Lecture Notes on “Channel Coding”

Prof. Peter A. Hoeher
Information and Coding Theory Lab
Faculty of Engineering
University of Kiel
Germany

ph@tf.uni-kiel.de
www.tf.uni-kiel.de/ict

Summer Term 2003
Last update: 14/05/2003

Contents

Chapter 0: Introduction

- What is channel coding?
- Tasks and applications of channel coding
- Error detection, error correction, error concealment
- Why digital transmission?
- Source coding, cryptology, channel coding
- Message, information, code, redundancy

What is Channel Coding?

Channel coding provides techniques in order to transmit data from a transmitter to a receiver with a minimum amount of errors.

Redundancy is added to the data, so that transmission errors can be detected and/or corrected at the receiver.

Main tasks:

- **error detection**
- **error correction**
- **error concealment**.

Without channel coding, robust data transmission via noisy communication channels as well as reliable storage is unthinkable.
Therefore, channel coding is applied in many different applications, particularly in **digital transmission systems** and in **digital storage systems**.

Applications of Channel Coding

Digital transmission systems

- mobile radio systems
- data modems, internet
- satellite communication systems, deep-space probes
- underwater communication systems
- optical communication systems

Digital storage systems

- compact disc (CD), digital versatile disc (DVD)
- digital audio tape (DAT)
- hard disc, magnetic storage systems

Why Digital Transmission?

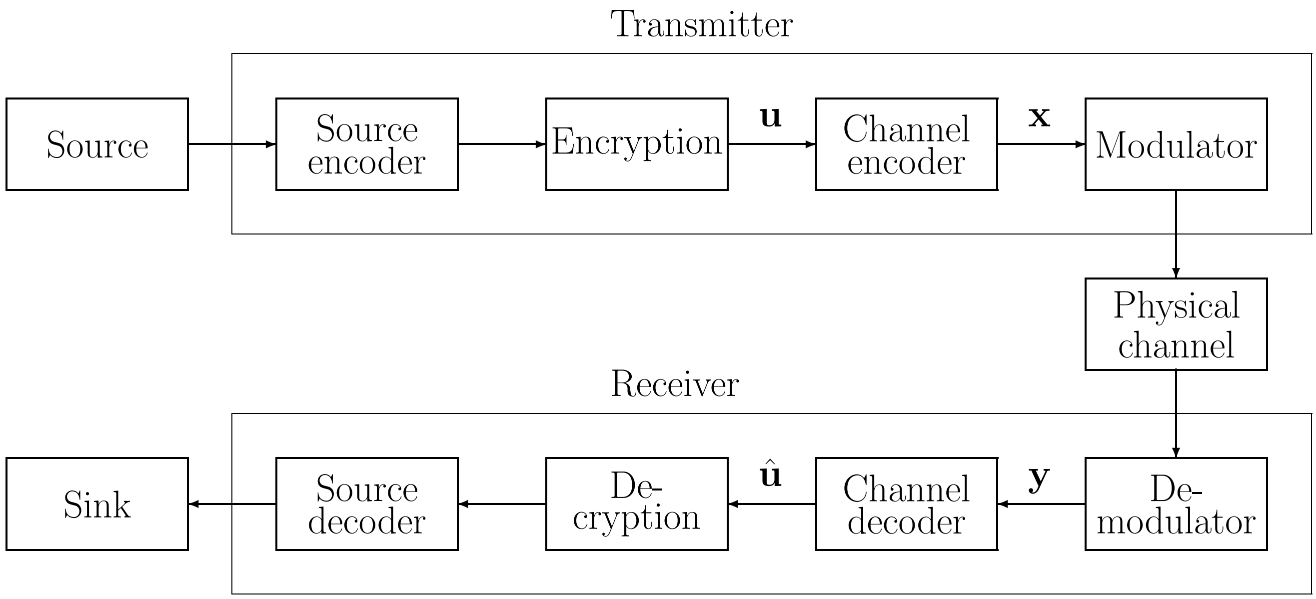
New transmission systems will exclusively be **digital**:

- Due to source coding, channel coding, and efficient modulation techniques significantly higher **power efficiency** compared to analog transmission (Example: FM broadcasting $\approx$ 100 kW Tx power, DAB $\approx$ 1 kW Tx power)
- Higher **bandwidth efficiency** due to non-binary modulation techniques
- **Flexibility**: Trade-off between power efficiency and bandwidth efficiency (e.g. HDTV, EDTV, SDTV, LDTV)
- **Regenerability** (e.g. repeaters in sea cables, satellite on-board processing)
- Higher **quality** (due to error correction, equalization, etc.)
- Higher **capacity** (due to macro/micro/pico cells, single frequency networks, etc.)
- Digital **recording media** (tapes, discs, RAM, CD, etc.)
- **Multimedia** (integration of speech, audio, data, images, video, etc.)
- Advances in digital **technology** (micro processors, DSP, RAM, etc.)

Quality Criteria of Digital Transmission Systems

- Data rate [bit/s]
- Data volume [byte]
- Quantitative quality criteria: e.g. bit error rate, word error rate, frame error rate
- Subjective (audio, visual) quality
- Power efficiency: necessary signal/noise ratio for a given quality constraint [decibel]
- Bandwidth efficiency: necessary bandwidth for a given data rate [bit/s/Hz]
- Network capacity [number of users/km^2/Hz]
- Spatial/temporal outage probability [percent]
- Computational complexity [number of flops, number of transistors, chip area]

Digital Transmission System



Shannon's Information Theory

Claude E. Shannon (1948)

- **Source coding:** Data compression
- **Cryptology:** Data encryption
- **Channel coding:** Error detection/correction/concealment

Separation theorem:
Source coding, encryption, and channel coding *may* be separated without information loss (note that the separation theorem holds for very long data sequences only)

Examples for Source Coding, Encryption, and Channel Coding

- **Source Coding**:
  1. Example: Characters A-Z encoded with $\lceil \log_2(26) \rceil = 5$ bits, no data compression
  2. Example: Characters A-Z encoded, with data compr. (e.g. Huffman algorithm)

No data compression:			With data compression:		
	A $\rightarrow$	[00000]		A $\rightarrow$	[11]
	B $\rightarrow$	[00001]		B $\rightarrow$	[001]
	C $\rightarrow$	[00010]		C $\rightarrow$	[0110]
	...			...	

- **Encryption**:
  Example: Add a key word to each source code word modulo 2
  e.g. source code word [0110] ($\doteq$ "C"), key word [1010] (random sequence)
  $\Rightarrow$ info word $[0110] \oplus [1010] = [1100]$

- **Channel coding**:
  Example: (2,1) repetition code with code rate $R = 1/2$
  e.g. info word [1100] $\Rightarrow$ code word [11 11 00 00]

Examples for Source Coding Techniques

Application	Rate without compr.	Rate with compr.	Technique
Speech coding	64 kbit/s (8 kHz · 8 bit/sample)	13 … 7 kbit/s 4.8 … 2.4 kbit/s	CELP vocoder
Audio coding	1.536 Mbit/s (2 · 48 kHz · 16 bit/sample)	256 … 128 kbit/s 96 kbit/s	MPEG-1 MPEG-2 AAC
Image coding	8 bit/pixel	0.25 … 1.25 bit/pix.	JPEG
Video coding	625 Mbit/s (HDTV) 163.9 Mbit/s (SDTV)	24 Mbit/s 6 Mbit/s	MPEG-2
Text compr.	–	factor 3:1 … 10:1	Lempel-Ziv

Tasks of Channel Coding

- **Error detection** and **error correction**

  ⇒ enhancement of error probability (**data security**) and/or

  ⇒ reduction of transmit power (enhancement of **power efficiency**)

- **Error concealment**
  (in conjunction with source coding)

  ⇒ improvement of subjective performance

- Enhancement of **bandwidth efficiency**
  (in conjunction with non-binary modulation techniques)

- **Unequal error protection**
  (in conjunction with source coding)

Fundamental Principles of Channel Coding

- **Forward error correction (FEC)**:
  In forward error correction schemes there is no feedback from the channel decoder to the channel encoder.
- **Automatic repeat request (ARQ)**:
  In automatic repeat request schemes there is feedback from the channel decoder to the channel encoder. For example, a code word may be repeated until the channel decoder does not detect any error. Alternatively, additional parity bits may be transmitted until the channel decoder does not detect any error. The additional decoding delay is not tolerable in all transmission schemes, such as in real-time speech transmission schemes.

Within this lecture our focus is on FEC techniques.

Fundamental Notions I

Message (general definition): Messages may be ideas, data, facts, lies, or nonsense, etc. Messages may or may not be known to the receiver in advance.

Information: Information may be data or facts, which are not known to the receiver in advance. Information does not contain redundancy.

According to **Shannon's information theory**, information is measurable. However, the impact on the receiver is not measurable.

Symbol: Messages and information are represented by symbols, which are known to all communication users.
(Examples: Characters, numbers, bits)

Message (formal definition): A message is a sequence of words.
Each word consists of symbols, which are known to all communication users.
In digital communications, a message consists of a sequence of zeros and ones.

Fundamental Notions II

Code: A code is a set of words over a well-defined symbol alphabet.
(Example: English words, German words, ASCII Code, genetic code.)

Redundancy: If, on average, more symbols are used to represent a message than actually needed for a perfect reconstruction, then the message contains redundancy besides information.
(Example: German Duden approximately 115 000 words,
number of possible words of length 8: $26^8 \approx 2.1 \cdot 10^{11}$)

Redundancy helps to protect information with respect to transmission errors.
(Example: Digital communica<u>d</u>ions makes fun)

Chapter 1: Block Codes I

- Definition of block codes
- Code rate, redundancy, error detection, error correction
- Systematic codes, parity check symbols, single parity check code, repetition code
- Hamming distance, minimum distance, Hamming weight
- Linear block codes, Hamming codes
- Hard-decision decoding, soft-decision decoding
- Word error probability, bit error probability
- MAP decoding, ML decoding, bounded minimum distance decoding
- Binary symmetrical channel, AWGN channel
- Coding gain
- Generator matrix, check matrix, dual code, syndrome, syndrome decoding
- Cyclic block codes, generator polynomial, parity check polynomial

Definition of Block Codes

We denote a sequence $\mathbf{u} := [u_0, u_1, \ldots, u_{k-1}]$ of k info symbols as an **info word**. The info symbols u_i, $i = 0, 1, \ldots k-1$, are defined over the alphabet $\{0, 1, \ldots, q-1\}$, where q is the number of elements ("cardinality") of the symbol alphabet.

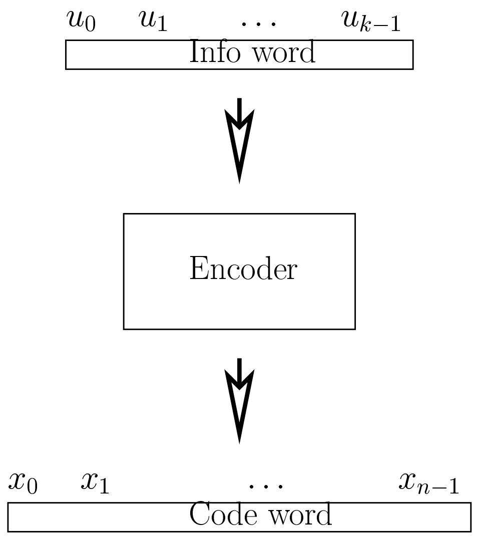
Definition (**block code**): An $(n, k)_q$ block code $\mathcal{C}$ maps an info word $\mathbf{u} = [u_0, u_1, \ldots, u_{k-1}]$ of length k onto a **code word** $\mathbf{x} := [x_0, x_1, \ldots, x_{n-1}]$ of length n, where $n > k$.

The code symbols x_i, $i = 0, 1, \ldots, n-1$, are assumed to be within the same alphabet $\{0, 1, \ldots, q-1\}$.

The assignment of code words with respect to the info words is

- unambiguous and reversible: For each code word there is exactly one info word
- time invariant: The mapping rule does not change in time
- memoryless: Each info word effects only one code word

Block Diagram of a Block Code



$u_i \in \{0, 1, \dots, q-1\},\ 0 \le i \le k-1$
$\mathbf{u} \in \{0, 1, \dots, q-1\}^k$

$x_i \in \{0, 1, \dots, q-1\},\ 0 \le i \le n-1$
$\mathbf{x} \in \{0, 1, \dots, q-1\}^n$

Redundancy, Error Detection, Error Correction

A **code** $\mathcal{C}$ is the set of all q^k code words.

Since n symbols are needed in order to transmit k info symbols, where $n > k$, the code contains **redundancy**, because only q^k of the q^n possible combinations are allowed.

This redundancy is used for **error detection**, **error correction**, or **error concealment** by the receiver.

The transmitted (possibly erroneous or noisy) code words are denoted as **received words $\mathbf{y}$**. For **hard-decision decoding** $y_i \in \{0, 1, \dots, q-1\}$, $i = 0, 1, \dots, n-1$, by definition.

The ratio

$$R := \frac{k}{n} < 1$$

is called **code rate**. The larger the code rate, the less is the redundancy given the same length n of the code word.

$\Rightarrow$ **Trade-off between bandwidth efficiency and power efficiency**.

Systematic Codes

Definition (**systematic code**): A code is called systematic, if the mapping between info words and code words is such that the info words are explicitly contained in the code words.

The $n-k$ remaining symbols are called **parity check symbols** ($q=2$: parity check bits).

Example 1: $(3,2)_2$-**single parity check (SPC) code**:
($q=2$, i.e., one symbol corresponds to one bit)

Info word $\mathbf{u}=[u_0,u_1]$	Code word $\mathbf{x}=[x_0,x_1,x_2]$
[00]	[000]
[01]	[011]
[10]	[101]
[11]	[110]

Parity check equation: $u_0 \oplus u_1 \oplus x_2 = 0$ ($\oplus$: modulo-q addition)

Code: $\mathcal{C} = \{[000],[011],[101],[110]\}$

Systematic Codes

Example 2: $(3,2)_{256}$-**single parity check (SPC) code**:
($q=2^8=256$, i.e., one symbol corresponds to one byte)

Info word $\mathbf{u}=[u_0,u_1]$	Code word $\mathbf{x}=[x_0,x_1,x_2]$
$[0,0]$	$[0,0,0]$
$[0,1]$	$[0,1,255]$
$[0,2]$	$[0,2,254]$
$\cdots$	$\cdots$
$[0,255]$	$[0,255,1]$
$[1,0]$	$[1,0,255]$
$\cdots$	$\cdots$
$[255,255]$	$[255,255,2]$

Parity check equation: $u_0 \oplus u_1 \oplus x_2 = 0$ ($\oplus$: modulo-q addition)

Code: $\mathcal{C} = \{[0,0,0],[0,1,255],\ldots,[255,255,2]\}$ ($q^k = 256^2 = 65536$ code words)

Hamming Distance

Definition (**Hamming distance**): The Hamming distance $d_H(\mathbf{x}, \mathbf{y})$ is the number of deviations between the components of $\mathbf{x}$ and $\mathbf{y}$, where $\mathbf{x} \in \{0, 1, \ldots, q-1\}^n$ and $\mathbf{y} \in \{0, 1, \ldots, q-1\}^n$ are words of length n. $\mathbf{x}$ and $\mathbf{y}$ may be code words or received words, respectively.

Example 1 ($q = 2$):
The Hamming distance between the code words
$\mathbf{x} = [000]$ and $\mathbf{y} = [110]$ is $d_H(\mathbf{x}, \mathbf{y}) = 2$
$\mathbf{x} = [011]$ and $\mathbf{y} = [101]$ is $d_H(\mathbf{x}, \mathbf{y}) = 2$.

Example 2 ($q = 3$):
The Hamming distance between the code words
$\mathbf{x} = [012]$ and $\mathbf{y} = [021]$ is $d_H(\mathbf{x}, \mathbf{y}) = 2$
$\mathbf{x} = [012]$ and $\mathbf{y} = [120]$ is $d_H(\mathbf{x}, \mathbf{y}) = 3$.

Minimum Distance

Definition (**minimum distance**): The minimum distance d_{min} of an (n, k) block code $\mathcal{C}$ is the minimum Hamming distance between all code words:

$$d_{min} := \min_{\text{all pairs of code words}} \{d_H(\mathbf{x}, \mathbf{y}) | \mathbf{x}, \mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}\}.$$

Example:
Let $\mathbf{x} = [00000]$, $\mathbf{y} = [01010]$, and $\mathbf{z} = [11111]$ be code words of a block code. We recognize that $d_H(\mathbf{x}, \mathbf{y}) = 2$, $d_H(\mathbf{x}, \mathbf{z}) = 5$, and $d_H(\mathbf{y}, \mathbf{z}) = 3$. Accordingly, $d_{min} = 2$.

In detailed notation, a block code $\mathcal{C}$ is denoted as an $(n, k, d_{min})_q$ code.

Example 1: A binary **single parity check code** is an $(n, n-1, 2)_2$ block code.

Example 2: A binary **repetition code** is an $(n, 1, n)_2$ block code.

Definition (**Hamming weight**): If the all-zeros code word exists, the Hamming weight $w_H(\mathbf{x})$ is the number of components of $\mathbf{x}$ which are different from zero.

Properties of the Hamming Distance and the Hamming Weight

The Hamming distance is a **metric** in mathematical sense, i.e., the following properties hold:

$$\begin{aligned} d_H(\mathbf{x},\mathbf{y}) &= d_H(\mathbf{y},\mathbf{x}) \\ 0 &\leq d_H(\mathbf{x},\mathbf{y}) \leq n \\ d_H(\mathbf{x},\mathbf{y}) = 0 &\Leftrightarrow \mathbf{x} = \mathbf{y} \\ d_H(\mathbf{x},\mathbf{y}) &\leq d_H(\mathbf{x},\mathbf{z}) + d_H(\mathbf{z},\mathbf{y}) \qquad \text{(triangle inequality)} \end{aligned}$$

If addition $\oplus$ and subtraction $\ominus$ are defined in the range of the symbols, the following properties apply for the Hamming weight:

$$\begin{aligned} w_H(\mathbf{x}) &= w_H(-\mathbf{x}) \\ 0 &\leq w_H(\mathbf{x}) \leq n \\ w_H(\mathbf{x}) = 0 &\Leftrightarrow \mathbf{x} = \mathbf{0} = [0,\ldots,0] \\ w_H(\mathbf{x} \oplus \mathbf{y}) &\leq w_H(\mathbf{x}) + w_H(\mathbf{y}) \end{aligned}$$

Modifications

Definition: An $(n, k, d_{min})_q$ block code can be modified to an $(n', k', d'_{min})_q$ block code as follows:

1. **Extending** means to append additional parity symbols:
   $n' > n$, $k' = k$, $R' < R$, $d'_{min} \geq d_{min}$.
2. **Puncturing** means to suppress parity symbols:
   $n' < n$, $k' = k$, $R' > R$, $d'_{min} \leq d_{min}$.
3. **Lengthening** means to append additional info symbols:
   $n' > n$, $k' > k$, $n' - k' = n - k$, $R' > R$, $d'_{min} \leq d_{min}$.
4. **Shortening** means to suppress info symbols:
   $n' < n$, $k' < k$, $n' - k' = n - k$, $R' < R$, $d'_{min} \geq d_{min}$.
   In case of shortening, multiple code words have to be avoided by a constraint.
   For example, for binary codes ($q = 2$) the constraint may be that all code words, where an info bit 0 is suppressed, shall be deleted.

Linear Block Codes

Definition (**linear block code**): An $(n, k, d_{min})_q$ block code $\mathcal{C}$ is called linear, if for any possible combination of two code words the modulo-q sum of their components yields a valid code word:

$$\mathbf{x}, \mathbf{y} \in \mathcal{C} \;\Rightarrow\; \mathbf{x} \oplus \mathbf{y} \in \mathcal{C}.$$

Additionally, for non-binary codes ($q \neq 2$) we require:

$$\mathbf{x} \in \mathcal{C},\; \alpha \in \{0, 1, \ldots, q-1\} \;\Rightarrow\; \alpha \cdot \mathbf{x} \in \mathcal{C}.$$

Example:
$\mathcal{C} = \{[000], [011], [101], [110]\}$ is linear
$\mathcal{C} = \{[000], [011], [110], [111]\}$ is non-linear.

Linear block codes are an important subset of block codes.

All block codes which are important in practice (such as repetition codes, single parity check codes, Hamming codes, BCH codes, Reed-Solomon codes) are linear codes.

Minimum Distance for Linear Codes

Theorem (**minimum distance**): The minimum distance of a linear $(n, k, d_{min})_q$ block code $\mathcal{C}$ is

$$d_{min} = \min_{\text{all code words}} \{w_H(\mathbf{x}) | \mathbf{x} \in \mathcal{C}, \mathbf{x} \neq \mathbf{0}\}.$$

Sketch of a proof: $d_H(\mathbf{x}, \mathbf{y}) = w_H(\mathbf{x} \ominus \mathbf{y})$, if the subtraction is defined in the range of the symbols.

Hence, an advantage of linear block codes is that for the calculation of d_{min} not all $q^k(q^k - 1)$ pairs are to be considered, but only $q^k - 1$ words.

The property of linearity may be used for an efficient encoding and decoding of linear block codes.

Decoding Spheres

Definition (**decoding sphere**): A decoding sphere $K_r(\mathbf{x})$ of radius r around a word $\mathbf{x}$ is the set of all words $\mathbf{y}$, which have a Hamming distance $d_H(\mathbf{x}, \mathbf{y}) \leq r$:

$$K_r(\mathbf{x}) := \{\mathbf{y} | d_H(\mathbf{x}, \mathbf{y}) \leq r\}.$$

Example: $(3, 1, 3)_2$ repetition code $\mathcal{C} = \{[000], [111]\}$

$K_1([000]) = \{[000], [100], [010], [001]\}$
$K_1([111]) = \{[111], [110], [101], [011]\}$

$K_2([000]) = \{[000], [100], [010], [001], [110], [101], [011]\}$
$K_2([111]) = \{[111], [110], [101], [011], [001], [010], [100]\}$

$K_3([000]) = K_3([111])$

Error Detection and Error Correction

The larger the minimum distance d_{min}, the more errors may be detected and/or corrected. This important fact will now be demonstrated for hard-decision decoding.

Theorem (**error detection**): A linear or non-linear $(n, k, d_{min})_q$ block code $\mathcal{C}$ is able to detect

$$t' = d_{min} - 1$$

symbol errors for sure. Although each symbol consists of $\log_2(q)$ bits, for $q > 2$ this does not mean that $(d_{min} - 1) \cdot \log_2(q)$ bit errors are detected for sure.

Example (single parity check code): Each SPC code is able to detect one error for sure.

Theorem (**error correction**): A linear or non-linear $(n, k, d_{min})_q$ block code $\mathcal{C}$ is able to correct t symbol errors for sure, if

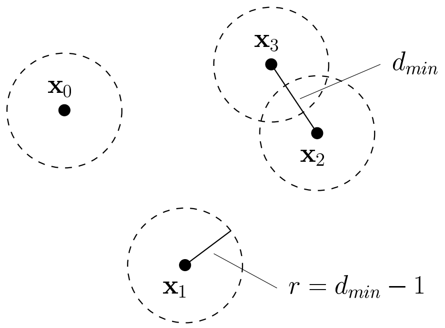
$$2t + 1 \leq d_{min}, \qquad \text{i.e.} \quad t = \lfloor (d_{min} - 1)/2 \rfloor .$$

Example (repetition code): A $(3, 1, 3)_2$ code is able to correct one error for sure.

Proof of Error Detection Theorem

- Each sphere of radius $r = d_{min} - 1$ around a code word includes no other code word.
- If at the most $d_{min} - 1$ errors occur, the received word is included in the decoding sphere around the transmitted code word.
- Since in this decoding sphere no further code word is contained, a confusion with an allowed code word is not possible.
- Therefore, an $(n, k, d_{min})_q$ block code can detect $t' = d_{min} - 1$ symb. errors for sure.

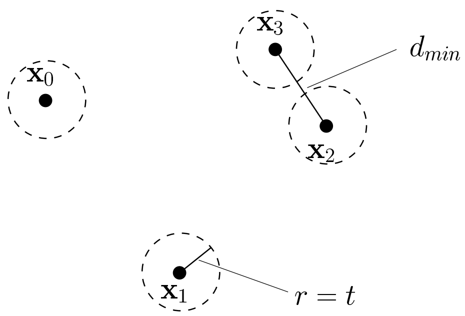
$\boxed{q = 2,\ k = 2\ \Rightarrow\ q^k = 4 \text{ code words}}$



Proof of Error Correction Theorem

- Now we consider decoding spheres of radius t.
- Hence, decoding spheres with a minimum distance of $2t + 1$ are disjunct.
- Therefore, an $(n, k, d_{min})_q$ block code can correct t symbol errors for sure, if $2t + 1 \leq d_{min}$, i.e., $t \leq \lfloor (d_{min} - 1)/2 \rfloor$.

$\boxed{q = 2,\ k = 2\ \Rightarrow\ q^k = 4 \text{ code words}}$



Hamming Codes

Definition (**Hamming codes**): $(n, k, d_{min})_q = (n, n-r, 3)_q$ Hamming codes of order r are defined as follows:

$$\begin{aligned} n &= \frac{q^r - 1}{q - 1} = 1 + q + q^2 + \ldots + q^{r-1} \\ k &= n - r \end{aligned}$$

Hamming codes exist for all orders $r \geq 2$.

All Hamming codes are linear and systematic and have a minimum distance of $d_{min} = 3$, i.e. $t = 1$ symbol error can be corrected for sure. Example $q = 2$:

r	n	k	R	d_{min}
2	3	1	1/3	3
3	7	4	4/7	3
4	15	11	11/15	3
5	31	26	26/31	3
6	63	57	57/63	3
r	$2^r - 1$	$2^r - r - 1$	k/n	3

Example: (7,4) Hamming code

The $(7, 4, 3)_2$ Hamming code consists of 16 code words of length 7:

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

Hard-Decision Decoding and Soft-Decision Decoding

We distinguish between

- **hard-decision (HD) decoding** (hard-input decoding) and
- **soft-decision (SD) decoding** (soft-input decoding),

depending whether the received symbols are within the set of the transmitted symbols or not. (The received symbols are the components of the received words.)

"Hard-decision decoding":
Let $\mathbf{y} := \mathbf{x} \oplus \mathbf{e}$, where the error components e_i, $i = 0, 1, \ldots, n-1$, are over the symbol alphabet and where the addition is done modulo q.

"Hard-decision decoding rule": $d_H(\mathbf{y}, \hat{\mathbf{x}}) \leq d_H(\mathbf{y}, \tilde{\mathbf{x}})$ for all $\tilde{\mathbf{x}} \in \mathcal{C}$.

"Soft-decision decoding":
Let $\mathbf{y} := \mathbf{x} + \mathbf{n}$, where $n_i \in \mathbb{R}$, $i = 0, 1, \ldots, n-1$, and the addition is defined over $\mathbb{R}$.

"Soft-decision decoding rule": $||\mathbf{y} - \hat{\mathbf{x}}|| \leq ||\mathbf{y} - \tilde{\mathbf{x}}||$ for all $\tilde{\mathbf{x}} \in \mathcal{C}$.

Hard-Decision Decoding

Example for the $(7, 4, 3)_2$ Hamming code:

Info word: $\mathbf{u} = [0010]$
Code word: $\mathbf{x} = [0010110]$
Error word: $\mathbf{e} = [0010000]$
Received word: $\mathbf{y} = [0000110]$
Decoded code word: $\hat{\mathbf{x}} = [0010110]$ $(*)$
Decoded info word: $\hat{\mathbf{u}} = [0010]$

$(*)$ This code word has the smallest Hamming distance with respect to the received word: $d_H(\mathbf{y}, \hat{\mathbf{x}}) = 1$, $d_H(\mathbf{y}, \tilde{\mathbf{x}}) > 1 \ \forall \ \tilde{\mathbf{x}} \neq \hat{\mathbf{x}}$. According to this example, we recognize that for systematic codes the parity check symbols should not be eliminated **before** decoding takes place.

Soft-Decision Decoding

In soft-decision decoding, usually the squared **Euclidean distance** is used:

$$\sum_{i=0}^{n-1}(y_i - \hat{x}_i)^2 \leq \sum_{i=0}^{n-1}(y_i - \tilde{x}_i)^2.$$

Example 1: $(7,4,3)_2$ Hamming code, binary antipodal modulation
("0" $\rightarrow +1$, "1" $\rightarrow -1$)

Info word: $\mathbf{u} = [0010]$
Transmitted sequence: $\mathbf{x} = [+1.0, +1.0, -1.0, +1.0, -1.0, -1.0, +1.0]$
Noise sequence: $\mathbf{n} = [+0.2, -0.3, +1.1, +0.2, -0.1, -0.2, +0.3]$
Received sequence: $\mathbf{y} = [+1.2, +0.7, +0.1, +1.2, -1.1, -1.2, +1.3]$
Decoded code word: $\hat{\mathbf{x}} = [+1.0, +1.0, -1.0, +1.0, -1.0, -1.0, +1.0]$ (*)
Decoded info word: $\hat{\mathbf{u}} = [0010]$

(*) This modulated code word has the smallest squared Euclidean distance with respect to the received sequence. Due to "analog decoding" often more than $t' = d_{min} - 1$ errors can be detected and more than $t = \lfloor (d_{min} - 1)/2 \rfloor$ errors can be corrected.

Soft-Decision Decoding

Example 2: $(3,2,2)_2$ SPC code, binary antipodal modulation ("0" $\rightarrow +1$, "1" $\rightarrow -1$)

Info word: $\mathbf{u} = [01]$
Transmitted sequence: $\mathbf{x} = [+1.0, -1.0, -1.0]$
Noise sequence: $\mathbf{n} = [+0.2, -0.3, +1.1]$
Received sequence: $\mathbf{y} = [+1.2, -1.3, +0.1]$

a) Exhaustive search: Decoding rule as described above.
b) Wagner decoding: Is parity check equation fulfilled?

- If yes: hard-decision decoding
- If no: hard-decision decoding after flipping the least reliable bit

Result of a) and b):

Decoded info word: $\hat{\mathbf{u}} = [01]$ $\Rightarrow$ The error has been corrected, although $t = 0$!

Error Probability

Definition (**word error probability**): By definition, the word error probability P_w is the probability that the decoded info word deviates from the transmitted info word in at least one symbol:

$$P_w := P(\hat{\mathbf{u}} \neq \mathbf{u}) = P(\hat{\mathbf{x}} \neq \mathbf{x}).$$

The actual number of bit errors (in case of binary codes) or the number of symbol errors (in case of non-binary codes) is not taken into account.

Definition (**symbol error probability**): By definition, the symbol error probability P_s is the average probability of a symbol error:

$$P_s := \frac{1}{k} \sum_{i=0}^{k-1} P(\hat{u}_i \neq u_i).$$

For binary codes ($q = 2$), the **symbol error probability**, P_s, is equal to the **bit error probability**, P_b.

Since the number of symbol errors per word error is between 1 and k,
$P_w/k \leq P_s \leq P_w$ holds.

Decoding Principles (for the example of block codes)

Let $\tilde{\mathbf{x}}$ be hypotheses of the code words of a block code $\mathcal{C}$ and let $\mathbf{y}$ be the received word. Furthermore, let $P(\tilde{\mathbf{x}}|\mathbf{y})$ be the conditional probability of $\tilde{\mathbf{x}}$ given $\mathbf{y}$.

Definition (**maximum a posteriori (MAP) decoding**):
The MAP rule is as follows: Select the code word $\hat{\mathbf{x}}$ such that for a given received word $\mathbf{y}$

$$\text{HD \& SD} : P(\hat{\mathbf{x}}|\mathbf{y}) \geq P(\tilde{\mathbf{x}}|\mathbf{y}) \quad \forall \quad \tilde{\mathbf{x}} \in \mathcal{C}.$$

Definition (**maximum-likelihood (ML) decoding**):
The ML rule is as follows: Select the code word $\hat{\mathbf{x}}$ such that for a given received word $\mathbf{y}$

$$\text{HD} \;:\; P(\mathbf{y}|\hat{\mathbf{x}}) \geq P(\mathbf{y}|\tilde{\mathbf{x}}) \quad \forall \quad \tilde{\mathbf{x}} \in \mathcal{C}$$
$$\text{SD} \;:\; p(\mathbf{y}|\hat{\mathbf{x}}) \geq p(\mathbf{y}|\tilde{\mathbf{x}}) \quad \forall \quad \tilde{\mathbf{x}} \in \mathcal{C}.$$

In both rules, all q^k possible code words $\tilde{\mathbf{x}}$ will be compared with the received word $\mathbf{y}$, at least conceptionally. The most likely code word (MAP rule) or the code word corresponding to the most likely received word (ML rule) is finally selected.

Decoding Principles (for the example of block codes)

Theorem: MAP and ML decoding are identical, if all q^k code words $\mathbf{x}$ are equiprobable.

Proof: According to Bayes' rule $P(\mathbf{x}|\mathbf{y}) = P(\mathbf{y}|\mathbf{x}) \cdot P(\mathbf{x})/P(\mathbf{y})$ or $P(\mathbf{x}|\mathbf{y}) = p(\mathbf{y}|\mathbf{x}) \cdot P(\mathbf{x})/p(\mathbf{y})$. Since the denominator on the right hand side is irrelevant with respect to a maximization, both decoding rules differ only in the use of a priori information $P(\mathbf{x})$. Therefore, for equiprobable code words both decoding rules are identical. q.e.d.

Definition (**bounded minimum distance (BMD) decoding**):
Let us given decoding spheres of radius $r \leq \lfloor (d_{min} - 1)/2 \rfloor$. Only those received words $\mathbf{y}$ will be considered for decoding, which are within a decoding sphere. In this case, the code word in the center of the corresponding decoding sphere is selected. For all received words $\mathbf{y}$ which are outside of the decoding spheres, an erasure is declared.

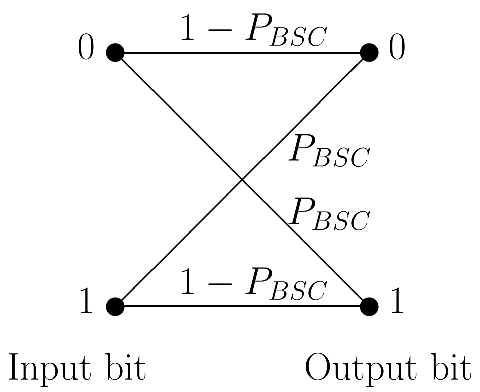
Definition (**perfect codes**):
A code is called perfect, if the ML and the BMD decoding principle are identical.

Binary Symmetric Channel, BSC

A so-called binary symmetric channel (BSC) models independent bit errors e_i:

$$y_i = x_i \oplus e_i, \qquad i = 0, 1, \ldots, n-1,$$

where $x_i, e_i, y_i \in \{0, 1\}$. The corresponding bit error probability is denoted as P_{BSC}.



Word Error Probability for the BSC

Now we consider a coded transmission system with hard-decision decoding given the example of a BSC with error probability P_{BSC}:

Theorem: For linear $(n, k, d_{min})_q$ block codes the word error probability for ML decoding can be approximated by the upper bound

$$P_w \leq 1 - \sum_{r=0}^{t} \binom{n}{r} P_{BSC}^r (1 - P_{BSC})^{n-r}.$$

For perfect codes (e.g., for binary Hamming codes) this bound is exact.
In this case ML and BMD decoding are identical.

Word Error Probability for the BSC

Proof:

$$P_w = 1 - P(\text{correct decoding}).$$

Assumption: Decoding is correct, if at most t errors occur:

$$P_w = 1 - P(w_H(\mathbf{e}) \leq t).$$

This corresponds to bounded minimum distance (BMD) decoding. Therefore,

$$P_w = 1 - \sum_{r=0}^{t} P(w_H(\mathbf{e}) = r).$$

Since ML decoding may be better than BMD, we obtain

$$P_w \leq 1 - \sum_{r=0}^{t} P(w_H(\mathbf{e}) = r).$$

The number of errors in a word of length n is binomial distributed:

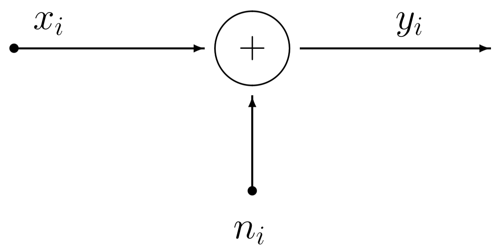
$$P(w_H(\mathbf{e}) = r) = \binom{n}{r} P_{BSC}^r (1 - P_{BSC})^{n-r}. \qquad \text{q.e.d.}$$

Additive White Gaussian Noise (AWGN) Channel

In an additive white Gaussian noise (AWGN) channel model, zero mean white Gaussian noise is linearly added to the channel input values x_i:

$$y_i = x_i + n_i, \qquad i = 0, 1, \ldots, n-1,$$

where $x_i, n_i, y_i \in I\!R$. The channel inputs x_i and the noise samples n_i are assumed to be statistically independent.
(Remark: In this lecture we only consider the discrete-time AWGN channel model.)



Additive White Gaussian Noise (AWGN) Channel

The noise n_i ($n_i \in I\!R$) is:

- **additive**: $y_i = x_i + n_i, \quad x_i,\, y_i \in I\!R, \quad E\{x_i^2\} = 1, \quad$ e.g., $x_i \in \{-1, +1\}$
- **white**: $E\{n_i \cdot n_j\} = 0$ if $i \neq j$
- **Gaussian distributed**: $p(n_i) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(n_i-\mu)^2}{2\sigma^2}}$ $\left(\text{i.e., } p(y_i|x_i) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(y_i-x_i)^2}{2\sigma^2}}\right)$

  with zero mean $\mu = E\{n_i\} = 0$
  and variance $\sigma^2 = E\{n_i^2\} = \frac{1}{2E_s/N_0}$,

    where E_s is the energy per symbol
    and N_0 is the single-sided noise power density: $N_0 = kT_{eff}$
    (k is here and only here the Boltzmann constant).

  E_s/N_0 is the so-called **signal/noise ratio** (SNR).
  The signal/noise ratio per info bit is $E_b/N_0 = R^{-1}E_s/N_0$.

(Note that white, Gaussian distributed random variables are statistically independent.)

Bit Error Probability for the AWGN Channel (uncoded)

For an uncoded transmission system ($R = 1$) with binary, identically distributed symbols $x_i \in \{+1, -1\}$ the bit error probability P_b of the AWGN channel can be calculated as follows:

$$P_b = \frac{1}{2}\int_{-\infty}^{0} p(y_i|x_i = +1)\; dy_i + \frac{1}{2}\int_{0}^{\infty} p(y_i|x_i = -1)\; dy_i = \int_{-\infty}^{0} p(y_i|x_i = +1)\; dy_i.$$

With

$$p(n_i) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{n_i^2}{2\sigma^2}}, \qquad \sigma^2 = \frac{N_0}{2E_s},$$

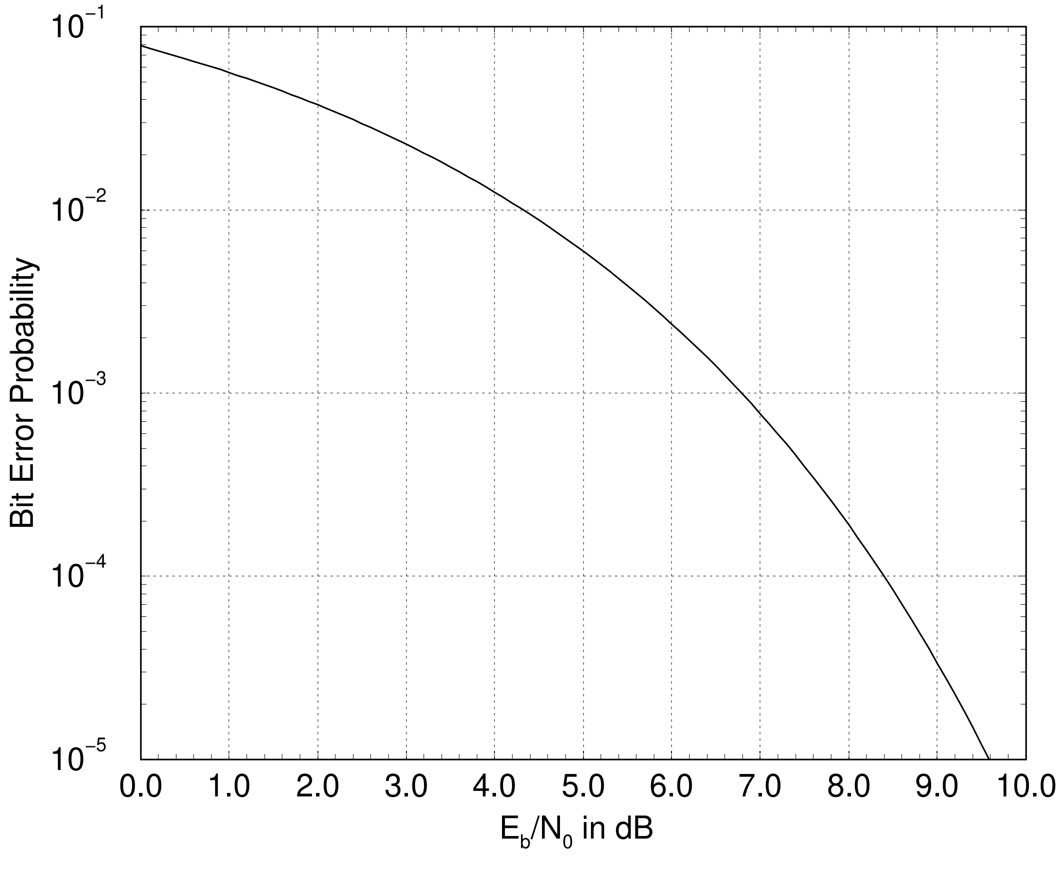
follows

$$p(y_i|x_i) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(y_i - x_i)^2}{2\sigma^2}}.$$

After substitution we obtain the final result

$$P_b = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\infty}^{0} e^{-\frac{(y_i-1)^2}{2\sigma^2}}\; dy_i := \frac{1}{2}\,\mathrm{erfc}\sqrt{\frac{E_s}{N_0}}, \text{ where } E_s = E_b.$$

Bit Error Probability for the AWGN Channel (uncoded)



Word Error Probability for the AWGN Channel

Now we consider a coded transmission system with soft-decision decoding given the example of an AWGN channel model with signal/noise ratio E_s/N_0:

Theorem: The word error probability for $(n, k, d_{min})_2$ block codes and binary antipodal transmission ($x_i \in \{+1, -1\}$) with ML decoding can be approximated by a lower and an upper bound:

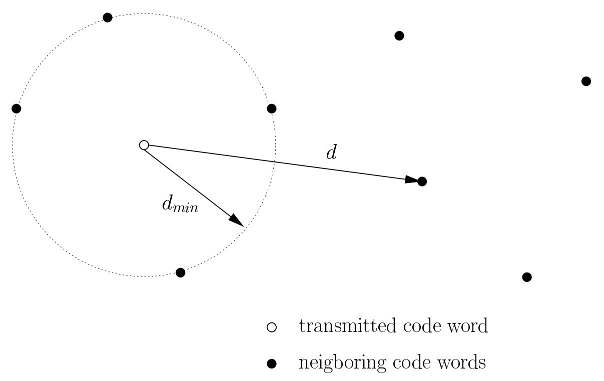
$$\frac{1}{2}\,\mathrm{erfc}\,\sqrt{d_{min}\frac{E_s}{N_0}} \le P_w \le \frac{1}{2}\sum_{d=d_{min}}^{n} a_d\,\mathrm{erfc}\,\sqrt{d\frac{E_s}{N_0}},$$

where $E_s/N_0 = RE_b/N_0$ and a_d is the number of all code words with Hamming distance d with respect to the transmitted code word.
For linear codes a_d is equal to the number of code words with Hamming weight d.
For linear codes the inequality holds for all q^k code words.

Example: $a_{d_{min}} = a_3 = 7$ for the $(7, 4, 3)_2$ Hamming code.

Asymptotically, i.e. for large E_s/N_0, the bounds are exact.

Word Error Probability for the AWGN Channel



Sketch of proof: For binary antipodal transmission, the probability that a code word with Hamming distance d is selected erroneously, is given by

$$P_d = \frac{1}{2}\,\mathrm{erfc}\,\sqrt{d\frac{E_s}{N_0}}$$

("two-word error probability").

The **lower bound** considers only one code word with Hamming distance d_{min}.

The **upper bound** considers all erroneous code words.

Coding Gain

Definition (**coding gain**): The coding gain is the difference [in decibel] between the signal/noise ratio for uncoded transmission and for coded transmission, respectively, given the same bit error probability (e.g., 10^{-5}).

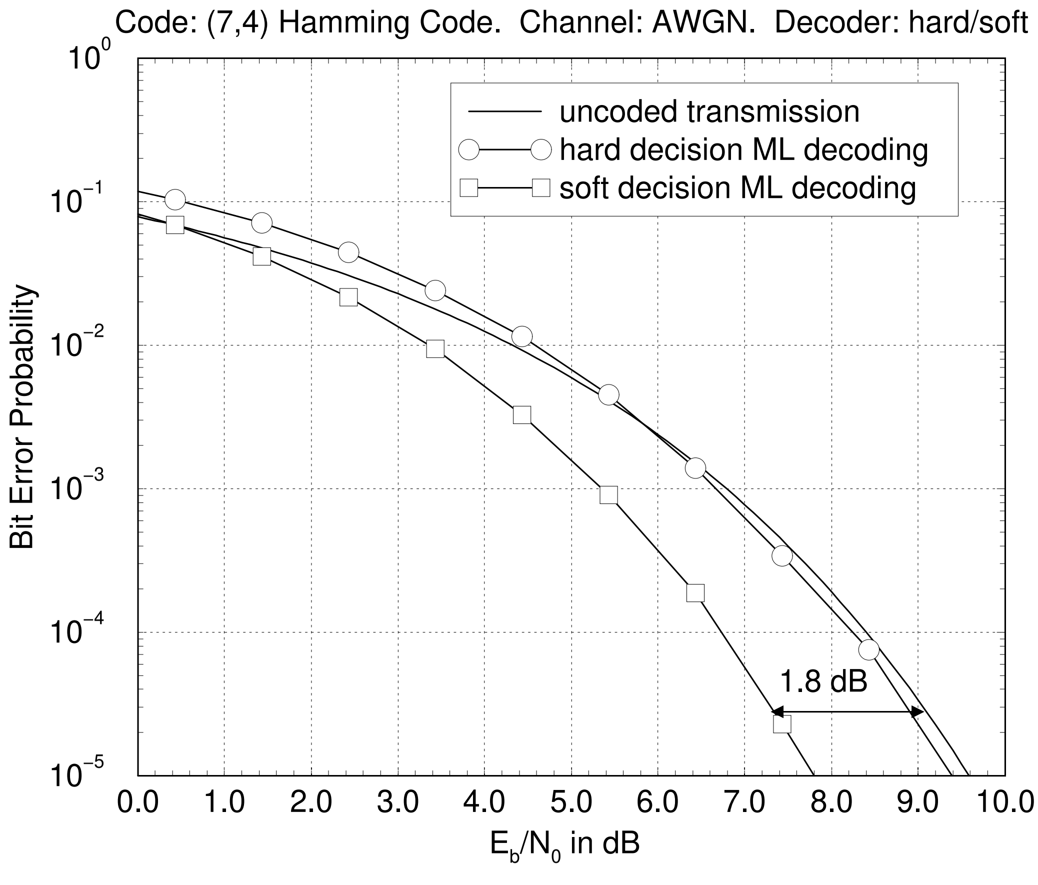
The **asymptotic coding gain** (asymptotic means: for large SNR) for the AWGN channel model and soft-decision decoding is given by

$$G_{asy} := 10\ \log_{10}(d_{min} \cdot R)\ \text{dB}.$$

$\Rightarrow$ With repetition codes, on the AWGN channel model no coding gain is possible: $d_{min} \cdot R = n \cdot 1/n = 1$ and $a_{d_{min}} = 1$.

$\Rightarrow$ Desirable is a large minimum distance d_{min} at a high code rate R, as well as a small number of code words with Hamming weight d_{min}.

Bit Error Probability for the $(7,4,3)_2$ Hamming Code



Generator Matrix

Theorem (**generator matrix**): The q^k code words of any linear $(n, k, d_{min})_q$ block code $\mathcal{C}$ can be generated by a vector/matrix multiplication:

$$\mathbf{x} := \mathbf{u} \cdot \mathbf{G},$$

where $\mathbf{u} = [u_0, u_1, \ldots, u_{k-1}]$ is a $(1 \times k)$ info word and $\mathbf{x} = [x_0, x_1, \ldots, x_{n-1}]$ the corresponding $(1 \times n)$ code word.

The $(k \times n)$ matrix $\mathbf{G}$ is called generator matrix.

For systematic linear block codes the generator matrix can be written as

$$\mathbf{G} = [\mathbf{I}_k | \mathbf{P}],$$

where $\mathbf{I}_k$ is the $(k \times k)$ identity matrix and $\mathbf{P}$ a $(k \times (n-k))$ matrix, which represents the parity check symbols.

Parity Check Matrix

Theorem (**parity check matrix**): For any linear $(n, k, d_{min})_q$ block code $\mathcal{C}$ an $((n-k) \times n)$ matrix $\mathbf{H}$ exists such that

$$\mathbf{x} \cdot \mathbf{H}^T := \mathbf{0},$$

if $\mathbf{x}$ is a code word of $\mathcal{C}$.
The all-zero word on the right hand side consists of $n - k$ elements.

The matrix $\mathbf{H}$ is called parity check matrix.

For systematic linear block codes the parity check matrix can be written as

$$\mathbf{H} = [-\mathbf{P}^T | \mathbf{I}_{n-k}],$$

where $\mathbf{I}_{n-k}$ is the $((n-k) \times (n-k))$ identity matrix and $\mathbf{P}^T$ an $((n-k) \times k)$ matrix representing the parity check symbols.

Dual Codes

Definition (**dual code**): The code $\mathcal{C}^\perp$ with generator matrix $\mathbf{G}^\perp$, where $\mathbf{G}^\perp$ is equal to the parity check matrix $\mathbf{H}$ of a code $\mathcal{C}$, is called the dual code $\mathcal{C}^\perp$ of code $\mathcal{C}$.

For binary block codes the **scalar product** leads to the following result:

$$\forall\, \mathbf{x} \in \mathcal{C}, \mathbf{y} \in \mathcal{C}^\perp \quad \Rightarrow \quad \left\langle \mathbf{x} \cdot \mathbf{y}^T \right\rangle := \sum_{i=0}^{n-1} x_i \cdot y_i \bmod 2 = 0.$$

Example:

$$\mathbf{G}^\perp = \mathbf{H} = \begin{bmatrix} 0\ 1\ 1\ 1 & 1\ 0\ 0 \\ 1\ 0\ 1\ 1 & 0\ 1\ 0 \\ 1\ 1\ 0\ 1 & 0\ 0\ 1 \end{bmatrix}, \qquad \mathbf{H}^\perp = \mathbf{G} = \begin{bmatrix} 1\ 0\ 0\ 0 & 0\ 1\ 1 \\ 0\ 1\ 0\ 0 & 1\ 0\ 1 \\ 0\ 0\ 1\ 0 & 1\ 1\ 0 \\ 0\ 0\ 0\ 1 & 1\ 1\ 1 \end{bmatrix}$$

Syndrome and Syndrome Decoding

Definition (**syndrome**): The syndrome $\mathbf{s}$ of an $(n, k, d_{min})_q$ block codes is defined as follows:

$$\mathbf{s} := \mathbf{y} \cdot \mathbf{H}^T.$$

The $n - k$ components of $\mathbf{s}$ are zero, if $\mathbf{y}$ is a code word.

Note that $\mathbf{s} = \mathbf{y} \cdot \mathbf{H}^T = (\mathbf{x} \oplus \mathbf{e}) \cdot \mathbf{H}^T = \mathbf{x} \cdot \mathbf{H}^T \oplus \mathbf{e} \cdot \mathbf{H}^T = \mathbf{e} \cdot \mathbf{H}^T$.

Syndrome decoding:

- Construct syndrome table (with q^{n-k} rows):

$$\hat{\mathbf{e}} = \arg \min_{\mathbf{e}:\ \mathbf{e}\,\mathbf{H}^T = \mathbf{s}} w_H(\mathbf{e})$$

- For each received word $\mathbf{y}$ compute the syndrome $\mathbf{s} = \mathbf{y} \cdot \mathbf{H}^T$
- Search syndrome $\mathbf{s}$ in table and hence obtain $\hat{\mathbf{e}}$
- Compute $\hat{\mathbf{x}} = \mathbf{y} \ominus \hat{\mathbf{e}}$

Cyclic Block Codes

Definition (**cyclic block code**): A linear $(n, k, d_{min})_q$ block code $\mathcal{C}$ is called cyclic, if each cyclic shift of a code word yields a valid code word:

$$[x_0, x_1, \ldots, x_{n-1}] \in \mathcal{C} \quad \Rightarrow \quad [x_{n-1}, x_0, \ldots, x_{n-2}] \in \mathcal{C}.$$

Counter example: $(7, 4, 3)_2$ Hamming code:

[0001111] is a code word.

[1111000] is no code word.

Correspondingly, this special Hamming code is not cyclic. (However, by means of a clever exchange and/or addition of the rows and columns of the generator matrix one can always obtain a cyclic Hamming code.)

It is not important whether we shift to the left hand side or to the right hand side, as can easily be seen when considering multiple shifts.

Info Polynomial and Code Polynomial

Each code word of an arbitrary block code $\mathcal{C}$ can be described by a **code polynomial** of degree $\leq n-1$:

$$x(D) := x_0 + x_1 D + \ldots + x_{n-2} D^{n-2} + x_{n-1} D^{n-1},$$

where $[x_0, \ldots, x_{n-1}]$ represent the n code symbols $x_i \in \{0, \ldots, q-1\}$, $i = 0, \ldots, n-1$.

Example:
The code word [110010] $(n = 6)$ can be represented by the code polynomial $x(D) = 1 + D + D^4$ of degree 4.

Accordingly, we define an **info polynomial** of degree $\leq k-1$:

$$u(D) := u_0 + u_1 D + \ldots + u_{k-2} D^{k-2} + u_{k-1} D^{k-1},$$

where $[u_0, \ldots, u_{k-1}]$ represent the k info symbols $u_i \in \{0, \ldots, q-1\}$, $i = 0, \ldots, k-1$.

Generator Polynomial and Parity Check Polynomial

Theorem (**generator polynomial**): Let $u(D)$ be an info polynomial of degree $\leq k-1$ and $g(D)$ be a so-called generator polynomial of degree $n-k$:

$$g(D) := 1 + g_1 D + \ldots + g_{n-k-1} D^{n-k-1} + 1 D^{n-k},$$

where $g_i \in \{0, \ldots, q-1\}$, $i = 1, \ldots, n-k-1$. The product $u(D) \cdot g(D)$ is a polynomial of degree $\leq n-1$ and corresponds to a code word of a linear $(n, k, d_{min})_q$ block code $\mathcal{C}$: $\quad x(D) = u(D) \cdot g(D) \qquad$ (Note that $\mathcal{C}$ is not necessarily cyclic).

Theorem (**parity check polynomial**): Let $g(D)$ be a generator polynomial of degree $n-k$ of a linear $(n, k, d_{min})_q$ block code $\mathcal{C}$. Then:

$$\mathcal{C} \text{ is cyclic} \qquad \Leftrightarrow \qquad g(D) \text{ is a divisor of } D^n - 1.$$

Therefore a polynomial $h(D) := h_0 + h_1 D + \ldots + h_{k-1} D^{k-1} + 1 D^k$ of degree k exists such that $g(D) \cdot h(D) = D^n - 1$, where $h_i \in \{0, \ldots, q-1\}$, $i = 0, \ldots, k-1$. The polynomial $h(D)$ is called parity check polynomial.

Golay Code

An example of a linear, cyclic block code is the $(23, 12, 7)_2$ **Golay code**.

Its generator polynomial is

$$g(D) = D^{11} + D^9 + D^7 + D^6 + D^5 + D + 1.$$

The corresponding parity check polynomial is

$$h(D) = D^{12} + D^{10} + D^7 + D^4 + D^3 + D^2 + D + 1.$$

Proof:

$$(D^{23}+1) : (D^{11}+D^9+D^7+D^6+D^5+D+1) = D^{12}+D^{10}+D^7+D^4+D^3+D^2+D+1.$$

Remark 1: For binary codes ($q = 2$) $D^n - 1 = D^n + 1$.

Remark 2: The $(23, 12, 7)_2$ Golay code is a perfect code.

CRC Code

Definition (**CRC code**): A cyclic $(2^r-1, 2^r-r-2, 4)_2$ code is called a cyclic redundancy check code, if the generator polynomial is of the form

$$g(D) := (1+D)\cdot p(D),$$

where $p(D)$ is a primitive polynomial of degree $r \geq 3$.

Examples of primitive polynomials:

Degree r	Primitive polynomial $p(D)$
1	$D+1$
2	D^2+D+1
3	D^3+D+1
4	D^4+D+1
5	D^5+D^2+1
6	D^6+D+1
7	D^7+D+1
8	$D^8+D^6+D^5+D^4+1$

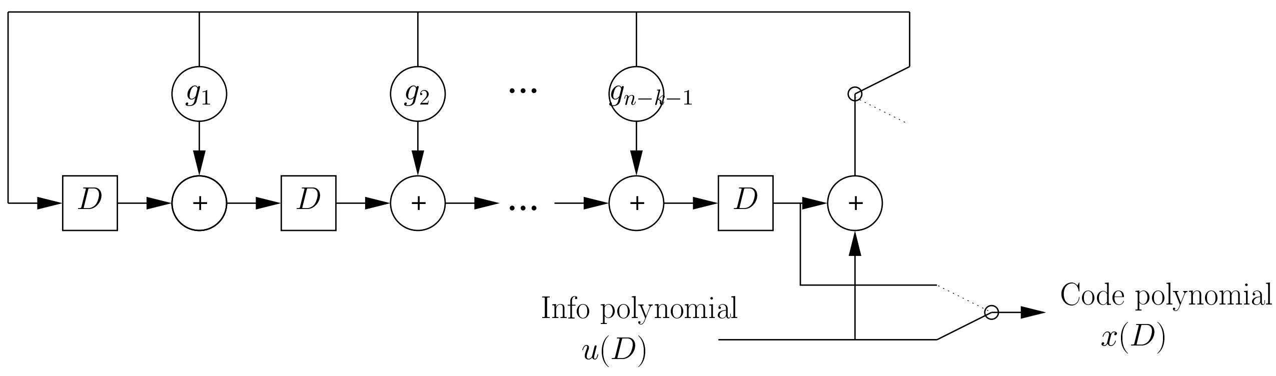
CCITT has standardized the following CRC codes, among others, for applications in the Open Systems Interconnection (OSI) data security layer:

$D^{16}+D^{12}+D^5+1 = (D+1)(D^{15}+D^{14}+D^{13}+D^{12}+D^4+D^3+D^2+D+1)$

$D^{12}+D^{11}+D^3+D^2+D+1 = (D+1)(D^{11}+D^2+1)$

$D^8+D^2+D+1 = (D+1)(D^7+D^6+D^5+D^4+D^3+D^2+1)$

Circuit for the Generation of Systematic, Cyclic Block Codes with Generator Polynomial $g(D)$



Reed-Solomon Codes

RS codes are characterized by the following parameters:

$$\begin{aligned} n &= p^m - 1 \\ n - k &= d_{min} - 1 = 2t \\ d_{min} &= 2t + 1 \\ q &= p^m, \end{aligned}$$

where m and t are arbitrary positive integer numbers and p is a prime number.
Often, $p = 2$ and $m = 8$ are chosen ($q = 256$), i.e., one symbol corresponds to one byte.

Example 1: $(255, 127, 129)_{256}$ RS code.
This $R \approx 1/2$ code can correct 64 bytes for sure.
The number of code words is $q^k \approx 10^{308}$.

Example 2: $(255, 239, 17)_{256}$ RS code.
This $R \approx 0.94$ code can correct 8 bytes for sure.
This code is often used as an outer code in concatenated coding systems.

Bose-Chaudhuri-Hocquenghem Codes

Binary BCH codes are characterized by the following parameters:

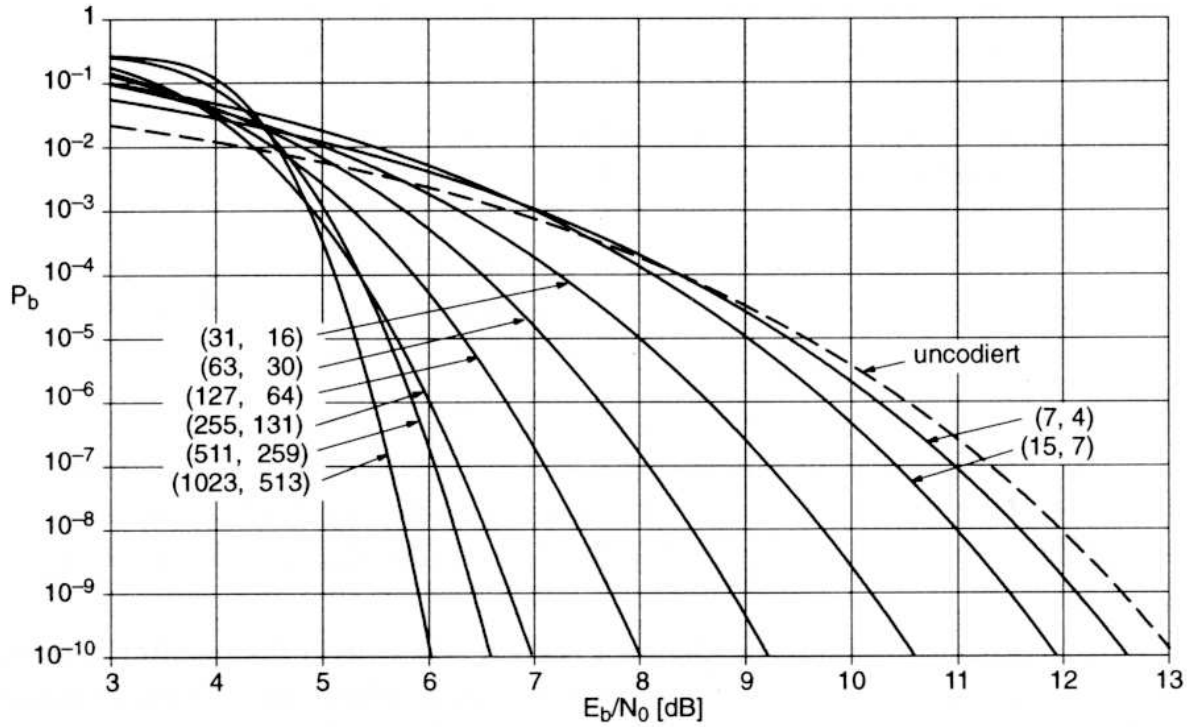
$$\begin{aligned} n &= 2^{m'} - 1 \\ n - k &\leq m' \cdot t \\ d_{min} &= 2t + 1 \\ q &= 2, \end{aligned}$$

where m' ($m' \geq 3$) and t are arbitrary positive integer numbers.

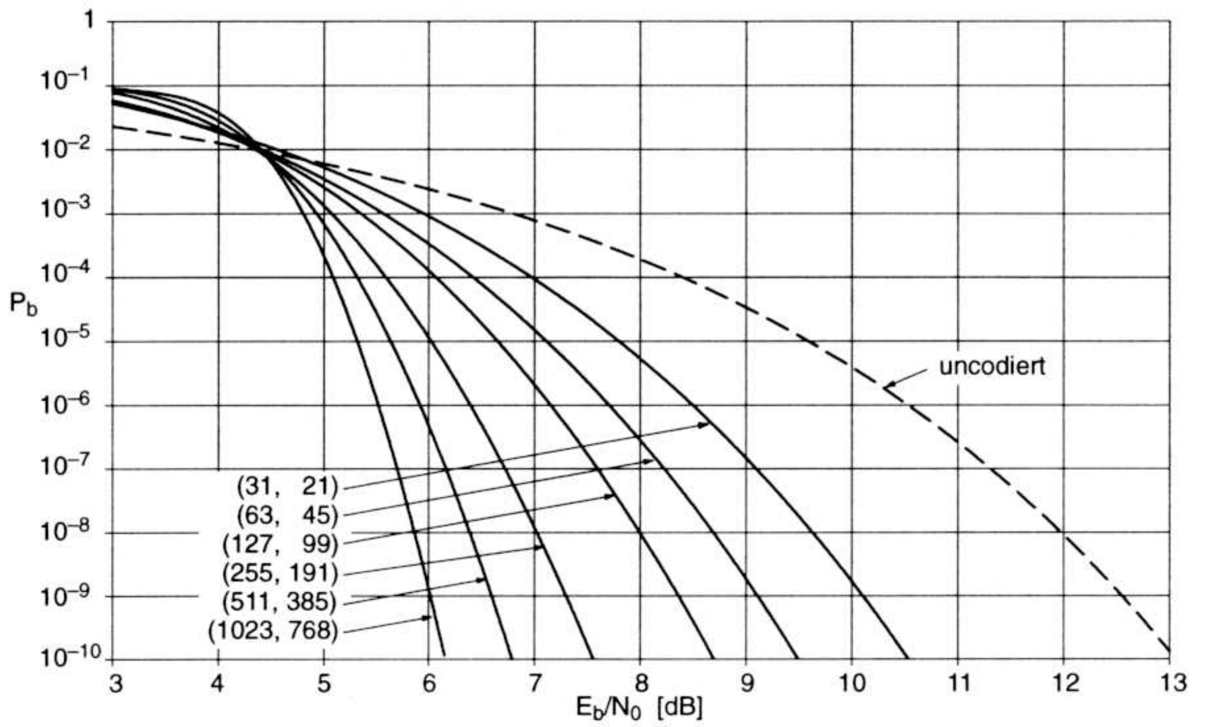
The code can be derived for a given minimum distance d_{min}.

The corresponding generator polynomials are tabulated.

Bit Error Probability for Binary $R \approx 1/2$ BCH Codes



Bit Error Probability for Binary $R \approx 3/4$ BCH Codes



Chapter 2: Convolutional Codes

- Definition of convolutional codes, memory length
- Shift register representation, state representation, trellis diagram
- Decoding of convolutional codes, Viterbi algorithm
- Error path, free distance, coding gain, distance spectrum
- Bit error probability for ML decoding, union bound
- Polynomial representation of convolutional codes
- Generator polynomial, info polynomial, generator matrix
- Recursive convolutional encoders
- Zero tailing, tail biting
- Punctured convolutional codes, unequal error protection
- Computation of the distance spectrum by means of the modified state diagram
- Catastrophic convolutional encoders

Coded Transmission System with Convolutional Codes

u_k: Info bits,	$u_k \in \{0,1\}$
x_n: Code bits,	$x_n \in \{0,1\}$
y_n: Received values,	hard-decision dec.: $y_n \in \{0,1\}$, soft-decision dec.: $y_n \in \mathbb{R}$
$\hat{u}_k$: Decoded info bits,	$\hat{u}_k \in \{0,1\}$
k: Index before encoder	
n: Index after encoder	

Convolutional Codes

Convolutional codes are able to encode the info bits continuously.

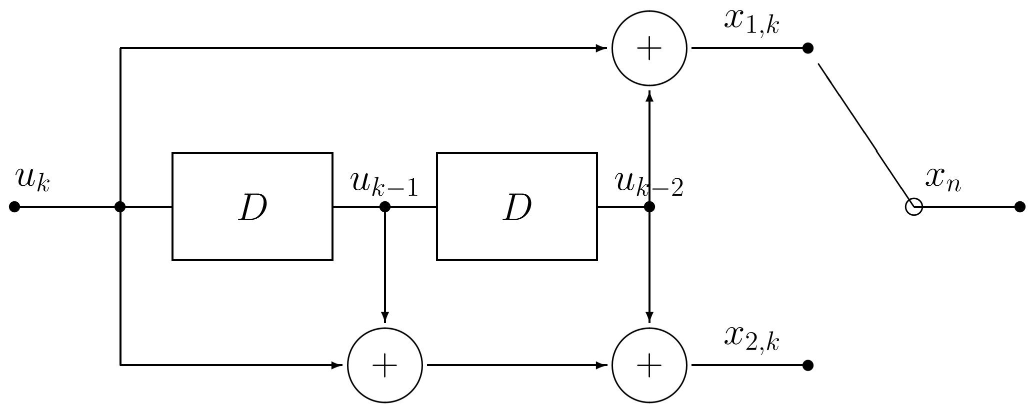
We restrict ourselves to binary block codes. The ratio between the number of info bits and the number of code bits is called coding rate R.

In practice, information is typically transmitted block-wise, rather than continuously.

The number of info bits per block is denoted as K, i.e., the index before the encoder is $0 \leq k \leq K-1$.

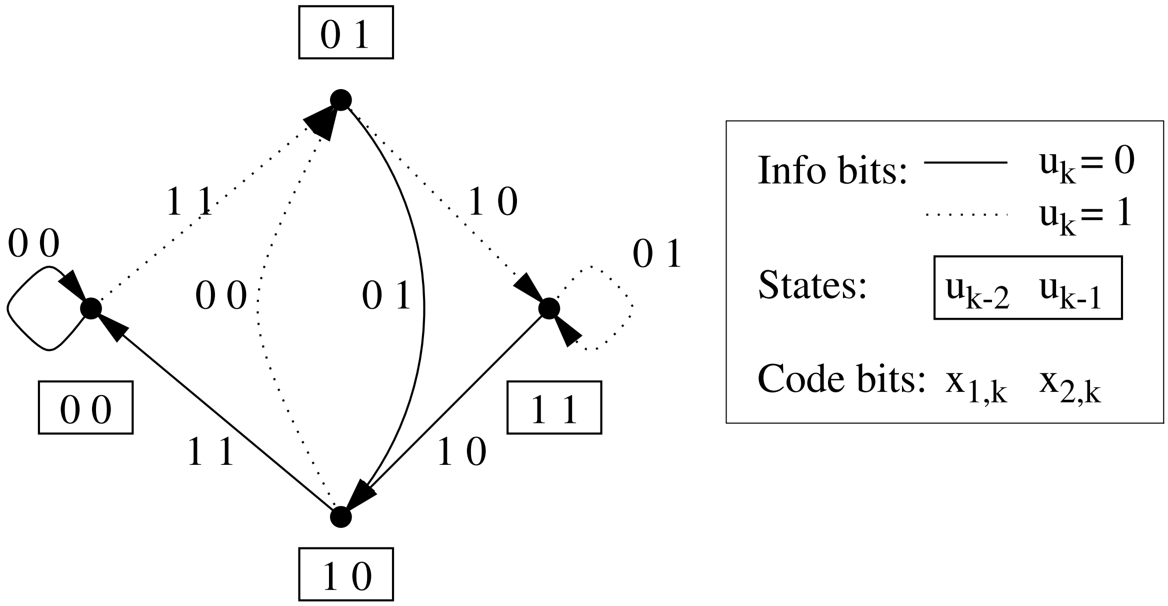
The number of coded bits per block is denoted as N, i.e., the index after the encoder is $0 \leq n \leq N-1$.

Shift Register Representation of a Binary, Non-Recursive $R = 1/2$ Convolutional Encoder with 4 States

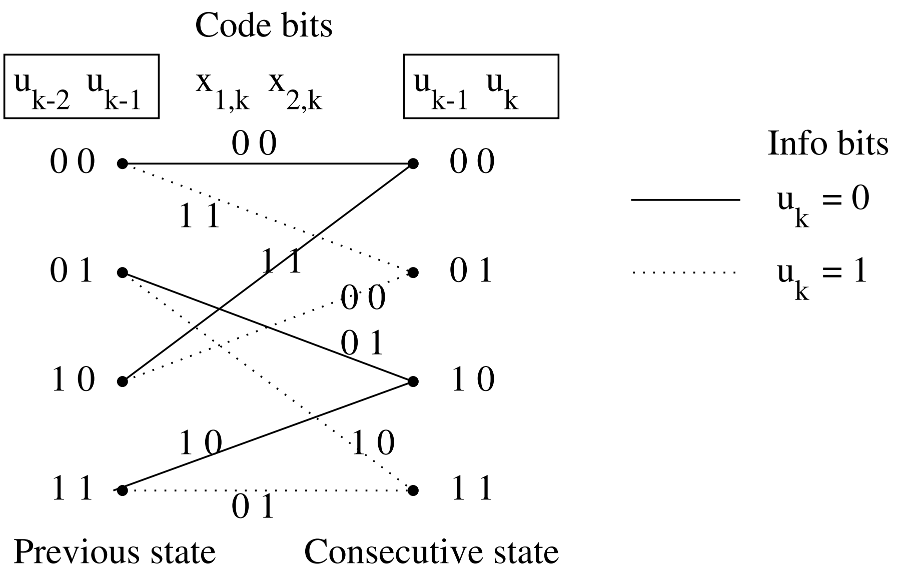


Memory length: $\nu = 2$
Number of states: $S = 2^{\nu}$

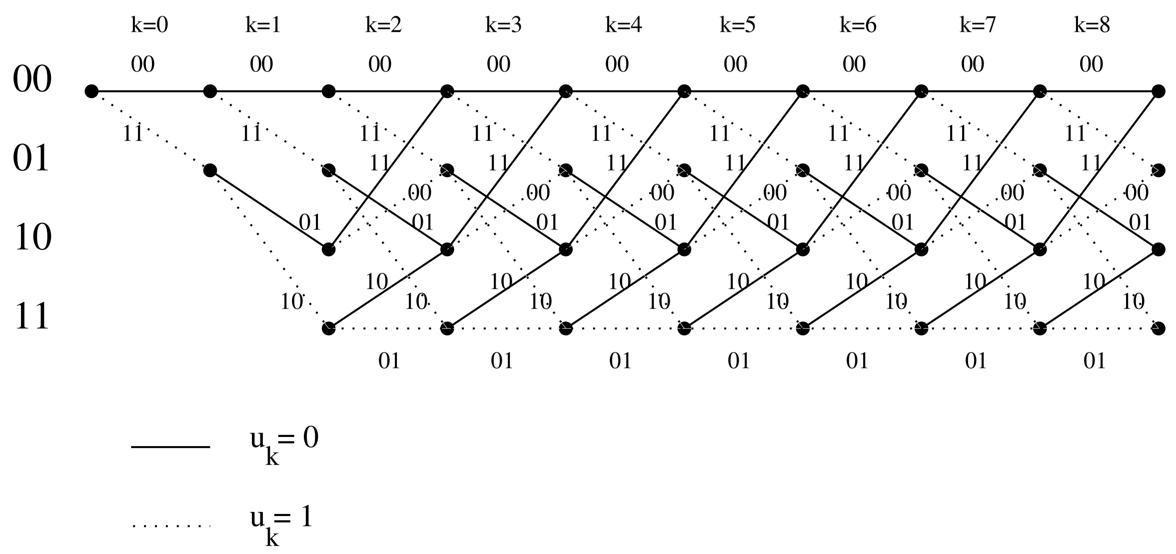
State Diagram of a Binary, Non-Recursive $R = 1/2$ Convolutional Encoder with 4 States



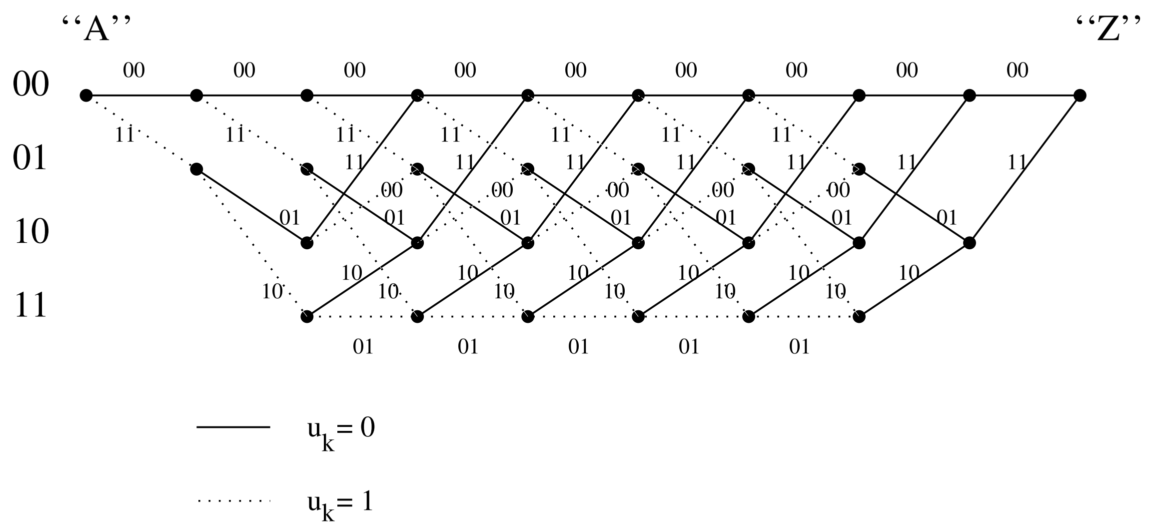
Trellis Segment of a Binary, Non-Recursive $R = 1/2$ Convolutional Encoder with 4 States



Trellis Diagram of a Binary, Non-Recursive $R = 1/2$ Convolutional Encoder with 4 States



Terminated Trellis Diagram of a Binary, Non-Recursive $R = 7/18 \approx 1/2$ Convolutional Encoder with 4 States



Decoding of Convolutional Codes

A trellis diagram is a so-called **directed graph**.

The **optimal decoder** in the sense of maximum-likelihood decoding searches the **most probable sequence** among all possible sequences within the trellis.

Hard-decision decoding: The path with the smallest Hamming distance with respect to the received sequence is selected.

Soft-decision decoding: The path with the smallest squared Euclidean distance with respect to the received sequence is selected.

This is an **optimization strategy**: Which path is best from “A” to “Z”, where path costs correspond to Hamming or squared Euclidean distances.

$\Rightarrow$ **Viterbi algorithm**

Viterbi Algorithm

1. **Initialization**: a) Compute all branch metrics $\lambda_k^{(\rho)}$, $0 \leq k \leq K-1$.
   Example: Euclidean branch metrics:
   $$\lambda_k^{(\rho)} = \sum_i (y_{i,k} - x_{i,k}^{(\rho)})^2, \qquad 0 \leq k \leq K-1,$$
   where $x_{i,k}^{(\rho)}$ are the encoded bits corresponding to the ρ-th path and $y_{i,k} \in \{0,1\}$ (hard decision) respectively $y_{i,k} \in I\!R$ (soft-decision) are the received values.
   b) Initialization of all path metrics Λ_{-1}, where $\Lambda_{-1} = 0$ for the initial state and $\Lambda_{-1} = \infty$ for all other states.
2. **Add-compare-select** operation (ρ and ρ' are concurrent paths):
   - add branch metrics: $\Lambda_k^{(\rho)} = \Lambda_{k-1}^{(\rho)} + \lambda_k^{(\rho)}$ and $\Lambda_k^{(\rho')} = \Lambda_{k-1}^{(\rho')} + \lambda_k^{(\rho')}$
   - compare the path metrics: $\Lambda_k^{(\rho)} <> \Lambda_k^{(\rho')}$?
   - select the better path: $\Lambda_k^{(\rho)}$ or $\Lambda_k^{(\rho')}$ remains
3. If in a terminated trellis diagram the final state is reached, only the ML path remains. This path is traced backwards (**back search** or **trace-back**), in order to obtain the decoded info bits $\hat{u}_k$.

Viterbi Algorithm

Theorem: For the AWGN channel model the Euclidean branch metric λ_k is optimal.
Proof: For the AWGN channel model $y_n = x_n + n_n$, $0 \leq n \leq N-1$, where

$$p_{Y|X}(y_n|x_n) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_n - x_n)^2}{2\sigma^2}}; \qquad \sigma^2 = \frac{1}{2E_s/N_0}.$$

Therefore,

$$p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) = \frac{1}{(2\pi\sigma^2)^{N/2}} \prod_{n=0}^{N-1} e^{-\frac{(y_n - x_n)^2}{2\sigma^2}}.$$

Hence,

$$\begin{aligned} -\log_e p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) &= \frac{N}{2}\log_e(2\pi\sigma^2) + \sum_{n=0}^{N-1} \frac{(y_n - x_n)^2}{2\sigma^2} \\ &= a + b\sum_{n=0}^{N-1}(y_n - x_n)^2, \end{aligned}$$

where a and $b > 0$ are constant factors.

Viterbi Algorithm

The optimal sequence in the sence of maximum-likelihood sequence estimation (MLSE) is

$$\begin{aligned} \hat{\mathbf{u}}_{MLSE} &= \arg\max_{\tilde{\mathbf{x}}} p_{\mathbf{Y}|\tilde{\mathbf{X}}}(\mathbf{y}|\tilde{\mathbf{x}}) \\ &= \arg\max_{\tilde{\mathbf{x}}} \log_e p_{\mathbf{Y}|\tilde{\mathbf{X}}}(\mathbf{y}|\tilde{\mathbf{x}}) \\ &= \arg\min_{\tilde{\mathbf{x}}} \left[-\log_e p_{\mathbf{Y}|\tilde{\mathbf{X}}}(\mathbf{y}|\tilde{\mathbf{x}})\right] \\ &= \arg\min_{\tilde{\mathbf{x}}} \left[a + b\sum_{n=0}^{N-1}(y_n - \tilde{x}_n)^2\right] \\ &= \arg\min_{\tilde{\mathbf{x}}} \sum_{n=0}^{N-1} \underbrace{(y_n - \tilde{x}_n)^2}_{:=\lambda_n} = \arg\min_{\tilde{\mathbf{x}}} \sum_{n=0}^{N-1} \lambda_n \\ &= \arg\min_{\tilde{\mathbf{x}}} \sum_{k=0}^{K-1} \underbrace{\sum_i (y_{i,k} - \tilde{x}_{i,k})^2}_{:=\lambda_k} = \arg\min_{\tilde{\mathbf{x}}} \sum_{k=0}^{K-1} \lambda_k. \end{aligned}$$

Therefore, $\lambda_k = \sum_i (y_{i,k} - \tilde{x}_{i,k})^2$ is the wanted branch metric. q.e.d.

Distance Properties

Convolutional codes are always **linear**, since the coded bits are obtained by a linear operation from the info bits. Due to linearity, we will assume without loss of generality that the all-zeros sequence is transmitted.

Definition (**error path**): An error path is a path, which deviates in the k'-th trellis segment for the first time from the all-zeros sequence and re-merges in the k''-th trellis segment with the all-zeros sequence, where $k'' > k'$.

Responsible for the error correction capability of convolutional codes is the **Hamming weight** of the error paths, rather than the lengths of the error paths.

Definition (**free distance**): The free distance, d_{free}, is equal to the minimum Hamming weight of all error paths.

Example (rate-1/2 convolutional code with $\nu = 2$): The free distance is $d_{free} = 5$.

Rate-1/2 Convolutional Codes with Maximal Free Distance

ν	Taps (octal)	d_{free}	Applications
2	5, 7	5	
3	15, 17	6	
4	23, 35	7	GSM
5	53, 75	8	
6	133, 171	10	DAB, DVB, Sat Comm.

Optimization of convolutional codes: Computer search

Rate-1/3 Convolutional Codes with Maximal Free Distance

ν	Taps (octal)	d_{free}	Applications
2	5, 7, 7	8	
3	13, 15, 17	10	
4	25, 33, 37	12	
5	47, 53, 75	13	
6	133, 145, 171	14	

Distance Spectrum

The number of error paths with distance d with respect to the all-zeros sequence, $d \geq d_{free}$, is denoted as a_d.

The corresponding sum of info bits, which are equal to one, is denoted as c_d.

The list of values a_d versus d and c_d versus d is called the **distance spectrum**.

Example (Rate-1/2 convolutional code with $\nu = 2$):

d	a_d	c_d
5	1	1
6	2	4
7	4	12
8	8	32
9	16	80
d	$2^{d-d_{free}}$	$a_d \cdot (d - d_{free} + 1)$

Bit Error Probability for ML-Decoding

For binary antipodal transmission via an AWGN channel, the bit error probability for soft-decision ML-decoding (or for MAP-decoding without a-priori information) can be lower and upper bounded as follows:

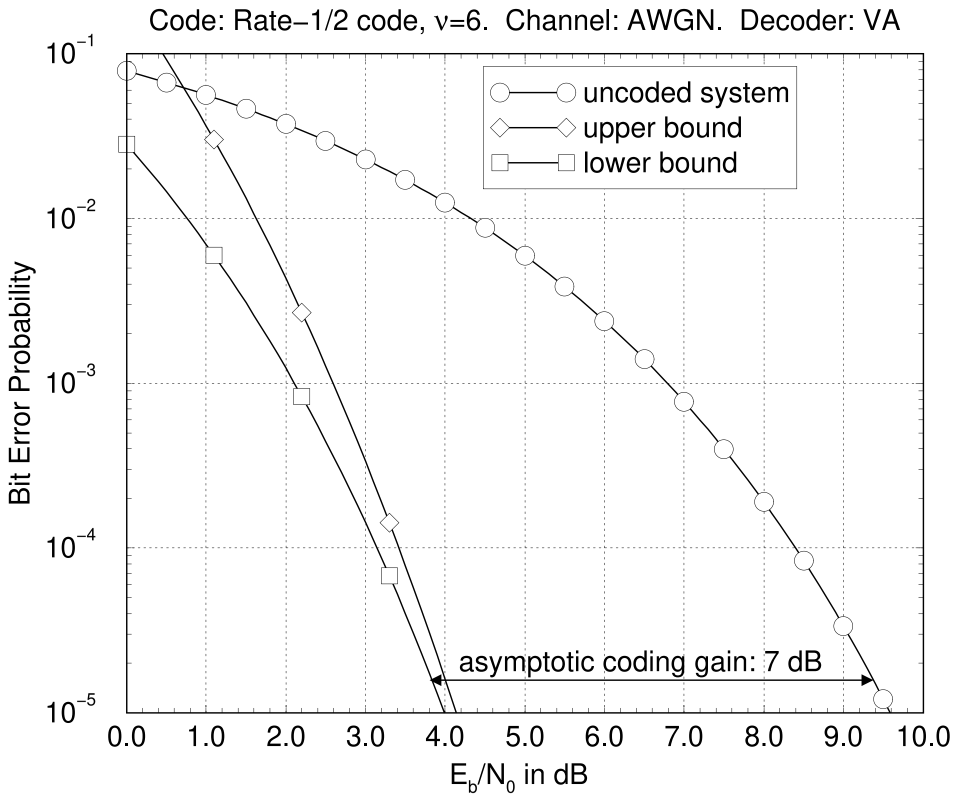
$$\frac{1}{2}\,\mathrm{erfc}\sqrt{d_{free} R \frac{E_b}{N_0}} \;\leq\; P_b \;\leq\; \frac{1}{2}\sum_{d=d_{free}}^{\infty} c_d\,\mathrm{erfc}\sqrt{dR\frac{E_b}{N_0}},$$

where $RE_b = E_s$.

The lower bound takes only the error event with the smallest Hamming distance into account, whereas the upper bound takes all error events into account (**union bound**).

For large E_b/N_0 both curves merge.

Bounds on the Bit Error Probability for the Rate-1/2 Convolutional Code with 64 States



Coding Gain

Definition (**coding gain**): The coding gain is the difference (in decibel) between the necessary E_b/N_0 for uncoded transmission and the necessary E_b/N_0 for coded transmission, in order to obtain the same bit error probability.

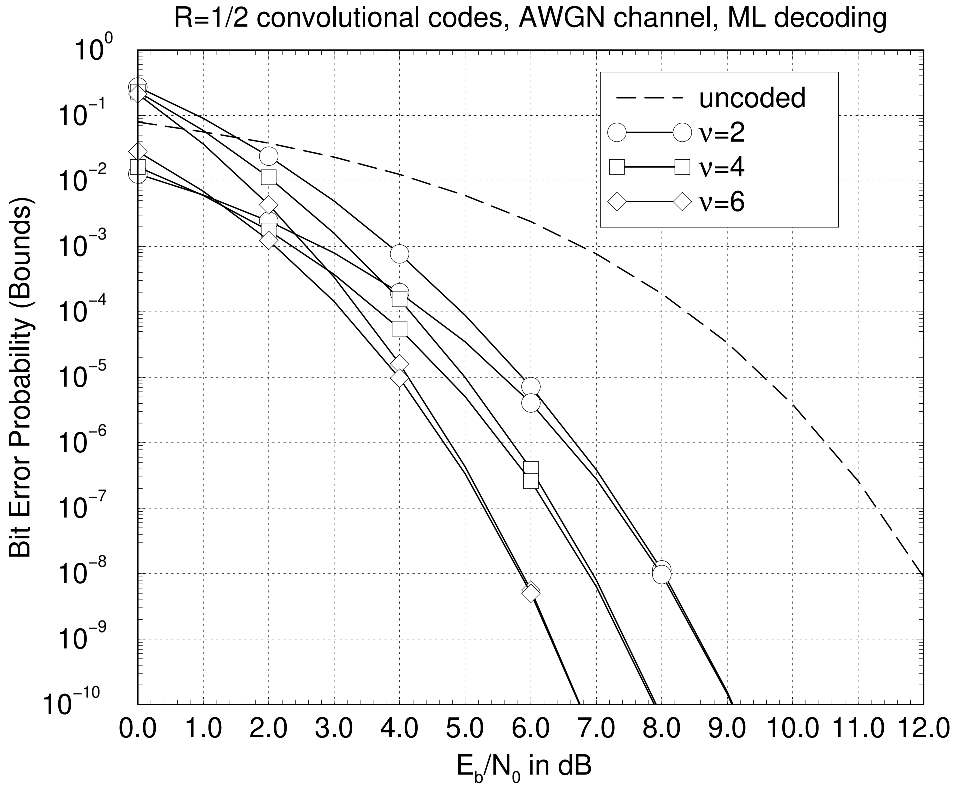
For convolutional codes, the AWGN channel, and ML decoding, the asymptotic coding gain (i.e., for $E_b/N_0 \to \infty$) is

$$G_{asy} = 10 \ \log_{10} \ (d_{free} \cdot R) \text{ dB}.$$

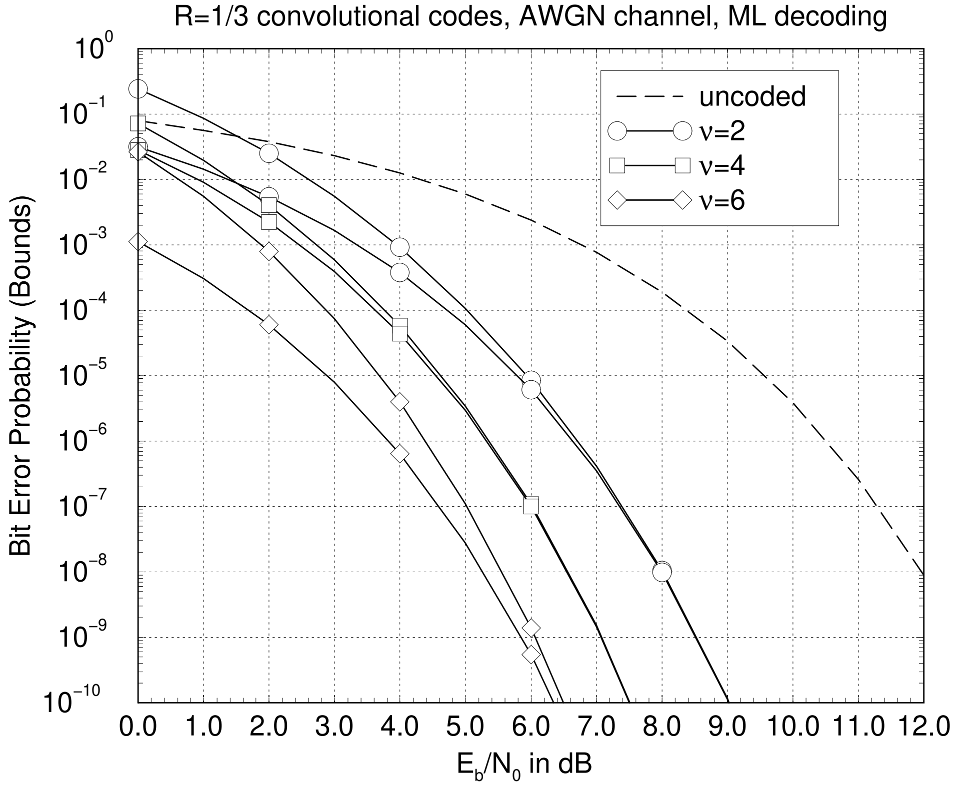
Examples:
The asymptotic coding gain for $R = 1/2$ convolutional codes with maximal free distance is

- 4 dB for $\nu = 2$
- 5.44 dB for $\nu = 4$
- 7 dB for $\nu = 6$.

Bounds on the Bit Error Probability of Rate-1/2 Convolutional Codes with Memory Lengths $\nu = 2, \ldots, 6$



Bounds on the Bit Error Probability of Rate-1/3 Convolutional Codes with Memory Lengths $\nu = 2, \ldots, 6$



Polynomial Representation of Convolutional Codes

Definition (**generator polynomials**): Non-recursive $R = 1/n$ convolutional encoders can be described by the generator polynomials

$$g_i(D) = \sum_{j=0}^{\nu} g_{i,j} D^j, \qquad i = 1, \ldots, n$$

where $g_{i,j} = 0$ if the corresponding modulo-2 addition does not exist and $g_{i,j} = 1$ if the corresponding modulo-2 addition exists.

Accordingly, the **memory length** is

$$\nu = \max_{1 \le i \le n} \ \deg g_i(D).$$

Definition (**info polynomial**): We define an info polynomial (which may be of infinite length)

$$u(D) = \sum_{k=0}^{\infty} u_k D^k, \qquad u_k \in \{0, 1\},$$

where $u_0, u_1, \ldots$ are the info bits.

Polynomial Representation of Convolutional Codes

Encoding corresponds to a multiplication of the polynomials $u(D)$ and $g_i(D)$:

$$x_i(D) = u(D) \cdot g_i(D) \qquad \text{for } 1 \leq i \leq n,$$

or equivalently

$$[x_1(D), \ldots, x_n(D)] = u(D) \cdot \underbrace{[g_1(D), \ldots, g_n(D)]}_{\mathbf{G}(D)}.$$

Definition (**generator matrix**): $\mathbf{G}(D) := [g_1(D), \ldots, g_n(D)]$ is often dubbed generator matrix.

The set of all code words (i.e., the code) can be written as

$$\mathcal{C} = \left\{ u(D)\mathbf{G}(D) \,\middle|\, u(D) = \sum_{k=0}^{\infty} u_k D^k,\ u_k \in \{0,1\} \right\}.$$

Correspondingly, convolutional codes are **linear**.

Recursive Convolutional Encoders

For each non-recursive convolutional encoder a corresponding recursive convolutional encoder generating a code with the same free distance d_{free} can be constructed.

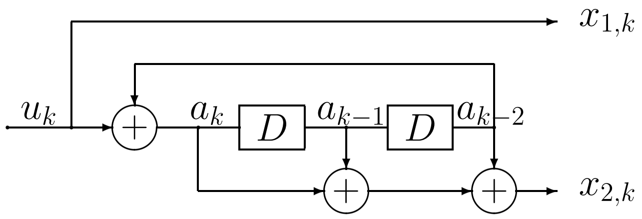
Example: Recursive, systematic $R = 1/2$ convolutional encoder with 4 states

Signal representation:

(1) $x_{1,k} = u_k$

(2) $a_k = u_k \oplus a_{k-2}$

(3) $x_{2,k} = a_k \oplus a_{k-1} \oplus a_{k-2}$



Polynomial representation:

(1') $x_1(D) = u(D)$, therefore $g_1(D) = 1$

(2') $a(D) = u(D) + D^2 a(D)$

(3') $x_2(D) = a(D) + Da(D) + D^2 a(D)$

(2') yields $a(D) = \frac{1}{1+D^2}\, u(D)$. Insertion into (3') yields: $x_2(D) = \underbrace{\dfrac{1 + D + D^2}{1 + D^2}}_{g_2(D)}\, u(D)$

Modifications for Convolutional Codes

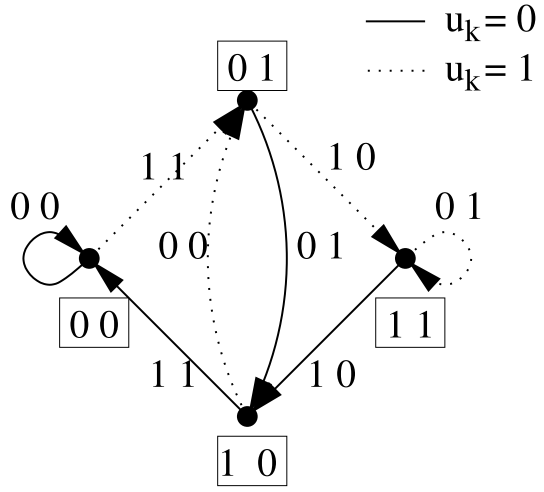
- **Continuous encoding**:
$$R = \frac{K}{N},$$
where K is the number of info bits and N is the number of code bits.
- **Terminated convolutional codes** (zero tailing):
$$R_{zero\ tailing} = \frac{K}{(K+\nu)/R} = R\frac{K}{K+\nu} < R,$$
where ν is the memory length.
- **Convolutional codes with tail biting**:
$$R_{tail\ biting} = \frac{K}{N}.$$
- **Punctured convolutional codes**:
$$R_{punctured} = \frac{P}{P/R - \rho} > R,$$
where P info bits form a set of bits. Out of the corresponding P/R code bits ρ bits are punctured. Therefore, P is called puncturing period.

Punctured Convolutional Codes

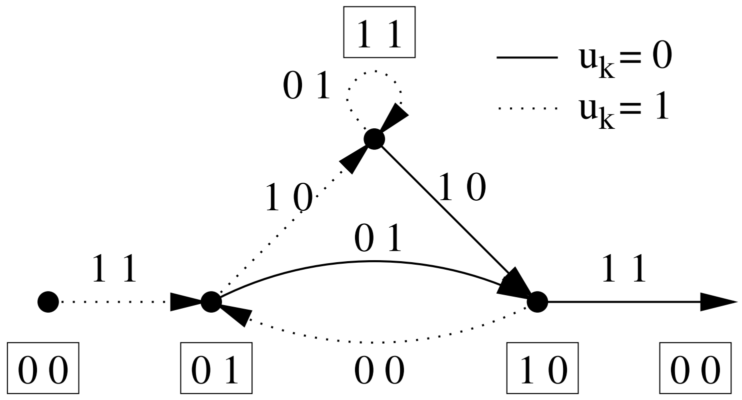
- Goal: Unequal error protection
- Applications: Mobile radio, SatCom, etc.
- Advantages:
  - flexible adaptation of the code rate
  - same (mother) encoder for all code rates
  - binary trellis (given a binary mother code)
  - same decoder
  - similar error correction properties compared to optimized codes of the same rate

Computation of the Distance Spectrum by means of the Modified State Diagram

Example: Non-recursive $R = 1/2$ convolutional encoder with $2^\nu = 4$ states (generator matrix $\mathbf{G}(D) = [1 + D^2, 1 + D + D^2]$)



State Diagram



Modified State Diagram

Computation of the Distance Spectrum by means of the Modified State Diagram

w_c: Hamming weight of the code bits

w_i: Hamming weight of the info bits

Weight enumeration function: $T(w_c, w_i) = \frac{\mu_{2^\nu}}{\mu_0}$

Example: Non-recursive $R = 1/2$ convolutional encoder with $2^\nu = 4$ states (generator matrix $\mathbf{G}(D) = [1 + D^2, 1 + D + D^2]$)

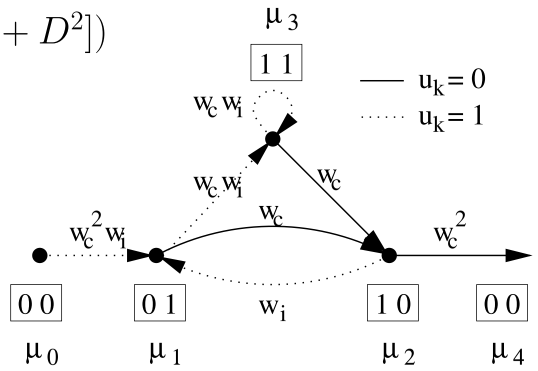
$\mu_1 = w_c^2 w_i \mu_0 + w_i \mu_2$

$\mu_2 = w_c \mu_1 + w_c \mu_3$

$\mu_3 = w_c w_i \mu_1 + w_c w_i \mu_3$

$\mu_4 = w_c^2 \mu_2$

$\Rightarrow T(w_c, w_i) = \frac{\mu_4}{\mu_0} = \frac{w_c^5 w_i}{1 - 2 w_c w_i}$



Computation of the Distance Spectrum by means of the Modified State Diagram

Theorem (**calculation of the distance spectrum**): Given the weight enumeration function, the distance spectrum can be calculated as follows:

$$T(w_c, 1) = \sum_{d=d_{free}}^{\infty} a_d w_c^d \qquad (1)$$

$$\left.\frac{\partial T(w_c, w_i)}{\partial w_i}\right|_{w_i=1} = \sum_{d=d_{free}}^{\infty} c_d w_c^d \qquad (2)$$

Proof:
According to (1): $w_i = 1$ eliminates the influence of irrelevant information.
According to (2): A partial derivation with respect to w_i affects that the exponent of w_i (i.e., the Hamming weight of the info bits) is separated. Due to a subsequent substitution of $w_i = 1$, the influence of irrelevant information is avoided.

Computation of the Distance Spectrum by means of the Modified State Diagram

Example: Non-recursive $R = 1/2$ convolutional encoder with $2^\nu = 4$ states (generator matrix $\mathbf{G}(D) = [1 + D^2, 1 + D + D^2]$)

$T(w_c, 1) = \frac{w_c^5}{1-2w_c} = w_c^5 + 2w_c^6 + 4w_c^7 + \ldots$

$\left.\frac{\partial T(w_c,w_i)}{\partial w_i}\right|_{w_i=1} = \frac{w_c^5}{1-4w_c(1-w_c)} = w_c^5 + 4w_c^6 + 12w_c^7 + \ldots$

Catastrophic Convolutional Encoders

As an example, we consider the $R = 1/2$ convolutional code with generator matrix $\mathbf{G}(D) = [1 + D, 1 + D^2]$, i.e., the taps $(6, 5)|_8$.

The info sequence $\mathbf{u}_0 = [0, 0, \ldots, 0]$ corresponds to the code sequence $\mathbf{x}_0 = [0, 0, 0, 0, 0, \ldots, 0]$.

The info sequence $\mathbf{u}_1 = [1, 1, \ldots, 1]$ corresponds to the code sequence $\mathbf{x}_1 = [1, 1, 0, 1, 0, \ldots, 0]$.

An important observation is that two info sequences with infinite Hamming distance exist, whose corresponding code sequences differ in a finite number of bits (here: 3 bits).

Definition (**catastrophic convolutional encoders**): Convolutional encoders are called catastrophic, if two info sequences with infinite Hamming distance exist, whose corresponding code sequences have a finite Hamming distance.

In the modified state diagram, catastrophic encoders are characterized by loops without any distance gain.

MAP and ML Decoding (revisited)

In the context of block codes, we already defined MAP decoding and ML decoding, respectively. Strictly speaking, we talked about **sequence estimation**.

Definition (**MAP sequence estimation**):
Select the code word $\hat{\mathbf{x}}$ so that

$$\text{HD \& SD}: \quad P(\hat{\mathbf{x}}|\mathbf{y}) \geq P(\tilde{\mathbf{x}}|\mathbf{y}) \quad \forall \quad \tilde{\mathbf{x}} \in \mathcal{C}$$

for a given received word $\mathbf{y}$. This rule is equivalent to selecting the info word $\hat{\mathbf{u}}$ so that

$$P(\hat{\mathbf{u}}|\mathbf{y}) \geq P(\tilde{\mathbf{u}}|\mathbf{y}) \quad \forall \quad \tilde{\mathbf{u}} \in \{0, 1, \ldots, q-1\}^k$$

for a given received word $\mathbf{y}$. Therefore,

$$\hat{\mathbf{u}} = \arg \max_{\tilde{\mathbf{u}}} \; P(\tilde{\mathbf{u}}|\mathbf{y}).$$

MAP and ML Decoding (revisited)

Definition (**ML sequence estimation**):
Select the code word $\hat{\mathbf{x}}$ so that

$$\text{HD}: \quad P(\mathbf{y}|\hat{\mathbf{x}}) \geq P(\mathbf{y}|\tilde{\mathbf{x}}) \quad \forall \quad \tilde{\mathbf{x}} \in \mathcal{C}$$

$$\text{SD}: \quad p(\mathbf{y}|\hat{\mathbf{x}}) \geq p(\mathbf{y}|\tilde{\mathbf{x}}) \quad \forall \quad \tilde{\mathbf{x}} \in \mathcal{C}$$

for a given received word $\mathbf{y}$. This rule is equivalent to selecting the info word $\hat{\mathbf{u}}$ so that

$$P(\mathbf{y}|\hat{\mathbf{u}}) \geq P(\mathbf{y}|\tilde{\mathbf{u}}) \quad \text{resp.} \quad p(\mathbf{y}|\hat{\mathbf{u}}) \geq p(\mathbf{y}|\tilde{\mathbf{u}}) \quad \forall \quad \tilde{\mathbf{u}} \in \{0, 1, \ldots, q-1\}^k$$

for a given received word $\mathbf{y}$. Therefore,

$$\hat{\mathbf{u}} = \arg \max_{\tilde{\mathbf{u}}} P(\mathbf{y}|\tilde{\mathbf{u}}).$$

- In the context of convolutional codes, “word” must be substituted by “sequence”.
- The Viterbi algorithm provides a recursive solution for the problem of sequence estimation.

MAP and ML Decoding (revisited)

If one is interested in a minimization of the symbol error probability or the bit error probability, respectively, **symbol-by-symbol estimation** is required.

Definition (**MAP symbol-by-symbol estimation**):
Select the info bit $\hat{u}_k$ so that

$$\text{HD \& SD}: \quad P(\hat{u}_k|\mathbf{y}) \geq P(\tilde{u}_k|\mathbf{y}) \quad \forall \quad \tilde{u}_k \in \{0, 1, \ldots, q-1\}$$

for a given received word $\mathbf{y}$. Therefore,

$$\hat{u}_k = \arg \max_{\tilde{u}_k} P(\tilde{u}_k|\mathbf{y}).$$

MAP and ML Decoding (revisited)

Definition (**ML symbol-by-symbol estimation**):
Select the info bit $\hat{u}_k$ so that

$$\text{HD}: \quad P(\mathbf{y}|\hat{u}_k) \geq P(\mathbf{y}|\tilde{u}_k) \quad \forall \quad \tilde{u}_k \in \{0, 1, \ldots, q-1\}$$

$$\text{SD}: \quad p(\mathbf{y}|\hat{u}_k) \geq p(\mathbf{y}|\tilde{u}_k) \quad \forall \quad \tilde{u}_k \in \{0, 1, \ldots, q-1\}$$

for a given received word $\mathbf{y}$. Therefore,

$$\hat{u}_k = \arg \max_{\tilde{u}_k} P(\mathbf{y}|\tilde{u}_k).$$

- MAP decoding and ML decoding differ only in the use of a priori information.
- In the context of convolutional codes, "word" must be substituted by "sequence".
- The so-called Bahl-Cocke-Jelinek-Raviv algorithm (BCJR algorithm) provides a recursive solution for the problem of symbol-by-symbol estimation.

Chapter 3: Concatenated Codes (“Turbo Codes”)

- Definition of the symbol-by-symbol MAP decoder
- Log-likelihood ratio
- Bahl-Cocke-Jelinek-Raviv algorithm
- A priori probability, a posteriori probability
- Information theoretical bound of the bit error rate
- The weather problem
- Serial concatenation of codes, parallel concatenation of codes
- Iterative decoding

Definition of the Log-Likelihood Ratio

Now, we consider binary codes ($q = 2$), in which case the s.b.s. MAP decoder computes $P(u_k = +1|\mathbf{y}) = 1 - P(u_k = -1|\mathbf{y})$

⇒ minimization of the bit error probability
⇒ soft-input/soft-output decoding

- Definition: Log-likelihood ratio (LLR)

$$L_k := C \log_{10} \frac{P(u_k = +1|\mathbf{y})}{P(u_k = -1|\mathbf{y})}, \qquad C > 0 \text{ arbitrary}, \quad 0 \le k \le K-1$$

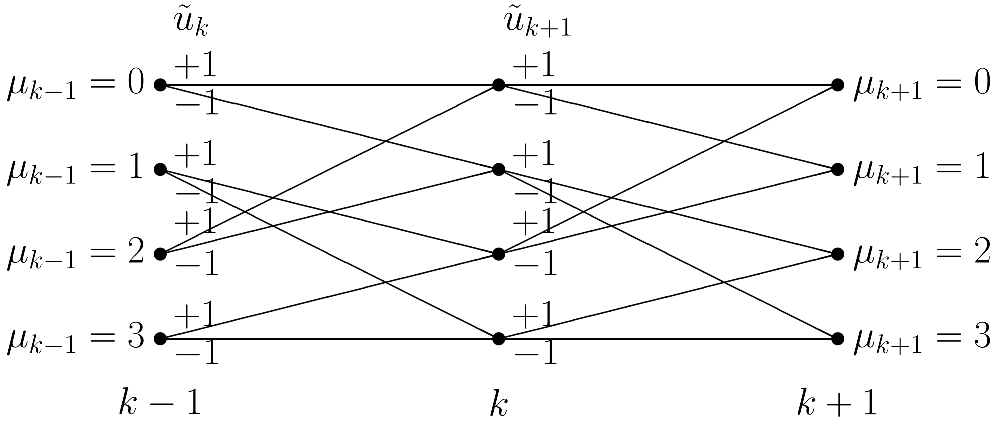
- Sign: Maximum a posteriori (MAP) decisions

$$\hat{u}_k = \begin{cases} +1 & \text{if } P(u_k = +1|\mathbf{y}) \ge 0.5 \\ -1 & \text{else} \end{cases}$$

- Magnitude: Reliability values $|L_k|$, $0 \le |L_k| < \infty$

Notation: “Tilde” marks a hypothesis, “hat” marks a decision

Principle of the Symbol-by-Symbol MAP Decoder



Bahl-Cocke-Jelinek-Raviv algorithm (BCJR algorithm):

1. step:

$$\begin{aligned} P(\mu_k = 1|\mathbf{y}) \ = \ & \big[P(\mu_{k-1}=0|\mathbf{y}) \cdot P(\mu_{k-1}=0 \to \mu_k = 1) \\ & +P(\mu_{k-1}=2|\mathbf{y}) \cdot P(\mu_{k-1}=2 \to \mu_k = 1)\big] \\ \cdot \ & \big[P(\mu_{k+1}=2|\mathbf{y}) \cdot P(\mu_{k+1}=2 \to \mu_k = 1) \\ & +P(\mu_{k+1}=3|\mathbf{y}) \cdot P(\mu_{k+1}=3 \to \mu_k = 1)\big] \end{aligned}$$

2. step:

$$L_k = C \log_{10} \frac{P(u_k = +1|\mathbf{y})}{P(u_k = -1|\mathbf{y})} = C \log_{10} \frac{P(\mu_k = 0|\mathbf{y}) + P(\mu_k = 2|\mathbf{y})}{P(\mu_k = 1|\mathbf{y}) + P(\mu_k = 3|\mathbf{y})}$$

Derivation of the Symbol-by-Symbol MAP Decoder

Assumptions:

We are given a trellis with a finite number of states.

The states are denoted as μ_k, $0 \leq k \leq K$.

Hypotheses with respect to the info bits u_k are denoted as $\tilde{u}_k$, $0 \leq k < K$.

Received values are denoted as y_n, $0 \leq n < N$.

The noise process is assumed to be white Gaussian.

In the following derivation we assume that $N = 2K$, i.e., $R = 1/2$.

S.b.s. MAP estimation of u_k corresponds to a maximization of $P(\tilde{u}_k|\mathbf{y}) = P(\tilde{u}_k|y_0^{N-1})$:

$$\hat{u}_k = \arg \max_{\tilde{u}_k} P(\tilde{u}_k|y_0^{N-1}).$$

According to Bayes' rule

$$P(\tilde{u}_k|y_0^{N-1}) = P(\tilde{u}_k, y_0^{N-1})/P(y_0^{N-1}).$$

Let $\mathcal{S}(\tilde{u}_k)$ be the set of all "relevant" (even or odd) states μ_k. Therefore,

$$P(\tilde{u}_k, y_0^{N-1}) = \sum_{\mu_k \in \mathcal{S}(\tilde{u}_k)} P(\mu_k, y_0^{N-1}) \quad \text{and} \quad P(y_0^{N-1}) = \sum_{\mu_k} P(\mu_k, y_0^{N-1}).$$

Derivation of the Symbol-by-Symbol MAP Decoder

Due to Bayes rule and for white noise, we obtain

$$P(\tilde{u}_k|y_0^{N-1}) = \frac{\sum\limits_{\mu_k \in \mathcal{S}(\tilde{u}_k)} P(\mu_k, y_0^{N-1})}{\sum\limits_{\mu_k} P(\mu_k, y_0^{N-1})} = \frac{\sum\limits_{\mu_k \in \mathcal{S}(\tilde{u}_k)} P(\mu_k, y_0^{2k-1}) \cdot P(y_{2k}^{N-1}|\mu_k)}{\sum\limits_{\mu_k} P(\mu_k, y_0^{2k-1}) \cdot P(y_{2k}^{N-1}|\mu_k)}. \qquad (*)$$

Both factors can be computed recursively. After an extension, due to Bayes rule and for white noise the first term in $(*)$ simplifies as

$$\begin{aligned} P(\mu_k, y_0^{2k-1}) &= \sum_{\mu_{k-1}} P(\mu_k, y_0^{2k-1}, \mu_{k-1}) \\ &= \sum_{\mu_{k-1}} P(\mu_{k-1}, y_0^{2k-3}) \cdot P(\mu_k, y_{2k-2}, y_{2k-1}|\mu_{k-1}, y_0^{2k-3}) \\ &= \sum_{\mu_{k-1}} P(\mu_{k-1}, y_0^{2k-3}) \cdot P(\mu_k, y_{2k-2}, y_{2k-1}|\mu_{k-1}) \\ &= \sum_{\mu_{k-1}} P(\mu_{k-1}, y_0^{2k-3}) \cdot P(\mu_k|\mu_{k-1}) \cdot P(y_{2k-2}, y_{2k-1}|\mu_{k-1}, \mu_k). \end{aligned}$$

Derivation of the Symbol-by-Symbol MAP Decoder

The last two terms represent the transition probabilities. Therefore,

$$P(\mu_k, y_0^{2k-1}) = \sum_{\mu_{k-1}} P(\mu_{k-1}, y_0^{2k-3}) \cdot P(\mu_{k-1} \to \mu_k).$$

Due to symmetry reasons, the second term in $(*)$ can be written as

$$P(y_{2k}^{N-1}|\mu_k) = \sum_{\mu_{k+1}} P(y_{2k}^{N-1}, \mu_{k+1}|\mu_k) = \sum_{\mu_{k+1}} P(y_{2k+2}^{N-1}|\mu_{k+1}) \cdot P(\mu_{k+1} \to \mu_k).$$

In summary, we obtain

$$P(\tilde{u}_k|y_0^{N-1}) = \frac{\sum\limits_{\mu_k \in \mathcal{S}(\tilde{u}_k)} P(\mu_k, y_0^{2k-1}) \cdot P(y_{2k}^{N-1}|\mu_k)}{\sum\limits_{\mu_k} P(\mu_k, y_0^{2k-1}) \cdot P(y_{2k}^{N-1}|\mu_k)}$$

with the forward recursion

$$P(\mu_k, y_0^{2k-1}) = \sum_{\mu_{k-1}} P(\mu_{k-1}, y_0^{2k-3}) \cdot P(\mu_{k-1} \to \mu_k),$$

Derivation of the Symbol-by-Symbol MAP Decoder

the backward recursion

$$P(y_{2k}^{N-1}|\mu_k) = \sum_{\mu_{k+1}} P(y_{2k+2}^{N-1}|\mu_{k+1}) \cdot P(\mu_{k+1} \to \mu_k),$$

and the transition probabilities (cf. AWGN channel model)

$$P(\mu_{k-1} \to \mu_k) = \frac{E_s}{\pi N_0} \underbrace{P(\mu_k|\mu_{k-1})}_{\text{a priori info}} \underbrace{\exp\left(-\frac{(y_{2k-2} - \tilde{x}_{2k-2})^2 + (y_{2k-1} - \tilde{x}_{2k-1})^2}{N_0/E_s}\right)}_{\text{channel information}}.$$

The Symbol-by-Symbol MAP Decoder (Summary)

S.b.s. MAP estimation of u_k corresponds to a maximization of $P(\tilde{u}_k|\mathbf{y}) = P(\tilde{u}_k|y_0^{N-1})$:

$$\hat{u}_k = \arg \max_{\tilde{u}_k} P(\tilde{u}_k|y_0^{N-1}) = \frac{\sum\limits_{\mu_k \in \mathcal{S}(\tilde{u}_k)} P(\mu_k, y_0^{2k-1}) \cdot P(y_{2k}^{N-1}|\mu_k)}{\sum\limits_{\mu_k} P(\mu_k, y_0^{2k-1}) \cdot P(y_{2k}^{N-1}|\mu_k)}.$$

The terms may be computed by means of a forward recursion

$$P(\mu_k, y_0^{2k-1}) = \sum_{\mu_{k-1}} P(\mu_{k-1}, y_0^{2k-3}) \cdot P(\mu_{k-1} \to \mu_k)$$

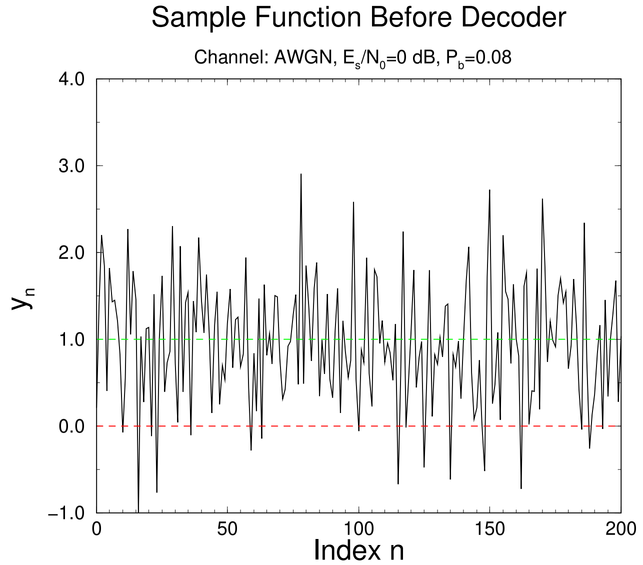
and a backward recursion

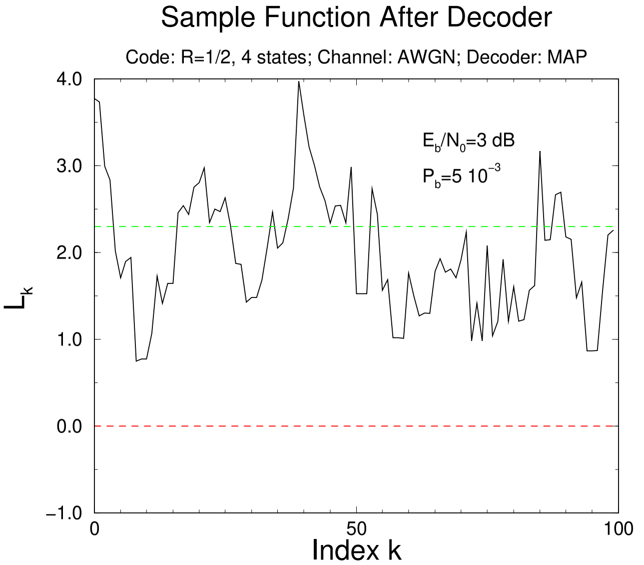
$$P(y_{2k}^{N-1}|\mu_k) = \sum_{\mu_{k+1}} P(y_{2k+2}^{N}|\mu_{k+1}) \cdot P(\mu_{k+1} \to \mu_k),$$

respectively. For the AWGN channel, the transition probabilities are

$$P(\mu_{k-1} \to \mu_k) = \frac{E_s}{\pi N_0} \underbrace{P(\mu_k|\mu_{k-1})}_{\text{a priori info}} \underbrace{\exp\left(-\frac{(y_{2k-2} - \tilde{x}_{2k-2})^2 + (y_{2k-1} - \tilde{x}_{2k-1})^2}{N_0/E_s}\right)}_{\text{channel information}}.$$

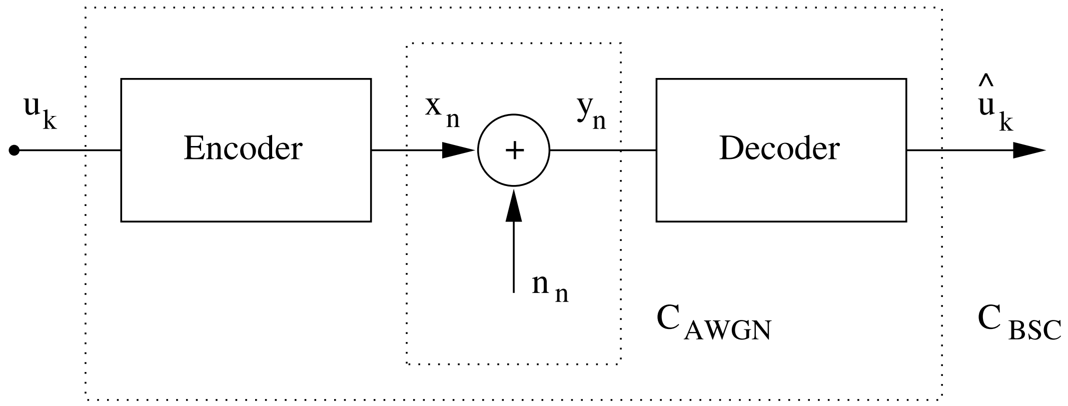
Interpretation of the Symbol-by-Symbol MAP Decoder





$\Rightarrow$ The MAP decoder may be interpreted as a **non-linear LLR filter**

Information Theoretical Bound of the Bit Error Probability



Information Theoretical Bound of the Bit Error Probability

Given a binary code of rate R and soft-input/hard-output decoding:
What is the information theoretical bound of the bit error probability as a function of the signal/noise ratio per info bit?

The channel capacity of an AWGN channel model with binary input symbols $\pm x$ is

$$C_{AWGN} = \frac{1}{2}\int_{-\infty}^{\infty} p(y|x)\log_2\frac{p(y|x)}{p(y)}\,dy + \frac{1}{2}\int_{-\infty}^{\infty} p(y|-x)\log_2\frac{p(y|-x)}{p(y)}\,dy.$$

The channel capacity of a binary symmetric channel (BSC) model with bit error rate P_{BSC} is

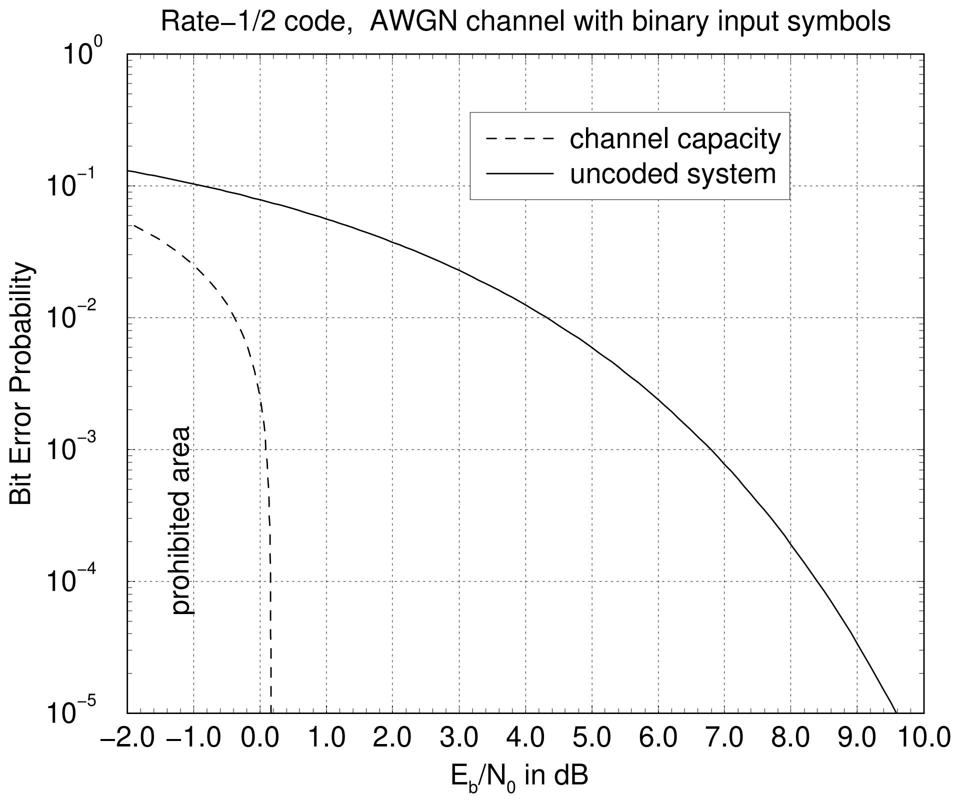
$$C_{BSC} = 1 + P_{BSC}\log_2(P_{BSC}) + (1-P_{BSC})\log_2(1-P_{BSC}).$$

According to a result from information theory

$$R\cdot \underbrace{C_{BSC}}_{f(P_{BSC})} \le \underbrace{C_{AWGN}}_{f(E_s/N_0)}.$$

The desired result is obtained by solving P_{BSC}.

Information Theoretical Bound of the Bit Error Probability



Log-Likelihood Ratio (LLR) for the AWGN Channel

$$
\begin{aligned}
L_k &= C \log_{10} \frac{P(u_k = +1|y_k)}{P(u_k = -1|y_k)}, \qquad \text{e.g., } C = \log_e(10) \\
&= \log_e \frac{P(y_k|u_k = +1) \cdot P(u_k = +1)}{P(y_k|u_k = -1) \cdot P(u_k = -1)} \\
&= \log_e \frac{\exp\{-\frac{E_s}{N_0}(y_k - 1)^2\}}{\exp\{-\frac{E_s}{N_0}(y_k + 1)^2\}} + \log_e \frac{P(u_k = +1)}{P(u_k = -1)} \\
&= \underbrace{4\frac{E_s}{N_0} y_k}_{\text{channel information}} + \underbrace{\log_e \frac{P(u_k = +1)}{P(u_k = -1)}}_{\text{a priori information}}
\end{aligned}
$$

Observation: Log-likelihood values of statistically independent observations of a random variable are additive

The Weather Problem

Assumption: Two unbiased weather stations observe the binary random variable “weather”, which may take the events “sunny” and “rainy”. Both weather stations use different measurement principles.
Weather station 1 (frog): 60 % sunny
Weather station 2 (satellite): 70 % sunny

Problem: What is the probability for sunny weather given both observations?

Answer:

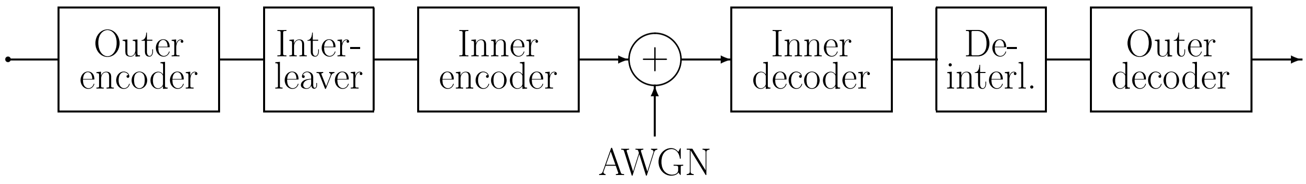
$$L_1 = \log_e \frac{0.6}{0.4}, \qquad L_2 = \log_e \frac{0.7}{0.3}$$

$$L = L_1 + L_2 = \log_e \frac{P}{1-P}$$

$$\Rightarrow \qquad P = \frac{\exp\{L_1 + L_2\}}{1 + \exp\{L_1 + L_2\}} \approx 0.777$$

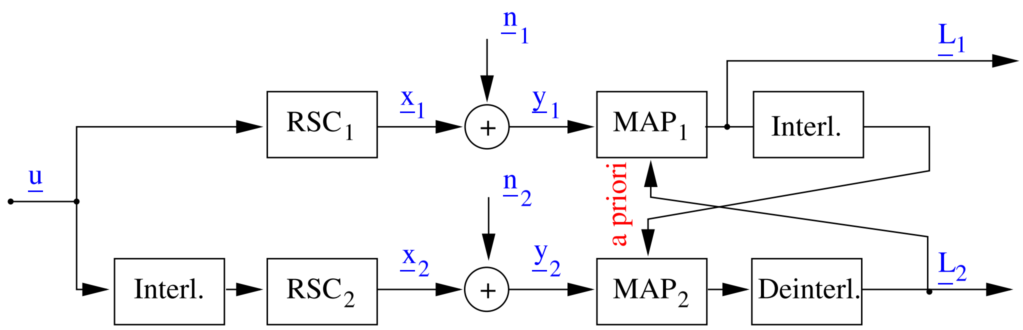
Hence, the probability for sunny weather is 77.7 % !

Serial Code Concatenation



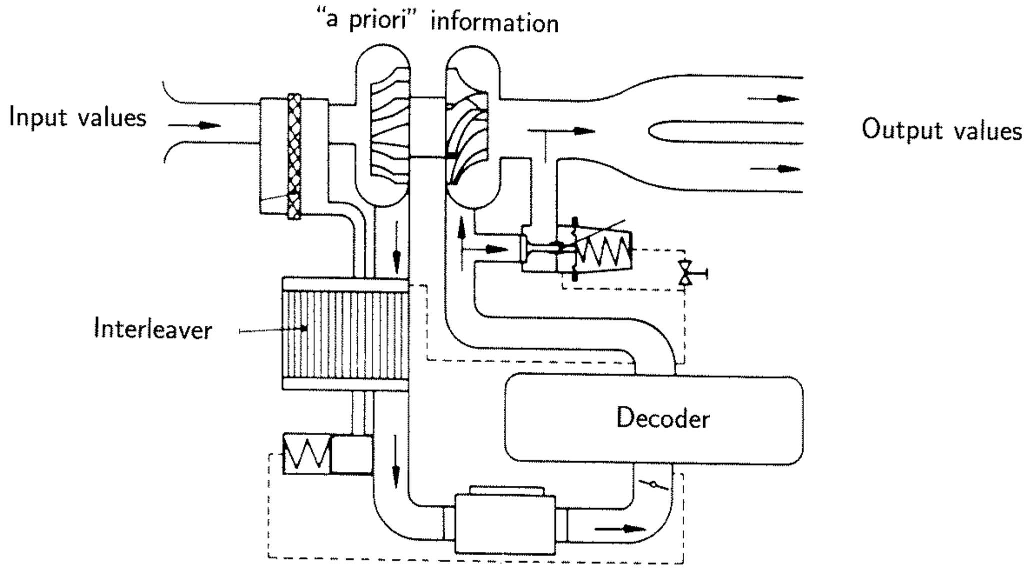
Parallel Code Concatenation ("Turbo Codes")

RSC code: Recursive Systematic Convolutional code

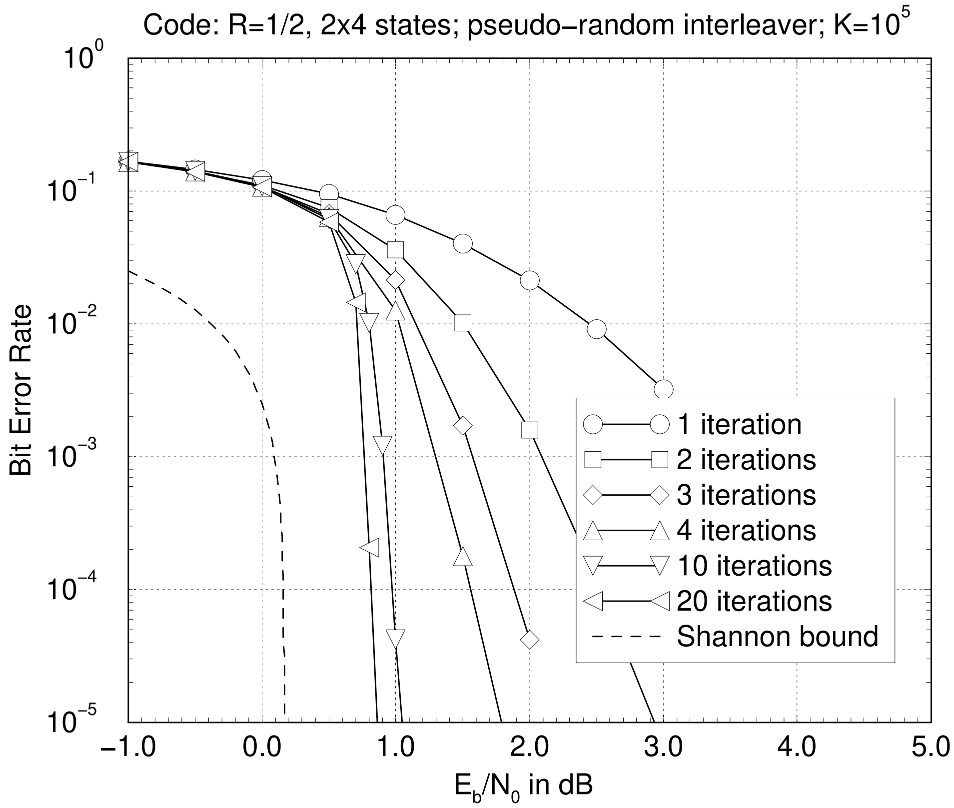


Parallel concatenated encoder and iterative ("turbo") decoder

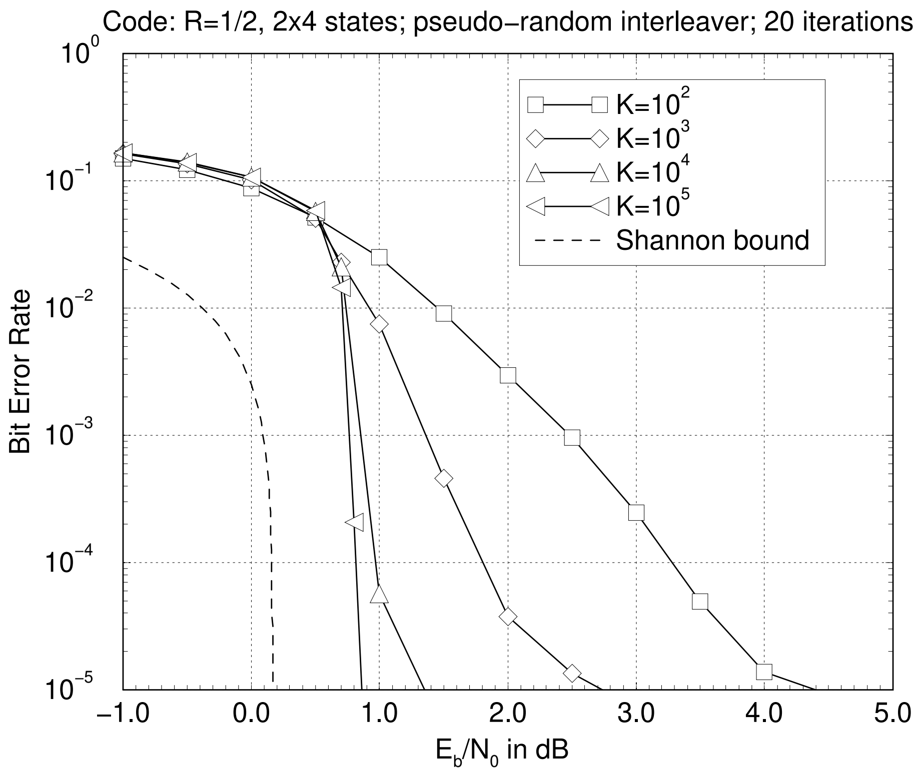
Block Diagram of a Turbo Charger



Bit Error Rate as a Function of the Number of Iterations



Bit Error Rate as a Function of the Info Word Length



Chapter 4: Block Codes II

- Group, field, Galois field, prime field
- Primitive element, order of an element
- Euclidean algorithm
- Extension field, irreducible polynomial, inverse polynomial
- Roots of a polynomial, primitive polynomial
- Reed-Solomon codes, Bose-Chaudhuri-Hocquenghem codes
- Algebraic decoding
- Further classes of block codes

Group

Definition (**group**): A group is a set $\mathcal{A}$ of elements (objects) if and only if it is possible to define an operation $*$, so that the following axioms apply:

1. Closure under the defining operation: $\forall\, a, b \in \mathcal{A}: \quad a * b \in \mathcal{A}$
2. Associative law: $\forall\, a, b, c \in \mathcal{A}: \quad a * (b * c) = (a * b) * c$
3. Existence of an identity element e: $\exists\, e \in \mathcal{A} \;\; \forall\, a \in \mathcal{A}: \quad a * e = a$
4. Existence of inverse elements: $\forall\, a \in \mathcal{A} \;\; \exists\, b \in \mathcal{A}: \quad a * b = e.$

If additionally the commutative law holds, i.e.,

$$\forall\, a, b \in \mathcal{A}: \quad a * b = b * a,$$

the group is called an **Abelian group**.

Field, Galois Field, Prime Field

Definition (**field**): A set $\mathcal{A}$ with two operations (the addition $\oplus$ and the multiplication $\odot$) is called a field, if the following axioms apply:

1. $\mathcal{A}$ is an Abelian group with respect to the addition
2. $\mathcal{A}\backslash\{0\}$ is a Abelian group with respect to the multiplication
3. Distributive law: $\forall\ a, b, c \in \mathcal{A}:\ \ a \odot (b \oplus c) = a \odot b \oplus a \odot c$

Definition (**Galois field**): A field with a finite number of elements is called Galois field (GF).

Definition (**prime field**): Let $p \in I\!N$ be a prime number. The set of elements $\{0, 1, \ldots, p-1\}$ with the operations $(\oplus, \odot)$ mod p fulfill the axioms of a field.
This field is called prime field and is denoted as GF(p).

Example: GF(5).

Primitive Element, Order of an Element

Definition (**primitive element**): An element $\alpha \in \mathrm{GF}(p)$, whose $p-1$ powers α^j, $j \in \{1, \ldots, p-1\}$, correspond to all possible elements $a \neq 0$ of a Galois field, is called a primitive element.

For each Galois field at least one primitive element exists. The inverse element of a primitive element is again a primitive element.

Example GF(5): The elements $\alpha = 2$ and $\alpha = 3$ are primitive elements.

Definition (**order of an element**): The order of an element $a \in \mathrm{GF}(p)$, $a \neq 0$, is the smallest exponent $r > 0$ so that $a^r = 1 \bmod p$.
For $r = p-1$, a is a primitive element.

Example GF(5): The order of the element $a = 4$ is $r = 2$.
Hence, $a = 4$ is not a primitive element.

Euclidean Algorithm

Theorem (**Euclidean algorithm**): Let $a, b \neq 0$ be two integer numbers, where $a < b$. The greatest common divisor of a and b, $\gcd(a, b)$, can be computed by means of the Euclidean algorithm:

$$\begin{aligned}
b &= q_1 \cdot a + r_1 \\
a &= q_2 \cdot r_1 + r_2 \qquad \text{with } a = r_0, b = r_{-1}: \\
r_1 &= q_3 \cdot r_2 + r_3 \qquad r_{j-1} = q_{j+1} \cdot r_j + r_{j+1} \\
&\dots \\
r_{n-1} &= q_{n+1} \cdot r_n + 0 \\
\gcd(a, b) &= r_n.
\end{aligned}$$

Theorem: For arbitrary integer numbers $a, b \neq 0$ there exist integer numbers v and w so that

$$\gcd(a, b) = b \cdot v + a \cdot w \qquad \text{(Remark: } v, w \text{ are not necessarily unique).}$$

Existence of an Inverse Element

Definition (**relatively prime**): Two integer numbers $a, b \neq 0$ are called relatively prime, if $\gcd(a, b) = 1$.

Theorem (**existence of an inverse element**): Given an arbitrary prime field $\mathrm{GF}(p)$, for each element $a \in \mathrm{GF}(p) \backslash \{0\}$ there exists a unique **inverse element** a^{-1}.

Proof: According to the definition of a prime number, each element $a < p$ is relatively prime with respect to p, i.e., for each $a \in \mathrm{GF}(p)$

$$\gcd(a, p) = 1.$$

According to the last theorem, there exist two integer numbers v and w such that

$$p \cdot v + a \cdot w = 1 \bmod p.$$

Therefore,

$$\begin{aligned}
a \cdot w &= 1 \bmod p \\
w &= a^{-1}. \qquad \text{q.e.d.}
\end{aligned}$$

Extension Field

So far, we have investigated only a small subset of Galois fields, namely prime fields GF(p).

In the following, our focus is on an extension of prime fields GF(p). Extension fields provide additional flexibility for the description of Galois fields.

In practice, usually extensions of GF(2) are used in order to exploit binary numbers.

Irreducible Polynomials

Definition (**irreducible polynomial**): A polynomial $p(D)$ with coefficients over GF(p) is irreducible with respect to GF(p), if it cannot be represented by a product of polynomials of smaller degree, whose coefficients are also over GF(p).

Theorem (**inverse polynomial**): If $p(D)$, $\deg p(D) = m$, $p_i \in$ GF(p), is irreducible with respect to GF(p), each of the $p^m - 1$ polynomials $b(D)\backslash\{0\}$ with $\deg b(D) < m$, $b_i \in$ GF(p), has a unique inverse polynomial $b^{-1}(D)$, i.e.,

$$b(D) \cdot b(D)^{-1} = 1 \bmod p(D).$$

Roots of a Polynomial

Definition (**roots of a polynomial**): Let $p(D)$, $p_i \in \mathrm{GF}(p)$, be an irreducible polynomial. Then, according to the definition of irreducible polynomials, $p(D)$ has no roots over $\mathrm{GF}(p)$. We define an element α over an **extension field** as

$$p(\alpha) = 0.$$

α is called root of $p(D)$.

Analogy: $D^2 + 1 = 0 \Rightarrow j$ is an element of the extension field $\mathbb{C}$ of $\mathbb{R}$.

Primitive Polynomial

Definition (**primitive polynomial, primitive element**): Let $p(D)$ be an irreducible polynomial, $\deg p(D) = m$, $p_i \in \mathrm{GF}(p)$, and let α be a root of $p(D)$ (i.e., $p(\alpha) = 0$) with the property that

$$\alpha^i \bmod p(\alpha),\ i = 0, 1, \ldots, p^m - 2$$

generates all $p^m - 1$ possible polynomials $f(\alpha) \neq 0$,
then $p(D)$ is called a primitive polynomial and α is called primitive element.

The root α is called a primitive element, if its powers generate all elements of the extension field $\mathrm{GF}(p^m)$.

Elements of an Extension Field

We assume the existence of a primitive polynomial $p(D)$ of degree m.

Definition (**elements of an extension field**): The elements of an extension field $\mathrm{GF}(p^m)$ are calculated by computing the powers

$$\alpha^i \bmod p(\alpha), \qquad i = -\infty, 0, 1, \ldots, p^m - 2,$$

where α is the root of the primitive polynomial $p(D)$ of degree m, i.e., $p(\alpha) = 0$.

Under the assumption that primitive polynomials exist, the elements of the extension field may be expressed by an **exponent representation** or a **component representation**, respectively.

Example: Galois Field GF(2^4) with Primitive Polynomial $p(D) = D^4 + D + 1$, $p_i \in \mathbf{GF}(2)$

The Galois field $\mathrm{GF}(2^4)$ with the primitive polynomial $p(D) = D^4 + D + 1$, $p_i \in \mathrm{GF}(2)$, and the primitive element α, $p(\alpha) = 0$, may be represented by the following table:

Exponent	Components				Calculation
$-\infty$				0	
0				1	
1			α		
2		α^2			
3	α^3				
4			α	$+1$	$\alpha^4 = \alpha + 1$
5		α^2	$+\alpha$		$\alpha^5 = \alpha \cdot \alpha^4 = \alpha^2 + \alpha$
6	α^3	$+\alpha^2$			$\alpha^6 = \alpha \cdot \alpha^5 = \alpha^3 + \alpha^2$
7	α^3		$+\alpha$	$+1$	$\alpha^7 = \alpha \cdot \alpha^6 = \alpha^4 + \alpha^3 = \alpha^3 + \alpha + 1$
8		α^2		$+1$	$\alpha^8 = \alpha \cdot \alpha^7 = \alpha^4 + \alpha^2 + \alpha = \alpha^2 + 1$
9	α^3		$+\alpha$		$\alpha^9 = \alpha \cdot \alpha^8 = \alpha^3 + \alpha$
10		α^2	$+\alpha$	$+1$	$\alpha^{10} = \alpha \cdot \alpha^9 = \alpha^4 + \alpha^2 = \alpha^2 + \alpha + 1$
11	α^3	$+\alpha^2$	$+\alpha$		$\alpha^{11} = \alpha \cdot \alpha^{10} = \alpha^3 + \alpha^2 + \alpha$
12	α^3	$+\alpha^2$	$+\alpha$	$+1$	$\alpha^{12} = \alpha \cdot \alpha^{11} = \alpha^3 + \alpha^2 + \alpha + 1$
13	α^3	$+\alpha^2$		$+1$	$\alpha^{13} = \alpha \cdot \alpha^{12} = \alpha^3 + \alpha^2 + 1$
14	α^3			$+1$	$\alpha^{14} = \alpha \cdot \alpha^{13} = \alpha^4 + \alpha^3 + \alpha = \alpha^3 + 1$
15				1	$\alpha^{15} = \alpha \cdot \alpha^{14} = \alpha^4 + \alpha = 1$

Multiplication: $\alpha^9 \cdot \alpha^7 = \alpha^{16 \bmod 15} = \alpha$

Addition: $\alpha^4 + \alpha^5 = 0011 \oplus 0110 = 0101 = \alpha^8$

Primitive Polynomials

Theorem: For each (prime) field GF(p) and each number $m \in I\!N$ there exists at least one primitive polynomial $p(D) = p_0 + p_1 D + \ldots + p_m D^m$, $p_i \in \mathrm{GF}(p)$.

m	Primitive polynomial $p(D)$	m	Primitive polynomial $p(D)$
1	$D+1$	9	D^9+D^4+1
2	D^2+D+1	10	$D^{10}+D^3+1$
3	D^3+D+1	11	$D^{11}+D^2+1$
4	D^4+D+1	12	$D^{12}+D^7+D^4+D^3+1$
5	D^5+D^2+1	13	$D^{13}+D^4+D^3+D+1$
6	D^6+D+1	14	$D^{14}+D^8+D^6+D+1$
7	D^7+D+1	15	$D^{15}+D+1$
8	$D^8+D^6+D^5+D^4+1$	16	$D^{16}+D^{12}+D^3+D+1$

Primitive polynomials for the design of extension fields GF(2^m) ($1 \le m \le 16$)

Properties of Extension Fields

Complex numbers are an extension field of real numbers.
If we consider the conjugate complex numbers $-j$, j, we get

$$(x-j)\cdot(x+j) = x^2+1.$$

The product of two polynomials with conjugate complex roots ($\in \mathbb{C}$)
results in a polynomial with coefficients over $I\!R$ rather than over $\mathbb{C}$.
Correspondingly, for extension fields the following theorem applies:

Theorem: Let $p(D)$, $\deg p(D) = m$, $p_i \in \mathrm{GF}(p)$, be irreducible with respect to GF(p), and let α be a root of $p(D)$, then together with α also

$$\alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{m-1}}$$

are roots of $p(D)$. The roots are called conjugate complex.

Order of an Element

In prime fields, we already have defined the order of elements $\alpha \in \mathrm{GF}(p)$.
Along the same line, we define for extension fields:

Definition (**order of an element**): Let $\beta \in \mathrm{GF}(p^m)$ and n be the smallest integer number for which $\beta^n = 1$. Then, n is called the order of β.

If $n = p^m - 1$, β is a primitive element.

It follows that all β^i, $0 \leq i < n$, are different.

RS Codes and BCH Codes

- Reed-Solomon (RS) codes and Bose-Chaudhuri-Hocquenghem (BCH) codes have been invented about 1960. Nowadays, they still belong to the most powerful block codes
- RS codes and BCH codes can be designed analytically
- The minimum distance is a design parameter of RS codes
- The weight enumeration function of RS codes can be computed analytically
- BCH codes can be interpreted as binary RS codes
- BCH codes are more suitable for the correction of single errors, whereas RS codes are more suitable for the correction of burst errors

Reed-Solomon Codes

In the following, we will restrict ourselves to prime fields $\mathrm{GF}(p)$ in order to design RS codes. However, RS codes may also be designed by means of extension fields $\mathrm{GF}(q) = \mathrm{GF}(p^m)$, where all algorithms are along the same line. In practice, extension fields with basis $p = 2$ are often used.

Definition (**RS codes**): Let $\alpha \in \mathrm{GF}(p)$ be an element of order n.
An $(n, k, d_{min})_q$ RS code $\mathcal{C}$, $q = p$, is defined by the set of info polynomials

$$A(D) = A_0 + A_1 D + A_2 D^2 + \ldots + A_{k-1} D^{k-1},\ A_j \in \mathrm{GF}(p),\ k < n$$

of degree $< k$ so that the p^k corresponding code words $\mathbf{a} = [a_0, a_1, \ldots, a_{n-1}]$ are given by $a_i = A(\alpha^i) \in \mathrm{GF}(p)$:

$$\mathcal{C} = \{\mathbf{a} \mid a_i = A(\alpha^i),\ i \in \{0, 1, \ldots, n-1\},\ \deg A(D) < k\}.$$

The minimum distance is $d_{min} = n - k + 1$.

Remark: $n = p - 1$ (generally: $n = p^m - 1$) is called primitive code word length.

Discrete Fourier Transform (DFT)

Given $\mathbf{a}$, how can we recover $A(D)$?

Definition (**Discrete Fourier Transform (DFT)**):
We define $a(D) = a_0 + a_1 D + a_2 D^2 + \ldots + a_{n-1} D^{n-1}$.
Let $\alpha \in \mathrm{GF}(p)$ be an element of order n and let $a(D)$ and $A(D)$ be polynomials of degree $< n$ and $< k$, respectively, with coefficients over $\mathrm{GF}(p)$. The DFT is defined as

$$A_j = n^{-1} \cdot a(\alpha^{-j}), \qquad j \in \{0, 1, \ldots, k-1\}.$$

Correspondingly, the **inverse DFT (IDFT)** is

$$a_i = A(\alpha^i), \qquad i \in \{0, 1, \ldots, n-1\}.$$

Short hand notation:

$$\begin{array}{cccll} a(D) & \circ\!\!-\!\!\bullet & A(D) & \text{DFT} & \text{(decoding)} \\ A(D) & \bullet\!\!-\!\!\circ & a(D) & \text{IDFT} & \text{(encoding)} \end{array}$$

Remark: In case of extension fields $\mathrm{GF}(2^m)$, $n^{-1} = 1$.

Convolution Theorem

Theorem (**convolution theorem**): The multiplication of two polynomials $\mathrm{mod}(D^n - 1)$ corresponds to a cyclic convolution:

$$a(D) \circ\!\!-\!\!\bullet\ A(D), \qquad b(D) \circ\!\!-\!\!\bullet\ B(D)$$

$$c_i = a_i \cdot b_i, \qquad c(D) \circ\!\!-\!\!\bullet\ C(D) = A(D) \cdot B(D) \bmod (D^n - 1)$$

$$C_i = A_i \cdot B_i, \qquad C(D) \bullet\!\!-\!\!\circ\ c(D) = \frac{1}{n} \cdot a(D) \cdot b(D) \bmod (D^n - 1).$$

Generator Polynomial

An RS code of length n, of dimension k, and with minimum distance $d_{min} = n - k + 1$ may alternatively be defined by all info polynomials $u(D)$ of degree $< k$.

The code polynomial can be calculated by a multiplication between the info polynomial $u(D)$ and the generator polynomial $g(D)$:

$$a(D) = u(D) \cdot g(D), \qquad \deg g(D) = n - k.$$

Hence, each code word is divisible by $g(D)$.

On the other hand, an RS code is defined by the set of all polynomials $A(D)$ of degree $< k$, i.e., $A_i = 0$ for $k \le i \le n - 1$.
Therefore,

$$A_i = n^{-1} \cdot a(\alpha^{-i}) = 0 \qquad \text{for } k \le i \le n - 1.$$

Since $n^{-1} \neq 0$, the code polynomial $a(D)$ must have roots α^{-i} for all i, $k \le i \le n - 1$.
In other words, the code polynomial must contain the factors $(D - \alpha^{-i})$.

Generator Polynomial

Hence, the code polynomial can be represented in the form

$$a(D) = \underbrace{a'(D)}_{u(D)} \cdot \underbrace{\prod_{i=k}^{n-1} (D - \alpha^{-i})}_{g(D)} .$$

Correspondingly, the product of the factors $(D - \alpha^{-i})$ establishes the generator polynomial $g(D)$ of degree $n-k$:

$$g(D) = \prod_{i=k}^{n-1} (D - \alpha^{-i}).$$

Accordingly, RS codes are linear codes.

Theorem: RS codes are cyclic codes.

Proof: For each code polynomial $a(D) \in \mathcal{C}$ we have

$$D \cdot a(D) = \underbrace{D \cdot u(D)}_{u'(D)} \cdot g(D) = \underbrace{u'(D)}_{\text{new info word}} \cdot g(D) \mod (D^n - 1) \in \mathcal{C}. \qquad \text{q.e.d.}$$

Parity Check Polynomial

The parity check polynomial $h(D)$ is defined as

$$a(D) \cdot h(D) = 0 \mod (D^n - 1) \text{ for all } a(D) \in \mathcal{C}.$$

A DFT results in

$$A_i \cdot H_i = 0, \qquad i = 0, 1, \ldots, n-1, \qquad h(D) \circ\!\!-\!\!\bullet\, H(D).$$

H_i must be zero at those positions, where A_i is not equal to zero or may be not equal to zero.

Parity Check Polynomial

Accordingly, the parity check polynomial may be represented as

$$h(D) = \prod_{i=0}^{k-1} (D - \alpha^{-i}), \qquad \deg h(D) = k$$

Hence, the roots of $g(D)$ and $h(D)$ are disjunct and contain, for $n = p - 1$, all α^i:

$$g(D) \cdot h(D) = \prod_{i=0}^{n-1} (D - \alpha^{-i}) = D^n - 1.$$

Encoding Methods for RS Codes

The following four methods generate the same code, however,
the mapping between info words and code words is different.

Methods 1 and 2 generate non-systematic codes:

1. The k info symbols are the coefficients of the polynomial
   $A(D) = A_0 + A_1 D + \ldots + A_{k-1} D^{k-1}$.
   The code word $a(D)$ is obtained by an inversive DFT: $a_i = A(\alpha^i), \; i = 0, \ldots, n-1$.
2. The k info symbols are the coefficients of the polynomial
   $u(D) = u_0 + u_1 D + \ldots + u_{k-1} D^{k-1}$.
   The code word $a(D)$ is obtained by a multiplication with the generator polynomial:
   $a(D) = u(D) \cdot g(D)$.

Encoding Methods for RS Codes

Methods 3 and 4 generate systematic codes:

3. The k info symbols are the coefficients $a_{n-k}, a_{n-k+1}, \ldots, a_{n-1}$.
   The $n-k$ parity check symbols are calculated as follows:
   $$(a_{n-1}D^{n-1} + \ldots + a_{n-k}D^{n-k}) : g(D) = u(D) + \text{remainder}(D)$$
   $$a(D) = a_{n-1}D^{n-1} + \ldots + a_{n-k}D^{n-k} - \text{remainder}(D).$$
4. The k info symbols are the coefficients $a_{n-k}, a_{n-k+1}, \ldots, a_{n-1}$.
   The $n-k$ parity check symbols are calculated as follows:
   $$a_j = -\frac{1}{h_0} \cdot \sum_{i=1}^{k} a_{n-i+j} \cdot h_i, \qquad j = 0, 1, \ldots, n-k-1.$$
   The index $n-i+j$ is taken mod n, and $h(D)$ is the parity check polynomial.

The error detection/correction capabilities are independent of the method.
Of course, given the received word the information is only recoverable if the method is known at the receiver.

Shift of the Parity Frequencies

An RS code $\mathcal{C}$ of length n is defined by the set of polynomials $A(D)$:
$$A(D) = A_0 + A_1D + A_2D^2 + \ldots + A_{k-1}D^{k-1}; \qquad A_k = A_{k+1} = \ldots = A_{n-1} = 0.$$
With $d_{min} = n - k + 1$ we obtain $k = n - d_{min} + 1$.
Therefore, $A_{n-d_{min}+1} = \ldots = A_{n-1} = 0$.
$d_{min} - 1$ consecutive coefficients are zero: “low-pass behavior”.

Now, we perform a cyclic shift of the coefficients by $n-k$ positions to the right:
$$\begin{aligned} A'(D) &= A(D) \cdot D^{n-k} \\ &= A_0D^{n-k} + A_1D^{n-k+1} + \ldots + A_{k-1}D^{n-1} \\ &= A'_{n-k}D^{n-k} + A'_{n-k+1}D^{n-k+1} + \ldots + A'_{n-1}D^{n-1}. \end{aligned}$$
Therefore, $A'_0 = A'_1 = \ldots = A'_{n-k-1} = 0$.
With $k = n - d_{min} + 1$ we finally obtain
$$A'_0 = A'_1 = \ldots = A'_{d_{min}-2} = 0 : \quad \text{“high-pass behavior”}.$$

Singleton Bound

Theorem (**Singleton bound**): For linear $(n, k, d_{min})_q$ codes

$$d_{min} \le n - k + 1.$$

Equality is obtained for so-called **maximum-distance separable** (MDS) codes.

Hence, RS codes are cyclic MDS codes. Therefore,

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

symbol errors can be corrected for sure.

Often, d_{min} is chosen to be an odd number. Then,

$$d_{min} = 2t + 1.$$

Theorem: A shortened MDS code is still an MDS code.

Algebraic Decoding

Let us consider an $(n, k, d_{min})_p$ RS code according to

$$a(D) \circ\!\!-\!\!\bullet\ A(D), \qquad \begin{array}{ll} A_0 = A_1 = \ldots = A_{n-k-1} = 0 & \text{“high-pass behavior”} \\ A_k = A_{k+1} = \ldots = A_{n-1} = 0 & \text{“low-pass behavior”}. \end{array}$$

We define a **code polynomial**: $a(D)$, $a_i \in \mathrm{GF}(p)$, $\deg a(D) < n$

error polynomial: $e(D)$, $e_i \in \mathrm{GF}(p)$, $\deg e(D) < n$

received polynomial: $y(D)$, $y_i \in \mathrm{GF}(p)$, $\deg y(D) < n$

The errors are assumed to be additive:

$y(D) = a(D) \oplus e(D) \qquad \circ\!\!-\!\!\bullet \qquad Y(D) = A(D) + E(D).$

Assumption: No more than $\lfloor (d_{min} - 1)/2 \rfloor$ symbol errors occur.

- If no error occurs ($e(D) = 0$):

$$Y(D) = A(D) \qquad \Rightarrow \qquad \begin{array}{ll} Y_0 = Y_1 = \ldots = Y_{n-k-1} = 0 & \text{“high-pass behavior”} \\ Y_k = Y_{k+1} = \ldots = Y_{n-1} = 0 & \text{“low-pass behavior”}. \end{array}$$

- If errors occur ($e(D) \neq 0$):

$$\begin{array}{lll} Y_i = E_i = S_i & \text{for } i \in \{0, 1, \ldots, n-k-1\} & \text{“high-pass behavior”} \\ Y_{k+i} = E_{k+i} = S_i & \text{for } i \in \{0, 1, \ldots, n-k-1\} & \text{“low-pass behavior”}. \end{array}$$

Algebraic Decoding

$S(D) = S_0 + S_1 D + \ldots + S_{n-k-1} D^{n-k-1}$ is called **syndrome polynomial**.
The syndrome polynomial depends on the error structure only.

Definition (**error positions**): The set of indices i for which $e_i \neq 0$ are called error positions, $0 \leq i < n$.

The problem is to calculate an error polynomial $\hat{e}(D)$ with a minimum number of non-zero coefficients. This property cannot be exploited algebraicly. Therefore, we propose the following solution:

Definition (**error position polynomial**): An error position polynomial $\hat{f}(D)$ has the property

$$\hat{f}_i = 0 \qquad \Leftarrow \qquad \hat{e}_i \neq 0, \qquad \text{i.e.,} \quad \hat{f}_i \cdot \hat{e}_i = 0 \quad \forall\, i \in \{0, \ldots, n-1\}.$$

The coefficients of $\hat{f}(D)$ are zero at the error positions and are arbitrary at all remaining positions, since at the latter positions the coefficients of $\hat{e}(D)$ are zero.

Algebraic Decoding

This rule defines a huge class of polynomials. Among this class of polynomials we select those polynomials, where additionally

$$\hat{f}_i \neq 0 \qquad \Leftarrow \qquad \hat{e}_i = 0,$$

i.e.,

$$\hat{f}_i = 0 \qquad \Leftrightarrow \qquad \hat{e}_i \neq 0 \quad \forall\, i \in \{0, \ldots, n-1\}.$$

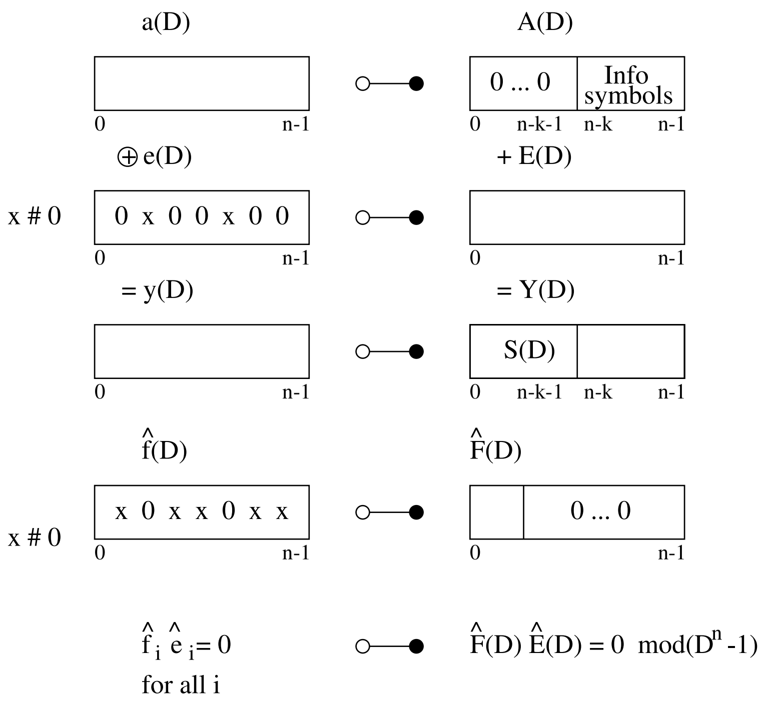
Since the number of coefficients $\hat{f}_i = 0$ is equal to deg $\hat{F}(D)$, where $\hat{f}(D) \circ\!\!-\!\!\bullet\ \hat{F}(D)$, we select

$$\hat{F}(D) := \prod_{\text{error positions } i} (D - \alpha^i).$$

If it is possible to calculate an error position polynomial $\hat{F}(D)$ with a degree as smallest as possible, the error positions i correspond to the roots α^i of $\hat{F}(D)$.
As a constraint, we have

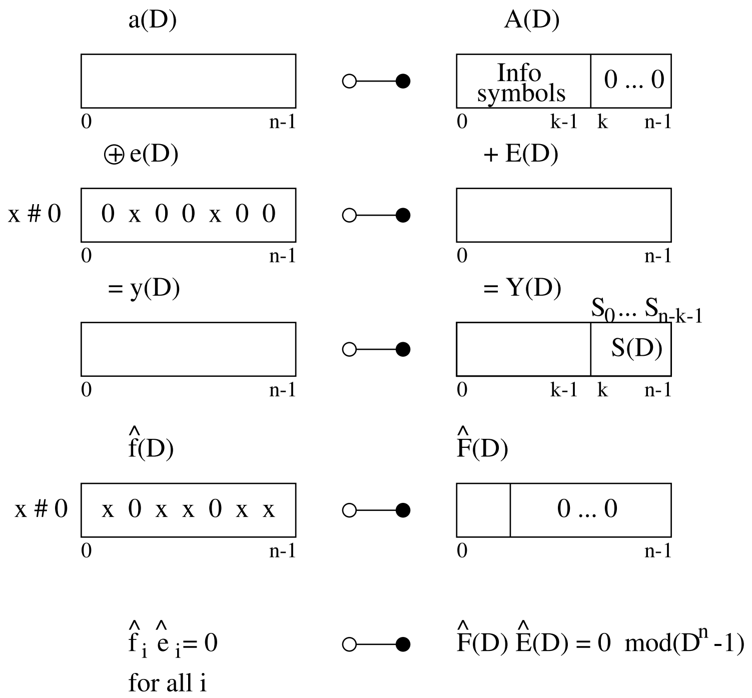
$$\hat{f}_i \cdot \hat{e}_i = 0 \qquad \circ\!\!-\!\!\bullet \qquad \hat{F}(D) \cdot \hat{E}(D) = 0 \quad \mathrm{mod}(D^n - 1).$$

Principle of Algebraic Decoding (High-Pass Behavior)



Given the syndrome polynomial $S(D)$, it is possible to calculate the polynomial $\hat{F}(D)$ with a degree as small as possible.

Principle of Algebraic Decoding (Low-Pass Behavior)



Given the syndrome polynomial $S(D)$, it is possible to calculate the polynomial $\hat{F}(D)$ with a degree as small as possible.

Algebraic Decoding – Summary

The principles of algebraic decoding are as follows:

1. **Calculation of the syndrome polynomial** $S(D)$
2. **Calculation of the error position polynomial** $\hat{F}(D)$ **and the error <u>positions</u>**:
   (linear equations, Berlekamp-Massey algorithm, Euclidean algorithm)
3. **Calculation of the error <u>values</u>**: Calculation of $\hat{E}(D)$ by means of $\hat{F}(D) \cdot \hat{E}(D) = 0 \mod(D^n - 1)$ followed by an inverse DFT: $\hat{e}_i = \hat{E}(\alpha^i)$.
4. **Error correction**: $\hat{a}(D) = y(D) \ominus \hat{e}(D)$. If $t \leq \left\lfloor \frac{d_{min}-1}{2} \right\rfloor$, $\hat{a}(D) = a(D)$.

If one knows which positions in a code word are likely to be wrong (for example by means of concatenated coding), it is only necessary to calculate the error values. The corresponding positions are denoted as **erasures**. Therefore, it is easy to consider erasures in RS decoders.

BCH Codes

Definition (**BCH codes**): A BCH code $\mathcal{C}$ with primitive code word length $n = p^m - 1$ $= q - 1$ is defined by the set of info polynomials

$$A(D) = A_0 + A_1 D + A_2 D^2 + \ldots + A_{k-1} D^{k-1},\ A_i \in \mathrm{GF}(p),\ k < n$$

of degree $< k$, where the p^k code words $\mathbf{a} = [a_0, a_1, \ldots, a_{n-1}]$ are given as $a_i = A(\alpha^i) \in \mathrm{GF}(p)$:

$$\mathcal{C} = \{\mathbf{a} \mid a_i = A(\alpha^i),\ i = 0, 1, \ldots, n-1,\ \deg A(D) < k\}.$$

The minimum distance is $d_{min} \leq n - k + 1$.

- The code symbols are elements of a prime field $\mathrm{GF}(p)$ instead of an extension field $\mathrm{GF}(p^m)$. Typically $p = 2$, i.e., BCH codes typically are binary codes.
- Correspondingly, BCH codes are more suitable for the correction of single codes, whereas RS codes are more suitable for the correction of burst errors.
- The minimum distance of BCH codes is no design parameter.

Further Classes of Block Codes

- Reed-Muller codes
- Golay codes
- CRC codes
- Goppa codes
- Simplex codes
- Fire codes
- Walsh codes
- ...

Differences between Block Codes and Convolutional Codes

Block Codes	Convolutional Codes
Block-wise encoding	Continuous encoding
Algebraic code design	Code design by means of computer search
Algebraic decoding	Viterbi or sequential decoding
Error detection and/or correction	Error correction only
Received word: Arbitrary error structure	Burst errors should be avoided
Powerful high-rate codes exist	High-rate codes are typically not powerful
Soft-decision decoding is difficult	Soft-decision decoding is simple
Applications: For moderate channel error rate	Applications, where soft-decision decoding is superior

Chapter 5: The Compact Disc

- Design of an audio CD
- Sampling of the audio signal
- Channel coding
- Interleaving
- Modulation

The Compact Disc

- The audio CD has been introduced 1982 on the market and is the first mass product employing channel coding
- The high quality is essentially based on error correction and error concealment
- Data is stored in form of
  - Holes (pits) $\Rightarrow$ scattering of the laser beam
  - Plain surface (lands) $\Rightarrow$ reflection of the laser beam
- Spiral track of about 5 km (!) length
- Length of a pit or a land, respectively, about 0.3 μm
- Track width about 0.6 μm
- Distance between tracks about 1 μm
- Sampling speed about 1.2 m/s
- Scattered or reflected laser beam is evaluated with a laser diode

The Compact Disc

- Data rate of the channel bits 4.3218 Mbit/s
- Playtime 74 min $\Rightarrow$ about $1.9 \cdot 10^{10}$ bits are stored (!)
- A channel bit corresponds to a track length of about 0.3 μm
- Scratches typically cause burst errors
- Material defects typically cause single errors
- Hence, the channel decoder must be able to correct single and burst errors
- 44.1 kHz sampling rate (20 kHz audio bandwidth)
- 16 bit A/D conversion $\Rightarrow 2 \cdot 16 \cdot 44.1\ 10^3 = 1.4112$ Mbit/s info data rate
- 6 samples from both stereo channels form an info word
- 8 bits are combined in one symbol (i.e., $q = 2^8$)
- This results in an info word length of $k = 2 \cdot 6 \cdot 16/8 = 24$ symbols

The Compact Disc

- Given a $(255, 251, 5)_{256}$ RS code, by means of shortening the following codes result:
  - $(28, 24, 5)_{256}$ RS code $\mathcal{C}_o$
  - $(32, 28, 5)_{256}$ RS code $\mathcal{C}_i$

  (Shortening means to suppress info bits)
- Since RS codes are maximum-distance separable, the minimal distance of the shortened codes is also $d_{min} = 5$
- The total rate of the serial-concatenated code is $R = R_o \cdot R_i = 24/32 = 3/4$
- The inner decoder is able to correct two single symbol errors (e.g., material defects)
- In order to convert burst errors into single errors, a convolutional deinterleaver ($N = 28$, $J = 4$, 8 bits/symbol) is used
- The single symbol errors (after deinterleaving) are corrected by the outer decoder
- Interleaver and deinterleaver store $N(N-1)J/2 = 1512$ symbols (12096 bits) each

The Compact Disc

- In order to ease clock synchronization, a so called **eight-to-fourteen modulation** (EFM) is used: Inside two ones between two and ten zeros must occur
- A one causes a transition from a pit to a land or vice versa
- Due to three coupling bits it can be maintained that also in a continuous data stream inside two ones between two and ten zeros occur
- Additionally, $24 + 3$ sync bits are inserted
- Altogether, **588 channel bits** per **192 info bits** are generated

  $\Rightarrow$ the effective rate is (only) $R_{eff} = 0.3265$, although the code rate is $R = 3/4$
- In contrast to sampling, coding, modulation, and interleaving the decoder has not been standardized
- Possible quality improvement: Joint EFM demodulation and channel decoding.

Chapter 6: Literature

S. Benedetto, E. Biglieri, *Principles of Digital Transmission*. Boston: Kluwer Academic / Plenum Publishers, 1999.

M. Bossert, *Channel Coding for Telecommunications*. New York: Wiley, 1999.

G.C. Clark, J.B. Cain, *Error-Correcting Coding for Digital Communications*. New York: Plenum Press, 1981.

E.A. Lee, D.G. Messerschmitt, *Digital Communications*. Boston: Kluwer, 1990.

S. Lin, D.J. Costello, *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs (NJ): Prentice Hall, 1983.

A.M. Michelson, A.H. Levesque, *Error-Control Techniques for Digital Communication*. New York: Wiley, 1985.

J.G. Proakis, *Digital Communications*. New York: McGraw-Hill, 4th edition, 2001.

S.B. Wicker, *Error Control Systems for Digital Communications and Storage*. Prentice Hall, 1995.

S.G. Wilson, *Digital Modulation and Coding*. Upper Saddle River (NJ): Prentice Hall, 1996.