uc3m | Universidad Carlos III de Madrid

Digital Communications

Grades in English

Chapter 5

Channel coding for error protection

Marcelino Lázaro

Departamento de Teoría de la Señal y Comunicaciones
Universidad Carlos III de Madrid

Index of contents

Introduction

- Errors happen in digital communication systems

$$P_e = P\{\hat{A}[n] \neq A[n]\} \qquad BER = P\{\hat{B}_b[\ell] \neq B_b[\ell]\} \qquad BER \approx \frac{P_e}{m}$$

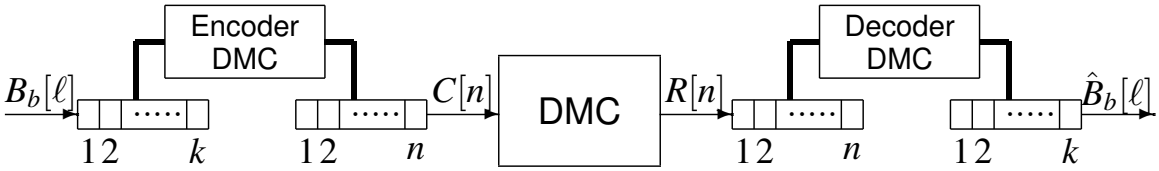
- Performance objective of the system
  - BER $<$ Required performance
- Alternatives to decrease probability of error
  - To increment signal energy (power)
    - ★ Limitations: Economical, physical, legal, interferencies, ...
  - Noisy-channel coding theorem (Shannon)
    - ★ Introduction of redundancy bits
    - ★ Coding rate: R

$$R = \frac{\text{number of information bits}}{\text{Number of transmitted bits (info+redundancy)}}$$

    - ★ Channel capacity: C (bits of information per channel use)
    - ★ BER can be made arbitrarily low as long as

$$R < C$$

Channel coding theorem



DMC: *Discrete Memoryless Channel*

Coding rate: $R = \frac{k}{n}$

Channel coding theorem (Shannon 1948):

$$\text{Channel capacity (DMC): } C = \max_{p_X(x_i)} I(X,Y)$$

$I(X,Y)$: mutual information between the input X and the output Y of the DMC

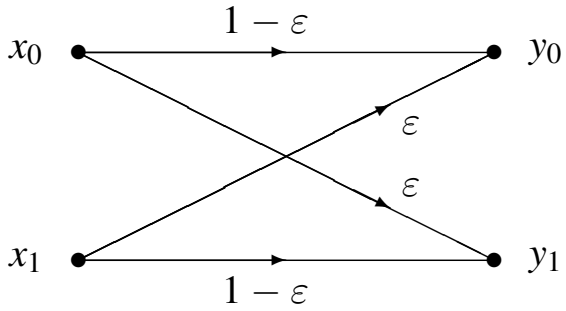
1. If transmission rate R is lower than C, then for any $\delta > 0$ exists a channel code with block length n long enough whose probability of error is lower than δ
2. If $R > C$, probability of error for every code with arbitrary block length is bounded by a non-null value
3. There exist channel codes allowing to achieve channel capacity $R = C$

Channel capacity - Digital channels

- Discrete memoryless channel (DMC) model
  - Input and output: random variables X and Y
  - Transition (conditional) probabilities $p_{Y|X}(y_j|x_i)$
- Channel capacity

$$C = \max_{p_X(x_i)} I(X,Y) \text{ bits/uso}$$

  - Example: Binary symmetric channel with BER=ε



$$C = 1 - H_b(\varepsilon) = 1 - \Omega(\varepsilon) \text{ bits/use}$$

Example - Binary symmetric channel

- Model for a binary digital channel with $BER = \varepsilon$
- Mutual information between input / output of the digital system

$$\begin{aligned} I(X,Y) =& H(Y) - H(Y|X) = H(Y) - \sum_{i=0}^{1} p_X(x_i)\, H(Y|X=x_i) \\ =& H(Y) - \sum_{i=0}^{1} p_X(x_i) \left[- \sum_{j=0}^{1} p_{Y|X}(y_j|x_i)\, \log p_{Y|X}(y_j|x_i) \right] \\ =& H(Y) - \sum_{i=0}^{1} p_X(x_i)\, [-\varepsilon\, \log(\varepsilon) - (1-\varepsilon)\, \log(1-\varepsilon)] \\ =& H(Y) - \sum_{i=0}^{1} p_X(x_i)\, H_b(\varepsilon) = H(Y) - H_b(\varepsilon) \end{aligned}$$

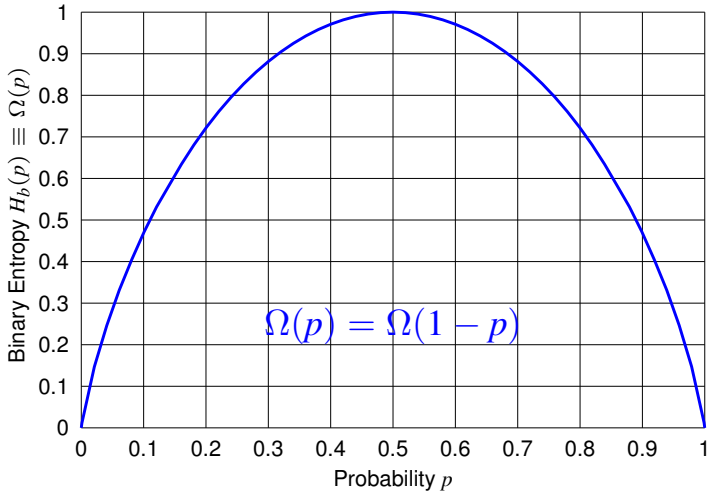
- Channel capacity
  - Search for maximum possible value of mutual information
    - ★ For this channel, is obtained when $H(Y)$ is maximum
    - ★ $H(Y)$ takes maximum value when output symbols are equiprobable
    - ★ For this channel, this happens when input symbols are equiprobable

$$C = 1 - H_b(\varepsilon) \text{ bits/uso} \qquad p_X(x_0) = p_X(x_1) = \frac{1}{2}$$

Binary Entropy: $H_b(p) \equiv \Omega(p)$

- Binary random variable
  - Alphabet: $\{x_0, x_1\}$
  - Probabilities: $\{p_X(x_0) = p, \quad p_X(x_1) = 1 - p\}$

$$H(X) \equiv H_b(p) \equiv \Omega(p) = -p \ \log_2(p) - (1-p) \ \log_2(1-p)$$
$$= p \ \log_2\left(\frac{1}{p}\right) + (1-p) \ \log_2\left(\frac{1}{1-p}\right) \text{ bits/symbol}$$



- Maximum value: $\max \Omega(p) = 1$ bit/symbol
  - It is achieved at $p = 0.5$ (reference value)

Channel capacity for a Gaussian channel

- Input / output relationship in a Gaussian channel

$$Y = X + Z$$

Z is a white and Gaussian random variable, zero mean and variance P_Z

- Channel capacity under the following constraints:
  - Power of the transmitted signal: P_X watt.
  - Available bandwidth: B Hz
    - Noise power: $P_Z = N_0 B$ watt.
- Channel capacity through mutual information

$$C = \max_{f_X(x) \mid E[X^2] \leq P_X} I(X,Y)$$

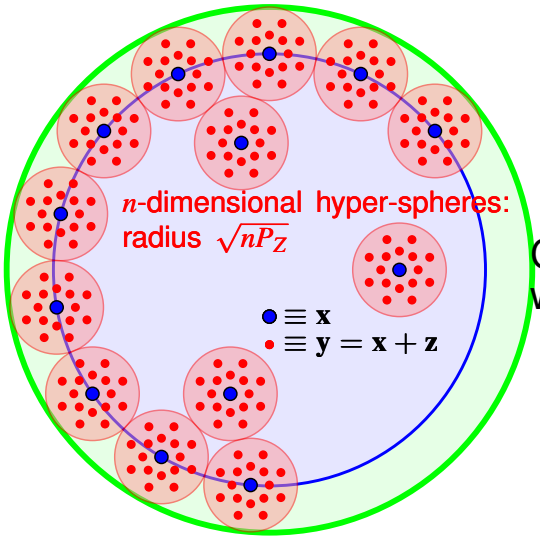
Constraint $E[X^2] \leq P_X$ given by signal power constraint

- Result

$$C = B \ \log_2\left(1 + \frac{P_X}{N_0 B}\right) \text{ bits/s}$$

It is obtained for a Gaussian $f_X(x)$

Channel capacity for a Gaussian channel (II)



n-dimensional hiper-sphere: radius $\sqrt{nP_X}$

n-dim. hyper-sphere: radius $\sqrt{n(P_X + P_Z)}$

Capacity of Gaussian channel under the following conditions:

- Transmitted Power: P_X watt.
- Bandwidth: B Hz
  - Noise Power: $P_Z = N_0 B$ watt.

Capacity: number of vector for n uses without overlapping (by noise)

$$M_{ss} = \left(1 + \frac{P_X}{P_Z}\right)^{n/2}$$

$$C = \frac{\log_2 M_{ss}}{n} = \frac{1}{2}\log_2\left(1 + \frac{P_X}{P_Z}\right)$$

$$C = \frac{1}{2}\log_2\left(1 + \frac{P_X}{N_0 B}\right) \text{ bits/use}$$

$$\boxed{C = B\,\log_2\left(1 + \frac{P_X}{N_0 B}\right) \text{ bits/s}}$$

Trends of channel capacity

$$\boxed{C = B\,\log\left(1 + \frac{P_X}{N_0 B}\right) \text{ bits/s}}$$

- Depends on two design parameters
  - Power of the transmitted signal, P_X
  - Available bandwidth in Hz, B
- Relationship of capacity and P_X

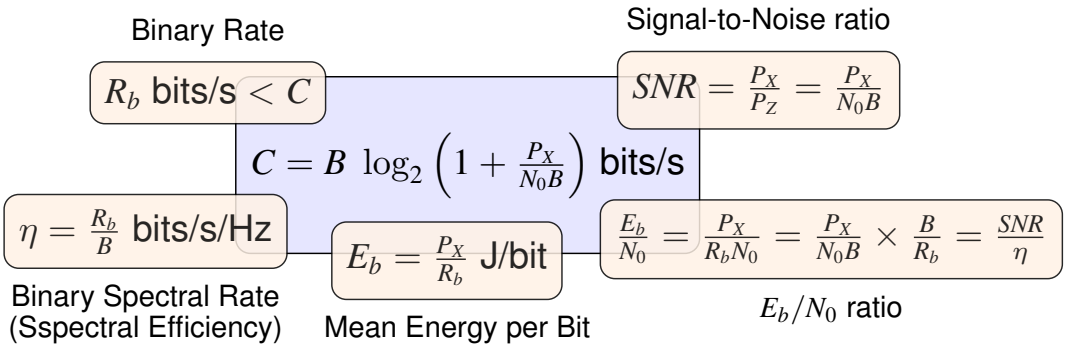
$$\boxed{\lim_{P_X \to \infty} C = \infty}$$

  - Capacity C can be increased arbitrarily by increasing P_X
  - Linear increase of C requires an exponential increase for P_X
- Relationship of capacity and bandwidth (B Hz)

$$\boxed{\lim_{B \to \infty} C = \frac{P}{N_0}\log_2(e) = 1.44\,\frac{P}{N_0}}$$

  - Increase of bandwidth B does not allow an arbitrary increase in C

Bounds for transmission through a Gaussian channel



- Practical communication system

$$R_b < C \rightarrow R_b < B \log_2(1 + SNR) \text{ bits/s}$$

  - Dividing by B in both sides and rearranging terms

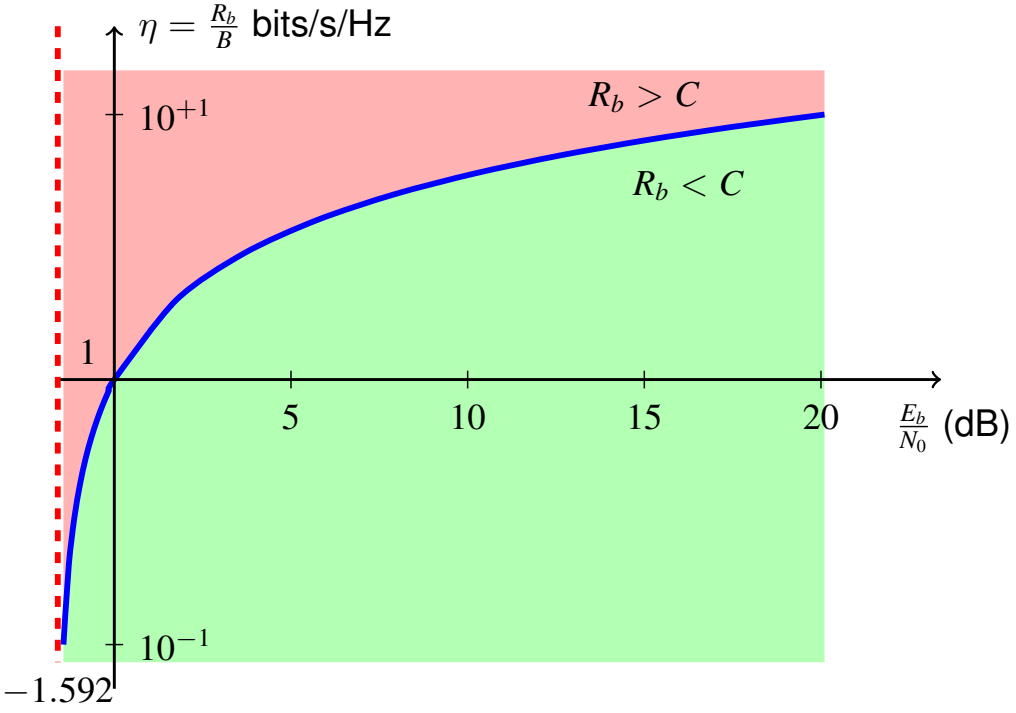
$$\eta < \log_2(1 + SNR), \qquad \eta < \log_2\left(1 + \eta \frac{E_b}{N_0}\right)$$

$$SNR > 2^{\eta} - 1, \qquad \frac{E_b}{N_0} > \frac{2^{\eta} - 1}{\eta}$$

$$\text{If } \eta \rightarrow 0 \text{ then } \frac{E_b}{N_0} = \ln 2 = 0.693 \approx -1.6 \text{ dB}$$

Spectral binary rate as a function of E_b/N_0

- The following curve is plotted in the η vs $\frac{E_b}{N_0}$ axis $\frac{E_b}{N_0} = \frac{2^{\eta}-1}{\eta}$
  - The plane is splitted in two parts: systems with $R_b > C$ (practical) and with $R_b > C$



Normalized signal to noise ratio

- Lower bound for SNR

$$SNR > 2^{\eta} - 1$$

- Definition of normalized SNR

$$SNR_{norm} = \frac{SNR}{2^{\eta} - 1}$$

- Lower bound for SNR_{norm}

$$SNR_{norm} > 1 \ (0 \text{ dB})$$

Classification of channel codes

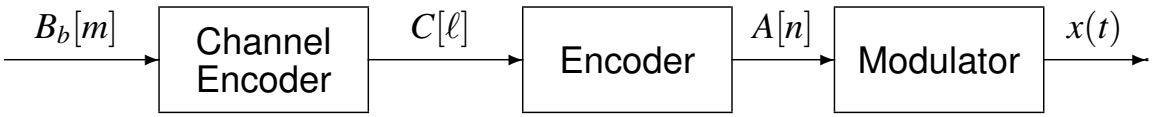
- Mechanism used to introduce redundancy
  - Block codes
    - ★ Independent coding of data blocks of fixed size (k bits)
    - ★ Dictionary: k uncoded bits / n coded bits (codewords)
    - ★ Key concept: distance between coded words
    - ★ Example: repetition code of order $n - 1$

Uncoded bits ($k = 1$)	Codewords (n)
1	11...1
0	00...0

  - Convolutional codes
    - ★ Continuous coding by using a digital filter bank
- Capability of code for detection and/or correction of errors
  - Codes with detection capability
  - Codes with correction capability
- Statistic used for detection
  - Hard output: decoding is performed from detected bits $\hat{C}[\ell]$
  - Soft output: decoding is performed from demodulator output $q[n]$
    - ★ Better performance with higher complexity
  - Bit erasure: bits or symbols with high error probability are labeled instead of decided (“erased”)

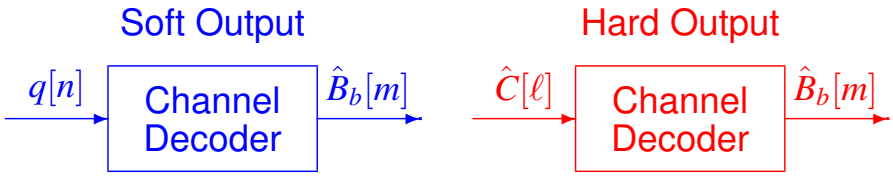
Soft output / Hard output

- Block diagram for transmitter



- Block diagram for receiver

- Soft output / Hard output channel decoder



$B_b[m]$: information (uncoded) bits

$C[\ell]$: encoded bits

Hard output / soft output

- Example: repetition code of order 2, 2-PAM modulation
  - Binary assignment: $0 \equiv -1$ / $1 \equiv +1$
  - Soft output: $\boldsymbol{q} = [-0.01,\ +1.2,\ -0.05]$
  - Hard output: $\hat{\mathbf{c}} = [0,\ 1,\ 0]$
- Decoding
  - Hard decoding: majority rule $\hat{B} = 0$
  - Soft decoding: comparison of $\boldsymbol{q}$ with:

$$\boldsymbol{q}_0 = [-1,-1,-1] \quad \text{and} \quad \boldsymbol{q}_1 = [+1,+1,+1]$$

$$d(\boldsymbol{q},\boldsymbol{q}_0) = \sqrt{(-0.01-(-1))^2+(+1.2-(-1))^2+(-0.05-(-1))^2} = 2.59$$

$$d(\boldsymbol{q},\boldsymbol{q}_1) = \sqrt{(-0.01-(+1))^2+(+1.2-(+1))^2+(-0.05-(+1))^2} = 1.47$$

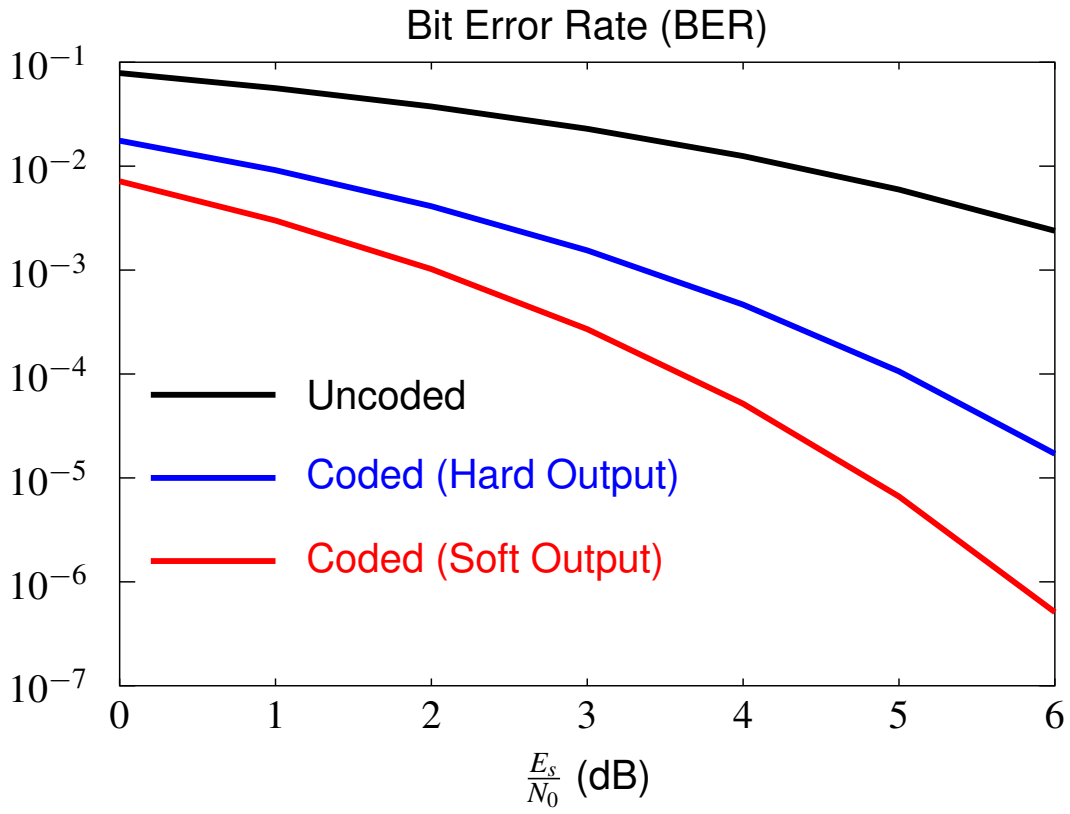
- Probability of error for BER=ε using the 2-PAM
  - Hard output

$$P_e^{Hard} = 3\,\varepsilon^2(1-\varepsilon)+\varepsilon^3, \qquad \varepsilon = Q\left(\frac{1}{\sqrt{N_0/2}}\right) = Q\left(\sqrt{2\frac{E_s}{N_0}}\right)$$

  - Soft output

$$P_e^{Soft} = Q\left(\frac{d(\boldsymbol{q}_0,\boldsymbol{q}_1)}{2\sqrt{N_0/2}}\right) = Q\left(\frac{\sqrt{3}}{\sqrt{N_0/2}}\right) = Q\left(\sqrt{\frac{6E_s}{N_0}}\right)$$

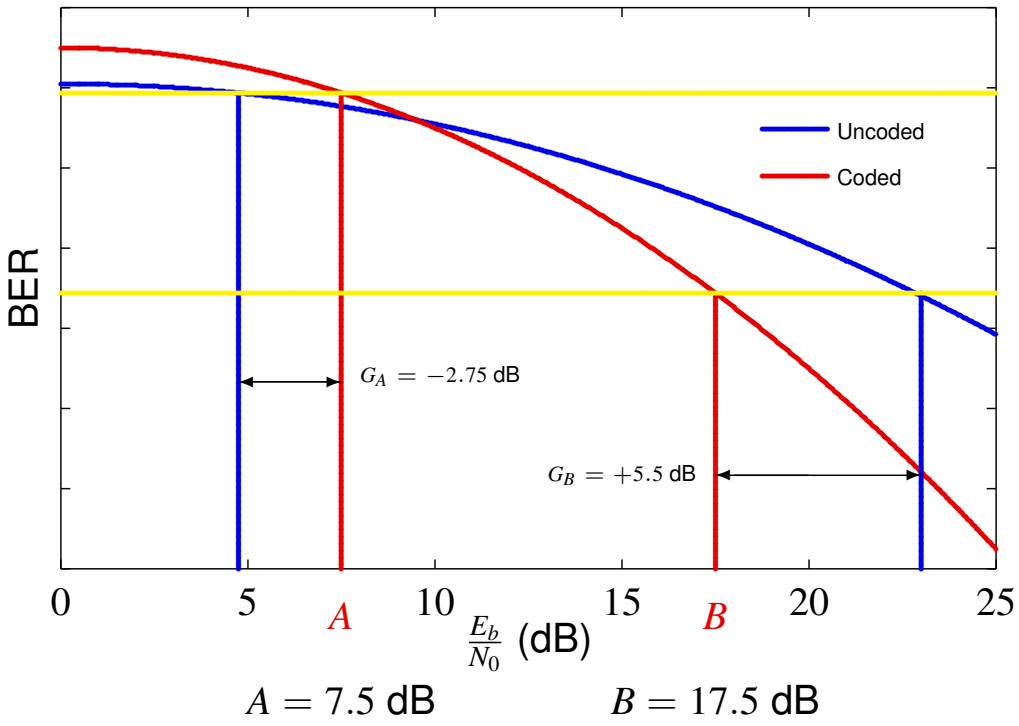
Hard output / Soft output



Coding gain

- Definition: for a code and a specific E_b/N_0 ratio (or equivalently a specific BER), difference in decibels over E_b/N_0 which is necessary to achieve the same BER without channel coding
  - E_b: Mean energy per information bit
- Coding gain allow to compare the performance of different channel coding techniques
- Coding gain depends on the objective BER (or on the E_b/N_0 work point)
  - Typically, it is referenced to a given value of E_b/N_0
- Gain can be negative for some range of E_b/N_0 ratio

Coding gain



Block codes - Definitions

- Coding is performed independently for blocks of k bits
  - Conversion in coded blocks of de n bits $\rightarrow$ Coding rate $R = k/n$
- Notation used to represent blocks of bits
  - Information bits (uncoded):

$$\mathbf{b}_i = [b_i[0], b_i[1], \cdots, b_i[k-1]],\ i = 0, 1, \cdots, 2^k - 1$$

  - Coded bits:

$$\mathbf{c}_i = [c_i[0], c_i[1], \cdots, c_i[n-1]],\ i = 0, 1, \cdots, 2^k - 1$$

  - Dictionary of the code: message word $\rightarrow$ coded word

$$\mathbf{b}_i \rightarrow \mathbf{c}_i$$

- Weight of a word $w(\mathbf{c}_i)$
  - Number of ones in the word
- Hamming distance for two coded words $d^H(\mathbf{c}_i, \mathbf{c}_j)$
  - Number of bits which are different
- Minimum distance of the code: $d^H_{min} \equiv d_{min}$
  - Minimum Hamming distance for two different words in the code

Block codes - Example

- Dictionary of the code: $k = 2$ $n = 6$

i	$\mathbf{b}_i$	$\mathbf{c}_i$
0	00	001110
1	01	010011
2	10	100100
3	11	111001

- Weight of the words

$$w(\mathbf{c}_0) = 3, \quad w(\mathbf{c}_1) = 3, \quad w(\mathbf{c}_2) = 2, \quad w(\mathbf{c}_3) = 4,$$

- Hamming distances between two words

$$\begin{aligned} d(\mathbf{c}_0, \mathbf{c}_1) &= 4, \quad d(\mathbf{c}_1, \mathbf{c}_2) = 5 \\ d(\mathbf{c}_0, \mathbf{c}_2) &= 3, \quad d(\mathbf{c}_1, \mathbf{c}_3) = 3 \\ d(\mathbf{c}_0, \mathbf{c}_3) &= 5, \quad d(\mathbf{c}_2, \mathbf{c}_3) = 4 \end{aligned}$$

- Minimum distance of the code

$$d_{min} = 3$$

Optimum decoding working with hard output

- The available observation conditioned to the transmission of $\mathbf{c}_i$ is

$$\mathbf{r} = \mathbf{c}_i + \mathbf{e}, \ \mathbf{e} = [e[0], e[1], \cdots, e[n-1]]$$

- Probabilistic model for the error pattern if $BER = \varepsilon$

$$p_E(e[j]) = \varepsilon^{e[j]} \ (1-\varepsilon)^{1-e[j]} = \begin{cases} \varepsilon, & e[j] = 1 \\ 1-\varepsilon, & e[j] = 0 \end{cases}$$

- Likelihood (coditional probability of observation)
  - Error: $e[j] = r[j] - c[j]$
  - Likelihood for each bit of the observation $r[j]$

$$p_{R[j]|C[j]}(r[j]|c[j]) = \varepsilon^{e[j]} \ (1-\varepsilon)^{1-e[j]} = \varepsilon^{r[j]-c_i[j]} \ (1-\varepsilon)^{1-(r[j]-c_i[j])}$$

  - Likelihood of a coded word for the observation $\mathbf{r}$

$$p_{\mathbf{R}|\mathbf{C}}(\mathbf{r}|\mathbf{c}_i) = \prod_{j=0}^{n-1} \varepsilon^{r[j]-c_i[j]} \ (1-\varepsilon)^{1-(r[j]-c_i[j])}$$

- Maximum likelihood (ML) estimation - Minimizing Hamming distance

$$\hat{\mathbf{c}} = \mathbf{c}_i = \arg \min_{\mathbf{c}_i} d^H(\mathbf{r}, \mathbf{c}_i)$$

Detection and correction capabilities with hard output

- Performance depends on Hamming distances
  - An error can not be detected if transmission errors transform a coded word in another coded word
  - An error happens when the number of erroneous in the transmission of the coded word makes the observation to be at lower Hamming distance of another coded word
- Performance is limited for minimum distance d_{min}
  - Capability of detecting (in n bits)

$$d = d_{min} - 1 \text{ errors}$$

  - Capability of correcting (in n bits)

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor \text{ errors}$$

Optimum decoding with soft output

- Depends on the constellation and binary assignment
  - M symbols: $m = \log_2(M)$ bits/symbol
- Sequence of symbols for a coded word

$$\mathbf{c}_i \rightarrow \boldsymbol{A}_i = [A_i[0], A_i[1], \cdots, A_i[n' - 1]],\ n' = \frac{n}{m}$$

- Probabilistic model for observation when $\mathbf{c}_i$ is transmitted

$$\boldsymbol{q} = \boldsymbol{A}_i + \mathbf{e},\ \mathbf{e} = [e[0], e[1], \cdots, e[n' - 1]]$$

- Probabilistic model for error pattern

$$f_E(e[j]) = \mathcal{N}(0, \sigma_z^2)$$

- Likelihood (conditional probability of observation)

$$f_{\boldsymbol{q}|\boldsymbol{A}}(\boldsymbol{q}|\boldsymbol{A}_i) = \mathcal{N}(\boldsymbol{A}_i, \sigma_z^2)$$

- Maximum likelihood estimation - Minimizing euclidean distance

$$\hat{\mathbf{c}} = \mathbf{c}_i, \quad i = \arg\min_i d^E(\boldsymbol{q}, \boldsymbol{A}_i)$$

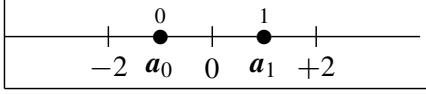
Example of block code and symbols assigment

- Dictionnary of the code: $k = 2$ $n = 6$

i	$\mathbf{b}_i$	$\mathbf{c}_i$
0	00	001110
1	01	010011
2	10	100100
3	11	111001

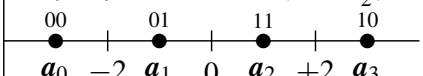
- Symbols assignment depends on the constellation and binary assignment

Ejemplo A: 2-PAM ($n' = n$)



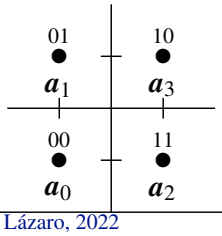
$$A_0 = [a_0, a_0, a_1, a_1, a_1, a_0] = [-1, -1, +1, +1, +1, -1]$$
$$A_1 = [a_0, a_1, a_0, a_0, a_1, a_1] = [-1, +1, -1, -1, +1, +1]$$
$$A_2 = [a_1, a_0, a_0, a_1, a_0, a_0] = [+1, -1, -1, +1, -1, -1]$$
$$A_3 = [a_1, a_1, a_1, a_0, a_0, a_1] = [+1, +1, +1, -1, -1, +1]$$

Ejemplo B: 4-PAM ($n' = \frac{n}{2}$)



$$A_0 = [a_0, a_2, a_3] = [-3, +1, +3]$$
$$A_1 = [a_1, a_0, a_2] = [-1, -3, +1]$$
$$A_2 = [a_3, a_1, a_0] = [+3, -1, -3]$$
$$A_3 = [a_2, a_3, a_1] = [+1, +3, -1]$$

Ejemplo C: 4-QAM ($n' = \frac{n}{2}$)



$$A_0 = [a_0, a_2, a_3] = \left[\begin{bmatrix}-1\\-1\end{bmatrix}, \begin{bmatrix}+1\\-1\end{bmatrix}, \begin{bmatrix}+1\\+1\end{bmatrix}\right]$$
$$A_1 = [a_1, a_0, a_2] = \left[\begin{bmatrix}-1\\+1\end{bmatrix}, \begin{bmatrix}-1\\-1\end{bmatrix}, \begin{bmatrix}+1\\-1\end{bmatrix}\right]$$
$$A_2 = [a_3, a_1, a_0] = \left[\begin{bmatrix}+1\\+1\end{bmatrix}, \begin{bmatrix}-1\\+1\end{bmatrix}, \begin{bmatrix}-1\\-1\end{bmatrix}\right]$$
$$A_3 = [a_2, a_3, a_1] = \left[\begin{bmatrix}+1\\-1\end{bmatrix}, \begin{bmatrix}+1\\+1\end{bmatrix}, \begin{bmatrix}-1\\+1\end{bmatrix}\right]$$

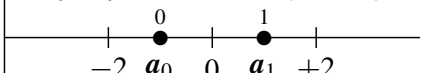
Example of decoding with soft output

- Dictionnary of the code: $k = 2$ $n = 6$

i	$\mathbf{b}_i$	$\mathbf{c}_i$
0	00	001110
1	01	010011
2	10	100100
3	11	111001

- Symbols assignment

Ejemplo A: 2-PAM ($n' = n$)



$$A_0 = [a_0, a_0, a_1, a_1, a_1, a_0] = [-1, -1, +1, +1, +1, -1]$$
$$A_1 = [a_0, a_1, a_0, a_0, a_1, a_1] = [-1, +1, -1, -1, +1, +1]$$
$$A_2 = [a_1, a_0, a_0, a_1, a_0, a_0] = [+1, -1, -1, +1, -1, -1]$$
$$A_3 = [a_1, a_1, a_1, a_0, a_0, a_1] = [+1, +1, +1, -1, -1, +1]$$

n	0	1	2	3	4	5
$q[n]$	+0.8	−1.2	−0.7	+0.2	−0.5	−1.1

$$d(\boldsymbol{q}, \boldsymbol{A}_0) = \sqrt{(+0.8-(-1))^2 + (-1.2-(-1))^2 + (-0.7-(+1))^2 + (+0.2-(+1))^2 + (-0.5-(+1))^2 + (-1.1-(-1))^2} = \sqrt{9.07}$$
$$d(\boldsymbol{q}, \boldsymbol{A}_1) = \sqrt{(+0.8-(-1))^2 + (-1.2-(+1))^2 + (-0.7-(-1))^2 + (+0.2-(-1))^2 + (-0.5-(+1))^2 + (-1.1-(+1))^2} = \sqrt{16.27}$$
$$d(\boldsymbol{q}, \boldsymbol{A}_2) = \sqrt{(+0.8-(+1))^2 + (-1.2-(-1))^2 + (-0.7-(-1))^2 + (+0.2-(+1))^2 + (-0.5-(-1))^2 + (-1.1-(-1))^2} = \sqrt{1.07}$$
$$d(\boldsymbol{q}, \boldsymbol{A}_3) = \sqrt{(+0.8-(+1))^2 + (-1.2-(+1))^2 + (-0.7-(+1))^2 + (+0.2-(-1))^2 + (-0.5-(-1))^2 + (-1.1-(+1))^2} = \sqrt{13.87}$$

Decisión: $\hat{\mathbf{c}} = \mathbf{c}_2 = 100100$

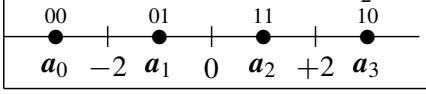
Example of decoding with soft output (II)

- Dictionnary of the code: $k = 2$ $n = 6$

i	$\mathbf{b}_i$	$\mathbf{c}_i$
0	00	001110
1	01	010011
2	10	100100
3	11	111001

- Symbols assignment

Ejemplo B: 4-PAM ($n' = \frac{n}{2}$)



$$A_0 = [a_0, a_2, a_3] = [-3, +1, +3]$$
$$A_1 = [a_1, a_0, a_2] = [-1, -3, +1]$$
$$A_2 = [a_3, a_1, a_0] = [+3, -1, -3]$$
$$A_3 = [a_2, a_3, a_1] = [+1, +3, -1]$$

n	0	1	2
$q[n]$	$+0.8$	$+2.2$	-0.7

$$d(q, A_0) = \sqrt{(+0.8 - (-3))^2 + (+2.2 - (+1))^2 + (-0.7 - (+3))^2} = \sqrt{29.57}$$
$$d(q, A_1) = \sqrt{(+0.8 - (-1))^2 + (+2.2 - (-3))^2 + (-0.7 - (+1))^2} = \sqrt{33.17}$$
$$d(q, A_2) = \sqrt{(+0.8 - (+3))^2 + (+2.2 - (-1))^2 + (-0.7 - (-3))^2} = \sqrt{20.37}$$
$$d(q, A_3) = \sqrt{(+0.8 - (+1))^2 + (+2.2 - (+3))^2 + (-0.7 - (-1))^2} = \sqrt{0.77}$$

Decisión: $\hat{\mathbf{c}} = \mathbf{c}_3 = 111001$

Linear block codes

- Channel code $C(k, n)$
- Basis for the code: k linearly independent coded words

$$\{\mathbf{g}_0, \mathbf{g}_1, \cdots, \mathbf{g}_{k-1}\}$$

$$\mathbf{g}_i = [g_i[0], g_i[1], \cdots, g_i[n-1]]$$

- Coding methodology: linear combination of k basis

$$\mathbf{c}_i = b_i[0]\ \mathbf{g}_0 + b_i[1]\ \mathbf{g}_1 + \cdots + b_i[k-1]\ \mathbf{g}_{k-1}$$

  Coefficients for the expansion: information bits
- Properties
  - All elements in the basis are coded words (belong to the code)

$$\mathbf{c}_i = \mathbf{g}_\ell \rightarrow w(\mathbf{b}_i) = 1, \quad b_i[\ell] = 1$$

  - $\mathbf{c}_0 = \mathbf{0} = [0, 0, \cdots, 0]$ word belongs to the code
    - ★ Associated to $\mathbf{b}_0 = \mathbf{0} = [0, 0, \cdots, 0]$
  - A linear combination of coded words belongs to the code
  - All coded words have at least another coded word at minimum distance d_{min}
  - Therefore, as $\mathbf{c}_0$ has at least a coded word at d_{min}

$$d_{min} = \min_{\mathbf{c}_i \neq \mathbf{c}_0} w(\mathbf{c}_i)$$

Generator matrix

- The basis of the code can be packed in a matrix of size $k \times n$

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_{k-1} \end{bmatrix} = \begin{bmatrix} g_0[0] & g_0[1] & \cdots & g_0[n-1] \\ g_1[0] & g_1[1] & \cdots & g_1[n-1] \\ \vdots & \vdots & \ddots & \vdots \\ g_{k-1}[0] & g_{k-1}[1] & \cdots & g_{k-1}[n-1] \end{bmatrix}$$

- Coded words can be obtained from this matrix

$$\mathbf{c}_i = \mathbf{b}_i \, \mathbf{G}$$

- Systematic: message $\mathbf{b}_i$ is part of the coded word $\mathbf{c}_i$

$$\mathbf{c}_i = [\mathbf{b}_i \mid \mathbf{p}_i] \rightarrow \mathbf{G} = [\mathbf{I}_k \mid \mathbf{P}]$$

$$\mathbf{c}_i = [\mathbf{p}_i \mid \mathbf{b}_i] \rightarrow \mathbf{G} = [\mathbf{P} \mid \mathbf{I}_k]$$

Generator matrix - An example

- Code $C(2,5)$

$$\mathbf{G} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

- Encoded Words

$\mathbf{b}_i$	$\mathbf{c}_i$
0 0	0 0 0 0 0
0 1	1 0 1 0 1
1 0	0 1 1 1 0
1 1	1 1 0 1 1

- Systematic code
- Minimum distance of the code: $d_{min} = 3$
  - Detects 2 errors (in $n = 5$ bits)
  - Corrects 1 error (in $n = 5$ bits)

Parity check matrix

- Matrix of size $(n-k) \times n$: orthogonal complement for $\mathbf{G}$

$$\mathbf{G}\,\mathbf{H}^T = \mathbf{0} \text{ matrix of } k \times (n-k) \text{ zeros}$$

- Systematic codes

$$\mathbf{G} = [\mathbf{I}_k \mid \mathbf{P}] \;\rightarrow\; \mathbf{H} = [\mathbf{P}^T \mid \mathbf{I}_{n-k}]$$

$$\mathbf{G} = [\mathbf{P} \mid \mathbf{I}_k] \;\rightarrow\; \mathbf{H} = [\mathbf{I}_{n-k} \mid \mathbf{P}^T]$$

- Identification of coded words

$$\mathbf{c}_i\,\mathbf{H}^T = \mathbf{b}_i\,\mathbf{G}\,\mathbf{H}^T = \mathbf{0} \text{ vector of } n-k \text{ zeros}$$

- Syndrome-based decoding
  - Transmission model: $\mathbf{r} = \mathbf{c}_i + \mathbf{e}$ ($\mathbf{e}$: error pattern)
  - Syndrome depends only on the error pattern

$$\mathbf{s} = \mathbf{r}\,\mathbf{H}^T = (\mathbf{c}_i + \mathbf{e})\,\mathbf{H}^T = \mathbf{e}\,\mathbf{H}^T$$

  - Decoding can be made from a syndrome table

Parity check matrix and syndrome table - An example

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

- Syndrome table: $\mathbf{s} = \mathbf{e}\,\mathbf{H}^T$

$\mathbf{e}$	$\mathbf{s}$
0 0 0 0 0	0 0 0
1 0 0 0 0	1 0 0
0 1 0 0 0	0 1 0
0 0 1 0 0	0 0 1
0 0 0 1 0	0 1 1
0 0 0 0 1	1 0 1
¿?	1 1 0
¿?	1 1 1

$\mathbf{s} = 110 \rightarrow \mathbf{e}_1 = 11000,\ \mathbf{e}_2 = 00011,\ \mathbf{e}_3 = 10110,\ \mathbf{e}_4 = 01101$

$\mathbf{s} = 111 \rightarrow \mathbf{e}_1 = 10010,\ \mathbf{e}_2 = 01001,\ \mathbf{e}_3 = 11100,\ \mathbf{e}_4 = 00111$

Solution: to choose one of the patterns with $t+1 = 2$ errors ($\mathbf{e}_1$ or $\mathbf{e}_2$)

- Having two erroneous bits has a higher probability than having three or more errors

Syndrome-based decoding

- Alternative process for decoding equivalent to maximun likelihood for hard-decoding
  - Steps to follow:
    1. Calculation of the syndrome
    $$\mathbf{s} = \mathbf{r}\,\mathbf{H}^T$$
    2. Identification of the error pattern (syndrome table)
    $$\mathbf{s} \rightarrow \mathbf{e}$$
    3. Error correction (decision of the codeword)
    $$\hat{\mathbf{c}} = \mathbf{r} + \mathbf{e} = \mathbf{c}_i$$
    4. Decoding
    $$\hat{\mathbf{c}} = \mathbf{c}_i \rightarrow \hat{\mathbf{b}} = \mathbf{b}_i$$

    Simpler in systematic codes

Advantage of working with $\mathbf{G}$ and $\mathbf{H}$ and with systematic codes

- Implementation of encoder and decoder
  - Number of coded words can be high: 2^k
    - ★ $k = 2, n = 5$
      - 4 coded words
    - ★ $k = 247, n = 255$ (Hamming Code)
      - $2^{247} \approx 2.26\ 10^{74}$ coded words
  - Number of symdromes: 2^{n-k}
    - ★ $k = 2, n = 5$ $(t = 1)$
      - 8 syndromes
    - ★ $k = 247, n = 255$ $(t = 1)$
      - 256 syndromes
- Alternative: structured codes
  - Implementation using shift registers
    - ★ Definition using generator polynomials

Gaussian elimination method - Example

- It can be used to transform a non-systematic generator matrix in a systematic-one
- Rows are replaced by linear combinations of rows
  - 1^{st} row: $1^{st} + 2^{nd}$ rows
  - 2^{nd} row: 1^{st} row
- Code $C(2,5)$

$$\mathbf{G} = \left[\begin{array}{ccc|cc} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \end{array}\right] \rightarrow \mathbf{G}' = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

- Coded words

$\mathbf{b}_i$	$\mathbf{c}_i$
0 0	0 0 0 0 0
0 1	1 0 1 0 1
1 0	0 1 1 1 0
1 1	1 1 0 1 1

$\rightarrow$

$\mathbf{b}'_i$	$\mathbf{c}'_i$
0 0	0 0 0 0 0
0 1	0 1 1 1 0
1 0	1 1 0 1 1
1 1	1 0 1 0 1

- Same coded words / different assignation to information bits
  - The same $\mathbf{H}$ can be used to generate syndrome table

Hamming bound

- Number of symdromes with redundancy $r = n - k$ bits is

$$2^{n-k} = 2^r$$

- Hamming bound: to correct t errors, the minimum necessary redundancy is

$$r \geq \log_2 V(n,t), \;\; V(n,t) = \sum_{j=0}^{t} \binom{n}{j}$$

  - $V(n,t)$: Hamming sphere with radius t
- Interpretation: the number of available symdromes is

$$\binom{0}{0} + \binom{n}{1} + \cdots + \binom{n}{t} \leq 2^r$$

  - Equality: Perfect codes

Perfect codes

- Repetition codes (majority decission)
  - Odd n, $k = 1$, $t = \frac{n-1}{2}$ (because $d_{min} = n$)
- Hamming codes
  - For $m \geq 3$, $n = 2^m - 1$, $k = 2^m - m - 1$, $t = 1$ $(d_{min} = 3)$
    - Redundancy: $n - k = m$
  - Parity check matrix: the n columns contain all the possible binary combinations of $(n-k)$ bits, except all zeros
    - Example: Hamming code (4,7)

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

- Golay code
  - $n = 23$, $k = 12$, $t = 3$ $(d_{min} = 7)$

Golay code

- Systematic generating matrix

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

- Extended Golay code
  - A parity bit is added: $d_{min} = 8$, (detects $d = 7$ errors)
    - It is NOT a perfect code
  - Code $k = 12$, $n = 24$, rate $R = \frac{1}{2}$

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

Performance - Hard decoding

- Assumption: bit error rate (BER) in transmission: ε
  - Errors happen when the correction capability is exceeded
- Perfect code correcting up to t errors in n bits

$$P_e = \sum_{e=t+1}^{n} \binom{n}{e} \varepsilon^e \, (1-\varepsilon)^{n-e}$$

- Non perfect code correcting
  - All patterns of up to t errors
  - Moreover, a patterns of $t+1$ errors

$$P_e = \left[\binom{n}{t+1} - a \right] \varepsilon^{t+1} \, (1-\varepsilon)^{n-t-1} + \sum_{e=t+2}^{n} \binom{n}{e} \varepsilon^e \, (1-\varepsilon)^{n-e}$$

- With Gray or pseudo-Gray bit assignement and high SNR

$$BER \approx \frac{1}{k} P_e$$

Performance - Soft decoding

- Probability of error

$$P_e \approx c \, Q\left(\frac{d^E_{min}}{2\sqrt{N_0/2}} \right)$$

  - d^E_{min}: minimum euclidean distance of symbol sequences corresponding to two different coded words

$$d^E_{min} = \min_{j \neq i} d^E(\boldsymbol{A}_i, \boldsymbol{A}_j)$$

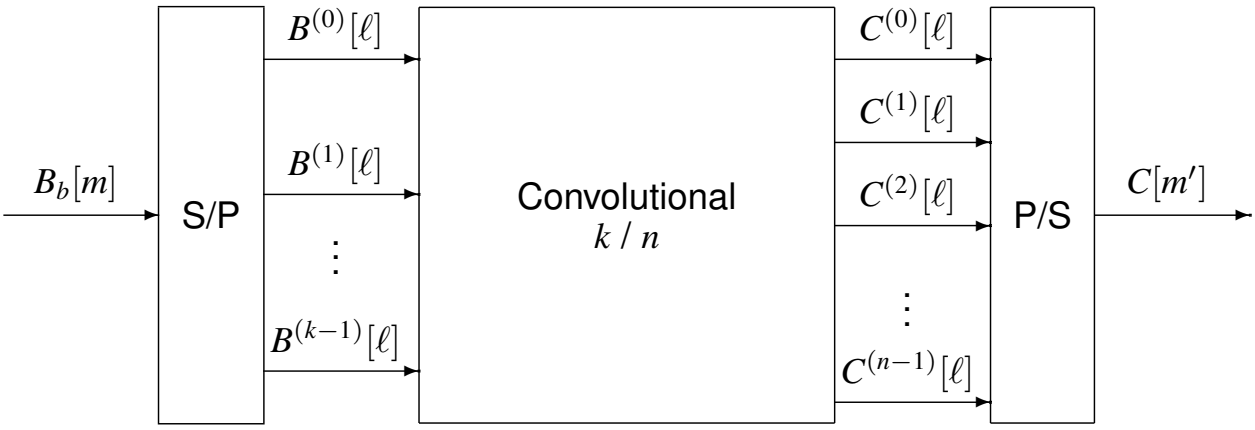
  - c: maximum number of coded words whose symbol sequence are at minimum euclidean distance of the symbol sequence for a given coded word
- In general, d^E_{min} depends on: constellation + binary assignment
  - For binary modulations (constellations of 2 symbols $\boldsymbol{a}_0$ and $\boldsymbol{a}_1$)

$$d^E(\mathbf{c}_i, \mathbf{c}_j) = d(\boldsymbol{a}_0, \boldsymbol{a}_1) \sqrt{d^H(\mathbf{c}_i, \mathbf{c}_j)}$$

$$P_e \approx c \, Q\left(\frac{d^E(\boldsymbol{a}_0, \boldsymbol{a}_1)}{2\sqrt{N_0/2}} \sqrt{d^H_{min}} \right)$$

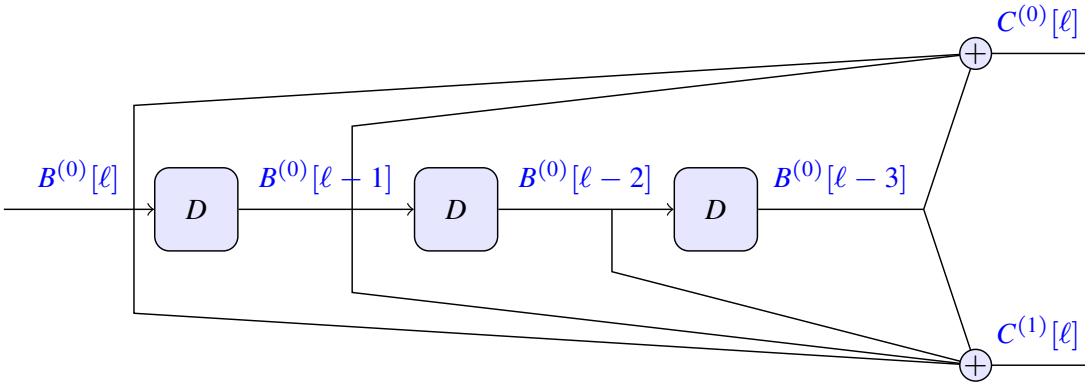
Convolutional codes

- Serial / parallel (S/P) converter
- Convolutional code
- Parallel / serial (P/S) converter



Convolutional codes

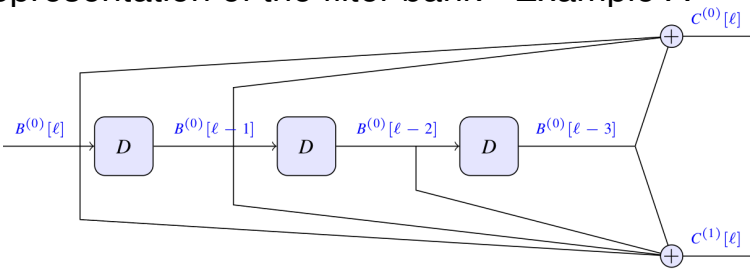
- Redundancy is introduced by digital filtering
  - Introduction of memory in the transmission
- Rate $R = k/n$: filter bank with
  - k inputs
  - n outputs



- Notation:
  - Sequences for inputs: $B^{(i)}[\ell]$, with $i = 0, 1, \cdots, k-1$
  - Sequences for outpus: $C^{(j)}[\ell]$, with $j = 0, 1, \cdots, n-1$

Representation of convolutional codes

- Schematic representation of the filter bank - Example A



- Analytic relationship of all outputs with respect to inputs

$$C^{(0)}[\ell] = B^{(0)}[\ell] + B^{(0)}[\ell-1] + B^{(0)}[\ell-3]$$

$$C^{(1)}[\ell] = B^{(0)}[\ell] + B^{(0)}[\ell-1] + B^{(0)}[\ell-2] + B^{(0)}[\ell-3]$$

- D polinomial representation of sequences - D Transform

$$B^{(i)}(D) = \sum_{\ell} B^{(i)}[\ell]\, D^{\ell}$$

  - Property of D representation with respecto to delays

$$B^{(i)}[\ell-d] \leftrightarrow B^{(i)}(D)\, D^{d}$$

Representations of convolutional codes (II)

- Notation using D polinomial representation

$$C^{(0)}(D) = B^{(0)}(D)\left\{1 + D + D^3\right\}$$

$$C^{(1)}(D) = B^{(0)}(D)\left\{1 + D + D^2 + D^3\right\}$$

- Matrix notation (polinomial):

$$\mathbf{C}(D)_{1\times n} = \mathbf{B}(D)_{1\times k}\, \mathbf{G}(D)_{k\times n}$$

- Generator matrix of size $k \times n$
  - Each element is a D polynomial
  - Element in row i column j: contribution into j-th output coming from i-th input
  - Examples
    - ★ Previous example (A): $k = 1$, $n = 2$

$$\mathbf{G}(D) = \left[1 + D + D^3,\ 1 + D + D^2 + D^3\right]_{1\times 2}$$

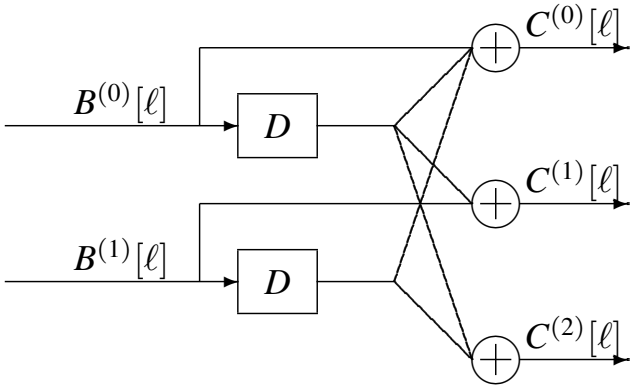
    - ★ Another example (B): $k = 2$, $n = 3$

$$\mathbf{G}(D) = \begin{bmatrix} 1+D & D & D \\ D & 1 & D \end{bmatrix}_{2\times 3}$$

Translation to squematic representation - Example B

$$\mathbf{G}(D) = \begin{bmatrix} 1+D & D & D \\ D & 1 & D \end{bmatrix}_{2\times 3}$$

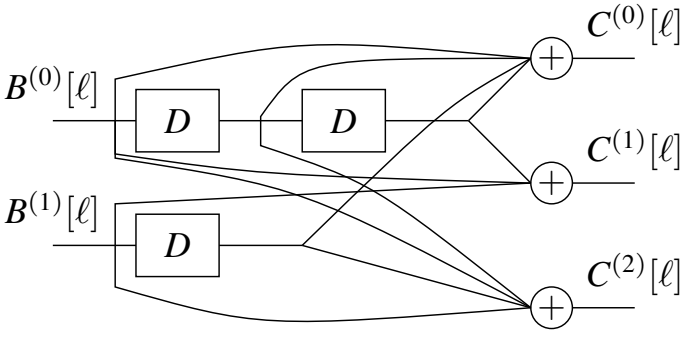
- Number of inputs of the bank:
  - Number of rows of $\mathbf{G}(D)$: $k = 2$
- Number of outputs of the bank:
  - Number of columns of $\mathbf{G}(D)$: $n = 3$
- Number of memories in each input:
  - Maximum degree of all polinomials in its corresponding row: $M^{(0)} = 1$, $M^{(1)} = 1$



Translation to schematic representation - Example C

$$\mathbf{G}(D) = \begin{bmatrix} 1+D+D^2 & 1+D^2 & 1+D \\ D & 1 & 1+D \end{bmatrix}_{2\times 3}$$

- Number of inputs of the bank:
  - Number of rows of $\mathbf{G}(D)$: $k = 2$
- Number of outputs of the bank:
  - Number of columns of $\mathbf{G}(D)$: $n = 3$
- Number of memories in each input:
  - Maximum degree of all polinomials in its corresponding row: $M^{(0)} = 2$, $M^{(1)} = 1$



Parameters of interest

- Total memory of the code: M_t
  - Total number of delay units (memories)

$$M_t = \sum_{i=0}^{k-1} M^{(i)}$$

  - Memory of the i-th input:

$$M^{(i)} = \max_j \text{degree}(g_{i,j}(D))$$

- Constraint length: K
  - Length of the inpulse response of the encoder (maximum number of discrete time instants where an input bit contributes to any system output)

$$K = 1 + \max_{i,j} \text{degree}(g_{i,j}(D)) = 1 + K_E$$

    K_E: memory of the impulse response of the encoder
  - In general, performance improves with higher K values

Systematic codes

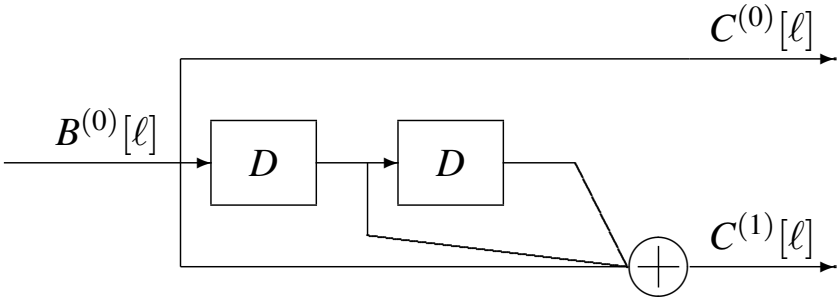
- Generator matrix

$$\mathbf{G}(D) = [\mathbf{I}_k \mid \mathbf{P}(D)]$$

$$\mathbf{G}(D) = [\mathbf{P}(D) \mid \mathbf{I}_k]$$

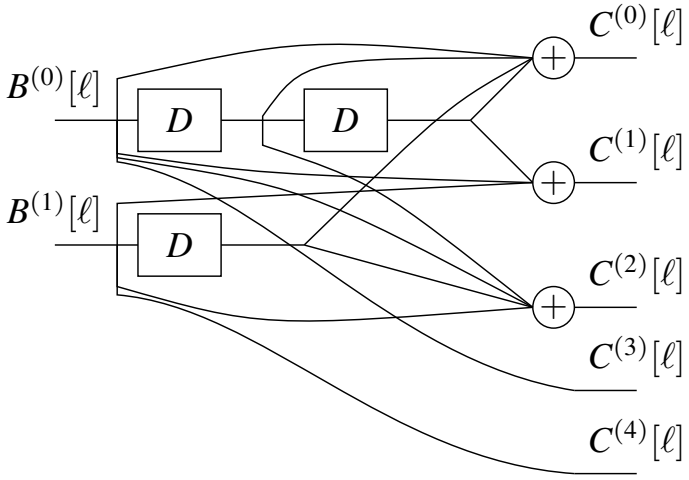
  - Inputs are “replicated” in a subset of outputs
  - Example (D)

$$\mathbf{G}(D) = \left[1,\ 1 + D + D^2\right]$$



Systematic codes - another example (E)

$$\mathbf{G}(D) = \left[\begin{array}{ccc|cc} 1+D+D^2 & 1+D^2 & 1+D & 1 & 0 \\ D & 1 & 1+D & 0 & 1 \end{array}\right]$$



Trellis diagram

- Definition for the system state in a given instant ℓ
  - Contents of the memories (M_t ordered bits)

$$\psi[\ell] = [B^{(0)}[\ell-1], \cdots, B^{(0)}[\ell-M^{(0)}], \cdots, B^{(k-1)}[\ell-1], \cdots, B^{(k-1)}[\ell-M^{(k-1)}]]$$

- Trellis diagram

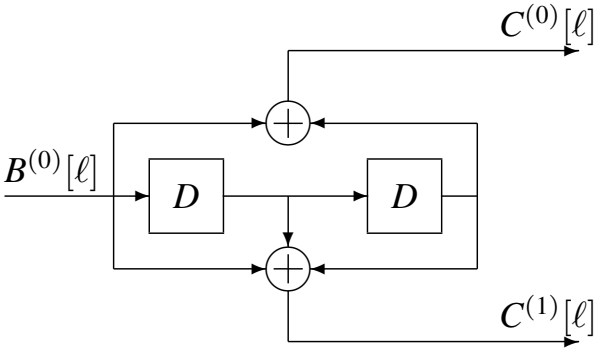
$$\psi[\ell] \xrightarrow{\text{label}} \psi[\ell+1]$$

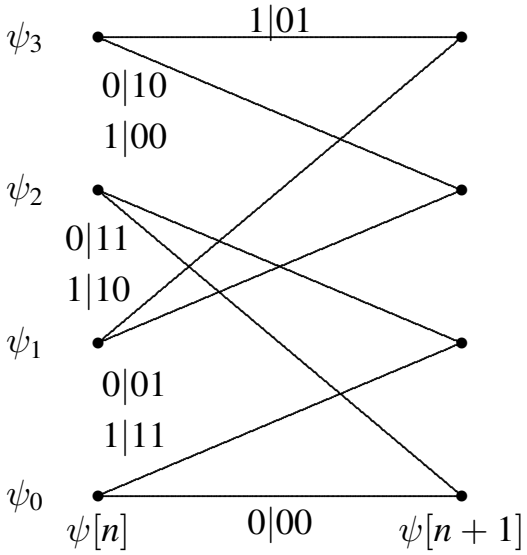
- Labelling of the trellis
  - Bits at the input (uncoded) | Bits at the output (encoded)

$$B^{(0)}[\ell], B^{(1)}[\ell], \cdots, B^{(k-1)}[\ell] \;\Big|\; C^{(0)}[\ell], C^{(1)}[\ell], \cdots, C^{(n-1)}[\ell]$$

Example - Convolutional (F)

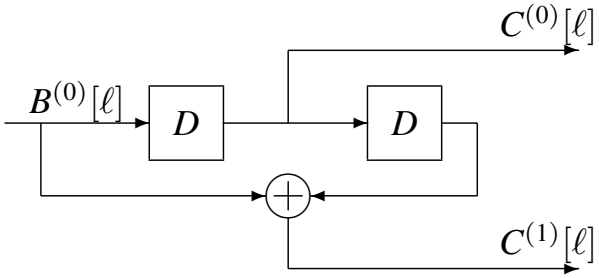
- State: $\psi[\ell] = \left[B^{(0)}[\ell-1], B^{(0)}[\ell-2]\right]$
- Values: $\psi_0 = [0,0]$, $\psi_1 = [1,0]$, $\psi_2 = [0,1]$, $\psi_3 = [1,1]$

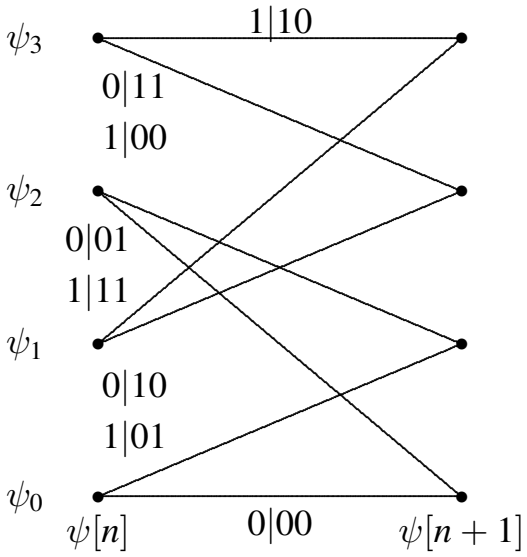




Example - Convolutional (G)

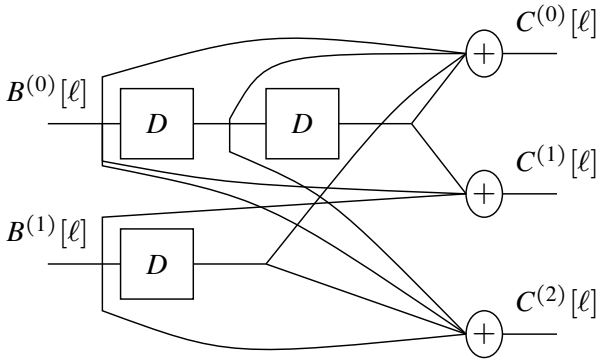
- State: $\psi[\ell] = \left[B^{(0)}[\ell-1], B^{(0)}[\ell-2]\right]$
- Values: $\psi_0 = [0,0]$, $\psi_1 = [1,0]$, $\psi_2 = [0,1]$, $\psi_3 = [1,1]$

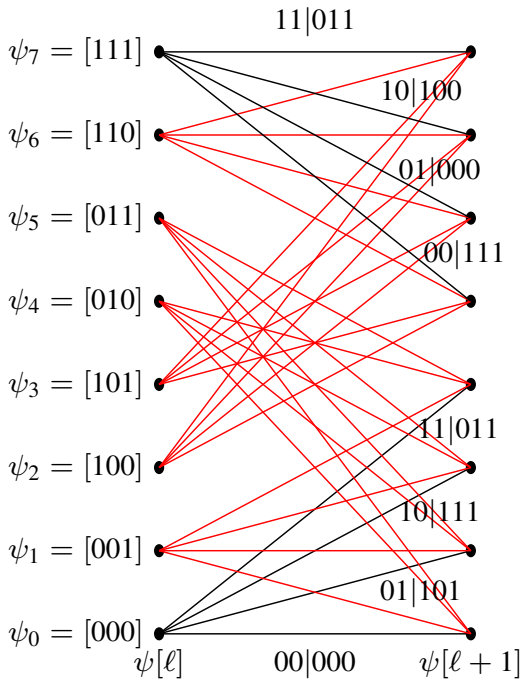




Ejemplo - Convolutional (C)

- State: $\psi[\ell] = \left[B^{(0)}[\ell-1], B^{(0)}[\ell-2], B^{(1)}[\ell-1]\right]$





Header and number of transitions in the trellis

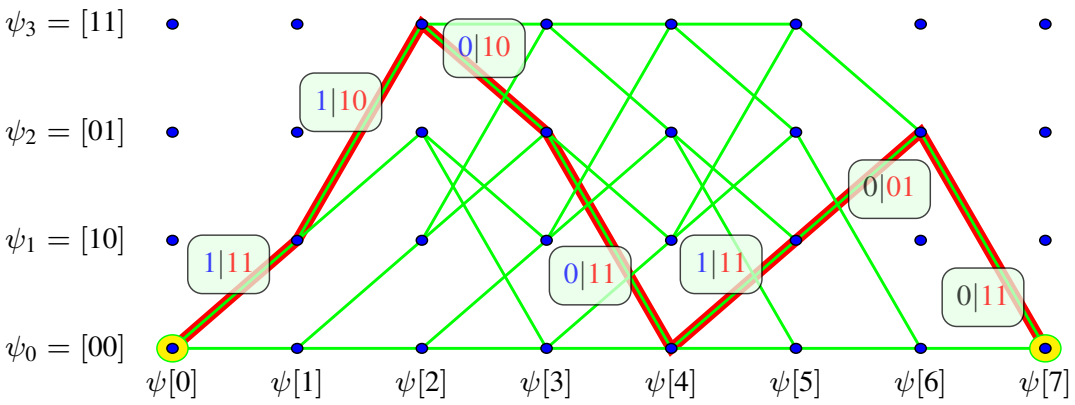
- Encoding of L blocks of k bits: $k \times L$ information bits
  - Serial to parallel conversion
    - L discrete instants for $B^{(i)}[\ell]$, $\ell \in \{0, 1, \cdots, L-1\}$
- Length of coded bits with information: $n \times L$
- Header to initialize the encoder
  - Header of $k \times K_E$ zeros
    - Zeros in all the k inputs for K_E discrete instants
    - Initial state: $\psi[0] = \psi_0 = [0, 0, \cdots, 0]$
    - Final state: $\psi[L + K_E] = \psi_0 = [0, 0, \cdots, 0]$
- Number of transition in the trellis diagram

  $L + K_E$ transitions
- Length of coded bits for $k \times L$ information bits

$$\underbrace{n \times L}_{\textit{information bits}} + \underbrace{n \times K_E}_{\textit{header bits}} = (L + K_E)n$$

Sequence of bits : path through the trellis

- Convolutional encoder F : $k = 1$, $n = 2$, $K_E = 2$
  - Header of $k \times K_E$ zeros: $00 \rightarrow$ $\psi[0] = \psi[L + K_E] = \psi_0$
- $L = 5$, block of $k \times L = 5$ bits: $2^{k \times L} = 2^5 = 32$ paths
  - Example: $B_b[m] = 11001$



- Encoded sequence: $n \times (L + K_E) = 14$ bits

$$C[m'] = 11\ 10\ 10\ 11\ 11\ 01\ 11$$

Decoding - Viterbi's algorithm

- Recovers the maximum likelihood (ML) data sequence
- Initial/final state
  - Reference header (usually zeros "*bit flushing*")
- Hard output (observation: detected bits $R[m]$)
  - Solution: encoded sequence $C[m]$ at minimum Hamming distance of the observation
    - ★ Branch metric in transition $\psi[\ell] \rightarrow \psi[\ell + 1]$:

$$d^H\left(\left[R^{(0)}[\ell], R^{(1)}[\ell], \cdots, R^{(n-1)}[\ell]\right], \left[C^{(0)}[\ell], C^{(1)}[\ell], \cdots, C^{(n-1)}[\ell]\right]\right)$$

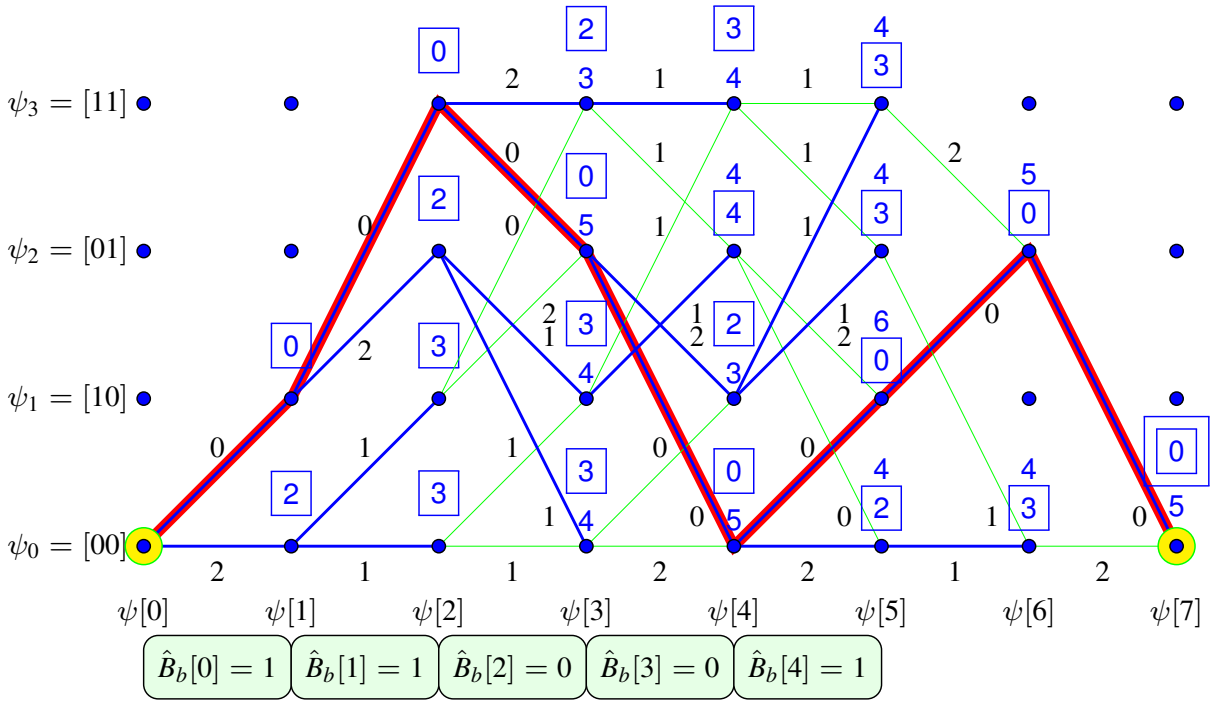
      Hamming distance between observation and encoded bits
- Soft output (observation: sequence $q[\ell]$)
  - Solution: sequence whose associated transmitted symbols are at minimum euclidean distance of the observation
    - ★ Branch metric: $|\boldsymbol{q}[\ell] - \boldsymbol{A}_i[\ell]|^2$

      The constellation and the binary assignment have to be taken into account to translate trellis labels to symbols of the constellation ($\boldsymbol{A}_i[\ell]$)
  - Better performance than working with hard output

Decoding working with hard output (Conv. F)

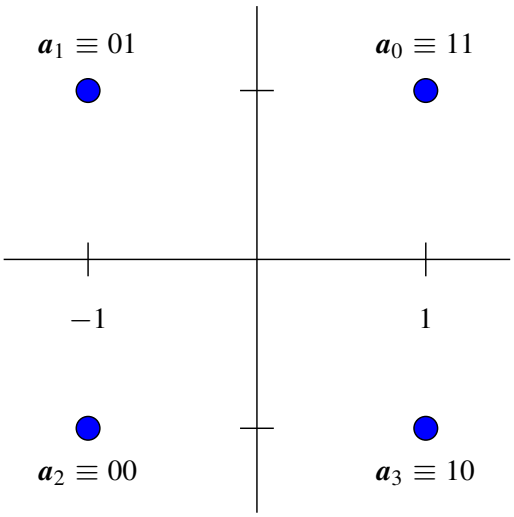
- Secuencia recibida

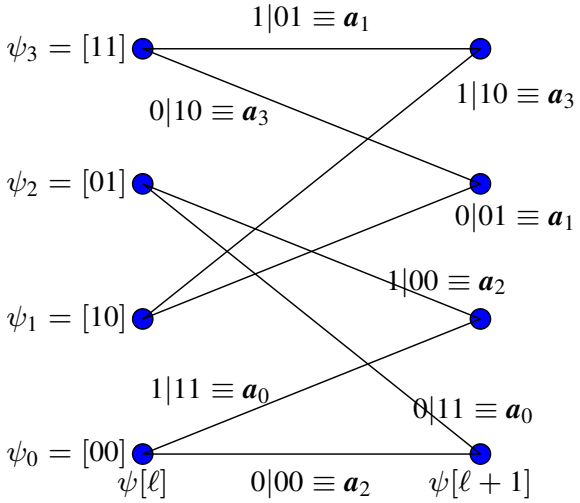
$$R[m'] = 11\ 10\ 10\ 11\ 11\ 01\ 11$$



Decoding working with soft output - Metric (Conv. F)

- Depends on the constellation and on the binary assignment
- Example 4-QAM

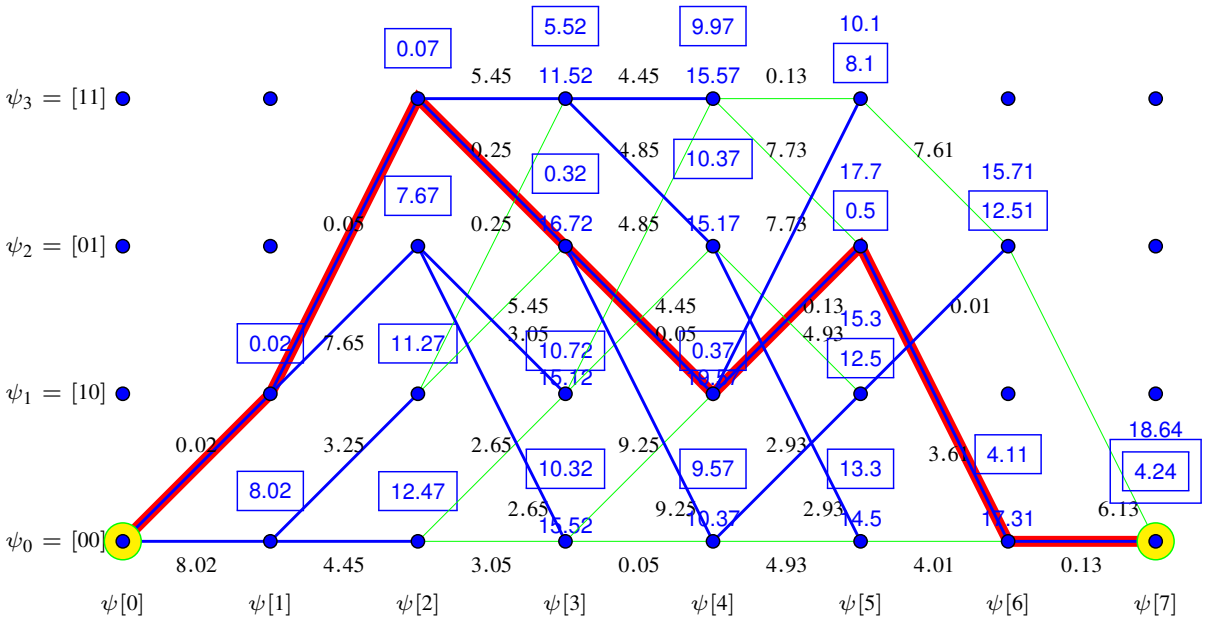




Decoding working with soft output - Example

- Received sequence

$$q[m] = \begin{bmatrix}+1.1\\+0.9\end{bmatrix}\ \begin{bmatrix}+1.1\\-0.8\end{bmatrix}\ \begin{bmatrix}+0.7\\-0.6\end{bmatrix}\ \begin{bmatrix}-1.2\\-1.1\end{bmatrix}\ \begin{bmatrix}-0.7\\+1.2\end{bmatrix}\ \begin{bmatrix}-0.9\\+1.0\end{bmatrix}\ \begin{bmatrix}+0.8\\+0.7\end{bmatrix}$$



$\hat{B}_b[0] = 1$ | $\hat{B}_b[1] = 1$ | $\hat{B}_b[2] = 0$ | $\hat{B}_b[3] = 1$ | $\hat{B}_b[4] = 0$

Performance

- Hard output

$$P_e \approx c \sum_{e=t+1}^{n\,z} \binom{n\,z}{e} \varepsilon^e \left(1-\varepsilon\right)^{n\,z-e}$$

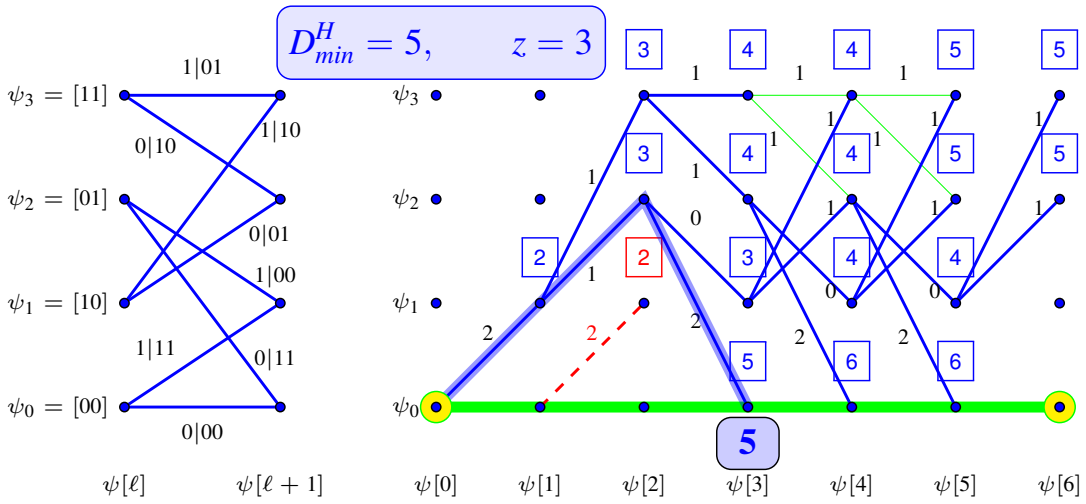
  - D^H_{min}: minimum Hamming distance for different coded sequences
  - z: length of erroneous event of minimum distance D^H_{min}
  - $t = \left\lfloor \frac{D^H_{min}-1}{2} \right\rfloor$ (correction capability over $n \times z$ bits)
  - ε: bit error rate during transmission (BER)
- Soft output

$$P_e \approx c\, Q\left(\frac{D^E_{min}}{2\sqrt{N_0/2}}\right)$$

  - D^E_{min}: minimum euclidean distance between sequences of symbols transmitted for two different data sequences

Calculation of D^H_{min}

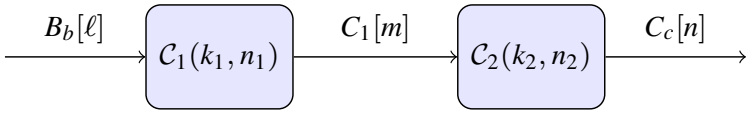
- Comparison with the encoded sequence of all zeros
  - Branch metric: number of ones in the set of n encoded bits
- Viterbi algorithm can be used to evaluate distances over erroneous events



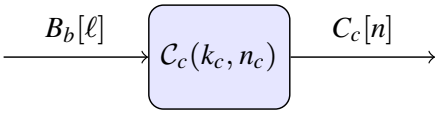
Concatenated codes

- A complex code can be implemented by the concatenation of two simpler codes
  - Serial concatenation of two encoders
  - Input code (outer) $\mathcal{C}_1(k_1, n_1)$, rate $R_1 = k_1/n_1$
  - Output code (inner) $\mathcal{C}_2(k_2, n_2)$, rate $R_2 = k_2/n_2$
  - Tasa del código concatenado

$$R_c = \frac{k_1\ k_2}{n_1\ n_2} = R_1 \times R_2.$$



- Usual relationships for code sizes
  - Case $n_1/k_2 = c \in \mathbb{Z}$
    - ★ In this case, there is an equivalent code $\mathcal{C}_c(k_1, c\ n_2)$
  - Case $k_2/n_1 = c \in \mathbb{Z}$
    - ★ In this case, there is an equivalent code $\mathcal{C}_c(c\ k_1, n_2)$



Example for block codes

- Codes $\mathcal{C}_1(2,3)$, $\mathcal{C}_2(3,6)$

$$\mathbf{G}_1 = \begin{bmatrix} 101 \\ 011 \end{bmatrix} \equiv$$

i	$\mathbf{b}_i$	$\mathbf{c}_i$
0	00	000
1	01	011
2	10	101
3	11	110

$$\mathbf{G}_2 = \begin{bmatrix} 100011 \\ 010101 \\ 001111 \end{bmatrix} \equiv$$

i	$\mathbf{b}_i$	$\mathbf{c}_i$
0	000	000000
1	001	001111
2	010	010101
3	011	011010
4	100	100011
5	101	101100
6	110	110110
7	111	111001

- Concatenated code $\mathcal{C}_c(2,6) = \mathcal{C}_1(2,3) - \mathcal{C}_2(3,6)$

i	$\mathbf{b}_i$	$\mathbf{c}_i$
0	00	000000
1	01	011010
2	10	101100
3	11	110110

$$\equiv \mathbf{G}_c = \begin{bmatrix} 101100 \\ 011010 \end{bmatrix}$$

$\mathbf{G}_c = \mathbf{G}_1 \times \mathbf{G}_2$, because $k_2 = n_1$

Another example with block codes

- Concatenated code $\mathcal{C}_c(3,12) = \mathcal{C}_2(3,6) - \mathcal{C}_2(3,6)$

$$\mathbf{G}_1 = \mathbf{G}_2 = \begin{bmatrix} 100011 \\ 010101 \\ 001111 \end{bmatrix} \equiv$$

i	$\mathbf{b}_i$	$\mathbf{c}_i$
0	000	000000
1	001	001111
2	010	010101
3	011	011010
4	100	100011
5	101	101100
6	110	110110
7	111	111001

i	$\mathbf{b}_i$	$\mathbf{c}_{1,i}$		$\mathbf{c}_{2,i}$	
0	000	000	000	000000	000000
1	001	001	111	001111	111001
2	010	010	101	010101	101100
3	011	011	010	011010	010101
4	100	100	011	100011	011010
5	101	101	100	101100	100011
6	110	110	110	110110	110110
7	111	111	001	111001	001111

$$\equiv \mathbf{G}_c = \begin{bmatrix} 100011011010 \\ 010101101100 \\ 001111111001 \end{bmatrix}$$

Another example with block codes (II)

- Código concatenado $\mathcal{C}_c(3,12) = \mathcal{C}_1(3,6) - \mathcal{C}_2(3,6)$

i	$\mathbf{b}_i$	$\mathbf{c}_i$
0	000	000000000000
1	001	001111111001
2	010	010101101100
3	011	011010010101
4	100	100011011010
5	101	101100100011
6	110	110110110110
7	111	111001001111

$$\equiv \mathbf{G}_c = \begin{bmatrix} 100011011010 \\ 010101101100 \\ 001111111001 \end{bmatrix}$$

Example with convolutional codes

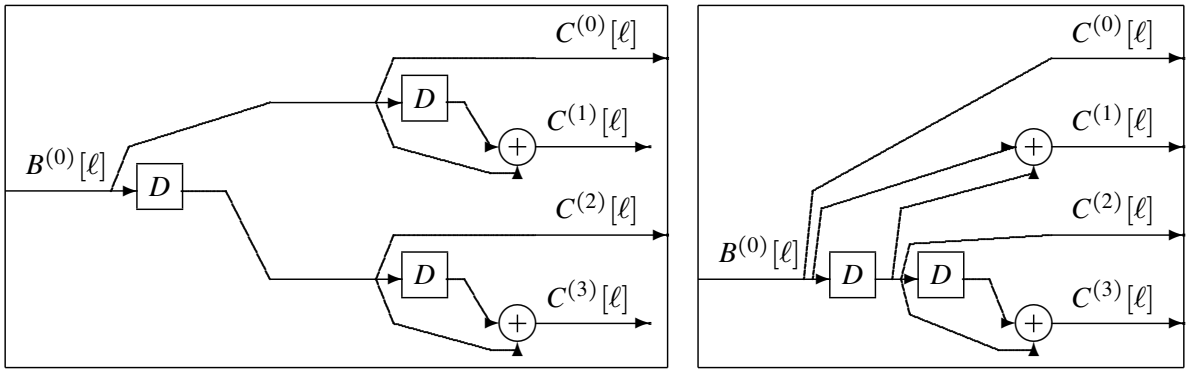
- Codes $\mathcal{C}_1(1,2)$, $\mathcal{C}_2(2,4)$

$$\mathbf{G}_1(D) = [1, D], \quad \mathbf{G}_2(D) = \begin{bmatrix} 1, & 1+D, & 0, & 0 \\ 0, & 0, & 1, & 1+D \end{bmatrix}$$

  - The code $\mathcal{C}_2$ is implemented with two codes $\mathbf{G}_2' = [1, 1+D]$ in parallel

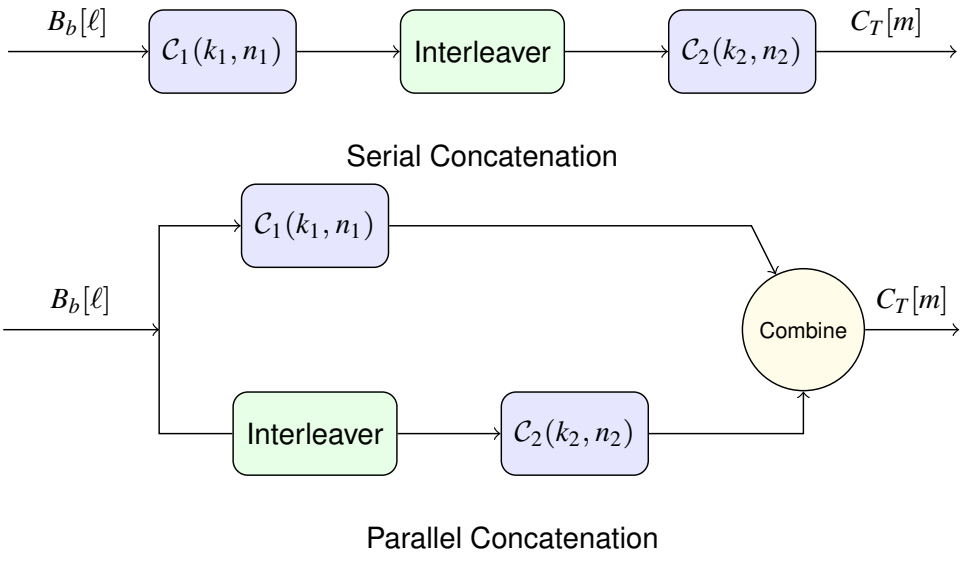
- Concatenated code $\mathcal{C}_c(1,4) = \mathcal{C}_1(1,2) - \mathcal{C}_2(2,4)$

$$\mathbf{G}_c = [1, \quad 1+D, \quad D, \quad D+D^2]$$



Turbo codes

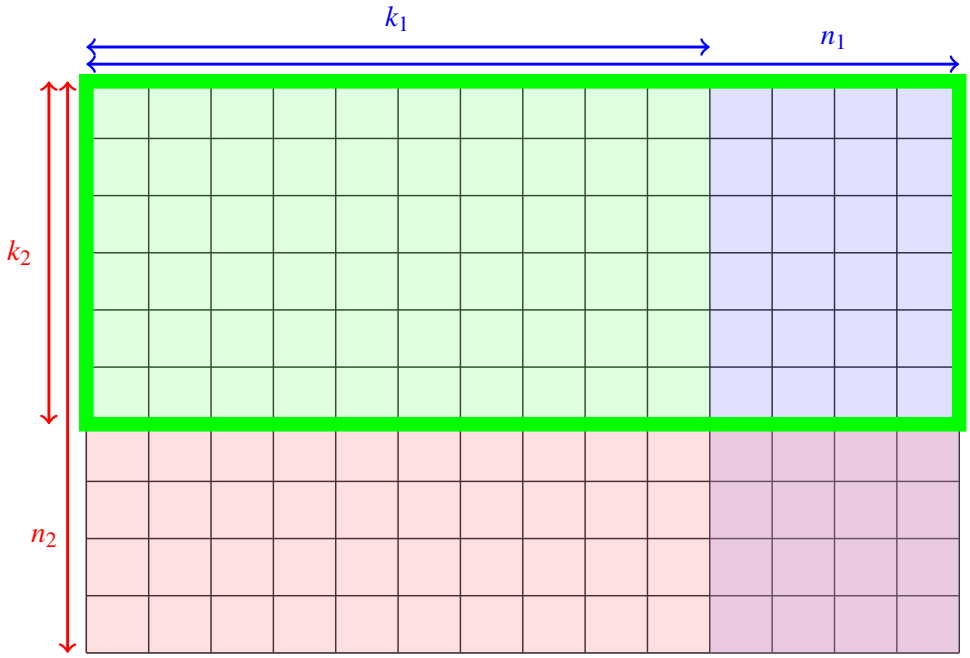
- Concatenation of codes using interleavers



  - Encoders
    - ★ Recursive Systematic Convolutional (RSC) encoders
  - Interleaver: re-arrangement of the bits
    - ★ Errors of one decoder in different coded words of the other one
    - ★ Determines the perfomance (pseudo-random interleavers are used)

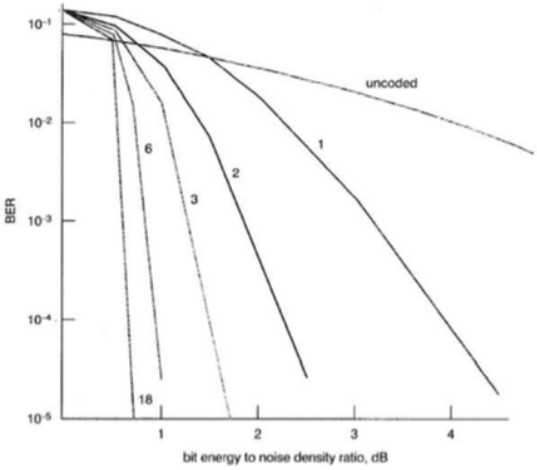
Interleaver - Example for serial concatenation ($k_2 \times n_1$)

- Block interleaver
  - Input of bits writing by rows
  - Output of bits reading by columns



Iterative decoding

- Iterative decoding (BCJR algorithm)
  - Each decoder provides a soft output
    - ⋆ LLR: Log-Likelihood Ratio for each bit
  - This information is exchanged between decoders
- Performance: close to the theoretical limit



LDPC codes

LDPC: *Low Density Parity Check*

- Linear block codes
  - Codes with big size
    - ⋆ Example: $\mathcal{C}(5000, 10000)$
  - Parity check matrix is sparse (few 1s)
- Representation by means of a bipartite graph (Tunner graph)
  - Two kind of nodes
    - ⋆ Bit nodes
    - ⋆ Parity-check nodes
  - Aplication of the Turbo principle: iterative decoding
    - ⋆ Soft outputs are provided
    - ⋆ Iterative algorithms of type "*belief propagation*" (BCJR, MAP, SOVA) are used
- Excelent performance
  - Current state of the art, along with Turbo codes