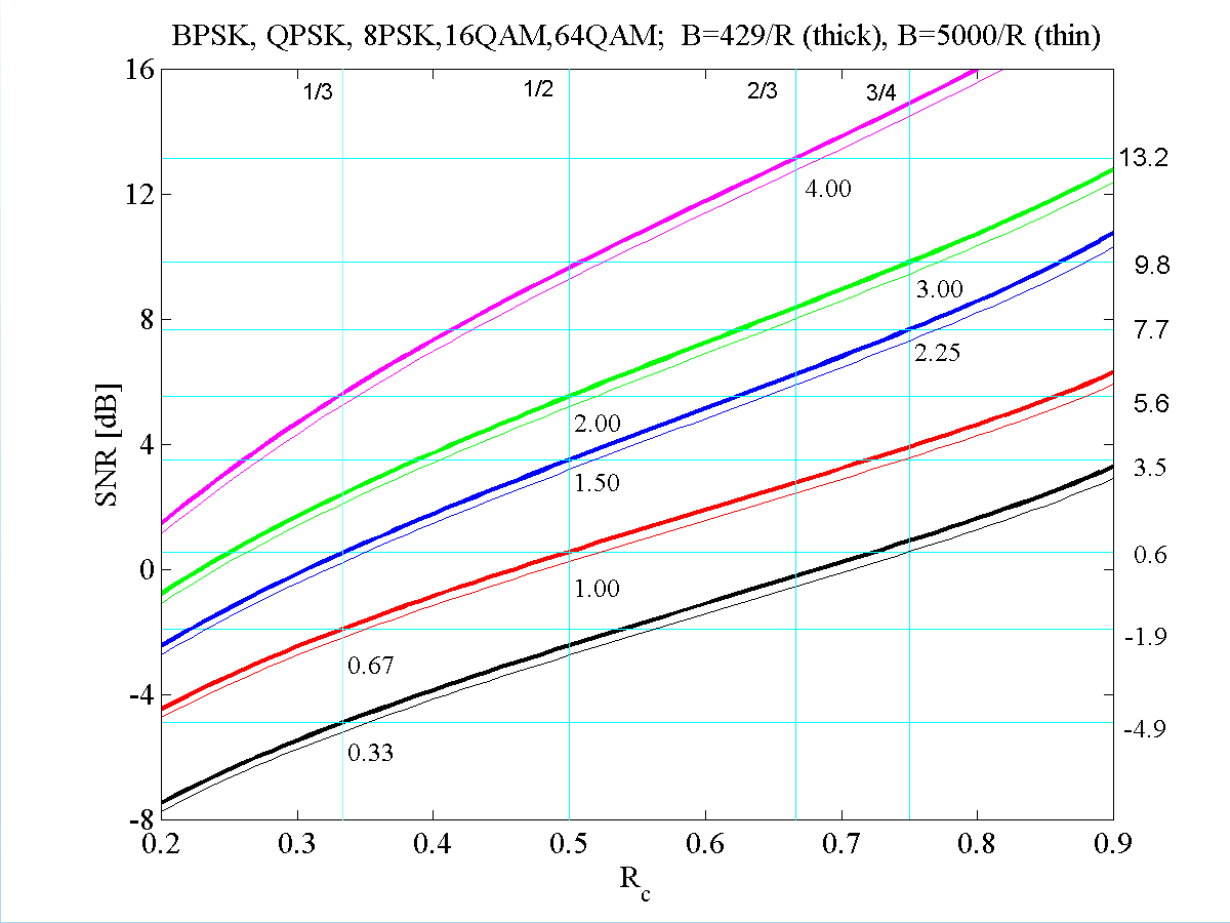


Adaptive System example

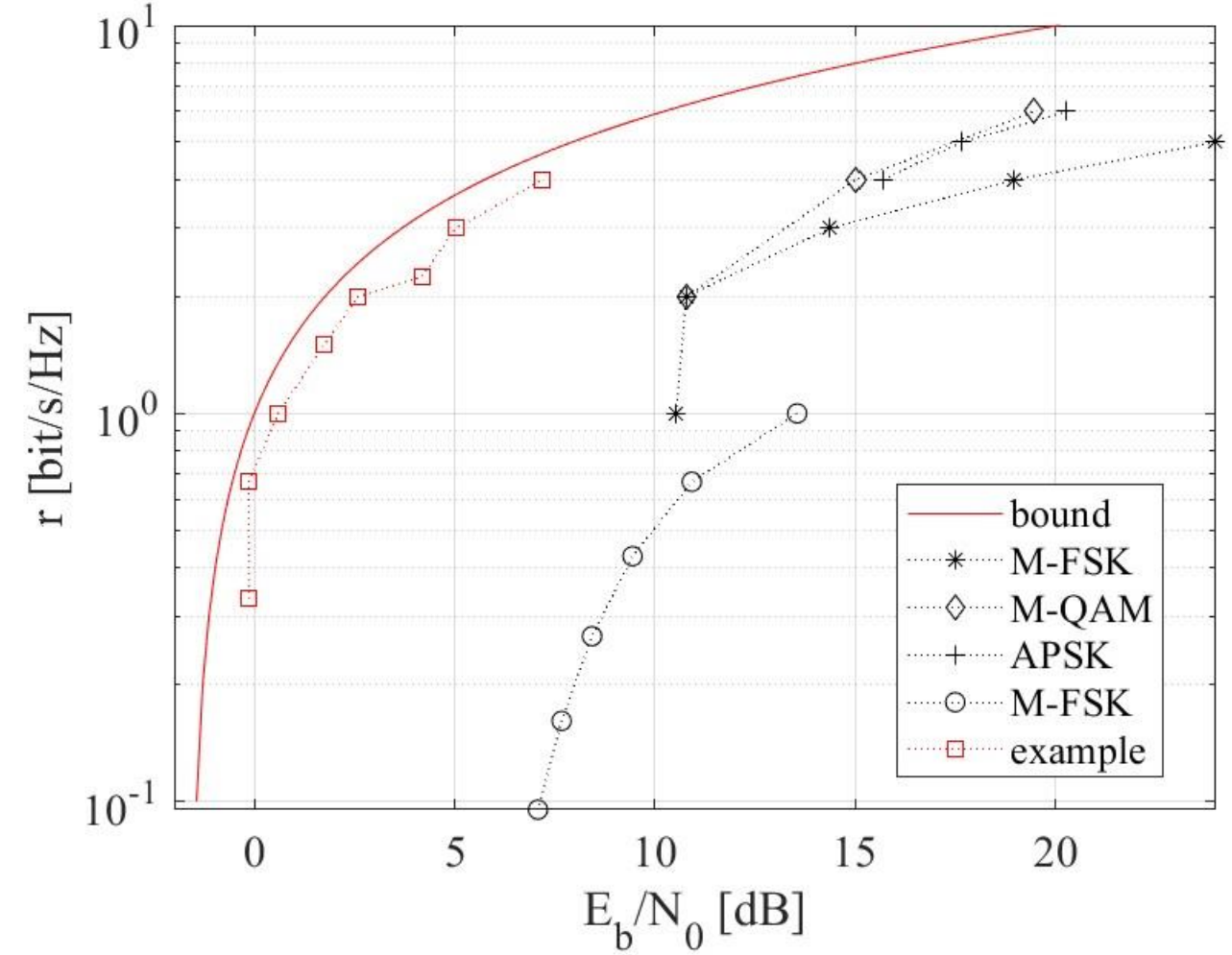


SNR threshold design (ON/OFF model)

- A possible set of thresholds, for block length N=429/R and N=5000/R, for R_c=1/3, 1/2, 2/3, 3/4.



Shannon bound comparison



Error Protection Techniques

Error protection techniques

- Error detection codes and retransmission algorithms (Automatic Repeat reQuest - ARQ).
- Self-correcting error codes (Forward Error Correction - FEC).
- Coding: *k* bits of information are associated with *n* bits for transmission on the channel, with *n*>*k*; redundancy is thus systematically introduced which can be used in reception to detect and/or correct errors.
- In the time *k* user bits are generated, *n* channel bits must be transmitted; therefore the time for the transmission of a bit is reduced and, consequently, the used bandwidth increases.
- Block codes: The *n*-*k* redundancy bits depend only on the current *k* user bits; the coding operation is, therefore, without memory.
- Convolutional codes: The *n*-*k* bits of redundancy depend on *k* current user bits and on (*N*-1)*k* previous user bits (*N* is called constraint length); the coding operation is, therefore, with memory.

Linear binary block codes

- The *n*-bit sequences produced by the encoder are called code words.
- The codes we will consider are linear codes, in which the sum (modulo 2) of two code words is still a code word.
- Weight of a code word: it is the number of bits other than 0 of the code word.
- Hamming distance between two code words: it is the number of bits in which they differ; it is also the weight of the word obtained by adding the code words modulo 2. Thus, the weight of the least weight word represents the minimum distance, d_{min}, between two codewords.

Performance of linear binary block codes

- Assume hard detection. The performance depend on the minimum distance. The decoding operation generally takes place in maximum likelihood, i.e. associating the closest code word to the n-tuple received.
- Consequently, a code with minimum distance $d_{\min}$ can correct at most $t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$ weight errors, and reveal at most $d_{\min}$ weight errors.
- A code is used to correct $t_c \leq \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$ weight errors at most, and detect l weight errors, being $t_c + 1 \leq l \leq d_{\min} - 1 - t_c$
- Perfect codes: each n-tuple has at least one code word at distance $\left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$. For them the relationship holds: $2^n = 2^k \sum_{i=0}^{t} \binom{n}{i}$
- The repetition codes $(n,1)$, with n odd ($d_{\min} = n$), the Hamming codes ($d_{\min} = 3$), and the Golay code ((23,12), $d_{\min} = 7$) are perfect codes.

Cyclic codes

- They are linear codes.
- A cyclic permutation of a codeword produces a codeword.
- Each binary sequence is associated with a polynomial whose coefficients are the bits that make up the sequence.
- Redundancy bits are found using a division algorithm. The divisor polynomial is denoted by $g(x)$, and has degree n-k. Therefore a cyclic code is characterized by n, k, $g(x)$.
- Systematic encoding.
  - The k-ple to be encoded is associated with a polynomial $u(x)$ of degree n-1, whose k most significant coefficients coincide with the user bits, while the other n-k bits are set to 0. Calculate the remainder, $r(x)$, of dividing u(x) by $g(x)$. The coefficients of $r(x)$ are the redundancy. The n-ple thus obtained is associated with a polynomial $c(x)$, divisible by g(x).
- The received sequence is associated with the polynomial $c_r(x)$. We divide by $g(x)$. If the remainder (syndrome) is null it is assumed that there have been no errors.

Cyclic codes for error detection

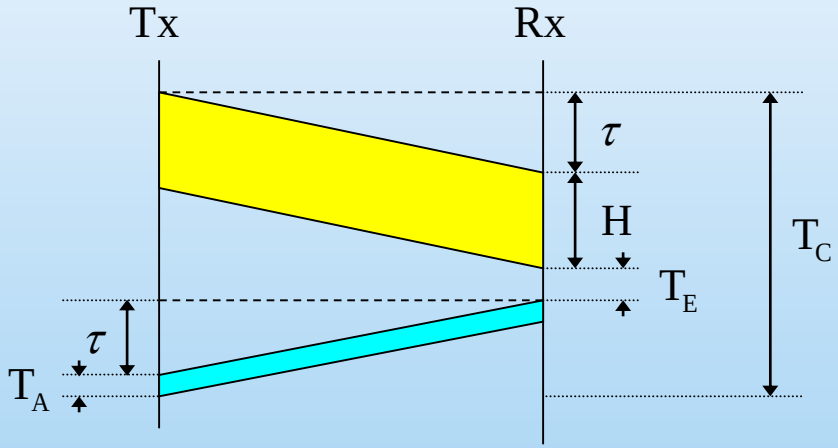
- Cyclic Redundancy Check (CRC)
- The generating polynomial is of the type $g(x)=g_1(x)(x+1)$, where $g_1(x)(x+1)$ is a polynomial of degree $m=n-k-1$ chosen so as to be a divisor of $x^{2^m-1}+1$, but not to be a divisor of any polynomial of the type x^h+1, with $h<2m-1$.
- With such choices, the code allows revealing all errors of odd weight (no polynomial with an odd number of terms can be divided by $x+1$) and all errors of weight 2 if $n<2^m-1$.
- The maximum code rate is $R_c = \left(2^m - m - 3\right) / \left(2^m - 2\right)$
- CRC is used by modern coding techniques to verify the correct decoding of a block.

CRC - Examples

$n-k$ [bit]	Generator	n [bit]	k [bit]	$R_c = \frac{k}{n}$	Org.
5	$g(x) = x^5 + x^3 + x + 1$	<15	<10	<2/3	ITU-T
8	$g(x) = x^8 + x^7 + x^3 + x^2 + 1$	<127	<119	<119/127	ITU-T
16	$g(x) = x^{16} + x^{12} + x^5 + 1$	<32767	<32751	≅1	
32	$g(x) = x^{32} + x^{26} + x^{23} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$	$\cong 10^9$	$\cong 10^9$	≅1	IEEE
64	$g(x) = x^{64} + x^4 + x^3 + x + 1$	$\cong 10^{18}$	$\cong 10^{18}$	≅1	ISO

Automatic Repeat reQuest

- Error detection.
- Bidirectional communication: availability of a return channel (feedback) on which ACKnowledgement (ACK) packets can be sent.
- Service tolerance for delay.



τ : propagation delay

2τ: round trip delay

H : packet transmission time

T_E : Elaboration time

T_A: ACK transmission time

T_C: cycle duration

$$T_C = 2\tau + H + T_E + T_A \Rightarrow \boxed{T_C \cong 2\tau + H}$$

$$T_E, T_A << \tau, H$$

The sustainable rate

- Assume that a codeblock of size N is transmitted in changing channel conditions.
- Assume that the codeblock may be divided into segments, each of which experiences a constant channel condition. With this assumption, the code performance may be determined as follows.
- Name with $[n_1, n_2, \ldots, n_L]$ the sequence of segment lengths and with $[\gamma_1, \gamma_2, \ldots, \gamma_L]$ the sequence of the relevant SNRs.
- Define sustainable code rate as the weighted average value of the equivalent code rates, that is given by

$$R_s = \sum_{j=1}^{L} \frac{R_j n_j}{N}$$

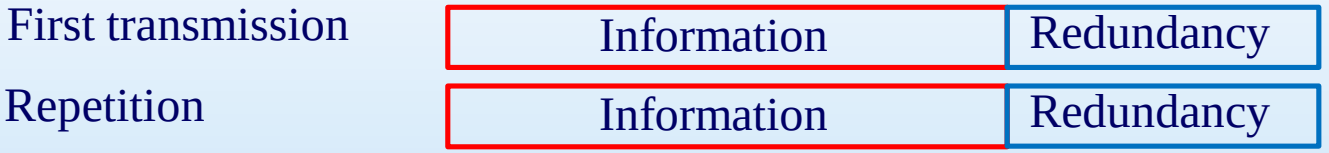
in which R_j is the equivalent rate of the j-th segments, which, assuming the ON-OFF model, represents the maximum rate which can be used with the SNR γ_j and block size N.

- The sustainable rate may be seen as the maximum rate that can be successfully used, given the block size the segment length sequence, and the SNR sequence.

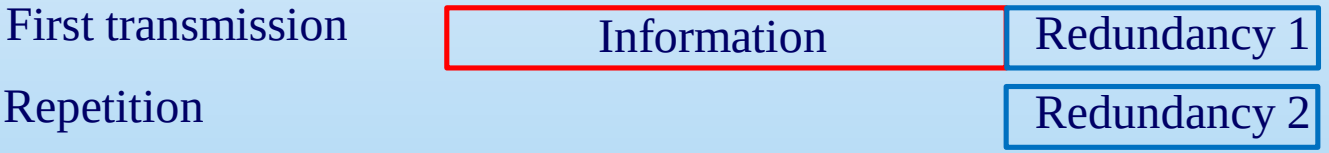
F. Babich, "On the Performance of Efficient Coding Techniques Over Fading Channels," IEEE Transactions on Wireless Communications, vol. 3, no. 1, pp. 290–299, Jan. 2004.

Hybrid ARQ

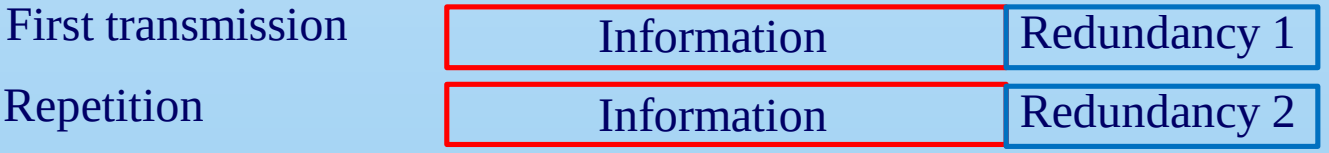
- Channel coding and repetition algorithm interact.
  - *Repetition Hybrid* ARQ *with soft (Chase) combining*: assume systematic encoding. The soft values of the received replicas are stored and combined to increase the probability of correct decoding.



  - *Incremental Hybrid* ARQ: Repetitions only include additional redundancy.



  - *Complementary Hybrid* ARQ: replications include information (systematic coding) and additional redundancy



Hybrid ARQ performance (I)

- Parameters
  - b: bit per symbol.
  - k: information bits
  - p_1, p_2 : redundancy bit first, second transmission.
  - $R_{c1}=k/(k+p_1)$: code rate first transmission.
  - $R_{c2}=k/(k+p_1+p_2)$: code rate second transmission.
  - γ_1, γ_2 : SNR first, second transmission.
  - $R_1=R_{c1}b$, $R_2=R_{c2}b$: rate first, second transmission.
- Limiting performance
- First transmission success condition:
  $R_1 < \log_2(1+\gamma_1)$, from which $\gamma_1 > 2^{R_1} - 1$
- Repetition ARQ (Chase combining): in case of failure of the first transmission, the limiting condition for the success of the second is :
  $R_1 < \log_2(1+\gamma_1+\gamma_2)$, from which $\gamma_2 > 2^{R_1} - 1 - \gamma_1$

Hybrid ARQ performance (II)

- Incremental ARQ: in case of failure of the first transmission be
  - $\alpha = \dfrac{k + p_1}{k + p_1 + p_2} = \dfrac{R_{c_2}}{R_{c_1}}$: fraction of bits sent in the first transmission;
  - limiting condition for the success of the second is :

    $R_2 < \alpha \log_2(1+\gamma_1) + (1-\alpha)\log_2(1+\gamma_2)$ from which: $\gamma_2 > 2^{\frac{R_2 - \alpha \log_2(1+\gamma_1)}{(1-\alpha)}} - 1$

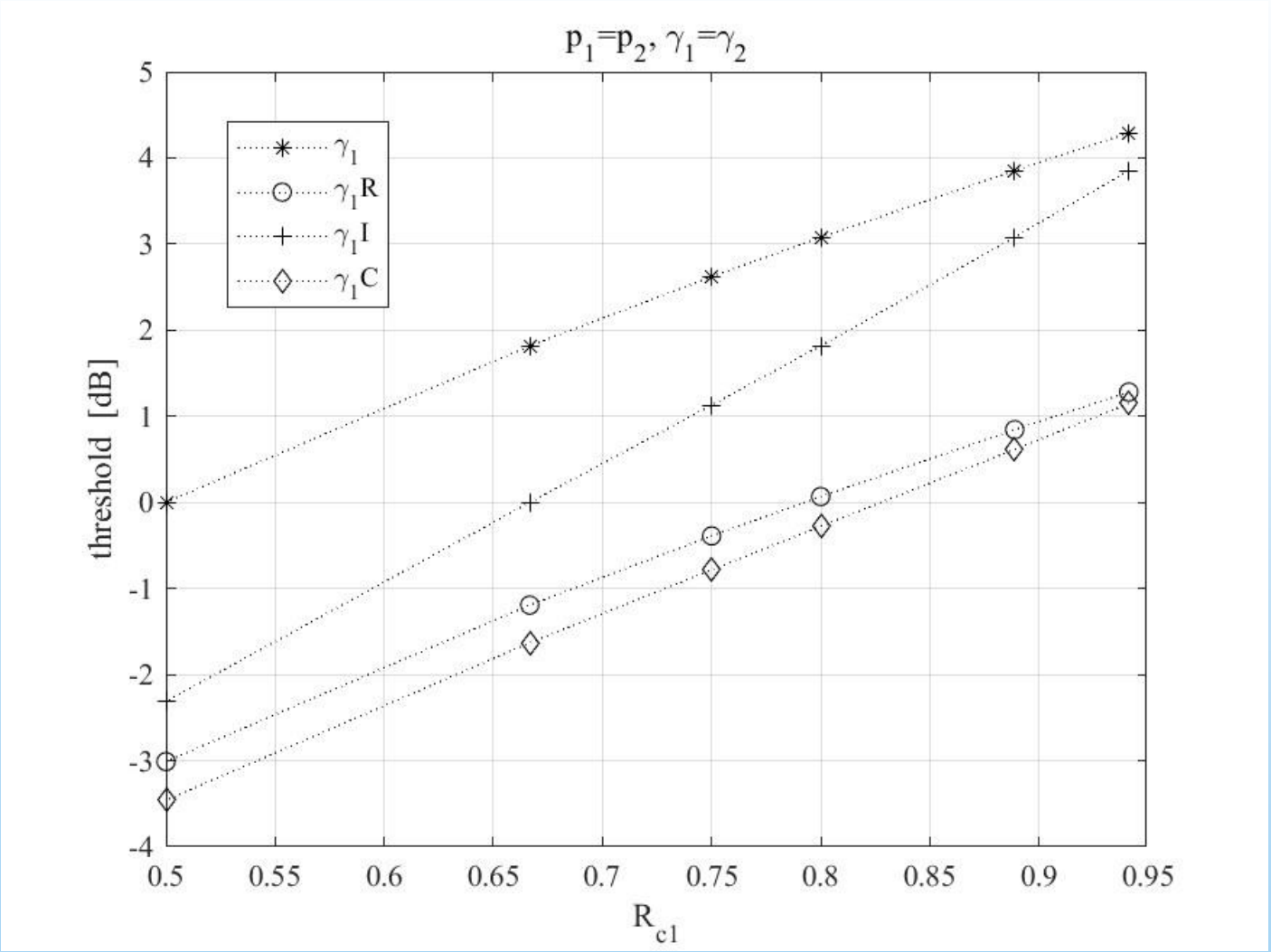
    if $p_2 = p_1$, $R_{c2}=R_{c1}/(2-R_{c1})$, we have: $\gamma_2 > 2^{\frac{bR_{c_1} - \log_2(1+\gamma_1)}{(1-R_{c_1})}} - 1$

- *Complementary* ARQ : in case of failure of the first transmission be

  $\alpha = R_{c_2}$ the fraction of bits of information, received with $\gamma_T = \gamma_1 + \gamma_2$;
  - hypothesizing $p_1 = p_2$, be $\beta=(1-\alpha)/2$ the fraction of parity received with both γ_1 and γ_2; the limiting condition for the success of the second is:

$$R_2 = b\alpha < \alpha \log_2(1+\gamma_1+\gamma_2) + \frac{(1-\alpha)}{2}\left[\log_2(1+\gamma_1) + \log_2(1+\gamma_2)\right]$$

Hybrid ARQ performance (III)



Channel coding techniques

Parity check codes

- Coding: it is done using the generator matrix **G** [*k* x *n*]; the encoded vector **x** is obtained from the unencoded vector **u** by means of the operation **x**=**uG**. If the first *k* bits of **x** coincide with **u**, the code is said to be systematic.
- The code consists of all possible sums (modulo 2) of the rows of the generator matrix.
- Error detection (parity check): this is done using the parity check matrix **H** [(*n*-*k*) x *n*]; if the received vector is **y**=**x**⊕**e**, where **e** is the error vector, **s**=**yH**‘=(**x**⊕**e**) **H**‘= **eH**‘ is calculated (where ⊕ it indicates the operation of addition modulo 2 and ' it indicates the transposition operation). If there were no errors, the vector **s** is the null vector. It turns out to be **GH'= 0**. The vector **s** is called syndrome.
- Error correction (hard): to each vector **s** it is associated a vector **e**, so that the decoded vector is **v**=**y**⊕**e**. The association takes place at a minimum distance (to each syndrome it is associated the minimum weight error sequence that produces that given syndrome). The non-zero syndromes (and therefore the correctable error sequences) are 2^{n-k}-1.

First example

- Consider a systematic (6,3) code able to correct all the single error and a given pattern of two errors.

$$\mathbf{G} = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right] \qquad \mathbf{H} = \left[\begin{array}{ccc|ccc} 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

- Being the code systematic $\mathbf{G} = \left[\mathbf{I}_k \middle| \mathbf{P}'\right]$ and $\mathbf{H} = \left[\mathbf{P} \middle| \mathbf{I}_{n-k}\right]$

- Decoding may be performed through the standard array.

Standard Array

- The first row represents the code words (the information bits are in red). The first column represents the correctable error patterns.

000000	100101	010111	110010	001011	101110	011100	111001
100000	000101	110111	010010	101011	001110	111100	011001
010000	110101	000111	100010	011011	111110	001100	101001
001000	101101	011111	111010	000011	100110	010100	110001
000100	100001	010011	110110	001111	101010	011000	111101
000010	100111	010101	110000	001001	101100	011110	111011
000001	100100	010110	110011	001010	101111	011101	111000
101000	001101	111111	011010	100011	000110	100100	010001

- There is the null word, 4 words with weight 3, 3 words with weight 4.
- Observe that the array includes all the possible $2^6=64$ *n*-tuples. The received n-tuple is associated to information bits reported in the first row (coset leader), being the word reported in the first row, the closest one to the received word.

Hamming codes

- Parity check codes with the following parameters: $n=2^m-1$, $n-k=m$, $d_{min}=3$, $t=1$, being $m\geq 3$ an integer.
- The columns of the parity check matrix consist of all the non zero m-length sequences.
- Example, $m=3$, $n=7$, $k=4$, systematic version.
- The hard decoding consists of identifying the single error position by comparing the syndrome with the columns of **H**.
- If more than one error occur the decoding is wrong.

$$\mathbf{H}=\left[\begin{array}{cccc|ccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]=\left[\mathbf{P}\,|\,\mathbf{I}_{n-k}\right]$$

$$\mathbf{G}=\left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array}\right]=\left[\mathbf{I}_k\,|\,\mathbf{P}'\right]$$

Hamming codes

- The Hamming codes may be used for correcting single errors, or for detecting single and double errors.
- If the error correction capability used the word correct reception probability is given by

$$P_C = \sum_{i=0}^{1} \binom{n}{i} p_b^i \left(1 - p_b\right)^{n-i}$$

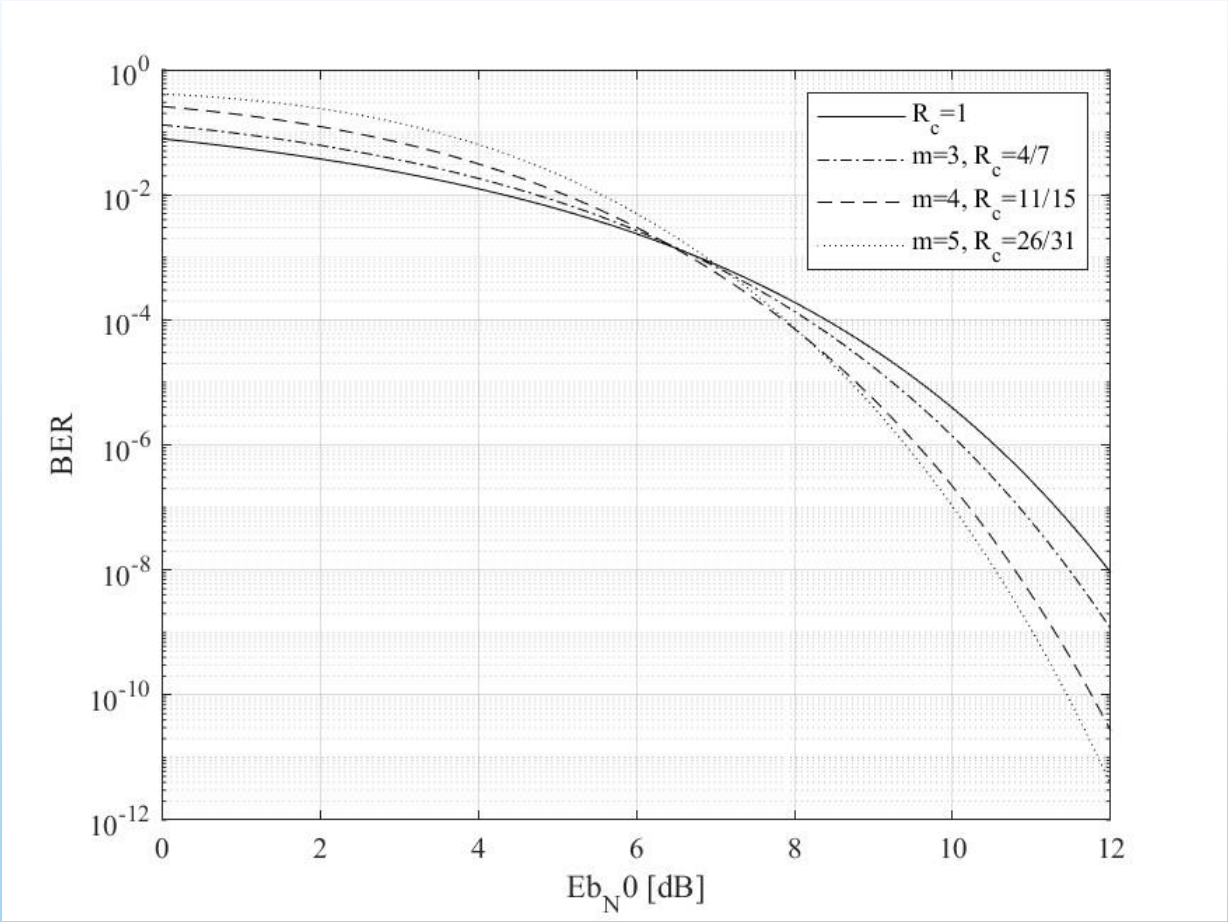
while the bit error probability is approximately given by $P_b = (1-P_C)/2$, assuming to substitute the correct *k*-tuple with another one chosen at random.

- If the error correction capability is not used, the probability of word correct reception, word error detection, or word wrong reception are respectively given by:

$$P_C = \left(1 - p_b\right)^n, \qquad P_D = \sum_{i=1}^{2} \binom{n}{i} p_b^i \left(1 - p_b\right)^{n-i}, \qquad P_E = 1 - P_C - P_D.$$

Hamming codes: performance

- t_c=1. By increasing *m*, the block size increases, the code rate approaches 1, and the coding gain, for low error rates, slightly increases.

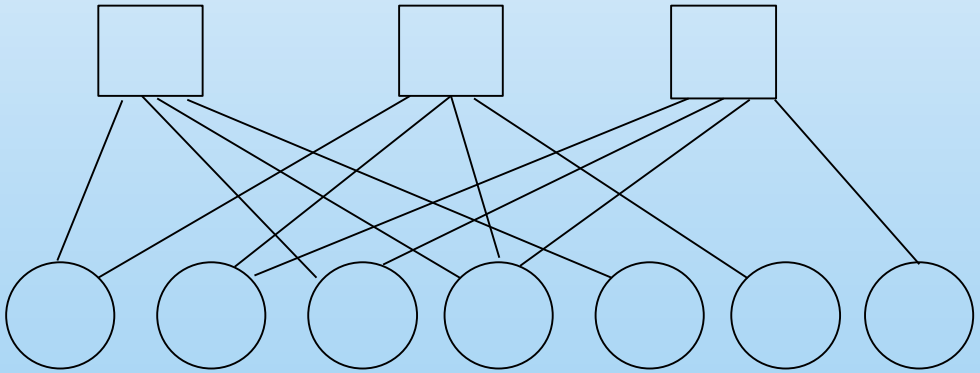


Graph Representations of Parity Codes
Tanner Graph

- A Tanner Graph is a bipartite graph which represents the parity check matrix of an error correcting code.
- **H** is the (*n*–*k*)-by-*n* parity check matrix. The Tanner graph has:
- *n* bit nodes (or variable nodes), represented by circles.
- *n*–*k* check nodes, represented by squares.
- There is an edge between bit node *i* and check node *j* if there is a one in row *i* and column *j* of **H**.
- The Tanner Graph will be a key tool for soft decoding.

$$\mathbf{H} = \left[\begin{array}{cccc|ccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

Hamming Code



Tanner graph

Weight Enumerating Function

- A more detailed description of a code may be given by the weight enumerating function

$$A(D) = \sum_{d=0}^{n} A_d D^d$$

in which A_d represents the number of word of weight d.

- For Hamming codes:

$$A(D) = \frac{1}{n+1}\left[(1+D)^n + n(1+D)^{(n-1)/2}(1-D)^{(n+1)/2}\right]$$

- For example, for m=3, n=7, we have

$$A(D) = 1 + 7D^3 + 7D^4 + D^7$$

There is the null word, 7 words with weight 3, 7 words with weight 4, and 1 word with weight 7.

Dual Codes

- A parity (n, k) check code has a dual (n, n-k) code in which **G** and **H** switch roles. More precisely **H** becomes the generating matrix and **G** becomes the parity check matrix.
- The weight enumerating function of the dual code is obtained from $A(D)$ by

$$A_{\text{DUAL}}(D) = 2^{-k}(1+D)^n A\left(\frac{1-D}{1+D}\right)$$

- For Hamming codes we have

$$A_{\text{DUAL Hamming}}(D) = \frac{2^{-k}(1+D)^n}{n+1}\left[\frac{2^n}{(1+D)^n} + n\frac{2^n D^{(n+1)/2}}{(1+D)^n}\right] = 1 + nD^{(n+1)/2}$$

All the n non zero words have weight $(n+1)/2$.

Extended Parity Check Codes

- Each Parity Check code may be extended by adding one parity digit that checks all the previous n digits of the word. We obtain a (n+1, k) code.
- See an example for Hamming codes.

$$\mathbf{H} = \left[\begin{array}{cccc|cccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

- This code is able to correct all the single errors and to detect all the even errors.
- Decoding may be performed as follows. Compute the syndrome.
  - If the last digit of syndrome is 1, the number of errors is odd, and the correction may performed as for Hamming codes.
  - Otherwise if the syndrome is not 0, an even error is detected.

Example

- Transmit **c**=[1 0 0 0 1 1 0 1]; **c H**'=[0 0 0 0].
- Receive **r**=**c** $\oplus$ [0 0 1 0 0 0 0 0]; **c H**'=[1 0 1 1].

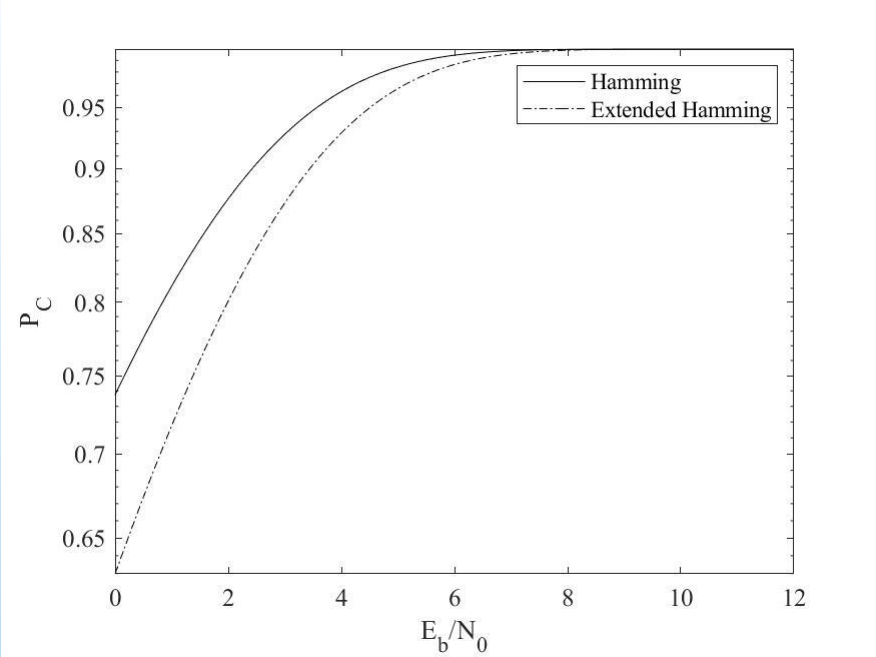
$$\mathbf{H} = \left[\begin{array}{cccc|cccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

The last digit is 1 (odd number of errors); the other values are 1 0 1 which identify the third column of **H**, showing that the error is in third position

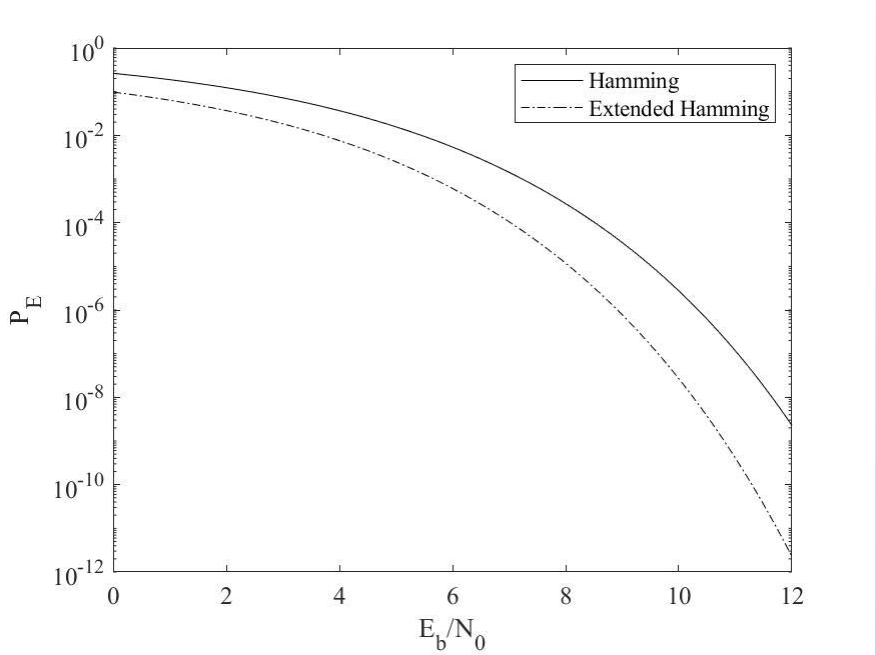
- Receive **r**=**c** $\oplus$ [0 0 1 0 1 0 0 0]; **c H**'=[0 0 1 0].
  The last digit is 0 (even number of errors); error detected.

Performance

- m=3.



Block correct reception probability



Block wrong reception probability

Cyclic codes

- They are linear codes.
- Any cyclic permutation of a code word produces a code word.
- Each binary sequence is associated with a polynomial whose coefficients are the bits that make up the sequence.
- Redundancy bits are found using a division algorithm. The divisor polynomial is denoted by $g(x)$, and has degree n-k. Therefore a cyclic code is characterized by n, k, $g(x)$.
- Systematic encoding.
  - The k-ple to be encoded is associated with a polynomial $u(x)$ of degree n-1, whose k most significant coefficients coincide with the user bits, while the other n-k bits are set to 0. Calculate the remainder, $r(x)$, of dividing u(x) by $g(x)$. The coefficients of $r(x)$ are the redundancy. The n-ple thus obtained is associated with a polynomial $c(x)$, divisible by $g(x)$.
- The received sequence, r, is associated with the polynomial $c_r(x)$. We divide by $g(x)$. If the remainder (syndrome) is null it is assumed that there have been no errors.

Factors of x^n+1

n	Factors
7	6,54,64
9	6,7,444
15	6,7,46,62,76
17	6,471,727
21	6,7,54,64,534,724
23	6,5342,6165
25	6,76,4102041
27	6,7,444,4004004
31	6,45,51,57,67,73,75
33	6,7,4522,6106,7776
35	6,54,64,76,57134,72364
39	6,7,57074.74364,77774
41	6,5747175,6647133
63	6,7,54,64,414,444,534,554,604,634,664,714,724
127	6,406,422,436,442,472,516,526,562,576,602,626,646,652,712,736,742, 756,772

Hamming code generators in red.

Some cyclic Hamming codes generators

m	n	G ($g_L\ g_{L-1} \dots g_0$)	G (ottale)
3	7	1 011	64
4	15	10 011	62
5	31	100 101	51
6	63	1 000 011	604
7	127	10 001 001	442
8	255	100 011 101	561
9	511	1 000 010 001	4204
10	1023	10 000 001 001	4402
11	2047	100 000 000 101	5001
12	4095	1 000 001 010 011	62404
13	8191	10 000 000 011 011	66002
14	16383	100 010 001 000 011	60421
15	32767	1 000 000 000 000 011	600004

Systematic code construction from $g(x)$

- The last row of **G** consists of k-1 zeros followed by the n-k+1 coefficients of $g(x)$.
- The upper row is built by shifting left of one position the lower one. If the bit in the k-th position is not 0, the last row is added before continuing. This way the new row consists of $g^{(1)}(x)+g(x)$ and it is still a code word,
- The process is repeated (always ensuring that the bit in k-th position is 0) until the full **G** matrix is built.
- Example: (7,4) code with $g(x)=x^3+x+1$.

  The last row is **0 0 0 1 0 1 1**

  Next one is **0 0 1 0 1 1 0**.

  Next one is 0 1 0 1 1 0 0 ⊕
  0 0 0 1 0 1 1 =
  0 1 0 0 1 1 1

  The next one 1 0 0 1 1 1 0 ⊕
  0 0 0 1 0 1 1=
  1 0 0 0 1 0 1

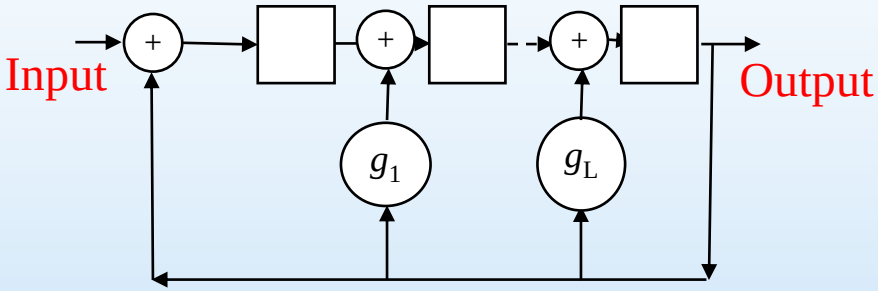
  The **G** and **H** matrices are shown on the right.

$$\mathbf{G} = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

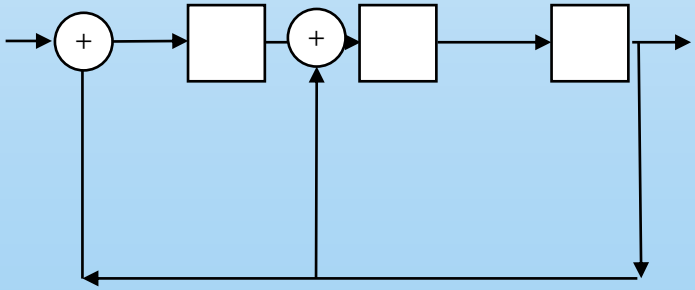
$$\mathbf{H} = \left[\begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

Encoding through shift registers

- The circuit shown in the figure may be used to perform the division ($L=n-k-1$). After entering the input values and $n-k$ zeros in the shift register we have the reminder.



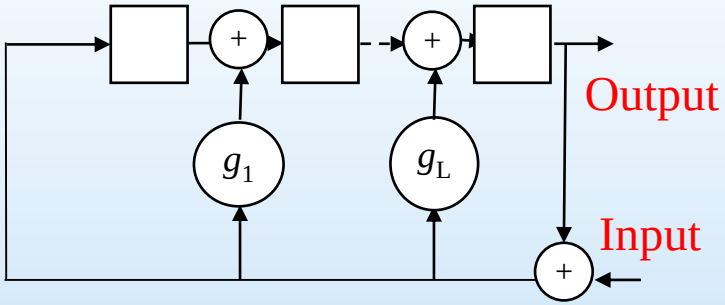
- Consider a (7,4) with $g(x)=x^3+x+1$. Suppose to encode di bits 0111 corresponding to x^2+x+1. We have to evaluate the reminder of $x^5+x^4+x^3$ which is x.
The content of the register, after entering 0111 000 is



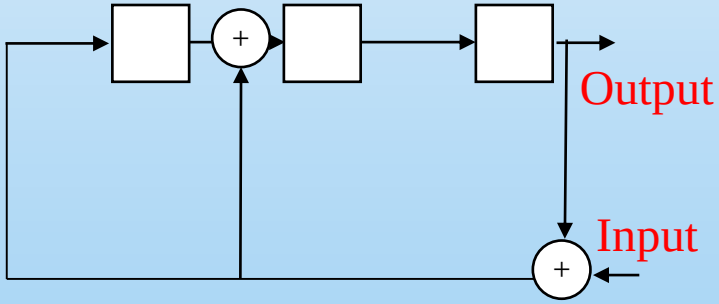
000
100
110
111
101
100
010

Encoding through shift registers

- The first *n-k* steps may be skipped by feeding the shift register from the output. This corresponds to pre-cycling the input sequence *n-k* times



- Consider again the (7,4) code with $g(x)=x^3+x+1$. Suppose to encode di bits 0111 The content of the register, after entering 0111 is

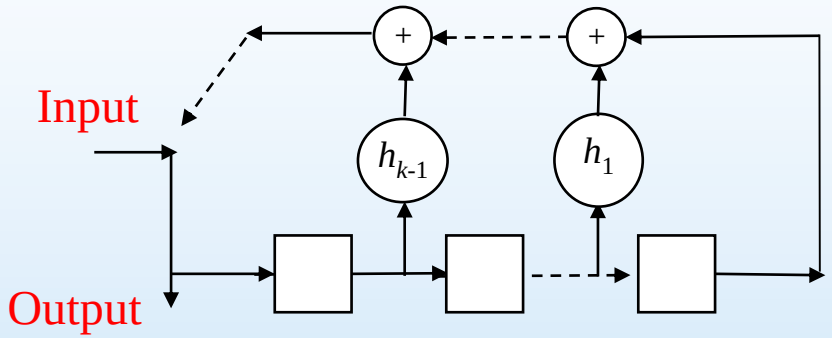


000
110
101
010

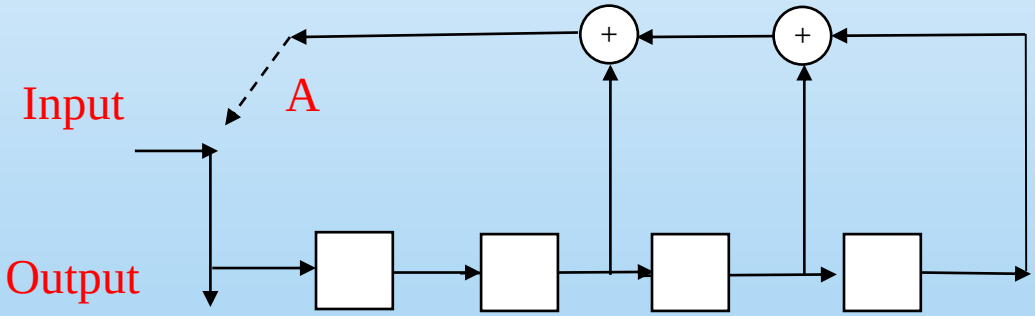
In the first step, the 4 digits enter the shift register and may transmitted to the channel. In the second step the content of the register may be transmitted.

Encoding through shift registers

- A different encoder may be considered taking into account the dual code generator $h(x)=(x_n+1)/g(x)$. First, the k information bits enter the register and go to the channel. After, the remaining n-k bits go the channel, after closing the dashed connection.



- Consider again the (7,4) code with $g(x)=x^3+x+1$. $h(x)= x^4+ x^2+x+1$. Suppose to encode di bits 0111.



	register	output
A open	0000	0
	1000	1
	1100	1
	1110	1
A closed	0111	0
	1011	1
	0101	0

Error detection through shift registers

- Theorem 1: if $s(x)$ is the syndrome associated with $e(x)$, the $e^{(i)}(x)$ syndrome (error shifted i positions to the left) is obtained by cycling the $s(x)$ syndrome i times in the division circuit that was used for determine it.

- Theorem 2: if the errors of $e(x)$ are confined to the n-k parity bits of $r(x)$, the syndrome $s(x)$ of $r(x)$ coincides with $e(x)$.

- Consider the first scheme. In the reception, after entering all bits, in the register we have the syndrome which should be the null vector if the received word is a code word. In that case the received word may be accepted. Otherwise the syndrome may be used for error correction.

- If we consider the second scheme instead, after receiving the n-k information bits, in the register we have the computed remainder which may be compared with the received one, for error detection. After entering also the parity bits, we obtain a shifted version of the syndrome (theorem 1) which is the null vector if the received vector is a code word, and which may be used for error correction otherwise.

Example

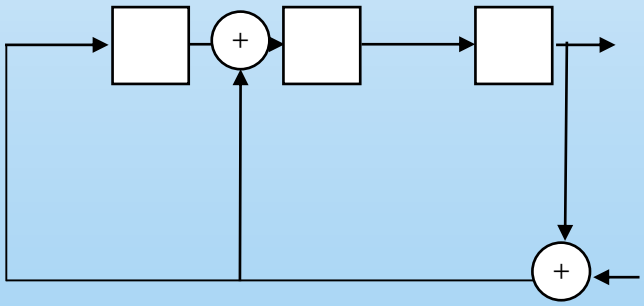
- Consider a (7,4) with $g(x)=x^3+x+1$

$$\mathbf{G} = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

$$\mathbf{H} = \left[\begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

Encode the input sequence **u**=1101. By using **G** we obtain **c**=1101001

By using the shift register we obtain



110
101
100
100

so that again **c**=1101001

Example

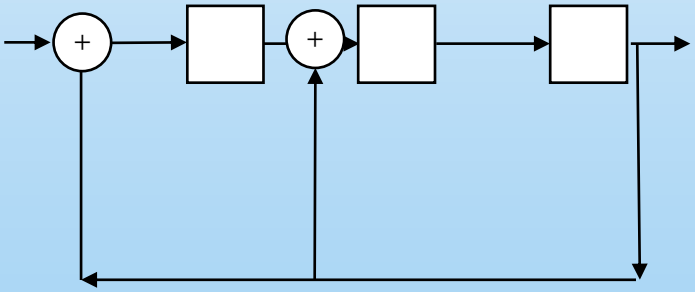
- Suppose now to receive **r**=1001001, with an error in second position.

$$\mathbf{G} = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

$$\mathbf{H} = \left[\begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

By using **H** we obtain **s**=111 which shows that the error is in second position.

By using the first shift register we obtain



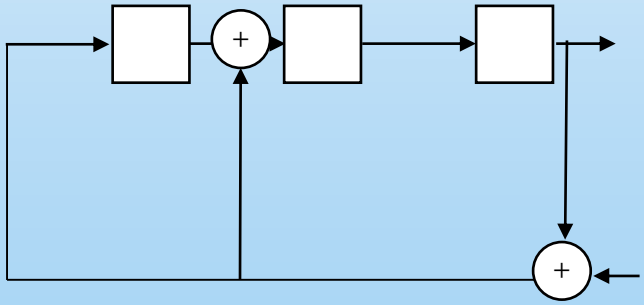
100
010
001
010
001
110
111 (syndrome)

Example

- Again, suppose now to receive **r**=1001001, with an error in second position.

$$\mathbf{G} = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

$$\mathbf{H} = \left[\begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$



By using the second shift register we obtain

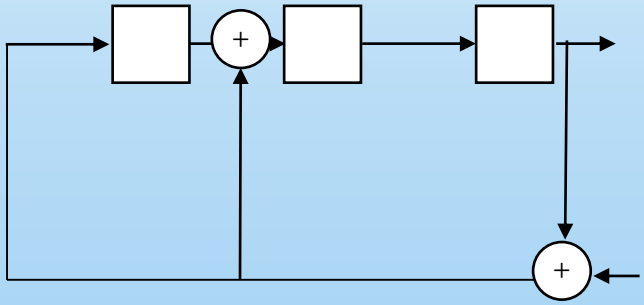
110
011
111
011 (wrong remainder)
111
101
010 (shifted syndrome)

Example

- If receive the correct word **r**=1101001.

$$\mathbf{G} = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

$$\mathbf{H} = \left[\begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$



By using the second shift register we obtain

110
101
100
100 (correct remainder)
010
001
000 (null shifted syndrome)

Other important cyclic codes

- Golay code (23,12): $g(x)=x^{11}+x^9+x^7+x^6+x^5+x+1$, $d_{min}=7$.

  The Golay code is able to correct 3 errors in a code word. This is the only non trivial perfect code able to correct more than one error (a code is perfect when all the code words have at least one other code word at d_{min} distance). The other perfect codes are the repetition and the Hamming codes.

- BCH (Bose, Chaudhuri, Hocquenghem) codes.

  Very efficient cyclic codes, which can be designed to correct t errors (Hamming codes are a special case, with t=1). Characterized by the property:

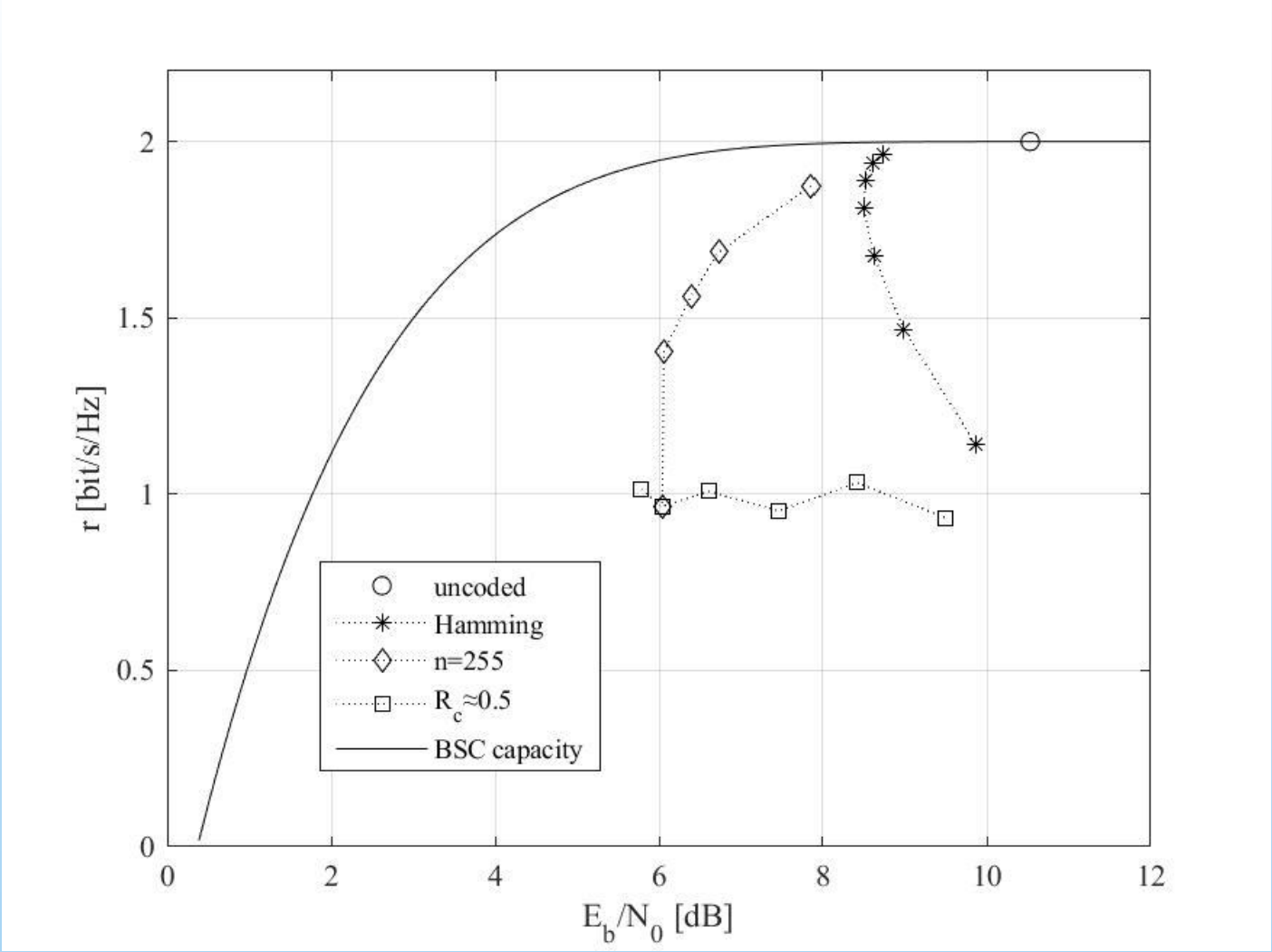
  $\forall\ m, t\ \exists$ BCH code with $n=2^m-1$, $n-k \leq mt$, $d_{min} \geq 2t+1$.

Some BCH codes

-

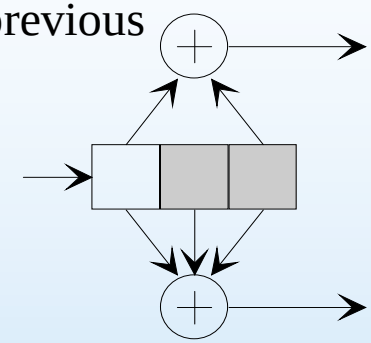
m	n	k	t	m	n	k	t	m	n	k	t	m	n	k	t	m	n	k	t
3	7	4	1	6	63	24	7	7	127	50	13	8	255	187	9	8	255	71	29
4	15	11	1			18	10			43	14			179	10			63	30
		7	2			16	11			36	15			171	11			55	31
		5	3			10	13			29	21			163	12			47	42
5	31	26	1			7	15			22	23			155	13			45	43
		21	2	7	127	120	1			15	27			147	14			37	45
		16	3			113	2			8	31			139	15			29	47
		11	5			106	3	8	255	247	1			131	18			21	55
		6	7			99	4			239	2			123	19			13	59
6	63	57	1			92	5			231	3			115	21			9	63
		51	2			85	6			223	4			107	22	9	511	502	1
		45	3			78	7			215	5			99	23			493	2
		39	4			71	9			207	6			91	25			484	3
		36	5			64	10			199	7			87	26			475	4
		30	6			57	11			191	8			79	27			466	5

BCH performance (p_b=10^{-6})

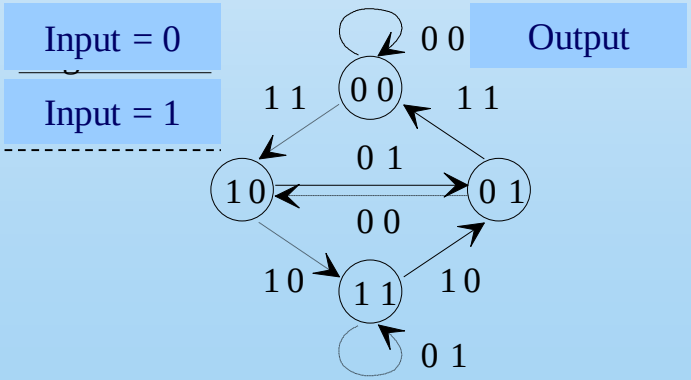


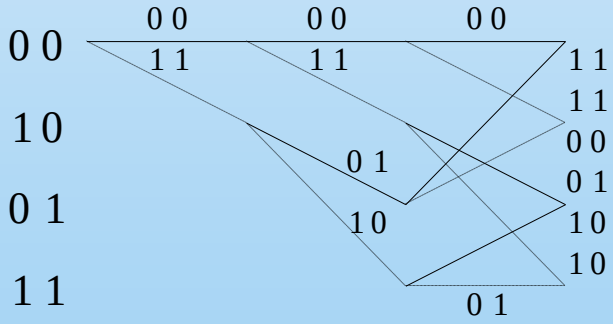
Convolutional Codes

- The coded sequences depends on the current and on the previous inputs.
- Example: R_c=1/2. The previous values of the input that have an influence on the current value of the output are highlighted (these values define the state of the system). The constraint length, L, is the size of the register (in the example L=3). The input/output relations define the generators which are indicated in octal notation. In the example g_1=(1 0 1) (indicated by 5), g_2=(1 1 1) (indicated by 7).



- Evolution is described by state diagram or trellis diagram.





Convolutional codes

- It is a linear code.
- The performance depends on the free distance, d_{free}, which is the weight of the lowest weight non zero sequence. It may me determined through the transfer function.
- In our example we obtain the transfer function $\frac{D^5}{1-2D} = D^5 + 2D^6 + 4D^7 + \ldots$

  There is a sequence of weight 5 (input 100 which produces output 110111), two sequences of weight 6 (input 1100 which produces output 11101011 and input 10100 which produces output 1101000111), 4 weighing 7, and so on.

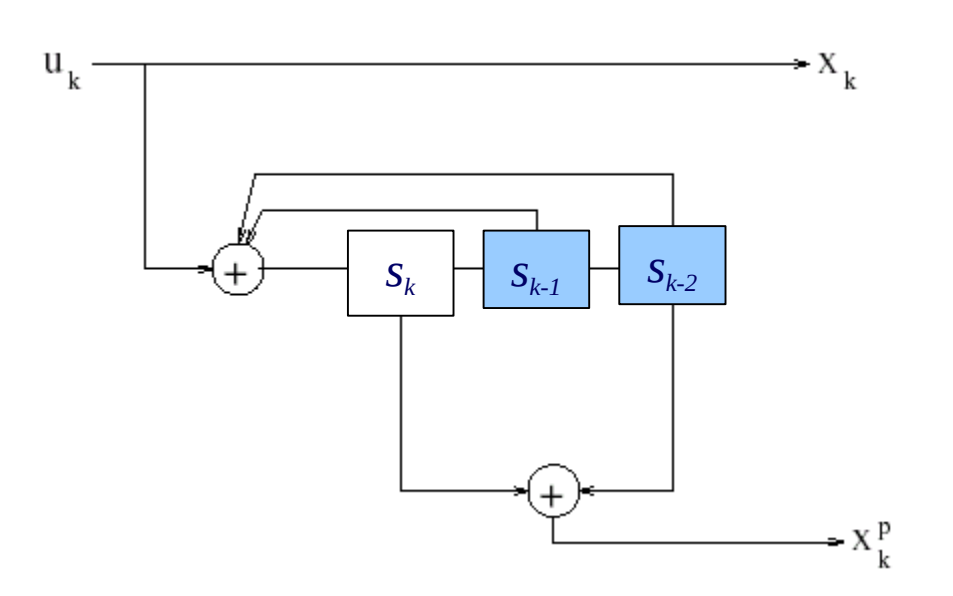
- Decoding takes place at a minimum distance, using the Viterbi algorithm.

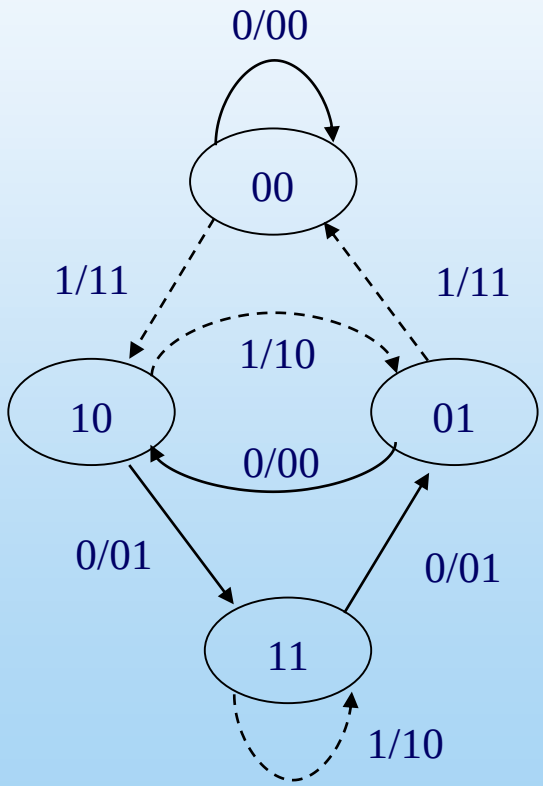
- The algorithm does not change if the soft decision is used. Euclidean distance is used instead of Hamming distance.

- Complexity grows linearly with length.

Recursive Convolutional Codes

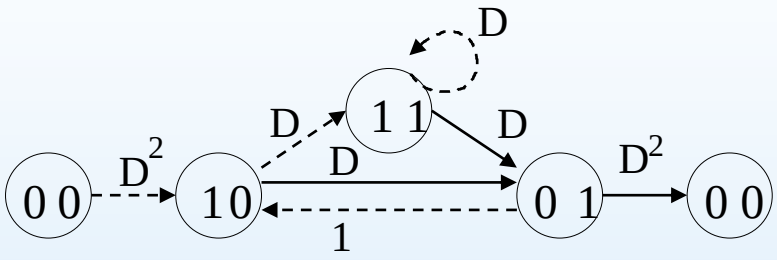
- It may be systematic (the output sequence contains the input sequence)
- It may be recursive.
- Example of systematic and recursive encoder with the relevant state diagram.
- It is the key component of the turbo encoder.





Transfer function

- The figure shows the graph from which to determine the transfer function. It is a state diagram in which state 0 has been isolated, from which we start to identify the word with minimum weight.
  D^h indicates that the input value produces an output pair of weight h).



- Graph modification rules: by using them we obtain the transfer function $\frac{D^5}{1-2D}$

A) $\alpha(D)$ $\beta(D)$ $\equiv$ $\alpha(D)\beta(D)$

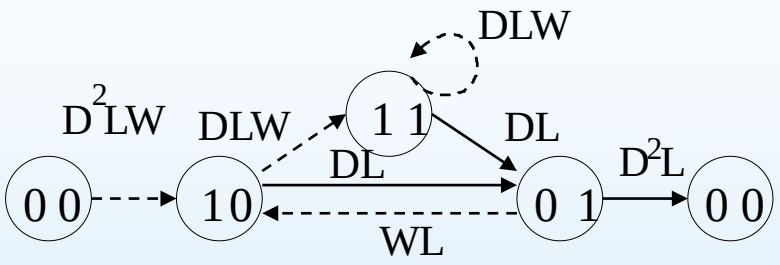
B) $\alpha(D)$, $\beta(D)$ $\equiv$ $\alpha(D)+\beta(D)$

C) $\alpha(D)$ $\gamma(D)$ $\beta(D)$ $\equiv$ $\alpha(D)\beta(D)/(1-\gamma(D))$

D) $\alpha(D)$ $\beta(D)$ $\delta(D)$, $\gamma(D)$ $\equiv$ $\alpha(D)\beta(D)\delta(D)/(1-\beta(D)\gamma(D))$

Input-output WEF

- A more comprehensive function may be obtained taking into account the input bits (the parameter *W* specifies that the transition is due to a '1' input bit) and the steps (the parameter *L* takes into account the step).



- By using the modification rules we obtain the input-output WEF.

$$T_3(W,D,L)=\frac{D^5WL^3}{1-DWL(1+L)}=D^5WL^3+D^6W^2L^4(1+L)+D^7W^3L^5(1+L)^2\ldots$$

showing that the path of weight 5 consists of 3 steps, with a input sequence with one '1', the two paths of weight 6 consist of 4 and 5 steps, both with a input sequence with two '1's, the four paths of weight 7 have length 5, 6 (two paths) and 7 steps, all with three '1's, and so on. The WEF may be obtained by putting *W*=1 and *L*=1.

Input-output WEF

- Sometimes the length of the input path is not important, and we may obtain another Input-Output weight enumerating function by putting L=1 in the previous one. In the previous example

$$T_2(W,D)=\frac{D^5W}{1-2DW}=D^5W+2D^6W^2+4D^7W^3\ldots$$

- Defining $T(D)=\sum_{d=d_{\text{free}}}^{\infty}A_dD^d$, $T_2(W,D)=\sum_{w=1}^{\infty}\sum_{d=d_{\text{free}}}^{\infty}B_{w,d}W^wD^d$

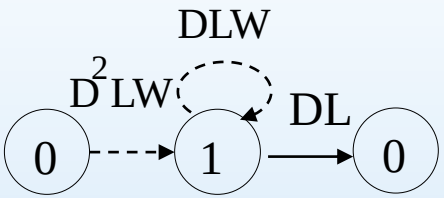
$$T_3(W,D,L)=\sum_{w=1}^{\infty}\sum_{d=d_{\text{free}}}^{\infty}\sum_{l=1}^{\infty}C_{w,d,l}W^wD^dL^l$$

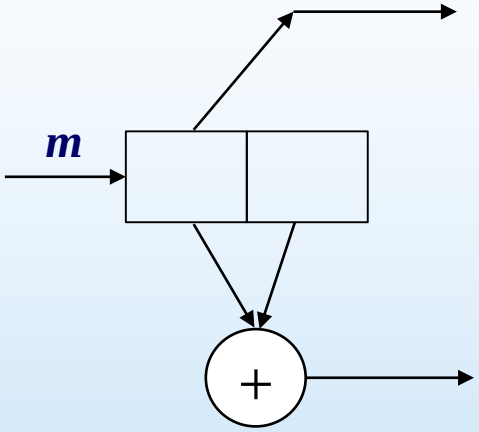
we have $A_d=\sum_{w=1}^{\infty}B_{w,d}=\sum_{w=1}^{\infty}\sum_{l=1}^{\infty}C_{w,d,l}$

and $B_{w,d}=\sum_{l=1}^{\infty}C_{w,d,l}$

Some definitions

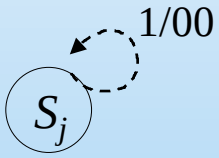
- A convolutional code is said to be systematic if the coded flow includes the information flow.

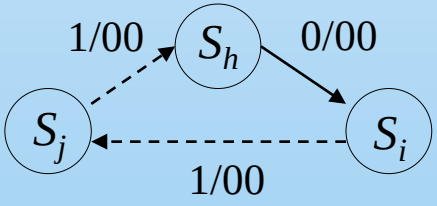




$$T_3(W,D,L) = \frac{D^3WL^2}{1-DWL} = D^3WL^2 + D^4W^2L^3 + \ldots$$

- A convolutional code is said to be catastrophic if a large number of bit errors occur when only a small number of channel bit errors is received. This type of code needs to be avoided and can be identified by the state diagram. A state diagram having a loop in which a nonzero information sequence corresponds to an all-zero output sequence identifies a catastrophic convolutional code. Systematic convolutional codes cannot be catastrophic.





Some definitions

- Feed forward (without recursions) non systematic, non catastrophic codes may achieve a higher free distance than systematic ones.

	D_f (R_c=1/2)		D_f (R_c=1/3)	
L	Systematic	Non Syst.	Systematic	Non Syst.
1	3	3	5	5
2	4	5	6	8
3	4	6	8	10
4	5	7	9	12
5	6	8	10	13
6	6	10	12	15
7	7	10	12	16

Recursive encoders

- However, for every non systematic, feed forward encoder, we may obtain a systematic, recursive one with the same performance.
- Consider a (2,1,L) non systematic encoder with generators $G_{1,1}(D)$ and $G_{1,2}(D)$. The output sequences are given by $c_1(D)= m(D)G_{1,1}(D)$, $c_2(D)= m(D)G_{1,2}(D)$.
- Consider now the two new coded sequences given by

$$\tilde{c}_1(D) = \frac{c_1(D)}{G_{1,1}(D)} = m(D) \qquad \tilde{c}_2(D) = \frac{c_2(D)}{G_{1,1}(D)} = m(D)\frac{G_{1,2}(D)}{G_{1,1}(D)}$$

- We have obtained a systematic, recursive encoder.
- If we define a new input sequence (simple reordering of the original one)

$$\tilde{m}(D) = \frac{m(D)}{G_{1,1}(D)}$$

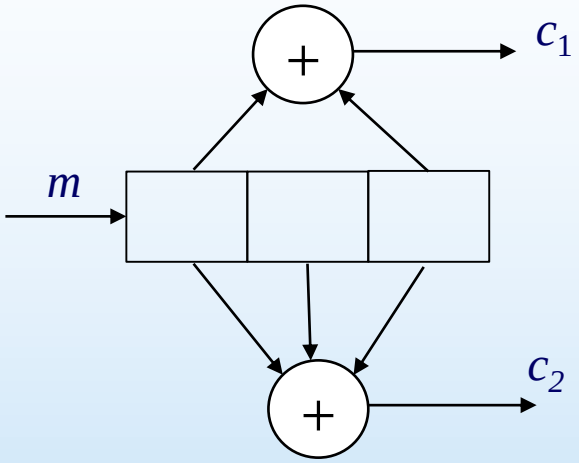
we obtain $\tilde{c}_1(D) = \tilde{m}(D)G_{1,1}(D)$ $\tilde{c}_2(D) = \tilde{m}(D)G_{1,2}(D)$

showing that the code sequences are the same (and WEF is the same).

Recursive encoders

- Consider again the (2,1,2) code in which $G_{1,1}=1+D^2$ and $G_{1,2}=1+D+D^2$.

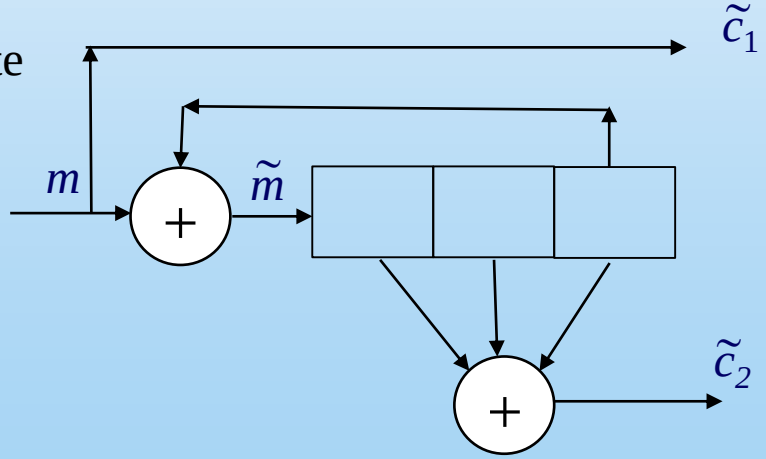
- The systematic, recursive version has generators

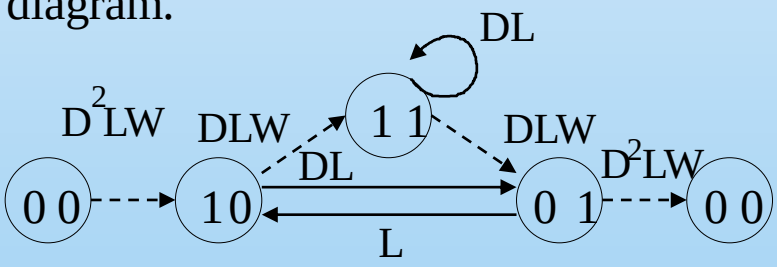


$$\tilde{G}_{1,1}(D)=1 \qquad \tilde{G}_{1,2}(D)=\frac{1+D+D^2}{1+D^2}$$

being $\tilde{m}(D)=\dfrac{m(D)}{1+D^2}$ from which $\tilde{m}_i = m_i \oplus \tilde{m}_{i-2}$.

Below are shown the encoder and the state diagram.





Recursive encoders

- The input-output WEF is given by:

$$T_3(W,D,L)=\frac{D^5W^2L^3\left(1+DW^2L-DL\right)}{D^2L^2\left(1-W^2\right)-DL\left(1+L\right)+L}$$

- For L=1 we obtain: $T_2(W,D)=\dfrac{D^5W^2\left(1+DW^2-D\right)}{D^2\left(1-W^2\right)-2D+1}$

  These functions are different from the functions obtained in the feed forward case, and this shows that the input sequences corresponding to minimum weight output sequence are different.

- Final, for W=1 we obtain $T(D)=\dfrac{D^5}{1-2D}$, showing that the WEF is the same

  of the function determined for the non recursive encoder, as expected.

Non systematic R_c=1/2 encoders

Memory, L	G1	G2	d_{free}
1	1	3	3
2	5	7	5
3	15	17	6
4	23	35	7
5	53	75	8
6	133	171	10
7	247	371	10
8	561	753	12
9	1131	1537	12
10	2473	3217	14
11	4325	6747	15
12	10627	16765	16

Non systematic R_c=1/3 encoders

Memory, L	G1	G2	G3	d_{free}
1	1	3	3	5
2	5	7	7	8
3	13	15	17	10
4	25	33	37	12
5	47	53	75	13
6	117	127	155	15
7	225	331	367	16
8	575	623	727	18
9	1167	1375	1545	20
10	2325	2731	3747	22
11	5745	6471	7553	24
12	10533	10675	17661	24

Performance

- Given the linearity assumption assume to transmit the all 0 sequence.
  The probability of error is bounded by (union bound):

$$P(e) \le \sum_{d=d_{\text{free}}}^{\infty} A_d P(X_0 \to X_d)$$

  in which A_d is the number of sequences of weight d.

- Hard decoding
  Assume that p is received bit error probability, and assume independent errors.
  The probability of mistaking a sequence of weight d with correct one is bounded by:

$$P(X_0 \to X_d) \le \left[\sqrt{4p(1-p)}\right]^d$$

- Soft decoding
  Assume for simplicity to use binary antipodal modulation.
  The probability of mistaking a sequence of weight d with correct one is given by

$$P(X_0 \to X_d) = Q\left(\frac{\text{distance}}{\sqrt{2N_0}}\right) = Q\left(\sqrt{2dR_c\frac{E_b}{N_0}}\right)$$

Performance

- Consider now the Output Weight Enumerating Function of the convolutional encoder

$$T(D) = \sum_{d=d_{\text{free}}} A_d D^d$$

being A_d the number of sequences of Hamming weight d.

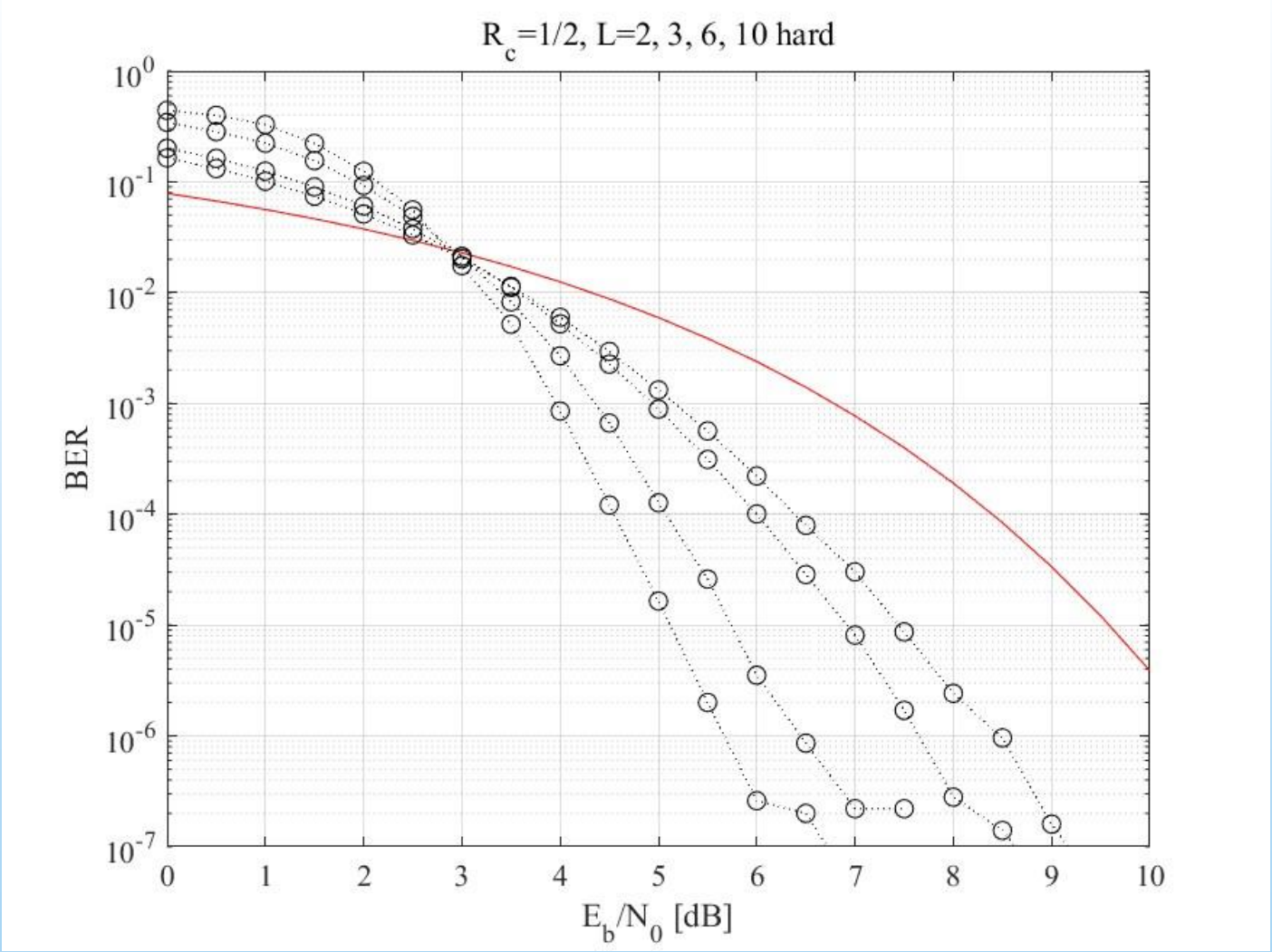
- Assuming soft decoding, and using the union bound we obtain:

$$P_w(e) \le \sum_{d=d_{\text{free}}} A_d \mathrm{Q}\left(\sqrt{2dR_c \frac{E_b}{N_0}}\right) < aT(D)\Big|_{D=\exp\left(-2bR_c E_b/N_0\right)}$$

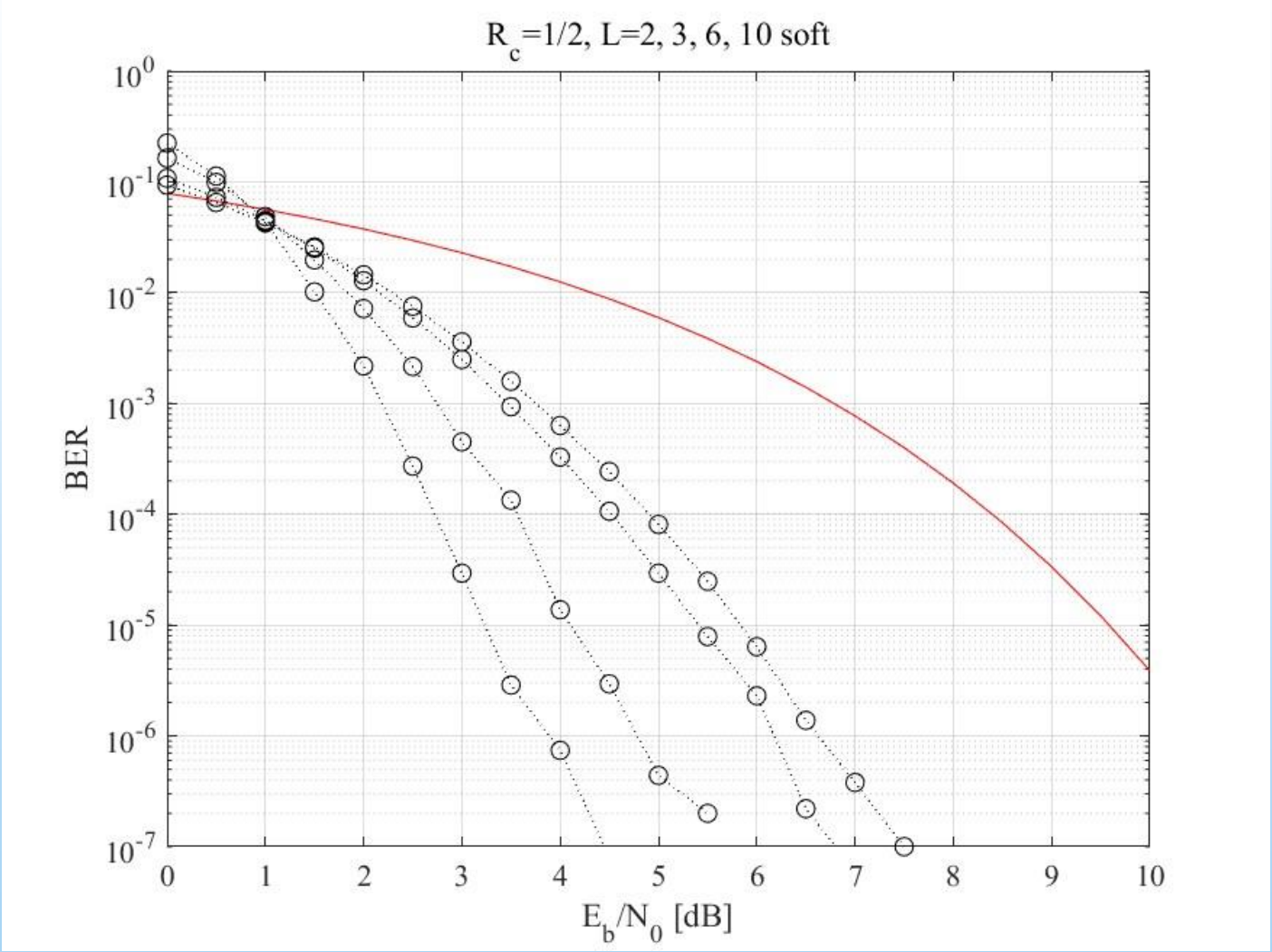
being $$\mathrm{Q}(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2} \mathrm{d}t \approx a \exp\left(-bx^2\right)$$

An excellent approximation is obtained by using a=0.24015 and b=0.5616.

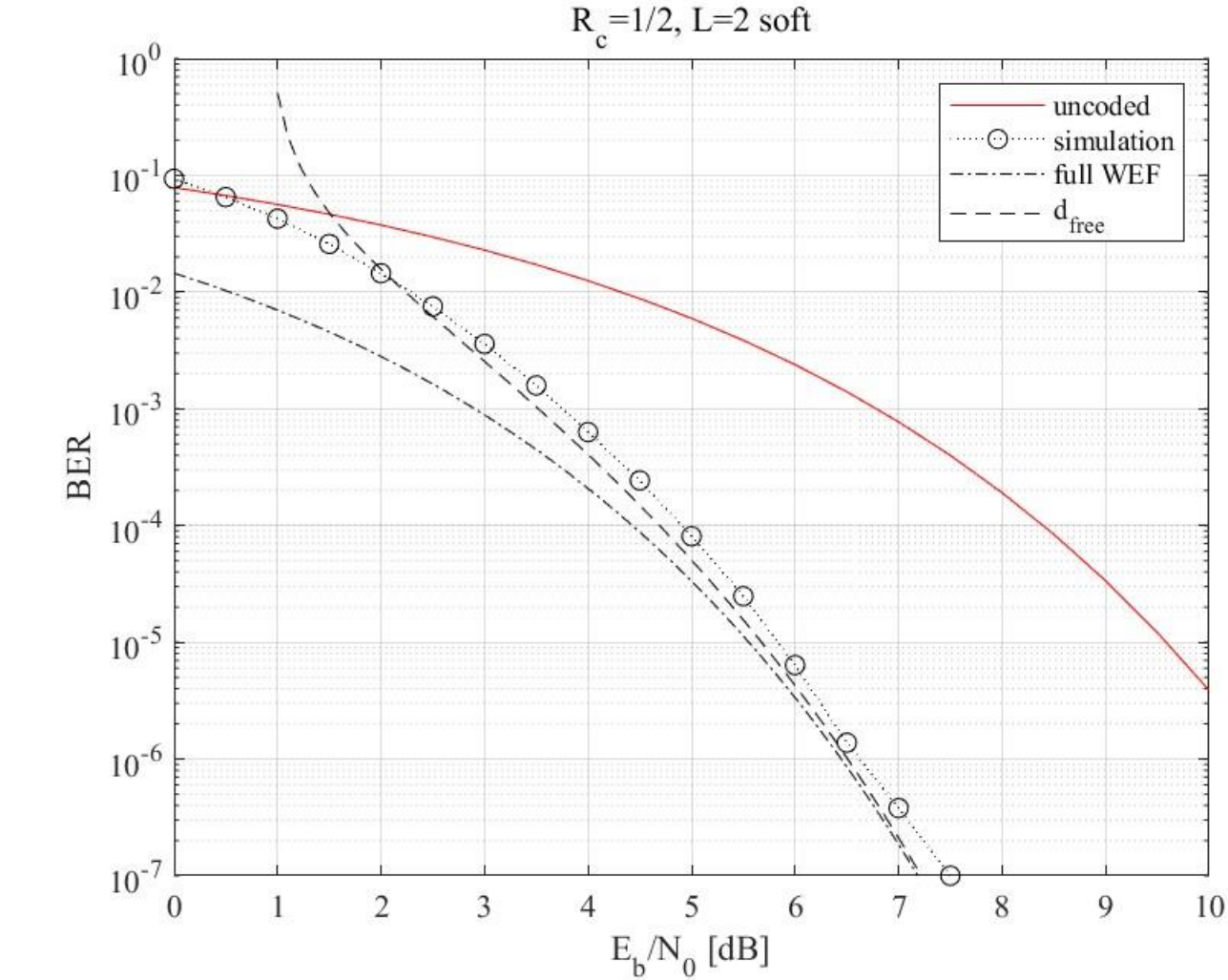
Performance



Performance



Performance

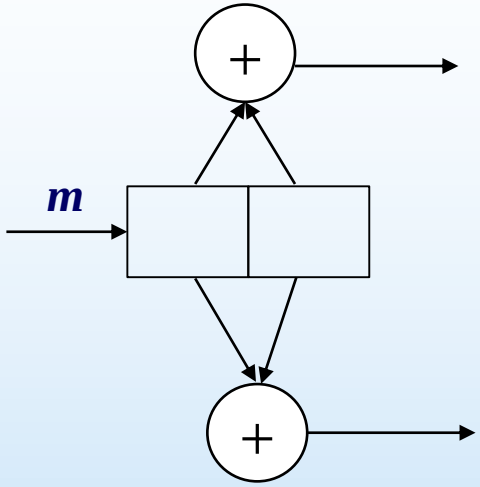


Code puncturing

- With a single decoder you can obtain various rates.
- Starting from a single "mother" convolutional code of rate 1/2, a set of convolutional codes of rate k/n is obtained, with $k>1$.
- Advantages: good solution from the point of view of complexity (only one Viterbi algorithm to implement, for the mother code of rate 1/2).
- Disadvantages: suboptimal performance (fixed k/n and number of states, with puncturing it is not always possible to obtain the best code).

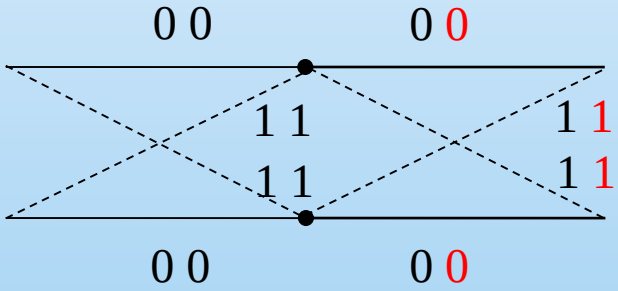
Example

- Mother code with rate R_c=1/2;



- Consider two trellis sections
  Assume to puncture (not transmit) the fourth output bit (in red), obtaining a R_c =2/3.

  Assume to start from upper state



Input	Output
0 0	0 0 0
0 1	0 0 1
1 1	1 1 1
1 0	1 1 0

Code puncturing

- In general, to create a k/n rate code:
- take a "mother" convolutional code of rate 1/2;
- consider k sections: k bits of information $2k$ bits of code;
- puncture $(2k\text{-}n)$ code bits;
- k bits of information correspond to n bits of code.
- Puncture pattern: the $2k$ sequence of binary values in which 1 correspond to a transmitted bit, while 0 correspond to a punctured bit.

- In the previous example the puncture pattern is 1110.

- To design a k/n rate code there are $\binom{2k}{n}$ different puncture patterns.

- It is possible to generate all of them and choose the best one.
- However, the best patterns are known and tabulated.

Code puncturing

- In the received stream, the punctured bits are replaced with neutral values (for example 0 values in a received soft sequence of antipodal values).

PRO

- It is sufficient to implement the Viterbi algorithm for the mother code.
- The code rate may be modified by simply changing the puncture pattern and inserting the received signal zero at the punctured bits.
- This is useful for reconfigurable applications, which require changing the code rate quickly depending on channel conditions.

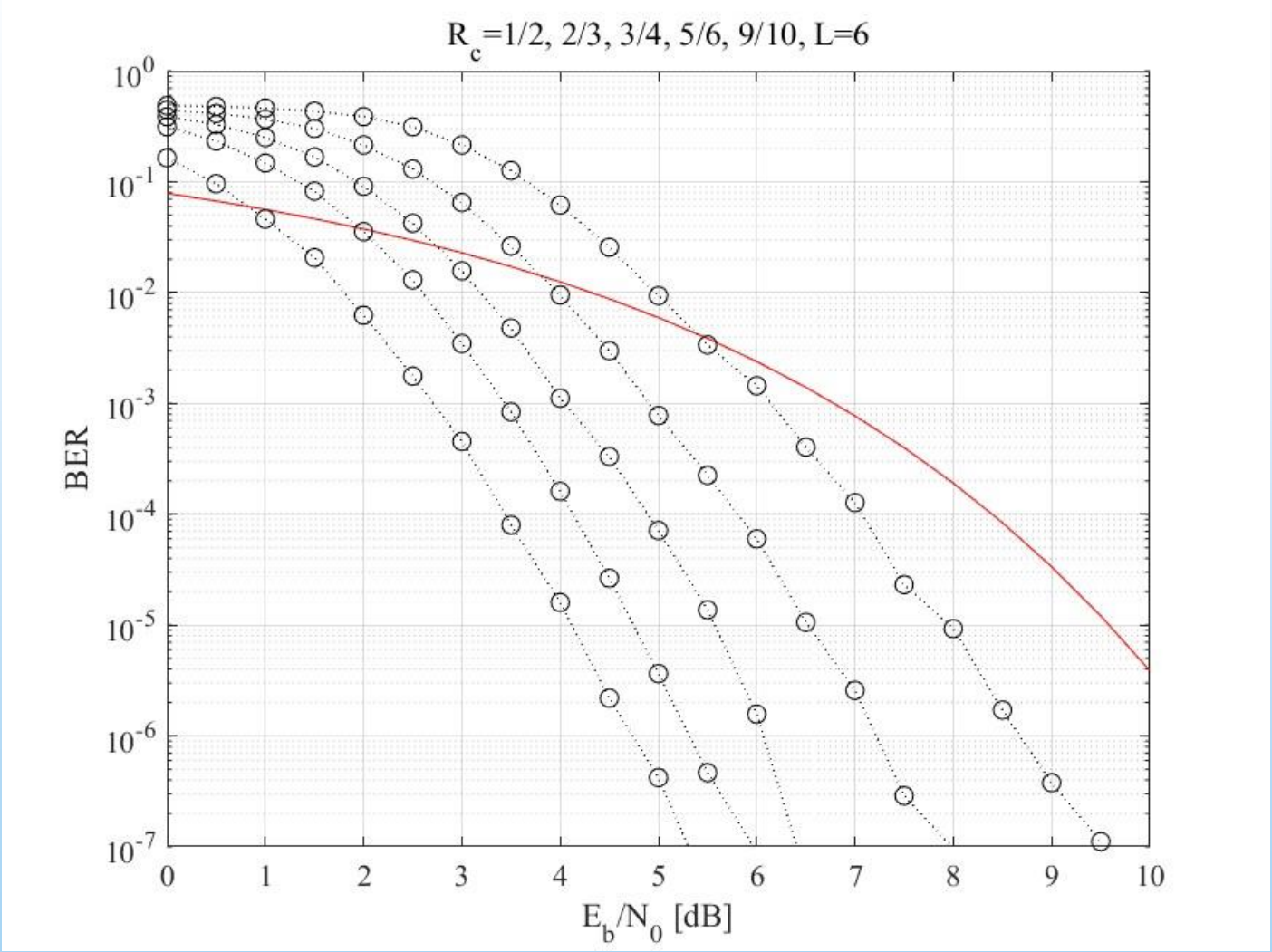
CONS

- Once the *k*/*n* code rate and the number of states are fixed, it is not always possible to obtain the best code by puncturing a 1/2 rate mother code.
- Possible loss of performance (but compared to the simplicity of implementation).

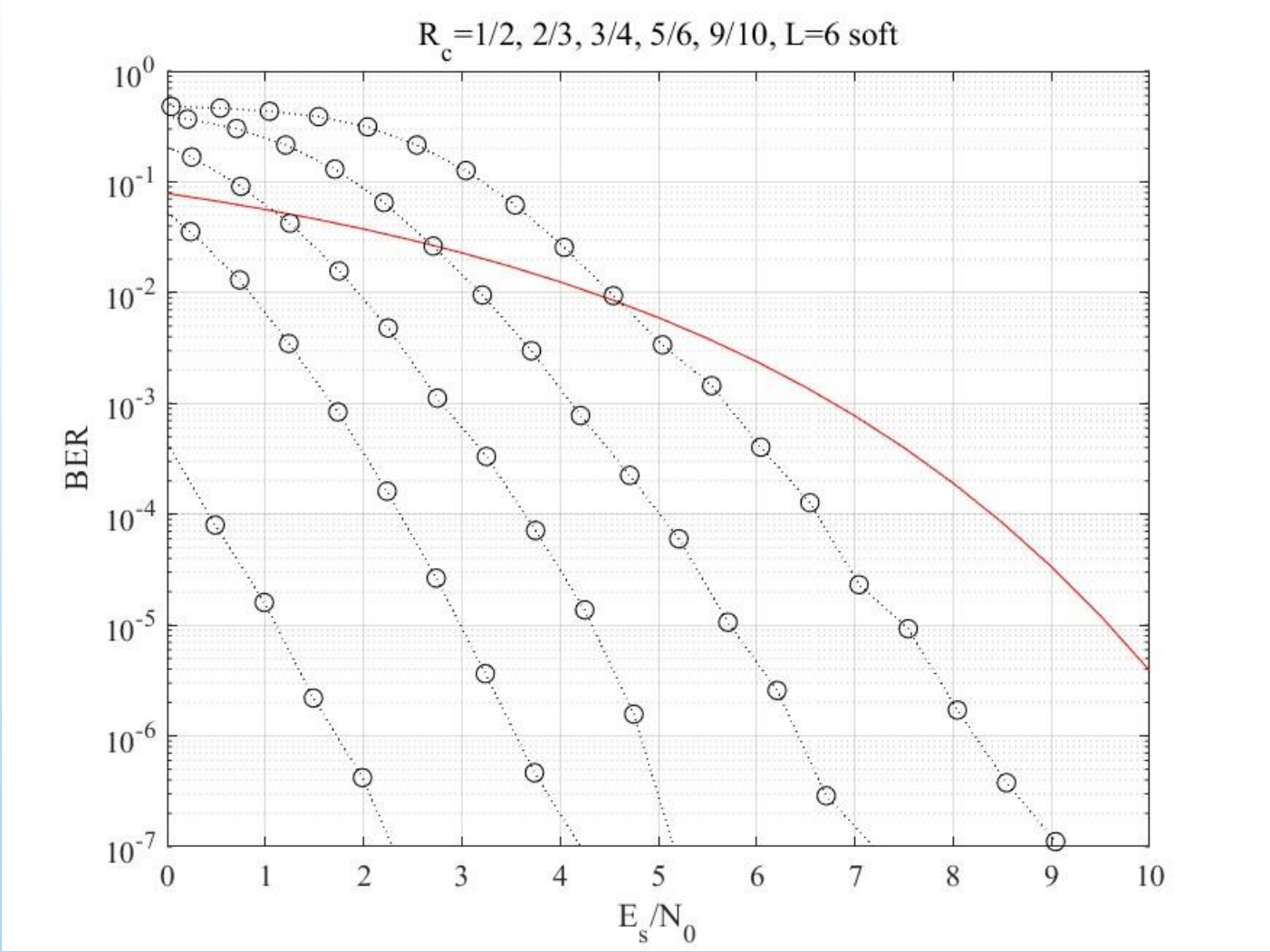
Best puncture patterns

R_c L	2	3	4	5	6	7	8
1/2	1 (5) 1 (7)	1 (15) 1 (17)	1 (23) 1 (35)	1 (53) 1 (75)	1 (133) 1 (171)	1 (247) 1 (371)	1 (561) 1 (753)
2/3	10 11	11 10	11 10	10 11	11 10	10 11	11 10
3/4	101 110	110 101	101 110	110 101	110 101	110 101	111 100
4/5	1011 1100	1011 1100	1010 1101	1000 1111	1111 1000	1010 1101	1101 1010
5/6	10111 11000	10100 11011	10111 11000	10000 11111	11010 10101	11100 10011	10100 11001
6/7	101111 110000	100011 111100	101010 110101	110110 101001	111010 100101	101001 110110	110110 101001
7/8	1011111 1100000	1000010 1111101	1010011 1101100	1011101 1100010	1111010 1000101	1010100 1101011	1101011 1010100
8/9	10111111 11000000	10000011 11111100	10100011 11011100	11100010 10011101	11110100 10001011	10110110 11001001	11100000 10011111
9/10	101111111 110000000	101000000 110111111	111110011 100001100	100001111 111110000	111101110 100010001	101100110 110011001	111000101 100111010

Code puncturing (performance)



Code puncturing (performance)



Bit Error Rate evaluation

- Consider the Input-Output Weight Enumerating Function of the first constituent encoder

$$T_3(W,D,L)=\sum_{w=1}^{\infty}\sum_{d=d_{\text{free}}}^{\infty}\sum_{l=1}^{\infty}C_{w,d,l}D^dW^wL^l$$

in which $C_{w,d,l}$ is the number of input sequences of length l, weight w which produces an output sequence of weight d.

- Using the union bound, for a (n_0, k_0) convolutional code the Bit Error Probability can be bounded as follows.

$$P_b(e)<\frac{1}{k_0}\sum_{w=1}^{\infty}\sum_{d=d_{\text{free}}}^{\infty}\sum_{l=1}^{\infty}wC_{w,d,l}Q\left(\sqrt{2dR_c\frac{E_b}{N_0}}\right)=\frac{1}{k_0}\sum_{w=1}^{\infty}\sum_{d=d_{\text{free}}}^{\infty}wB_{w,d}Q\left(\sqrt{2dR_c\frac{E_b}{N_0}}\right)$$

- Defining the bit multiplicity $A_d^{(b)}=\frac{1}{k_0}\sum_{w=1}^{\infty}wB_{w,d}$

we obtain $$P_b(e)<\sum_{d=d_{\text{free}}}^{\infty}A_d^{(b)}\mathrm{Q}\left(\sqrt{2dR_c\frac{E_b}{N_0}}\right)$$

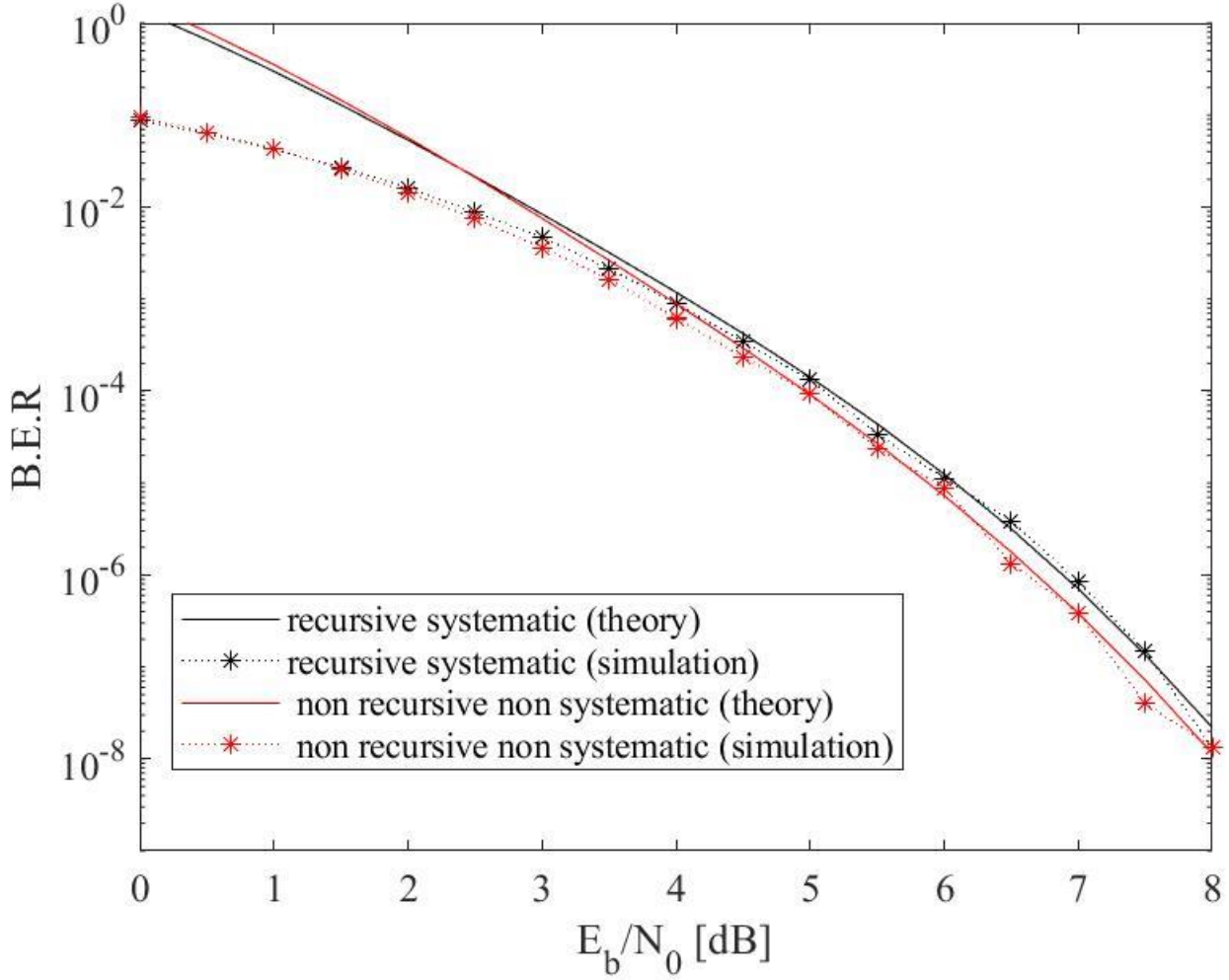
Matlab Example

- clear
- % Create a trellis structure to represent the encoder. Set the constraint length of the upper path to 5 and the constraint length of the lower path to 4. The octal representation of the code generator matrix corresponds to the taps from the upper and lower shift registers. The trellis structure serves as an input to the distspec function to represent the rate 2/3 convolutional code.
- rc=1/3; EbN0dB=0:10;EbN0=10.^(EbN0dB/10);pe=0;
- trellis = poly2trellis([5 4],[23 35 0; 0 5 13]);
- % Use the distspec function to compute the distance spectrum for a rate 2/3 convolutional code (10 terms)
- terms=10; spect = distspec(trellis,terms);
- % spect = struct with fields:
  % dfree: 5
  % weight: [1 6 28 142]
  % event: [1 2 8 25];
- % Determine the bound on the bit error probability
- for i=1:terms,pe=pe+1/2*spect.weight(i)*erfc((rc*EbN0*(spect.dfree+i-1)).^.5);end

Matlab Example (recursive)

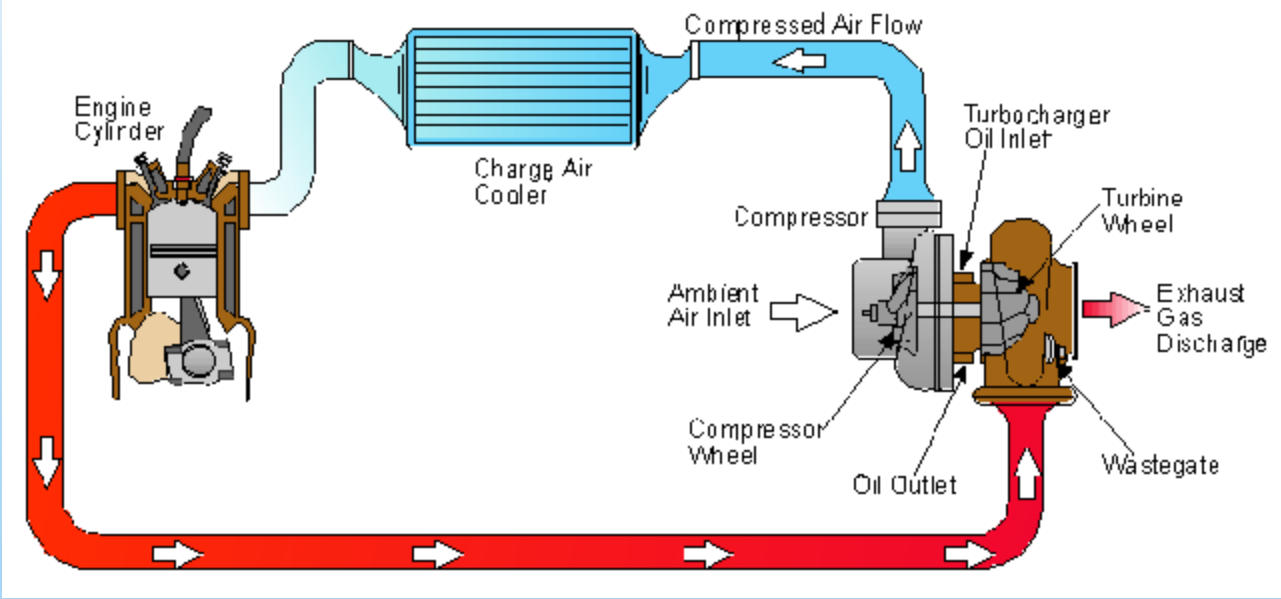
- clear
- % Create a trellis structure to represent the encoder. Set the constraint length of the upper path to 3. The octal representation of the code generator matrix corresponds to the taps from the upper and lower shift registers. The trellis structure serves as an input to the distspec function to represent the rate 1/2 convolutional code.
- rc=1/2; EbN0dB=0:10;EbN0=10.^(EbN0dB/10);pe=0;
- trellis = poly2trellis(3,[5 7],5);
- % Use the distspec function to compute the distance spectrum for a rate 2/3 convolutional code (10 terms)
- terms=10; spect = distspec(trellis,terms);
- % spect = struct with fields:
  % dfree: 5
  % weight: [2 6 14 32 72 160 352 768 1664 3584]
  % event: [1 2 4 8 16 32 64 128 256 512]
- % Determine the bound on the bit error probability
- for i=1:terms,pe=pe+1/2*spect.weight(i)*erfc((rc*EbN0*(spect.dfree+i-1)).^.5);end

Performance (B.E.R.)



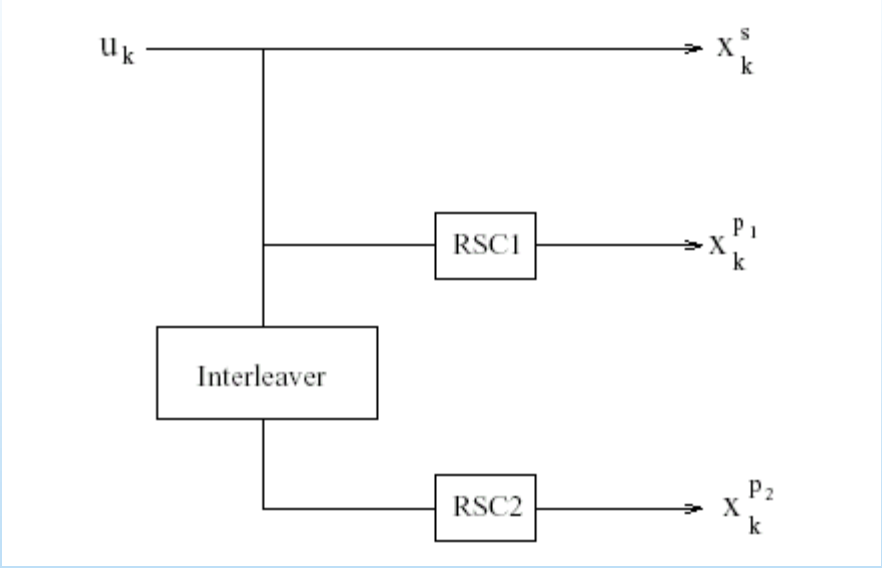
Iterative (turbo) decoding

- In 1993 Berrou and Glavieux proposed the concatenate iterative decoding (turbo decoding).
- Turbo codes owe their name to the decoding algorithm, which uses feedback, like the turbo engine: this algorithm is called iterative decoding.

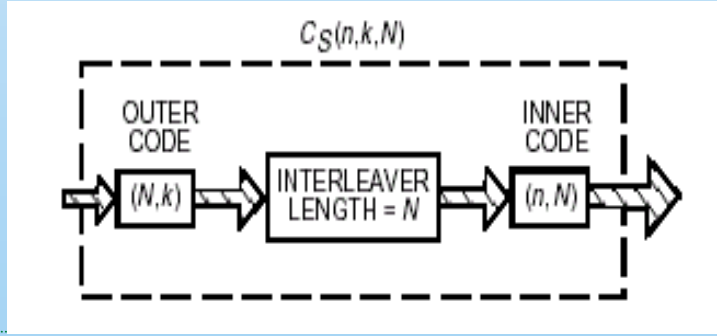


Parallel Concatenated Convolutional Codes

- A turbo encoder consists of the parallel concatenation of two recursive convolutional codes (PCCC: Parallel Concatenated Convolutional Codes) by means of a interleaver.



- A serial concatenation is also possible (and even preferable) with different properties and performance (SCCC)



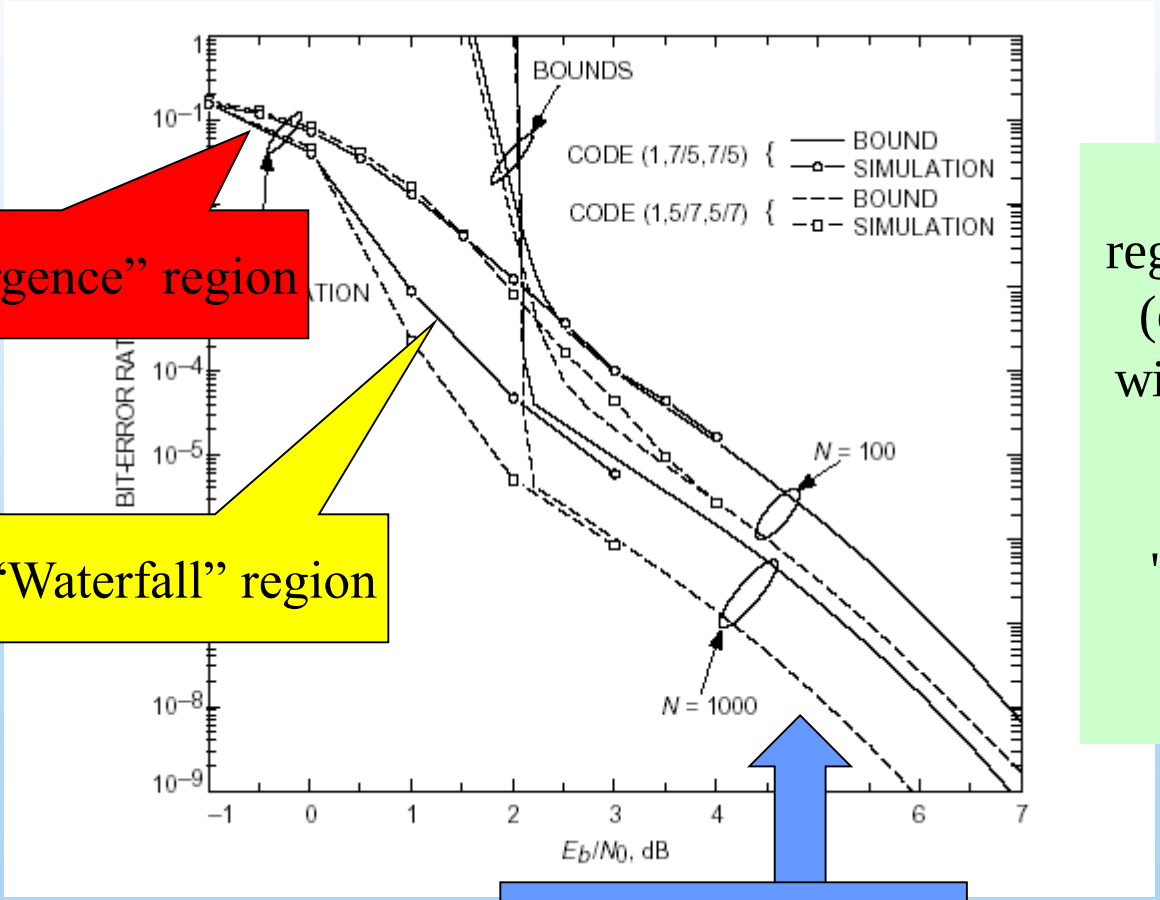
The role of the interleaver

- Name *w* the weight of an input sequence that may produce a low weight output sequence.
- For non recursive convolutional encoder w_{min}=1 (an input sequence consisting of single 1 produces a low weight output sequence).
- For recursive convolutional encoder w_{min}=2 (an input sequence consisting of single 1 produces an oscillating output sequence). The interleaver may be carefully designed for counteract the effects of *w*=2 sequences, providing the interleaver gain.

Recursive CC: w_{min} =2 → *Interleaver* Gain: N^{-1}

Non recursive CC: w_{min} =1 → *Interleaver* Gain independent from *N*

Performance



"Non convergence" region

"Waterfall" region

"Error floor" region

Three distinct regions of the BER (or FER) curves with respect to the SNR: the non-convergence, "waterfall" and "error floor" regions

Design parameters

- Parallel versus serial concatenation
- Number of concatenated codes
- Memory (# of states) of each code
- Code rate and generating polynomials for each constituent code
- Block length
- Excellent versus suboptimal decoding algorithms
- Trellis termination techniques
- Interleaver design
- Early iteration stopping techniques
- Using external block codes to lower the error floor

Design considerations

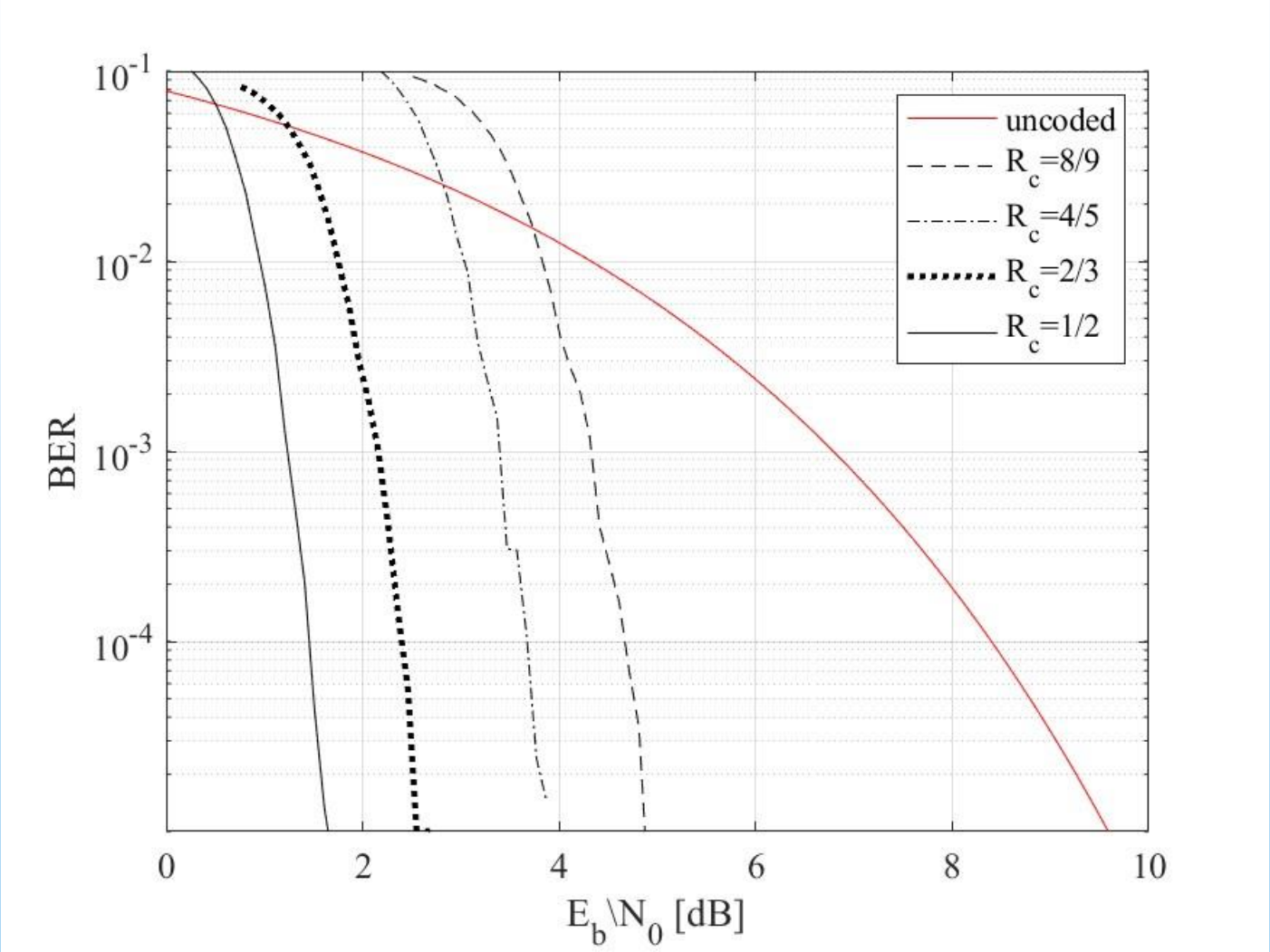
- Assume that the minimum weight of the input sequences used for leaving and going back to the state 0 is 2. The asymptotic expression (with respect to the interleaver length N) of the bit error probability is:

$$P_b(e) \approx \frac{1}{2} N^{-1} N_{C_1 \text{eff}} N_{C_2 \text{eff}} \operatorname{erfc}\left(\sqrt{\frac{d_{\text{Peff}} R_c E_b}{N_0}} \right)$$

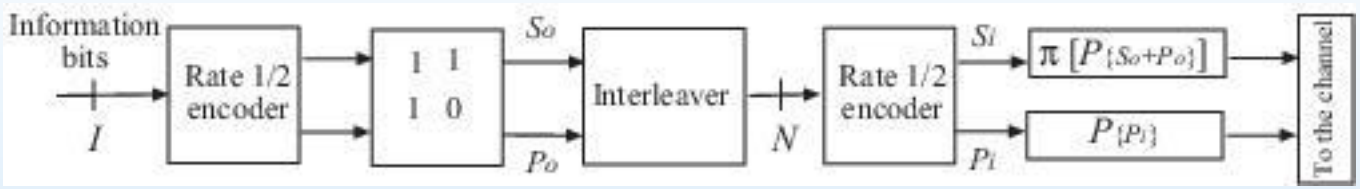
- being d_{Peff} is the effective free distance of the PCCC, i.e. the minimum weight of the PCCC sequences generated from input sequences of weight 2, and $N_{C i \text{eff}}$ (i=1,2) is the multiplicity of error events of the two CCs with weight equal to the effective free distance.

- It follows that, to improve the turbo code performance is necessary to increase d_{Peff} and to decrease $N_{C i \text{eff}}$.

PCCC coding gain

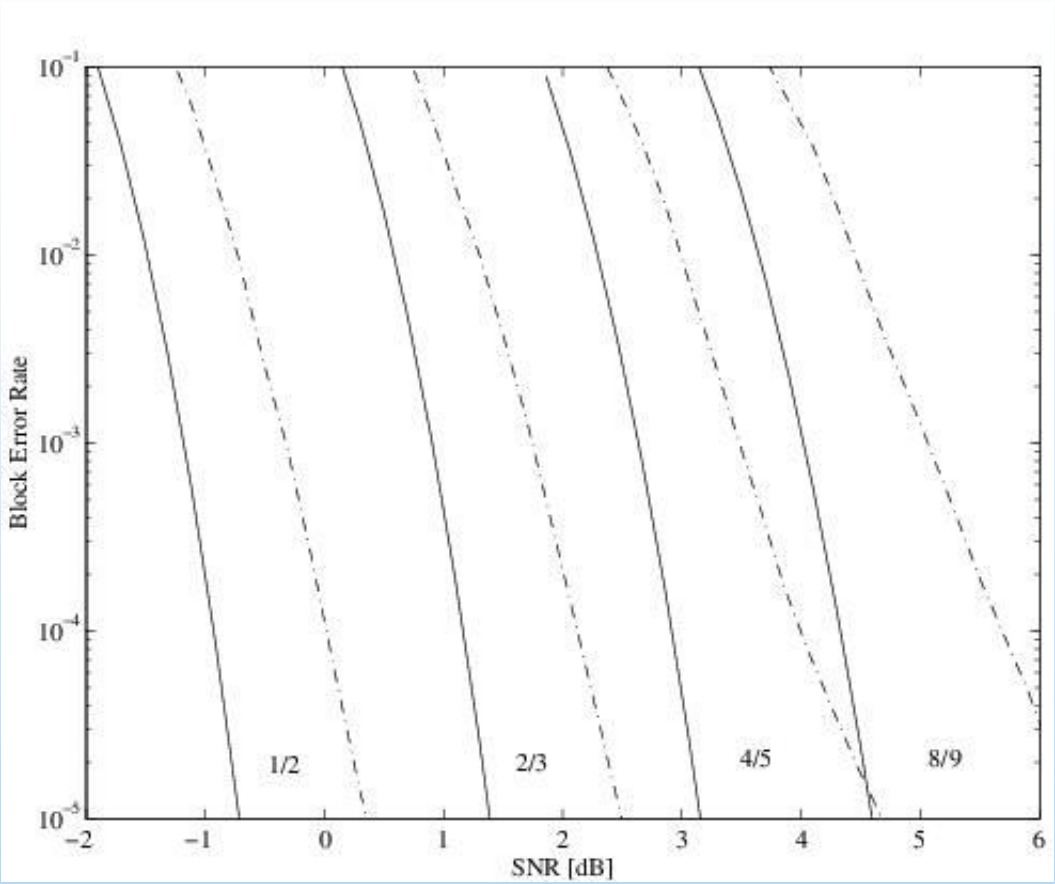


SCCC example



- Comparison of some rate compatible puncturing schemes, obtained starting from a rate 1/3 SCCC mother code formed by serially concatenating a 4-states rate 2/3 outer systematic recursive convolutional code (SRCC) and a 4-states rate 1/2 inner SRCC. A non periodic puncturing has been applied on inner output bits only. The frame length, measured in terms of input information bits, has been set to 200 using a random interleaver.

SCCC performance



- N=200, random interleaver
- The actual performance is compared with the limiting one

Low Density Parity Check Codes

- LDPC codes are parity check codes with sparse (low density) check matrix. They were first proposed by Gallagher in his 1960 doctoral dissertation.
- Most standards adopt LDPC codes instead of Turbo Codes. this derives partly from some patent issues related to Turbo Codes.
- But the LDPC have demonstrated to achieve a performance that is very close to the Shannon bound with simpler decoding algorithms.
- The decoding algorithms are independent from the coding rate (this is also true for the Turbo Codes).
- Both LDPC and Turbo Codes have a poor performance for short packets.

PhD in Industrial and Information Engineering

Recursive decoding for Parity Check Codes

MAP in the presence of a code

- Assume that **y** is a sequence of received noisy values, obtained by transmitted a sequence **x** of antipodal, binomial values. The MAP criterion may be applied as follows.

$$\log\frac{p(x_i=+|\mathbf{y})}{p(x_i=-|\mathbf{y})}=\log\frac{p(\mathbf{y}|x_i=+)\,p(x_i=+)}{p(\mathbf{y}|x_i=-)\,p(x_i=-)}=\log\frac{p(y_1,y_2,\ldots,y_n|x_i=+)\,p(x_i=+)}{p(y_1,y_2,\ldots,y_n|x_i=-)\,p(x_i=-)}=$$

$$=\log\frac{p(y_1,y_2,..,y_{i-1},y_{i+1},..,y_n|x_i=+)\,p(y_i,|x_i=+)\,p(x_i=+)}{p(y_1,y_2,..,y_{i-1},y_{i+1},..,y_n|x_i=-)\,p(y_i,|x_i=-)\,p(x_i=-)}$$

$$=\log\frac{p(y_1,y_2,..,y_{i-1},y_{i+1},..,y_n|x_i=+)}{p(y_1,y_2,..,y_{i-1},y_{i+1},..,y_n|x_i=-)}$$ Code

$$+\log\frac{p(y_i,|x_i=+)}{p(y_i,|x_i=-)}$$ Channel

$$+\log\frac{p(x_i=+)}{p(x_i=-)}$$ A priori

Parity codes: APP math

- Name $L = \log \frac{p_+}{p_-} = \log \frac{p_+}{1-p_+}$. We have $p_+ = \frac{e^L}{1+e^L}$.
- Consider a pair of (antipodal) bits, u_1, u_2. u_1u_2=+ se u_1=u_2.

$$p_+(u_1u_2) = p(u_1^+)p(u_2^+) + (1-p(u_1^+))(1-p(u_2^+)) = 1 - p(u_1^+) - p(u_2^+) + 2p(u_1^+)p(u_2^+)$$

$$p_-(u_1u_2) = p(u_1^+) + p(u_2^+) - 2p(u_1^+)p(u_2^+)$$

- Substituting, after some simple calculations we get

$$p_+(u_1u_2) = \frac{1+e^{L(u_1)}e^{L(u_2)}}{(1+e^{L(u_1)})(1+e^{L(u_2)})}$$

$$p_-(u_1u_2) = \frac{e^{L(u_1)}+e^{L(u_2)}}{(1+e^{L(u_1)})(1+e^{L(u_2)})}$$

$$L(u_1u_2) = \log\left[\frac{1+e^{L(u_1)}e^{L(u_2)}}{e^{L(u_1)}+e^{L(u_2)}}\right]$$

APP math II

- By induction it is straightforward to demonstrate that

$$L\left(u_1 u_2 \ldots u_J\right) = \log\left[\frac{\prod_{j=1}^{J}\left(e^{L(u_j)}+1\right)+\prod_{j=1}^{J}\left(e^{L(u_j)}-1\right)}{\prod_{j=1}^{J}\left(e^{L(u_j)}+1\right)-\prod_{j=1}^{J}\left(e^{L(u_j)}-1\right)}\right]$$

define $\alpha = \dfrac{\prod_{j=1}^{J}\left(e^{L(u_j)}-1\right)}{\prod_{j=1}^{J}\left(e^{L(u_j)}+1\right)} = \dfrac{\prod_{j=1}^{J}\left(e^{L(u_j)/2}-e^{L(u_j)/2}\right)}{\prod_{j=1}^{J}\left(e^{L(u_j)/2}+e^{L(u_j)/2}\right)} = \prod_{j=1}^{J}\tanh\left(L\left(u_j\right)/2\right)$

we have $L\left(u_1 u_2 \ldots u_J\right) = \log\left[\dfrac{1+\alpha}{1-\alpha}\right]$

from which $\alpha = \dfrac{e^{L(u_1 u_2 \ldots u_J)}-1}{e^{L(u_1 u_2 \ldots u_J)}+1} = \tanh\left(L\left(u_1 u_2 \ldots u_J\right)/2\right)$

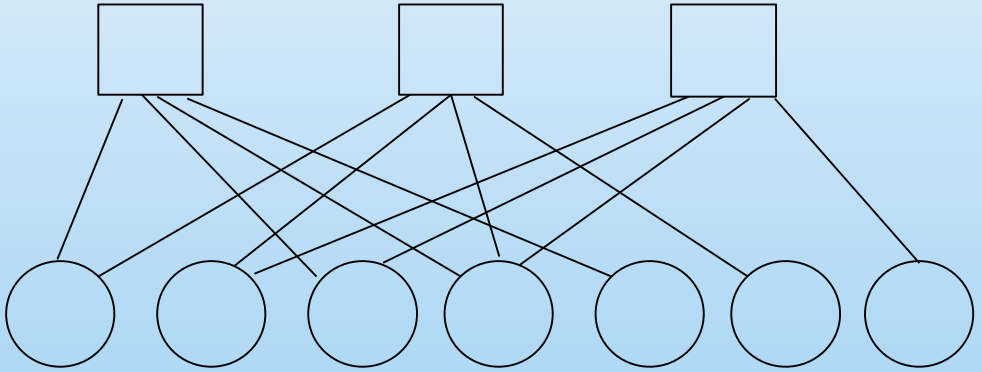
We finally obtain $L\left(u_1 u_2 \ldots u_J\right) = 2\ \operatorname{arctanh}\left(\alpha\right) = 2\ \operatorname{arctanh}\left(\prod_{j=1}^{J}\tanh\left(L\left(u_j\right)/2\right)\right)$

Graph Representations of Codes
Tanner Graph

- A Tanner Graph is a bipartite graph which represents the parity check matrix of an error correcting code.
- **H** is the (*n*–*k*)-by-*n* parity check matrix. The Tanner graph has:
- *n* bit nodes (or variable nodes), represented by circles.
- *n*–*k* check nodes, represented by squares.
- There is an edge between bit node *i* and check node *j* if there is a one in row *i* and column *j* of **H**.

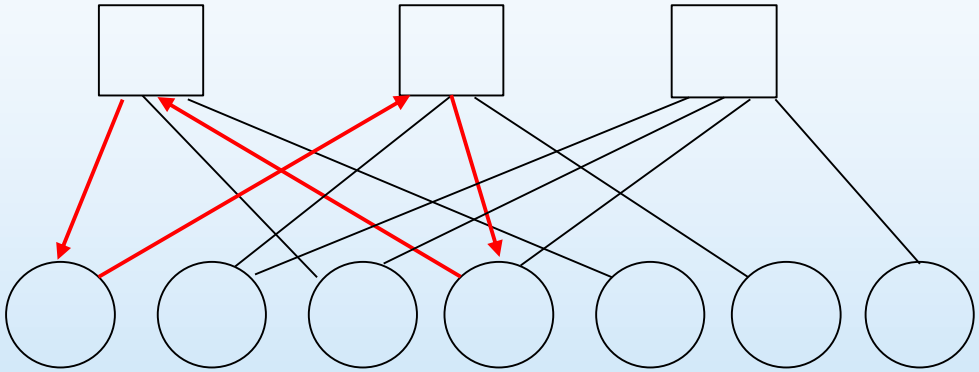
$$\mathbf{H} = \left[\begin{array}{cccc|ccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

Hamming Code



Tanner graph

Cycles and Girth of Tanner Graphs



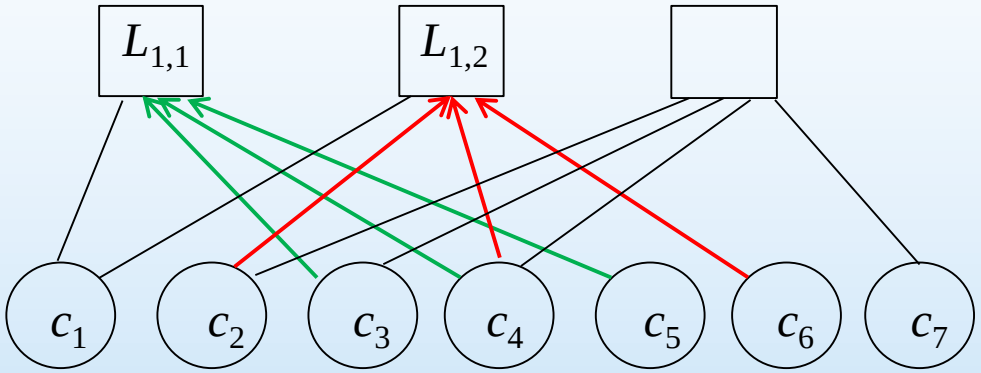
$$\mathbf{H} = \left[\begin{array}{ccccccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$

A cycle of length L in a Tanner graph is a path of L edges which closes back on itself

The girth of a Tanner graph is the minimum cycle length of the graph.

Message passing decoding

$$\mathbf{H} = \left[\begin{array}{cccc|ccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$



$c_1 \oplus c_3 \oplus c_4 \oplus c_5 = 0$

$c_1 \oplus c_2 \oplus c_4 \oplus c_6 = 0$

$c_2 \oplus c_3 \oplus c_4 \oplus c_7 = 0$

$c_1 = c_3 \oplus c_4 \oplus c_5$

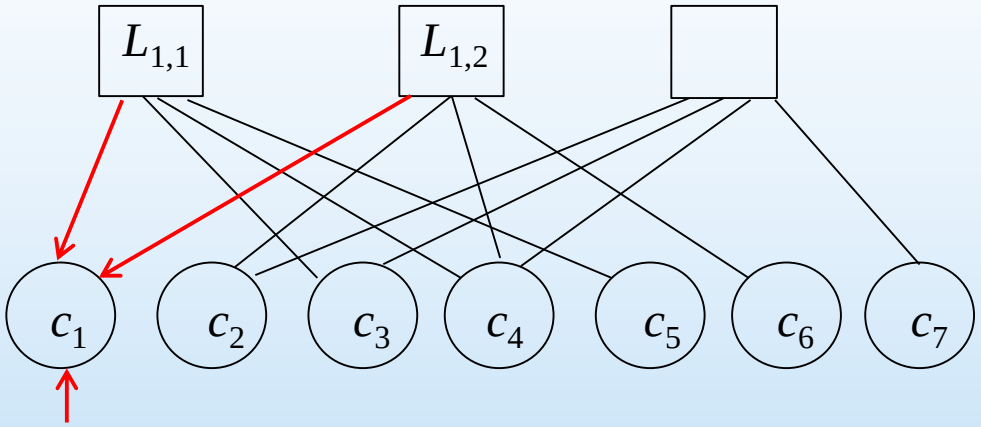
$$L_{1,1} = 2\,\operatorname{arctanh}\left(\tanh\left(\frac{L(c_3)}{2}\right)\tanh\left(\frac{L(c_4)}{2}\right)\tanh\left(\frac{L(c_5)}{2}\right)\right)$$

$c_1 = c_2 \oplus c_4 \oplus c_6$

$$L_{1,2} = 2\,\operatorname{arctanh}\left(\tanh\left(\frac{L(c_2)}{2}\right)\tanh\left(\frac{L(c_4)}{2}\right)\tanh\left(\frac{L(c_6)}{2}\right)\right)$$

Message passing decoding

$$\mathbf{H} = \left[\begin{array}{cccc|ccc} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{array}\right]$$



$L_{1,0}$ from the channel

$$L(c_1) = L_{1,0} + L_{1,1} + L_{1,2}$$

Decoding through message passing

- The algorithm continues iteratively until all the equations are satisfied, otherwise it stops after a given number of steps (error detected).
- Message passing is optimal if there are no cycles in the Tanner graph, otherwise it is suboptimal.
- The performance approach the optimal one if the girth is more than 4 (if there are no 4-cycles in the Tanner graph.

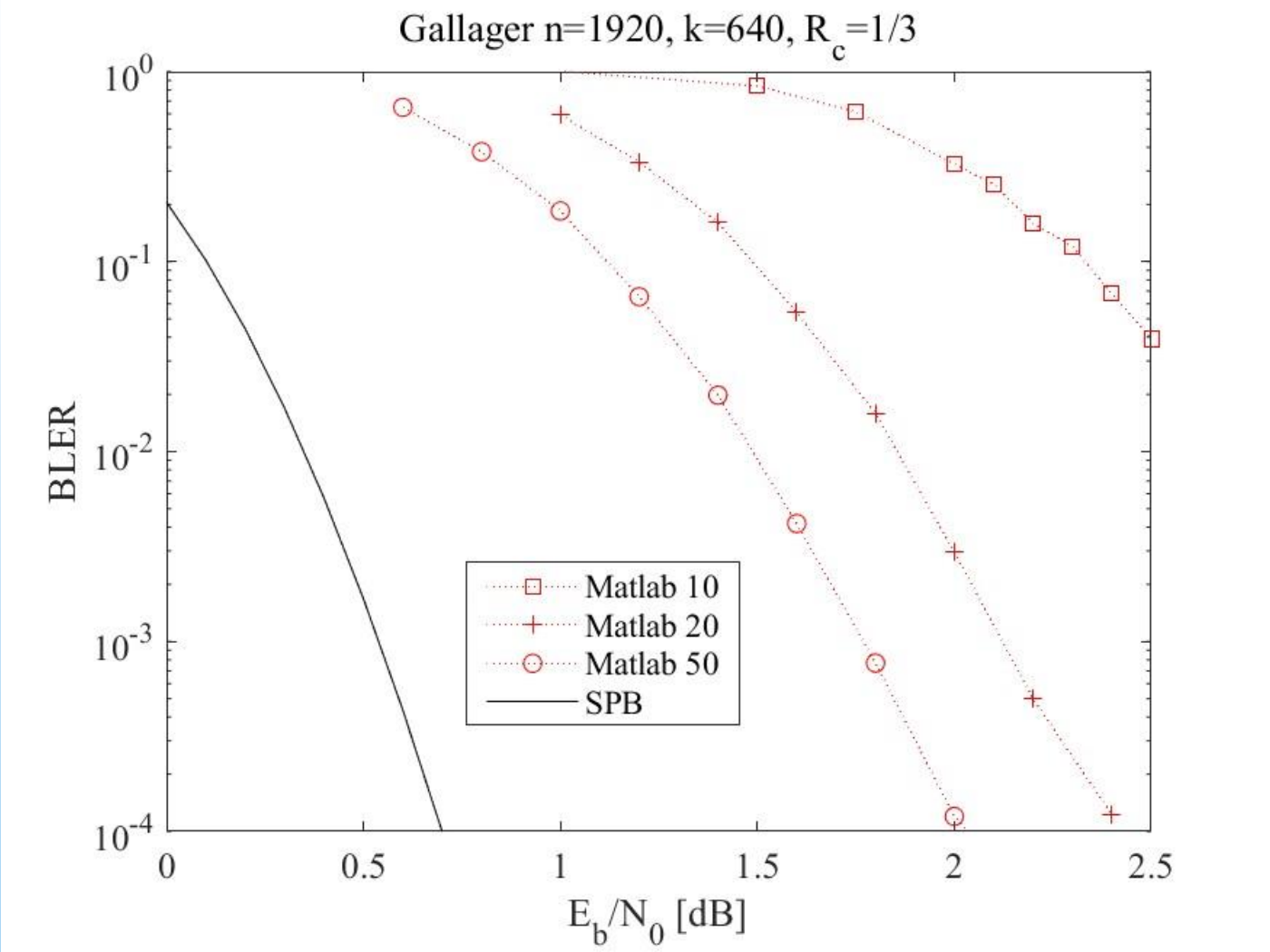
Regular (Gallager) LDPC codes

- Degree of a node: the number of edges connected to it.
- Regular (Gallager) LDPC codes: all variable nodes have the same degree and all the check nodes have the same degree.
- Parameters (n, j, i)
- n = codeword length
- j = number of parity-check equations involving each code bit = degree of each variable node
- i = code bits involved in each parity-check equation = degree of each check node.
- Locations of 1's in **H** (size n-k, n) may be chosen randomly, subject to (j, i) constraints.
- $n=mi$, $n-k=mj=nj/i$, $n(1-j/i)=k$, $R_c=n/k=1-j/i$
- Irregular codes may have a better performance.

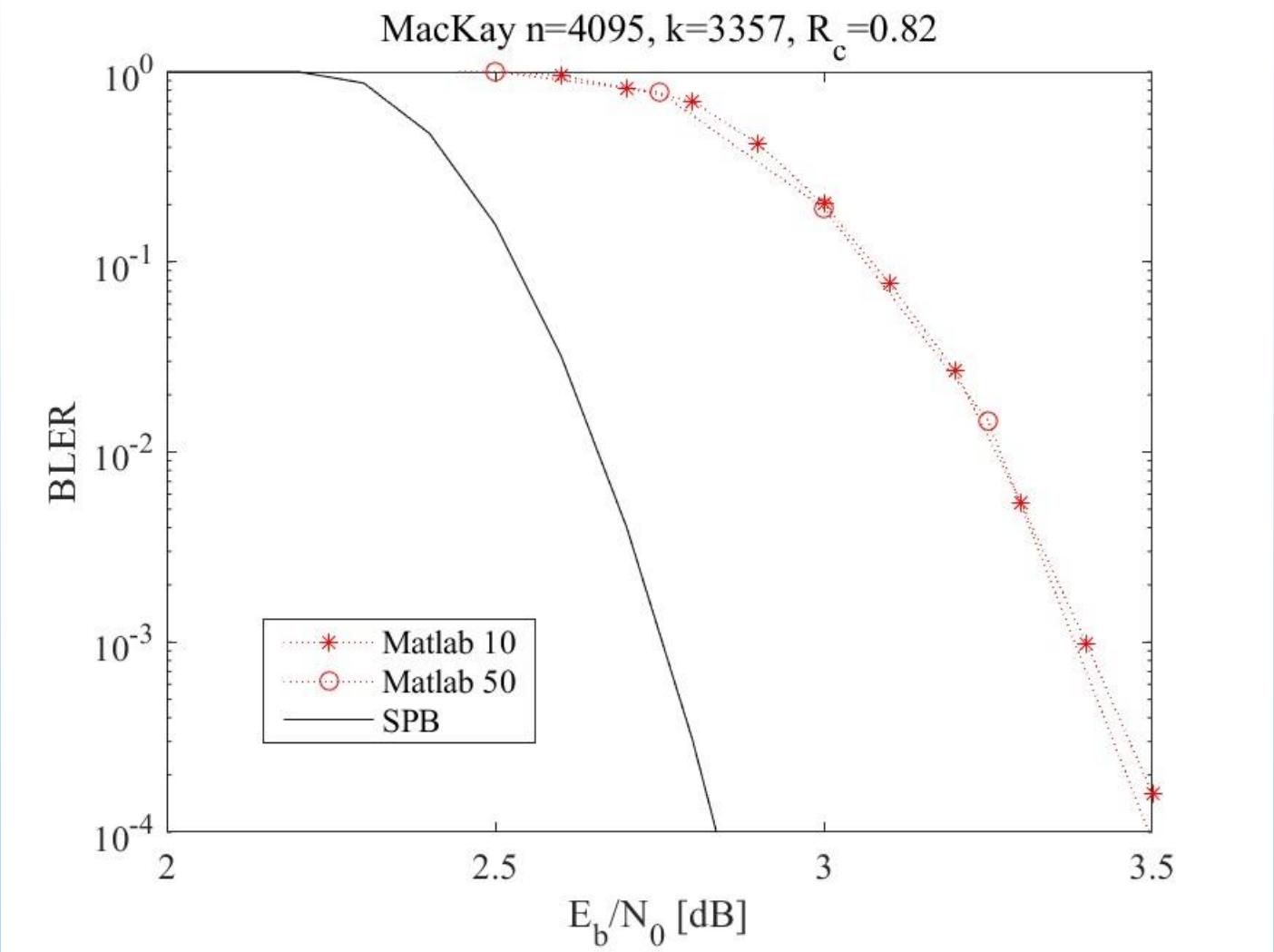
Example (n=20; j=3; i=4; R_c=1/4)

$$\mathbf{H} = \begin{bmatrix}
1 & 1 & 1 & 1 & & & & & & & & & & & & & & & & \\
 & & & & 1 & 1 & 1 & 1 & & & & & & & & & & & & \\
 & & & & & & & & 1 & 1 & 1 & 1 & & & & & & & & \\
 & & & & & & & & & & & & 1 & 1 & 1 & 1 & & & & \\
 & & & & & & & & & & & & & & & & 1 & 1 & 1 & 1 \\
1 & & & & 1 & & & & 1 & & & & 1 & & & & & & & \\
 & 1 & & & & 1 & & & & 1 & & & & & & & 1 & & & \\
 & & 1 & & & & 1 & & & & & & & 1 & & & & 1 & & 1 \\
 & & & 1 & & & & & & & 1 & & & & 1 & & & & 1 & \\
 & & & & & & & 1 & & & & 1 & & & & 1 & & & & 1 \\
1 & & & & & 1 & & & & & 1 & & & & & & 1 & & \\
 & 1 & & & & & 1 & & & 1 & & & & & 1 & & & & & \\
 & & 1 & & & & & 1 & & & & & 1 & & & & & 1 & \\
 & & & 1 & & & & & 1 & & & & & 1 & & & & & \\
 & & & & 1 & & & & & 1 & & & & & 1 & & & & 1
\end{bmatrix}$$

Performance example



Performance example



Single Parity Check Codes (I)

- Multi-dimensional product codes (M-SPC)
- Data are arranged in a hypercube of dimension D [1].
- The length of each dimension is k_i, corresponding to a encoded length $n_i = k_i + 1$. Before puncturing $R_c = \prod_{i=1}^{D} k_i / (k_i + 1)$
- Limited flexibility.
- Interleaving may be not used.
- Puncturing is effective.
- Good performance at average/low rates.

[1] M. Ranking and T. A. Gulliver, "Single parity check product codes", IEEE-COM, Vol. 49, n. 8, Aug 2001, pp. 1354-1362.

Single Parity Check Codes (II)

Multiple parity-check codes (P-SPC, E-SPC)

- SPC: Data are arranged in a LC matrix.
  - P-SPC: P parity columns; $R_c=C/(C+P)$ [2].
  - E-SPC: added 1 parity row; $R_c \approx C/(C+P)$, for $L>>1$.
- More flexibility (N depends on L; R_c does not depend on L).
- Interleaving is required (no interleaver gain).
- Puncturing is less effective.
- Good performance at high rates (best for $R_c=3/4$).

[2] J.S.K. Tee, D.P. Taylor, P. A. Martin, "Multiple serial and parallel concatenated single parity-check codes", IEEE-COM, Vol. 51, n. 10, Oct 2003, pp. 1666-1675.

Design (short blocks)

PCCCs

Modulation	Rc	R	B	Thr. [dB]
BPSK	1/3	1/3	1286	-4.9
QPSK	1/3	2/3	642	-1.9
QPSK	1/2	1	428	0.6
8-PSK	1/2	3/2	285	3.5
16-QAM	1/2	2	214	5.6
8-PSK	3/4	9/4	191	7.7
16-QAM	3/4	3	143	9.8
64-QAM	2/3	4	107	13.2

SPCs

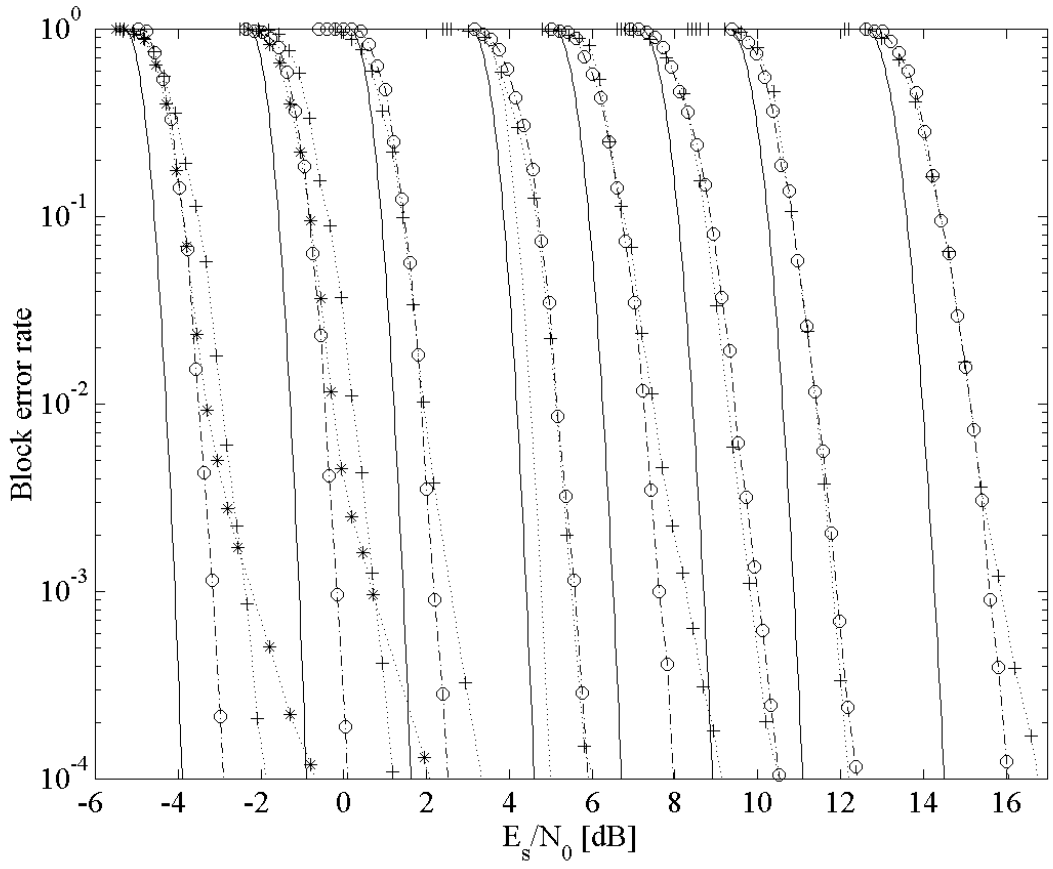
Modulation	Rc	R	Type	K	B	Thr. [dB]
BPSK	1/3	1/3	M	4,4,3,3,3	1296	-4.9
QPSK	1/3	2/3	M	4,4,3,3,3	648	-1.9
QPSK	1/2	1	E	65,7	455	0.6
QPSK	3/4	3/2	E	29,15	290	3.8
16-QAM	1/2	2	E	65,7	227	5.5
8-PSK	3/4	9/4	E	29,15	194	7.6
16-QAM	3/4	3	E	27,17	153	9.8
64-QAM	2/3	4	E	45,10	112	13.2

Before puncturing R_c=0.27

M-SPCs at low rates, E-SPCs at intermediate and high rates

AWGN: code performance

N=429/*R*. Bound (solid lines: PCCCs, dotted lines: SPCs); actual performance (dash-dotted lines with 'o'PCCCs; dotted lines with '*': M-SPCs; with '+': E-SPCs)



Water-fall performance affects efficiency (goodput)
Error floor performance affects residual error rate

Design (long blocks)

PCCCs

Modulation	Rc	R	B	Thr. [dB]
BPSK	1/3	1/3	165	-5.2
QPSK	1/3	2/3	8183	-2.1
QPSK	1/2	1	5458	0.3
8-PSK	1/2	3/2	3639	3.1
16-QAM	1/2	2	2729	5.2
8-PSK	3/4	9/4	2426	7.3
16-QAM	3/4	3	1820	9.4
64-QAM	2/3	4	1365	12.8

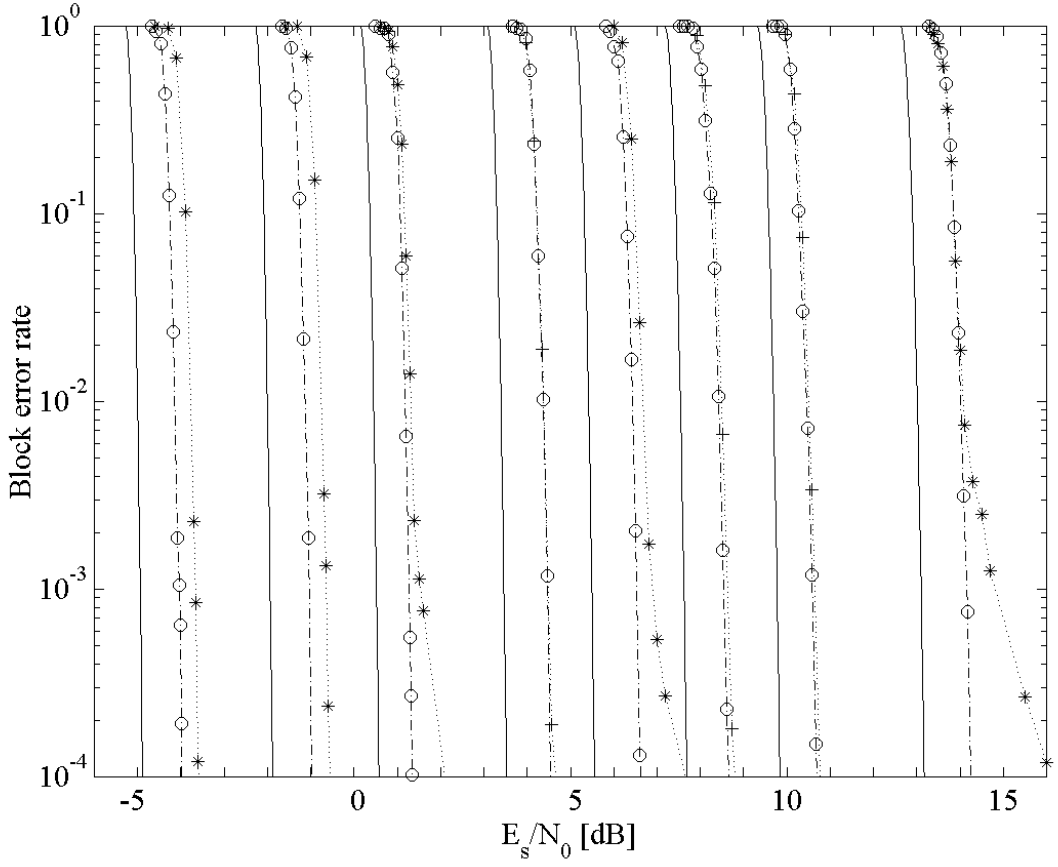
SPCs

Modulation	Rc	R	Type	K	B	Thr. [dB]
BPSK	1/3	1/3	M	5,4,4,4,4,4	15360	-5.2
QPSK	1/3	2/3	M	5,4,4,4,4,4	7680	-2.1
QPSK	1/2	1	M	6,6,5,5,5	4500	0.3
QPSK	3/4	3/2	E	303,18	3648	3.5
16-QAM	1/2	2	M	6,6,5,5,5	2250	5.3
8-PSK	3/4	9/4	E	303,18	2432	7.3
16-QAM	3/4	3	E	303,18	1824	9.4
64-QAM	2/3	4	M	11,9,8,7	1326	12.8

Boundary moves at higher rates for longer blocks

AWGN: code performance

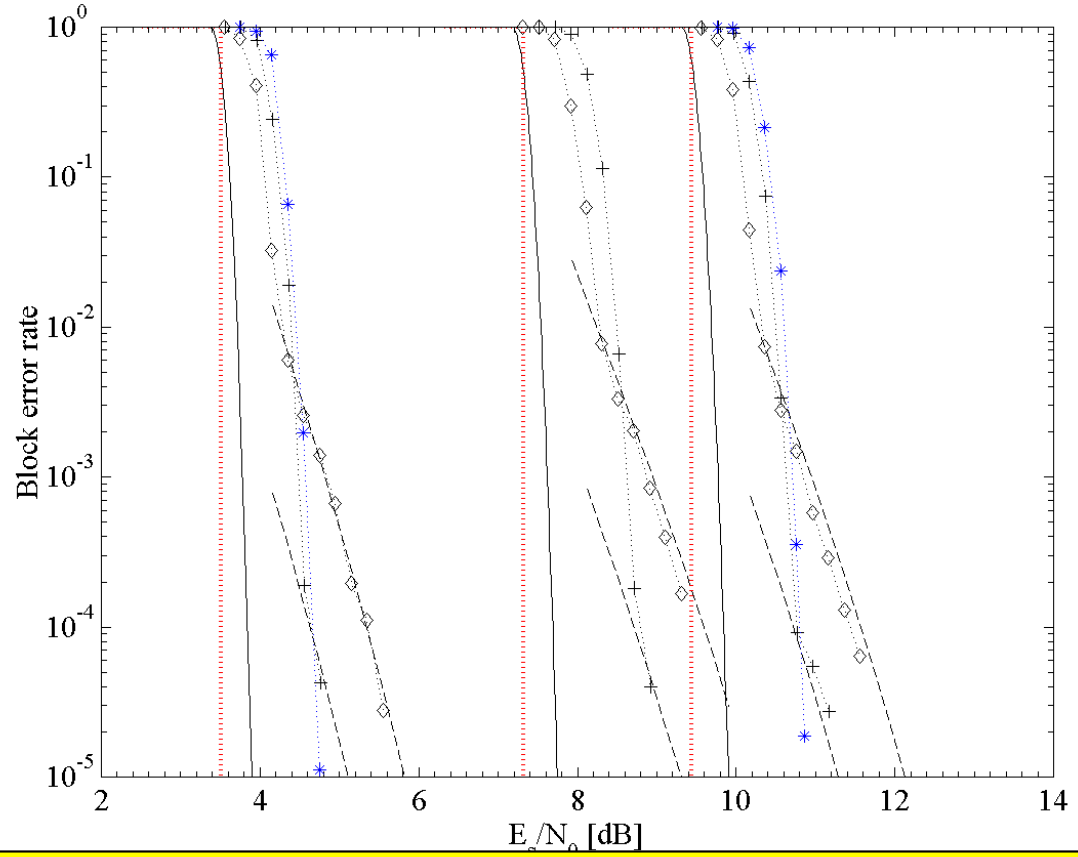
N=5000/*R*. Bound (solid lines: PCCCs, dotted lines: SPCs); actual performance (dash-dotted lines with 'o'PCCCs; dotted lines with '*': M-SPCs; with '+': E-SPCs)



Slightly different water-fall performance (PCCCs and M-SPCs)
Improved error floor performance for E-SPCs

AWGN: SCCC code performance

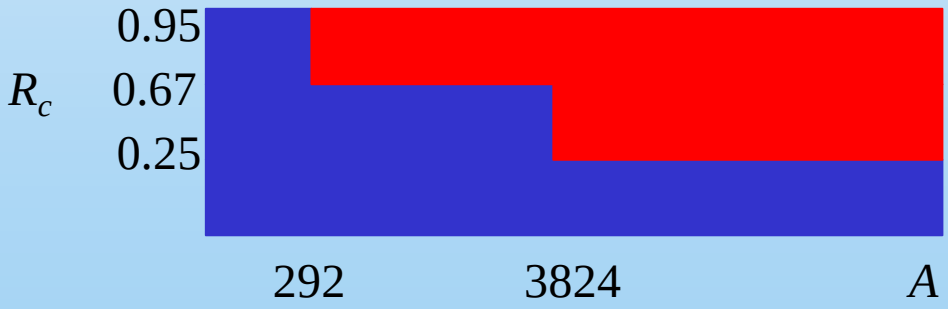
R_c=3/4, QPSK, 8PSK 16QAM.
Dotted lines with '◊': best waterfall performance; with '+': chosen candidates.



Residual error rate may be reduced adopting serial versions (in blue)

Error Control for 5G: LDPC

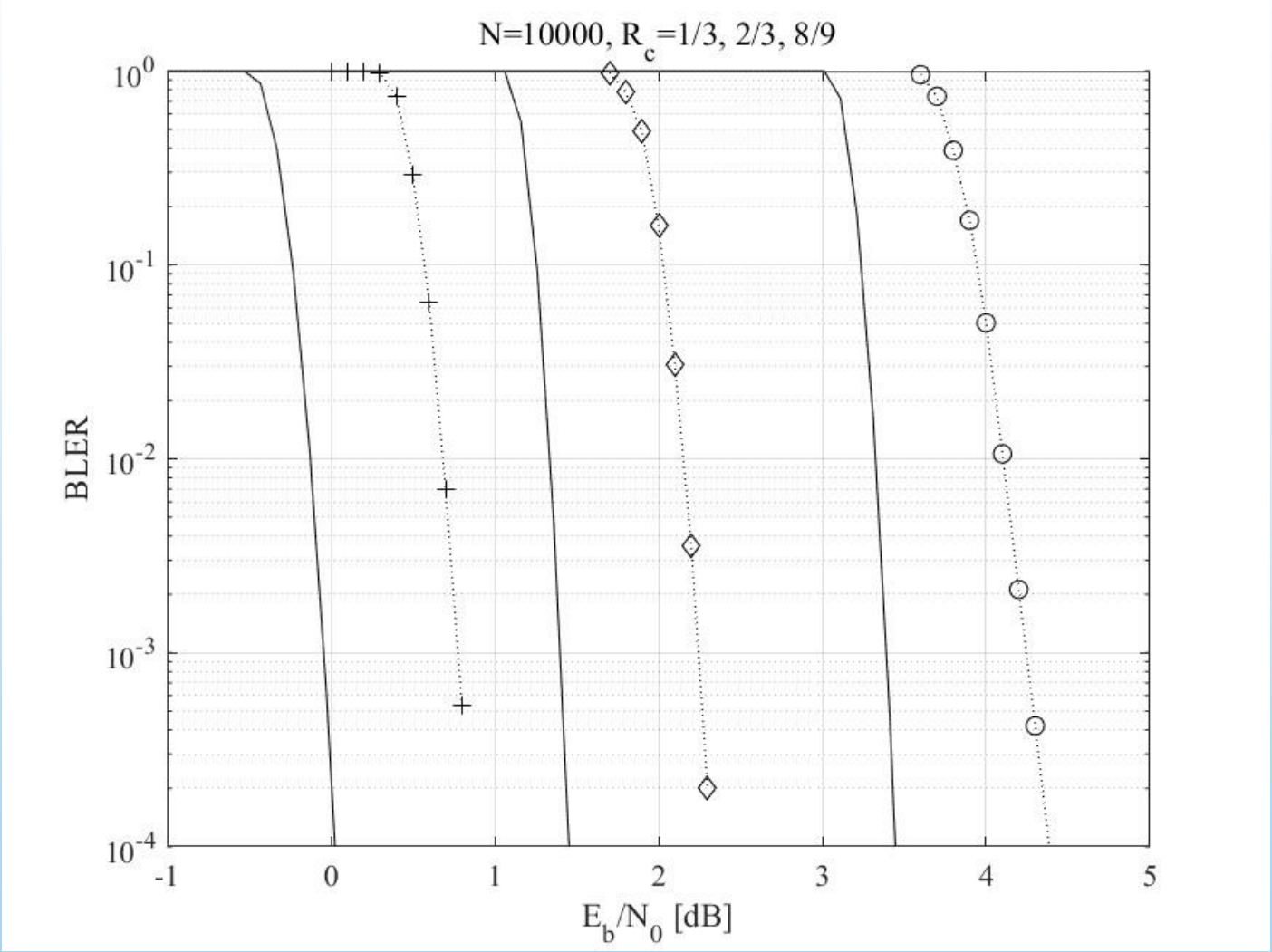
- Name *A* the payload size.
- The information block, *B*, is obtained by adding to the payload *L* bits of CRC, being *L*=24 if *A*>3824, and *L*=16 otherwise,.
- The information block is then segmented in blocks of size *K*, for channel encoding.
- The rules for selecting among base graph 1 and base graph 2 are
  - $A \leq 292$ base graph 2 (blue)
  - $A \leq 3824$, $R_c \leq 0.67$ base graph 2
  - $R_c \leq 0.25$ base graph 2
  - Else base graph 1 (red)
-



Error Control for 5G: LDPC

- Name N the number of channel bits per block.
- For base graph 1, K=22 Z_c, while N=66 Z_c. Thus, the basic rate is R_c=1/3.
- For base graph 2, K=10 Z_c, while N=50 Z_c. Thus, the basic rate is R_c=1/5.
- $N \leq 8448 = 22 \times 384$ for base graph 1 and $N \leq 3840 = 10 \times 384$ for base graph 2.
- Different rates are obtained through a Rate Matching algorithm.
- 5G supports hybrid-ARQ, by adopting incremental redundancy schemes.

Limiting performance



A comment

- LDPC limiting performance is close to the Shannon bound.
- Thus, they may be used instead of Turbo Codes which have a similar performance but are more complex.
- Are we satisfied?
- Not completely because we need reliable codes for short packets, and LDPC have a poor performance for short packets.
- But there is a solution: the Polar codes.

Error Control for 5G: Polar Codes (1)

- Polar codes [3] are error-correcting codes, which are able to achieve the capacity of binary-input memoryless symmetric (BMS) channels. This means that one can transmit at the highest possible rate over that class of channels. In addition, the encoding and decoding operations can be performed with low complexity, thanks to recursive techniques.

- Polar codes exploit channel polarization. More precisely, the BMS channel is characterized by the transition probability $W(y|x)$ (the probability that, having transmitted x, y was received). The polarization technique of channels consists in recursively building, starting from W, $N = 2^n$ binary input channels $W_N^{(1)}$, …, $W_N^{(N)}$. These channels are said to be polarized. They behave asymptotically as perfect channels or useless channels, allowing you to create a method of coding that sends information only through asymptotically perfect channels.

- [3] E. Arıkan, "Channel polarization: a method for constructing capacity-achieving codes for symmetric binary-input memoryless channels", IEEE Trans. Inf. Theory, vol. 55, no. 7, pp. 3051-3073, July 2009.

Error Control for 5G: Polar Codes (2)

- Formally, a specific polar code is fully defined by a 4-tuple (N, R_c, A, u_{Ac}) where:
  - N is the block length, i.e. the total number of bits transmitted over the channel.
  - R_c is the code rate, $R_c \in [0,1]$
  - A is the information set, $A \subset \{1,...,N\}$ i.e. the set of positions which contains the information bits.
  - u_{Ac} are the frozen bits, $u_{Ac} \in \{0,1\}^{N(1-Rc)}$, i.e. bits which have fixed values, shared between the encoder and the decoder.
- The coding matrix has a recursive structure: $G_N = B_N G_2^{\otimes \log_2 N} = B_N G_2^{\otimes n}$

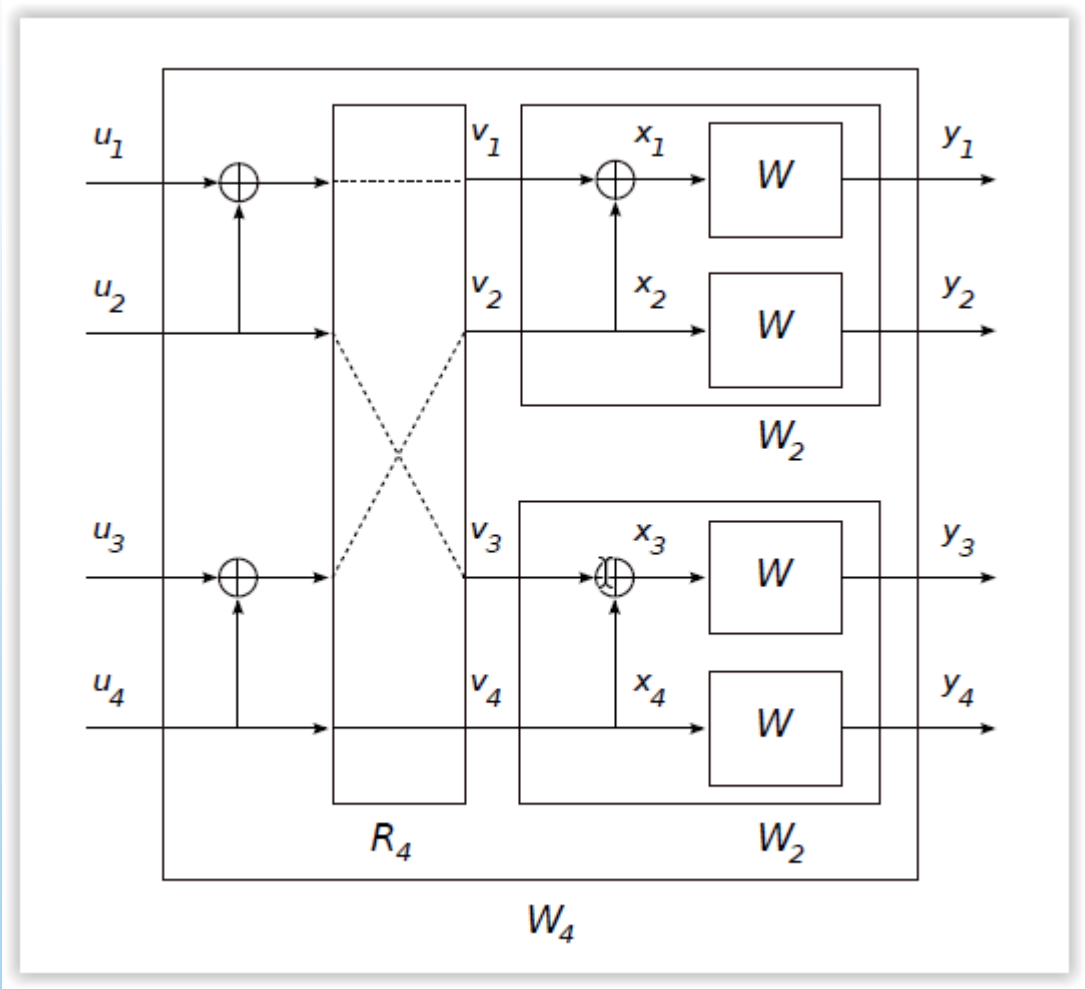
in which B_2 is a bit reordering matrix, $G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$, and $\otimes$ is the Kronecker product: $G_2 \otimes G_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$. After the reordering $G_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$

Error Control for 5G: Polar Codes (3)

- x^N_1 is transmitted over the channel W_N (which corresponds to N uses of the channel W), and y^N_1 is received.
- From y^N_1 the successive cancellation (SC) decoder produces an estimate of u^N_1 (making also use of the frozen bits values).
- The complexity of this operation is O(N log N).
- Finally, only the components of u^N corresponding to information bits are kept, yielding u^A

Error Control for 5G: Polar Codes (4)

- Encoder and transmission process for N=4;



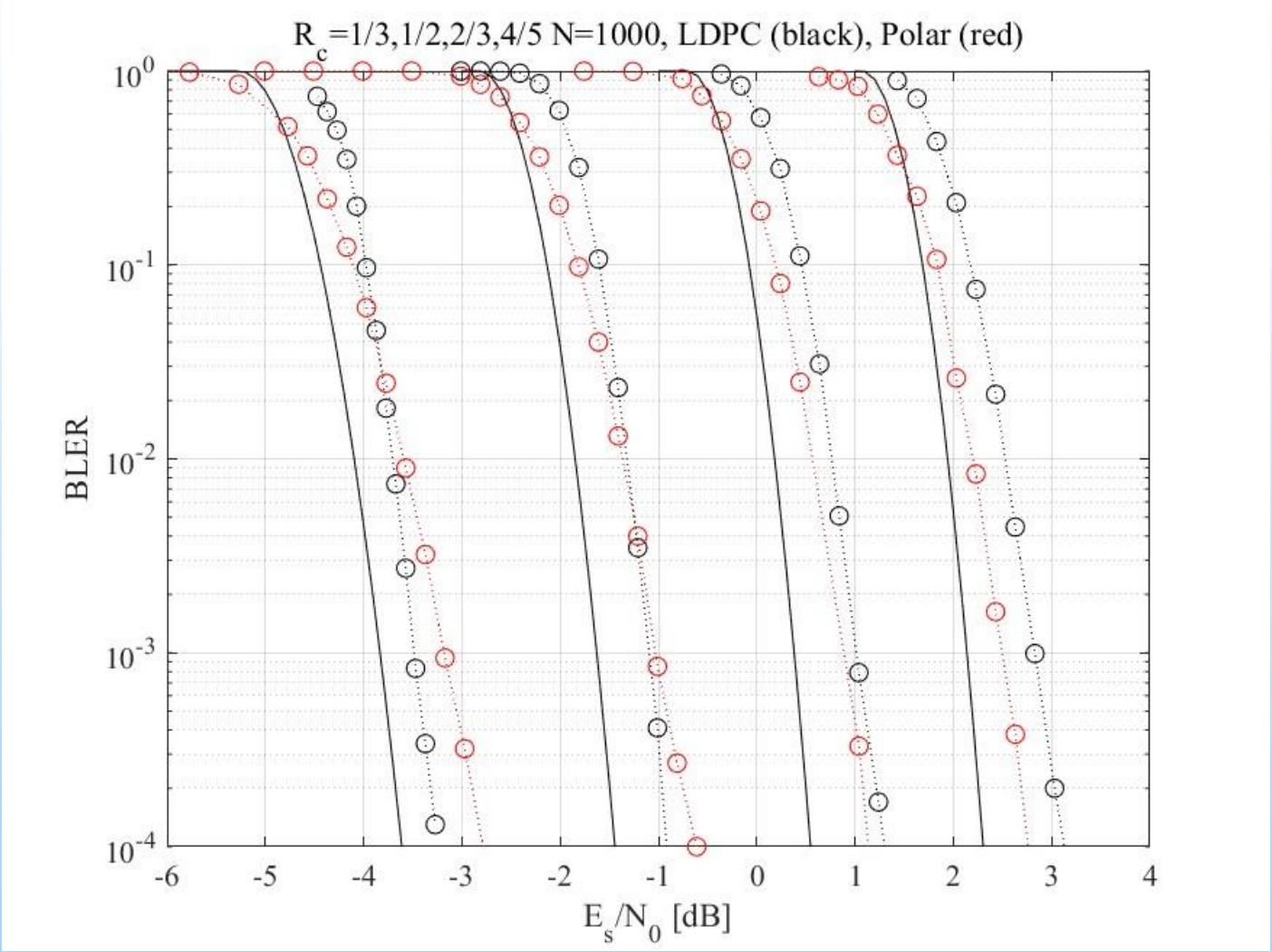
Capacity of individual channels

- Consider, for simplicity, the erasure channel with erasure probability e.
- At every iteration step, each channel of capacity C_n originates two channels of capacity respectively $2C_n e - C_n^2$ (the good one) and C_n^2 (the bad one), and the total capacity becomes $2C_n$.
- Example with e=0.5; C_1=1-e=0.5.

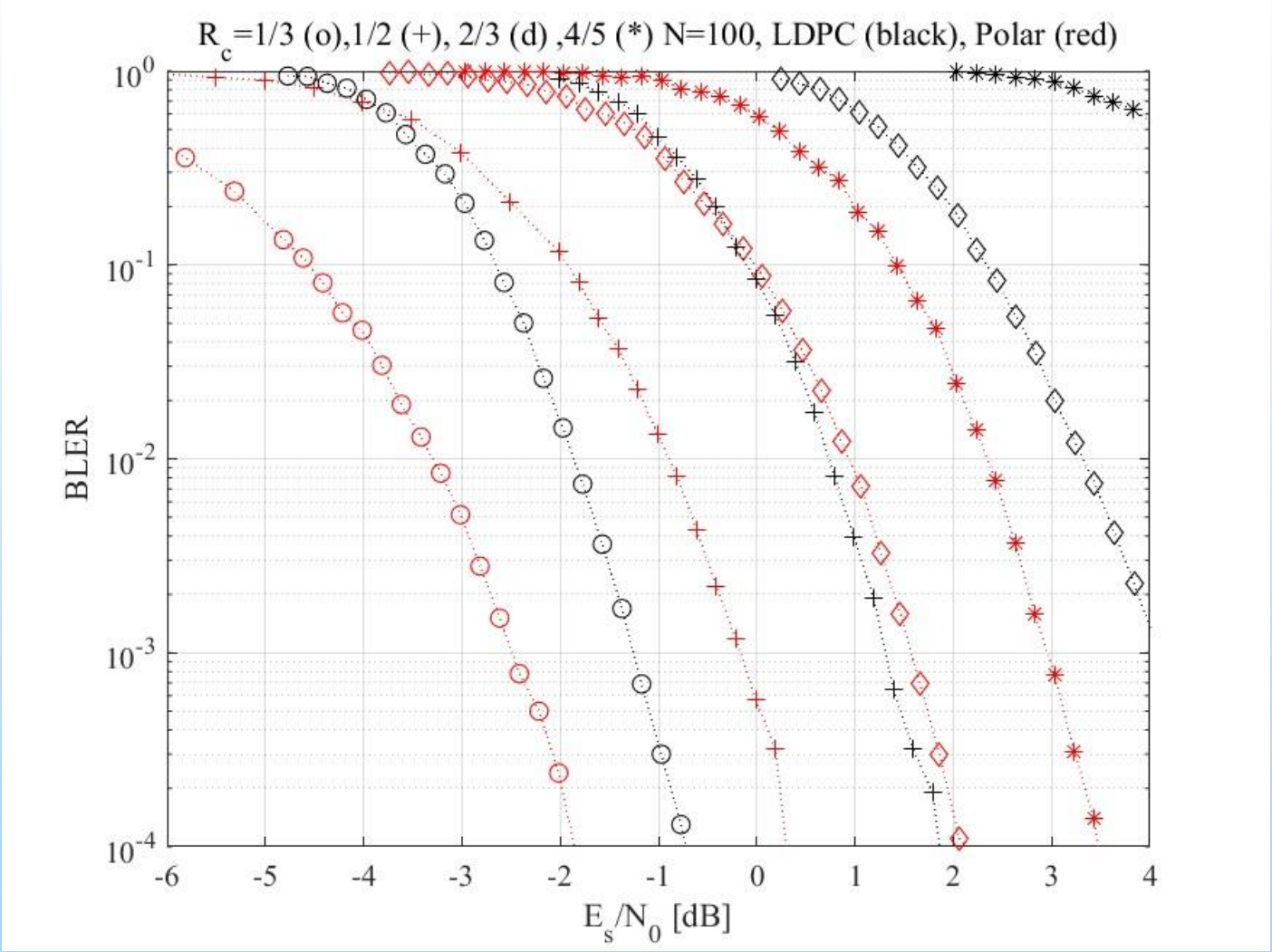
Capacity of individual channels

C_1	C_2	C_3	C_4
			0.99609375
		0.9375	
			0.87890625
	0.75		
			0.80859375
		0.5625	
			0.31640625
0.5			
			0.68359375
		0.4375	
			0.19140625
	0.25		
			0.12109375
		0.0625	
			0.00390625

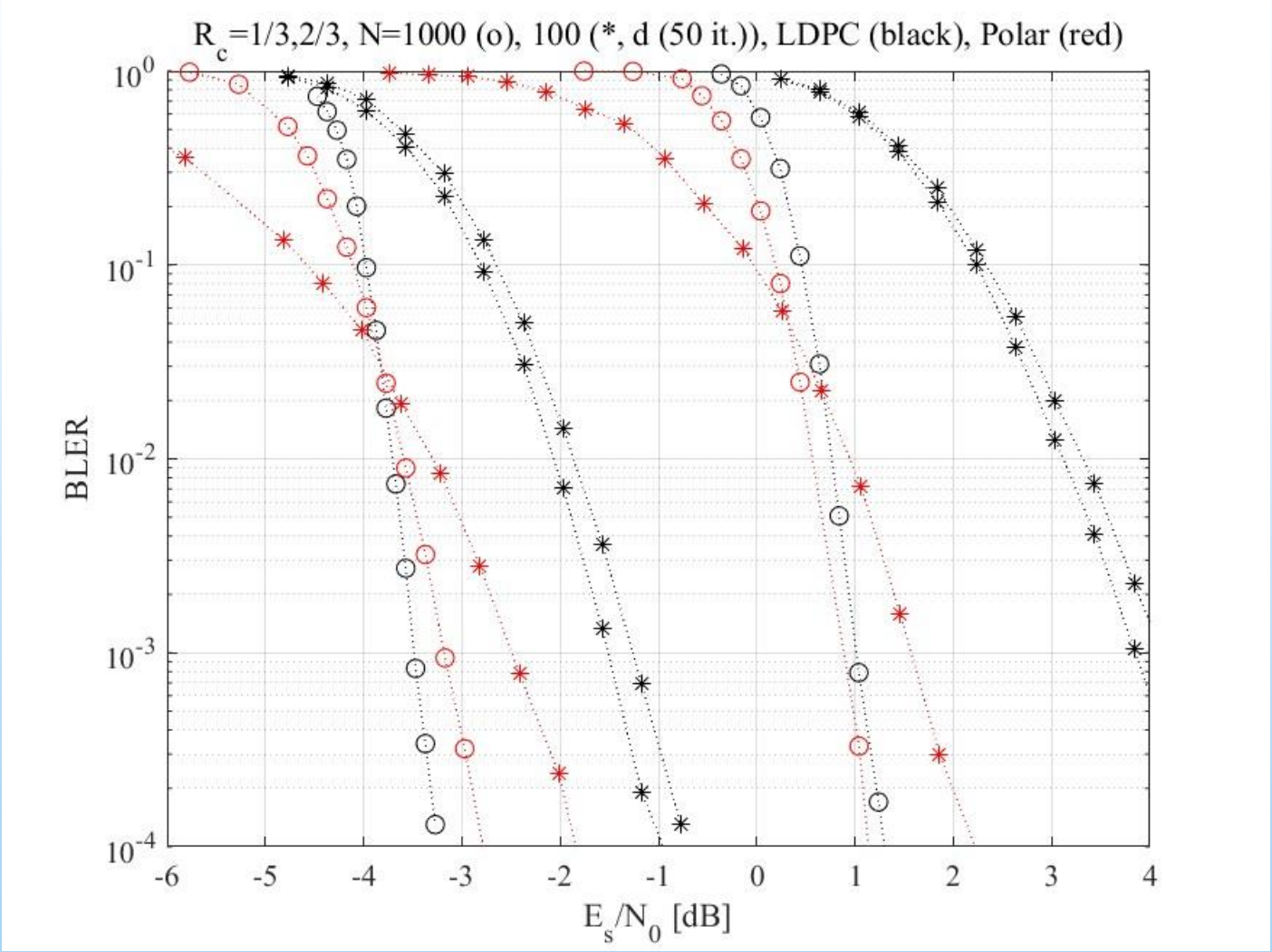
A performance comparison (1)



A performance comparison (2)



A performance comparison



A performance comparison (4)

- For medium length packets LDPC may have a slightly better performance, specially for the high SNR values.
- For short packets, polar codes are significantly better, specially for moderate block error rates.
- Increasing the number of iterations does not allow one to improve the performance of LDPC codes significantly,
- It may be concluded that polar codes are more suitable for IoT applications.

Fountain (rateless) codes

- Fountain codes (also known as rateless erasure codes) are a class of erasure codes with the property that a potentially limitless sequence of encoding symbols can be generated from a given set of source symbols such that the original source symbols can ideally be recovered from any subset of the encoding symbols of size equal to or only slightly larger than the number of source symbols. The term fountain or rateless refers to the fact that these codes do not exhibit a fixed code rate.

- A fountain code is optimal if the original *k* source symbols can be recovered from any *k* successfully received encoding symbols (i.e., excluding those that were erased).

- Raptor codes are the most efficient fountain codes, having linear time encoding and decoding algorithms, and requiring only a small constant number of XOR operations per generated symbol for both encoding and decoding.

Fountain (rateless) codes

- Fountain codes (also known as rateless erasure codes) are a class of erasure codes with the property that a potentially limitless sequence of encoding symbols can be generated from a given set of source symbols such that the original source symbols can ideally be recovered from any subset of the encoding symbols of size equal to or only slightly larger than the number of source symbols. The term fountain or rateless refers to the fact that these codes do not exhibit a fixed code rate.

- A fountain code is optimal if the original *k* source symbols can be recovered from any *k* successfully received encoding symbols (i.e., excluding those that were erased).

- Raptor codes are the most efficient fountain codes, having linear time encoding and decoding algorithms, and requiring only a small constant number of XOR operations per generated symbol for both encoding and decoding.