Block Codes - an introduction

Valdemar C. da Rocha Jr.

Communications Research Group - CODEC
Department of Electronics and Systems
Federal University of Pernambuco
Recife, BRAZIL

Campinas, 19-20 January 2015

Summary

Introduction

Motivation

- Reliable transmission of data at high speed has represented a constant challenge for both engineers and researchers in telecommunications.
- Error-correcting codes have contributed in a significant way for both the theoretical and technological advances in this area.
- The storage and recovery of large amounts of data in semiconductor memories have also benefited from error-correcting coding techniques.

Introduction

Noise

Frequently in the context of digital communications we face problems of detection or correction of errors caused by noise during transmission, or which have affected stored data.

Example

In a banking data transmission network ideally errors should never occur.

Introduction

Communication systems

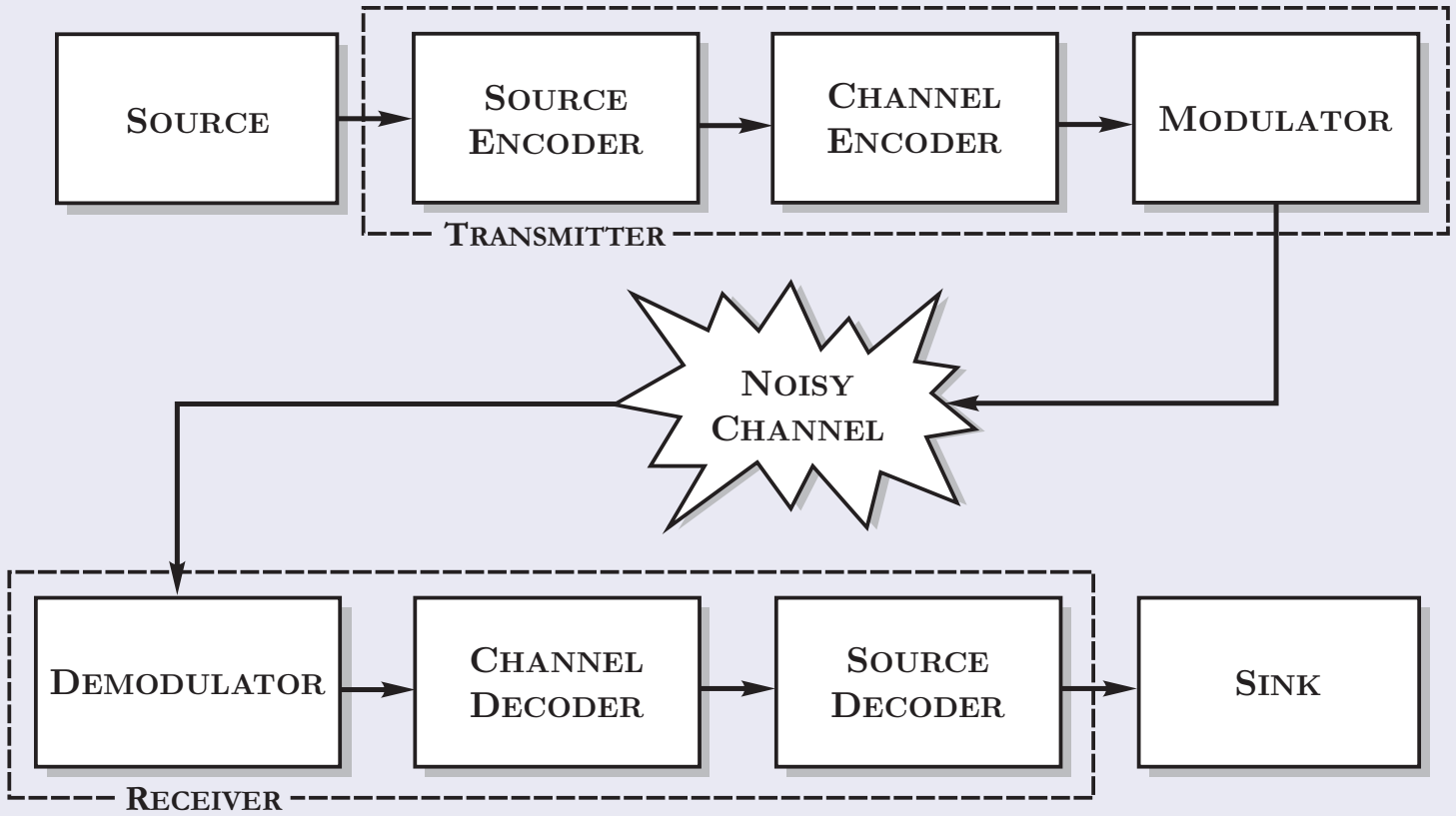
Digital communication systems keep changing their appearance as far as circuits and components are concerned, as a consequence of changes in technology.

Example

Old communication systems evolved from the electromechanical relay to thermionic valves, later to transistors, and so on. A closer look at such systems reveals that, in general, they can be represented by a block diagram as shown next.

Introduction

Digital communication system



Introduction

Errors

Due to the presence of noise errors may occur during transmission or storage of data.

Types of errors

Errors may occur sporadically and independently, in which case they are referred to as *random errors*, or else errors can appear in bursts of many errors each time it occurs, and are called *burst errors*, in which case the channel is said to have a memory.

Introduction

Channel models

- Ideally the receiver should be able to process a continuous signal received from the channel. This situation is modeled by a channel with a discrete input and a continuous output.

- For practical reasons very often the receiver output needs to be quantized into a finite number of levels, typically 8 or 16 levels, which situation is modeled by a discrete channel.

Introduction

Channel models - example

- Two typical discrete channel models are the binary symmetric channel (BSC) and the binary erasure channel (BEC).
- Each binary digit at the BSC output is either correct or assumes its complementary value, while the BEC outputs are either correct binary digits or are erased digits.

Introduction

Linear codes

The parity-check digits in a codeword result from a linear combination involving information digits.

Nonlinear codes

The parity-check digits may result from nonlinear logical operations on the information digits of a codeword, or else result from nonlinear mappings over a given finite field or finite ring, of linear codes over a finite field or finite ring of higher dimension.

Introduction

Types of codes

Depending on how the digits of redundancy are appended to the digits of information, two different types of codes result: block codes and convolutional codes.

Comment

Block codes and convolutional codes are competitive in many practical situations. The final choice of one of them depends on factors such as data format, delay in decoding, system complexity necessary to achieve a given error rate, etc..

Introduction

Block codes

Codes for which redundancy in a block of digits checks the occurrence of errors only in that particular block are called *block codes*.

Convolutional codes

Codes where the redundancy in a block checks the presence or absence of errors in more than one block are called *convolutional codes*, and are a special case of tree codes.

Introduction

Comment

The practical problem in coding theory is not the provision of error-free communications but the design of systems that have an error rate sufficiently low for the user.

Example

An error rate of 10^{-4} for the letters of a book is perfectly acceptable while that same error rate for the digits of a computer operating electronic funds transfer would be disastrous.

Introduction

Comment

The maximum potential of error-correcting codes was established in 1948, with the Shannon coding theorem for a noisy channel.

Theorem

For any memoryless channel, with a discrete input alphabet, there are codes with information rate R (nats/symbol), with codewords of length n digits, for which the probability of decoding error employing maximum likelihood is bounded by $P_e < e^{-nE(R)}$, where $E(R) > 0,\ 0 \leq R < C$, is a decreasing convex-$\cup$ function, specified by the channel transition probabilities and C is the channel capacity.

Introduction

Comments

- The coding theorem proves the existence of codes that can make the probability of erroneous decoding very small, but gives no indication of how to construct such codes.
- Notice that P_e decreases exponentially when n is increased, which usually entails an increase in system complexity.

Goals of coding theory

- Finding long and efficient codes.
- Finding practical methods for encoding and efficient decoding.

Introduction

Comments

- Recent developments in digital hardware technology have made it possible the use of sophisticated coding procedures, and the corresponding circuits can be rather complex.
- The current availability of complex processors makes the advantages derived from the use of codes become even more accessible.

Exercise

- A source produces eight equally likely messages which are encoded into eight distinct codewords as 0000000, 1110100, 1101001, 1010011, 0100111, 1001110, 0011101, 0111010.
- The codewords are transmitted through a BSC with probability of error p, $p < 1/2$.
- Calculate the probability that an error pattern will not be detected at the receiver.

Exercise

Solution

- An error pattern will not be detected if the received word coincides with a codeword.
- For the set of codewords given, notice that the modulo 2 bit by bit addition of any two codewords produces a valid codeword.
- Therefore we conclude that if an error pattern coincides with a codeword it will not be detected at the receiver.
- The probability of undetected error is thus: $(1-p)^7 + 7p^4(1-p)^3$.

Block codes

- Block codes can be easily characterized by their encoding process.
- The process of encoding consists in segmenting the message to be transmitted in blocks of k digits and appending to each block $n-k$ redundant digits.
- These $n-k$ redundant digits are determined from the k-digit message and are intended for just detecting errors, or for detection and correction of errors, or correcting erasures which may appear during transmission.

Block codes

- Block codes may be linear or nonlinear.
- For linear block codes, as mentioned in earlier, the redundant digits are calculated as linear combinations of information digits.
- Linear block codes represent the most well-developed part of the theory of error-correcting codes. One could say that this is in part due to the use of mathematical tools such as linear algebra and the theory of finite fields, or *Galois fields*.

Block codes

- Due to their importance in practice, in what follows we consider mostly binary linear block codes, unless indicated otherwise.
- In general, the code alphabet is q-ary, where q denotes a power of a prime. Obviously for binary codes we have $q = 2$.
- A q-ary (n, k, d) *linear block code* is defined as follows.

Definition

A q-ary (n, k, d) linear block code is a set of q^k q-ary n-tuples, called codewords, where any two distinct codewords differ in at least d positions, and the set of q^k codewords forms a subspace of the vector space of all q^n q-ary n-tuples.

Block codes

Matrix representation

- The code rate R, or code efficiency, is defined as $R = k/n$.
- The codewords can be represented by vectors with n components.
- The components of these vectors are generally elements of a finite field with q elements, represented by $\mathrm{GF}(q)$, also called a *Galois field*.
- Very often we use the binary field, the elements of which are represented by 0 and 1, i.e., $\mathrm{GF}(2)$.

Block codes

Matrix representation

- As already mentioned, a linear code constitutes a subspace and thus any codeword can be represented by a linear combination of the basis vectors of the subspace, i.e., by a linear combination of linearly independent vectors.
- The basis vectors can be written as rows of a matrix, called the code *generator matrix*.
- Given a generator matrix $\mathbf{G}$ of a linear code with k rows and n columns, we can form another matrix $\mathbf{H}$, with $n-k$ rows and n columns, such that the row space of $\mathbf{G}$ is orthogonal to $\mathbf{H}$,

Block codes

Matrix representation

- The $\mathbf{G}$ matrix can be written as

$$\mathbf{G} = [I_k : \mathbf{g}], \tag{1}$$

where $\mathbf{g}$ denotes a $k \times (n-k)$ matrix and I_k denotes the $k \times k$ identity matrix.

- The form of $\mathbf{G}$ in (1) is called the reduced echelon form.

Block codes

Matrix representation

- If $\mathbf{v}_i$ is a vector in the row space of $\mathbf{G}$ then

$$\mathbf{v}_i \mathbf{H}^T = \mathbf{0},\ 0 \leq i \leq 2^k - 1.$$

- The $\mathbf{H}$ matrix is called the code *parity-check matrix* and can be represented as

$$\mathbf{H} = [\mathbf{h} : I_{n-k}],$$

where $\mathbf{h}$ denotes an $(n-k) \times k$ matrix and I_{n-k} is the $(n-k) \times (n-k)$ identity matrix.

Block codes

Matrix representation

- The $\mathbf{g}$ and $\mathbf{h}$ matrices are related by the expression $\mathbf{g} = \mathbf{h}^T$.
- Since the rows of $\mathbf{H}$ are linearly independent, they generate a $(n, n-k, d')$ linear code called the *dual code* of the (n, k, d) code generated by $\mathbf{G}$.
- The code $(n, n-k, d')$ can be considered as the dual space of the (n, k, d) code generated by $\mathbf{G}$.

Block codes

Matrix representation

- Using matrix representation, we find that an encoder has the function of performing the product $\mathbf{mG}$ of a row matrix $\mathbf{m}$, with k elements which represent the information digits, by the $\mathbf{G}$ matrix.
- The result of such an operation is a linear combination of the rows of $\mathbf{G}$ and thus a codeword.
- The ability of simply detecting errors, or error detection and error correction of a code is directly linked to a quantity, defined later, that is its *minimum distance*.

Block codes

Hamming weight and Hamming distance

Before defining minimum distance we define *Hamming weight* of a vector and the *Hamming distance* between two vectors.

Definition

The Hamming weight $W_H(\mathbf{v})$ of a vector $\mathbf{v}$ is the number of non-zero coordinates in $\mathbf{v}$.

Definition

The Hamming distance $d_H(\mathbf{v}_1, \mathbf{v}_2)$ between two vectors, $\mathbf{v}_1$ and $\mathbf{v}_2$, having the same number of coordinates, is the number of positions is which these two vectors differ.

Block codes

Hamming distance

We observe that the Hamming distance is a metric.

Definition

The minimum distance of a code is the smallest Hamming distance between pairs of distinct codewords.

Exercise

Consider the vectors $\mathbf{v}_1 = (0, 1, 0, 0, 2)$ and $\mathbf{v}_2 = (1, 1, 0, 3, 2)$. Compute their respective Hamming weights, $W_H(\mathbf{v}_1)$ and $W_H(\mathbf{v}_2)$, and the Hamming distance $d_H(\mathbf{v}_1, \mathbf{v}_2)$.

Solution

The Hamming weights of $\mathbf{v}_1$ and $\mathbf{v}_2$ are, respectively, $W_H(\mathbf{v}_1) = 2$ and $W_H(\mathbf{v}_2) = 4$ and the Hamming distance between $\mathbf{v}_1$ and $\mathbf{v}_2$ is $d_H(\mathbf{v}_1, \mathbf{v}_2) = 2$.

Block codes

Matrix representation

- Denote by $q = p^m$ the cardinality of the code alphabet, where p is a prime number and m is a positive integer.
- Due to the linearity property, the modulo-q sum of any two codewords of a linear code results in a codeword.
- Suppose that $\mathbf{v}_i, \mathbf{v}_j$ and $\mathbf{v}_l$ are codewords such that $\mathbf{v}_i + \mathbf{v}_j = \mathbf{v}_l$.

Block codes

Matrix representation

- From the definitions of Hamming distance and Hamming weight it follows that $d_H(\mathbf{v}_i, \mathbf{v}_j) = W_H(\mathbf{v}_l)$.
- Hence we conclude that, to determine the minimum distance of a linear code means to find the minimum nonzero Hamming weight among the codewords.
- This last remark brings a great simplification to computing the minimum distance of a linear code because if the code has M codewords, instead of making C_M^2 operations of addition modulo-q and the corresponding Hamming weight calculation, it is sufficient to calculate the Hamming weight of the $M-1$ nonzero codewords only.

Exercise

If d is an odd number, show that by adding an overall parity-check digit to the codewords of a binary linear (n, k, d) code, a $(n+1, k, d+1)$ code results.

Solution

The minimum nonzero weight of a linear code is equal to d, which in this problem is an odd number. Extending this binary code by appending an overall parity-check digit to each codeword will make the weight of every codeword an even number and thus the minimum nonzero weight will become $d+1$. Therefore the minimum distance of the extended code is $d+1$.

Block codes

Matrix representation

- In special situations where the code, besides linearity, has an additional mathematical structure, the determination of the minimum distance, or the determination of upper or lower bounds for the minimum distance can be further simplified.
- In a code with minimum distance d, the minimum number of changes necessary to convert a codeword into another codeword is at least d.
- Therefore, the occurrence of up to $d-1$ errors per codeword during a transmission can be detected, because the result is an n-tuple that does not belong to the code.

Block codes

Matrix representation

- Regarding error correction it is important to note that after detecting the occurrence of errors, we must decide which codeword is more likely to have been transmitted.
- Assuming that the codewords are equiprobable, we decide for the codeword nearest (in terms of Hamming distance) to the received n-tuple.
- Obviously this decision will be correct as long as an error pattern containing up to t errors per codeword occurs, satisfying the relation $2t + 1 \leq d$.

Block codes

Error syndrome and decoding

- Suppose a codeword $\mathbf{v}$ of a linear code with generator matrix $\mathbf{G}$ and parity-check matrix $\mathbf{H}$ is transmitted through a noisy channel.
- The signal associated with $\mathbf{v}$ arriving at the receiver is processed to produce an n-tuple $\mathbf{r}$ defined over the code alphabet.
- The n-tuple $\mathbf{r}$ may differ from $\mathbf{v}$ due to the noise added during transmission.

Block codes

Error syndrome and decoding

- The task of the decoder is to recover $\mathbf{v}$ from $\mathbf{r}$.
- The first step in decoding is to check whether $\mathbf{r}$ is a codeword.
- This process can be represented by the following expression

$$\mathbf{r}\mathbf{H}^T = \mathbf{s},$$

where $\mathbf{s}$ denotes a vector with $n - k$ components, called *syndrome*.

Block codes

Error syndrome and decoding

- If $\mathbf{s} = \mathbf{0}$, i.e., a vector having the $n - k$ components equal to zero, we assume that no errors occurred, and thus $\mathbf{r} = \mathbf{v}$.
- However if $\mathbf{s} \neq \mathbf{0}$, $\mathbf{r}$ does not match a codeword in the row space of $\mathbf{G}$, and the decoder uses this error syndrome for detection, or for detection and correction.
- The received n-tuple $\mathbf{r}$ can be written as

$$\mathbf{r} = \mathbf{v} + \mathbf{e},$$

where $+$ denotes componentwise addition and $\mathbf{e}$ is defined over the code alphabet, denoting an n-tuple representing the error pattern.

Block codes

Error syndrome and decoding

- The decoding process involves a decision about which codeword was transmitted.
- Considering a binary code, a systematic way to implement the decision process is to distribute the 2^n n-tuples into 2^k disjoint sets, each set having cardinality 2^{n-k}, so that each one of them contains only one codeword.
- Thus the decoding is done correctly if the received n-tuple $\mathbf{r}$ is in the subset of the transmitted codeword.

Block codes

Error syndrome and decoding

- The 2^n binary n-tuples are separated into cosets as follows.
- The 2^k codewords are written in one row then, below the all-zero codeword, put an n-tuple $\mathbf{e}_1$ which is not present in the first row.
- Form the second row by adding modulo-2 to $\mathbf{e}_1$ the elements of the first row, as illustrated next.

$$\begin{array}{cccccc} 0 & v_1 & v_2 & \cdots & v_{2^k-1} \\ e_1 & e_1 \oplus v_1 & e_1 \oplus v_2 & \cdots & e_1 \oplus v_{2^k-1}, \end{array}$$

where $\oplus$ denotes modulo-2 addition of corresponding coordinates.

Block codes

Error syndrome and decoding

- Subsequent rows are formed similarly, and each new row begins with an element not previously used.
- In this manner we obtain the *standard array*:

$$
\begin{array}{lllll}
0 & v_1 & v_2 & \cdots & v_{2^k-1} \\
e_1 & e_1 \oplus v_1 & e_1 \oplus v_2 & \cdots & e_1 \oplus v_{2^k-1} \\
e_2 & e_2 \oplus v_2 & e_2 \oplus v_2 & \cdots & e_2 \oplus v_{2^k-1} \\
\vdots & \vdots & \vdots & \cdots & \vdots \\
e_{2^{n-k}-1} & e_{2^{n-k}-1} \oplus v_1 & e_{2^{n-k}-1} \oplus v_2 & \cdots & e_{2^{n-k}-1} \oplus v_{2^k-1}
\end{array}
$$

Block codes

Error syndrome and decoding

- The standard array rows are called *cosets* and the leftmost element in each coset is called a *coset leader*.
- The procedure used to construct the given linear code standard array is called the *coset decomposition* of the vector space of n-tuples over $\mathrm{GF}(q)$.
- In order to use the standard array it is necessary to find the row, and therefore, the associated leader, to which the incoming n-tuple belongs.

Block codes

Error syndrome and decoding

- This is usually not easy to implement because 2^{n-k} can be large, so that the concept of the standard array is most useful as a way to understand the structure of linear codes, rather than a practical decoding algorithm.
- Methods potentially practical for decoding linear codes are presented next.

Block codes

Maximum likelihood decoding

- If the codewords of a (n, k, d) code are selected independently and all have the same probability of being sent through a channel, an optimum way (in a sense we will explain shortly) to decode them is as follows.
- On receiving an n-tuple $\mathbf{r}$, the decoder compares it with all possible codewords. In the binary case, this means comparing $\mathbf{r}$ with 2^k distinct n-tuples that make up the code.
- The codeword nearest to $\mathbf{r}$ in terms of the Hamming distance is selected, i.e., we choose the codeword that differs from $\mathbf{r}$ in the least number of positions.

Block codes

Maximum likelihood decoding

- This chosen codeword is supposedly the transmitted codeword.
- Unfortunately, the time necessary to decode a received n-tuple may become prohibitively long even for moderate values of k.
- It should be noted that the decoder must compare $\mathbf{r}$ with 2^k codewords, for a time interval corresponding to the duration of n channel digits.
- This fact makes this process of decoding inappropriate in many practical cases.

Block codes

Maximum likelihood decoding

- A similar conclusion holds if one chooses to trade search time by a parallel decoder implementation, due to high decoder complexity.
- Let $\mathbf{v}$ denote a codeword and let $P(\mathbf{r}|\mathbf{v})$ denote the probability of $\mathbf{r}$ being received when $\mathbf{v}$ is the transmitted codeword.
- If all codewords have the same probability of being transmitted then the probability $P(\mathbf{v}, \mathbf{r})$ of the pair $(\mathbf{v}, \mathbf{r})$ occurring is maximized when we select that $\mathbf{v}$ which maximizes $P(\mathbf{r}|\mathbf{v})$, known in statistics as the *likelihood function*.

Block codes

Decoding by systematic search

- A general procedure for decoding linear block codes consists of associating each non-zero syndrome with one correctable error pattern.
- One of the properties of the standard array is that all n-tuples belonging to the same coset have the same syndrome.
- Furthermore, each coset leader should be chosen as the most likely error pattern in the respective coset.

Block codes

Decoding by systematic search

- Based in these standard array properties, it is possible to apply the following procedure for decoding.

  1) Calculate the syndrome for the received n-tuple.
  2) By systematic search, find the pattern of correctable errors, i.e., the coset leader, associated with the syndrome of the received n-tuple.
  3) Subtract from the received n-tuple the error pattern found in the previous step, in order to perform error-correction.

Block codes

Decoding by systematic search

- In order to implement this procedure it is necessary to generate successively all configurations of correctable errors and feed them into a combinational circuit, which outputs the corresponding syndromes.

- Using a logic gate with multiple entries, we can detect when the locally generated syndrome coincides with the syndrome of the received n-tuple.

Block codes

Decoding by systematic search

- If a (n, k, d) code corrects t errors per block then the number of distinct configurations of correctable errors that are necessary to generate by this systematic search process is given by

$$C_n^1 + C_n^2 + \cdots + C_n^t = \sum_{i=1}^{t} C_n^i \leq 2^{n-k} - 1. \tag{2}$$

- We observe in (2) that the number of distinct configurations grows rapidly with n and t.

- For this reason, this decoding technique is of limited applicability.

Block codes

Probabilistic decoding

- In recent years, there have appeared in the literature various decoding algorithms of a probabilistic nature, which in principle can operate on unquantized coordinate values of the received n-tuple.
- For practical reasons channel output quantization is employed.
- If the code used is binary and the number of channel output quantization levels is 2 than the decoding technique is called *hard-decision*, otherwise it is called a *soft-decision* decoding technique.

Block codes

Probabilistic decoding

- Hartmann and Rudolph (1976) introduced a probabilistic decoding algorithm which is optimal in the sense that it minimizes the probability of error per digit, when the codewords are equiprobable and are transmitted in the presence of additive noise in a memoryless channel.
- This algorithm is exhaustive in that every codeword of the **dual code** is used in the decoding process.
- This feature makes it practical for use with high rate codes, contrary to what happens with most conventional techniques.

Block codes

Probabilistic decoding

- Another decoding algorithm was introduced (Wolf, 1978) which is a rule to walk in a trellis type structure, and depends on the code $\mathbf{H}$ matrix.
- The received n-tuple is used to determine the most probable path in the trellis, i.e., the transmitted codeword.
- Trellis decoders for block codes for practical applications are addressed in the literature.

Block codes

Simple codes

- In the sequel we present some codes of relatively simple structure, which will allow an understanding of more sophisticated coding mechanisms in future.

Block codes

Binary repetition code

- The code parameters are: $k = 1,\ n - k = c \geq 1$ and $n = k + c = 1 + c$.
- Because $k = 1$, this code has only two codewords, one is a sequence of n zeros and the other is a sequence of n 1's.
- The parity-check digits are all identical and are a repetition of the information digit.

Block codes

Binary repetition code

- A simple decoding rule in this case is to declare as the information digit transmitted the one that most often occurs in the received word. This will always be possible when n is odd.
- If n is even, and there is a tie in the count of occurrence of zeros and 1's, we simply detect the occurrence of errors.
- The minimum distance is $d = n$ and the efficiency (or code rate) is $R = 1/n$. Obviously any pattern with $t \leq \lfloor n/2 \rfloor$ errors is correctable.

Block codes

Binary single parity-check code

- As the title indicates, this code has a single redundant digit per codeword.
- The redundant digit is calculated so as to make even the number of 1's in the codeword.
- We count the number of 1's in the information section. If the result is odd the parity-check digit is made equal to 1, otherwise it is made equal to zero.

Block codes

Binary single parity-check code

- The code parameters are: $k \geq 1,\ n - k = 1$, i.e., $n = k + 1$.
- The Hamming distance and efficiency are respectively $d = 2$ and $R = k/n = k/(k + 1)$.
- The rule used for decoding single parity-check codes is to count the number of 1's in the received word.

Block codes

Binary single parity-check code

- If the resulting count is even, the block received is assumed to be free of errors and is delivered to the recipient.
- Otherwise, the received block contains errors and the recipient is then notified of the fact.
- These codes, while allowing only to detect an odd number of errors, are effective when used in systems that operate with a return channel to request retransmission of messages, or when decoded with soft-decision.

Block codes

Binary Hamming code

We now consider the construction of the $(7, 4, 3)$ Hamming code.

The ideas described are easily generalized to any $(n, k, 3)$ Hamming code.

The number of parity-check digits of the $(7, 4, 3)$ code is $n - k = 7 - 4 = 3$.

Block codes

Binary Hamming code

- Consider the distinct non-zero binary numbers that can be formed with $n - k = 3$ binary digits. That is,

$$\begin{array}{ccccccc} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ c_1 & c_2 & k_1 & c_3 & k_2 & k_3 & k_4. \end{array}$$

- Hamming associated the numbers of the form $2^j,\ j = 0, 1, 2, \ldots$ with parity-check positions.
- The unused numbers were associated with information digits.

Block codes

Binary Hamming code

- Looking alongside the rows in the list, the parity-check equations denoted as $c_i, 1 \leq i \leq 3$, are written as modulo-2 sums of information positions where a 1 appears in the particular row considered, i.e.,

$$\begin{aligned} c_1 &= k_1 \oplus k_2 \oplus k_4 \\ c_2 &= k_1 \oplus k_3 \oplus k_4 \\ c_3 &= k_2 \oplus k_3 \oplus k_4. \end{aligned}$$

- Upon receiving a word, the decoder recalculates the parity-check digits and adds them modulo-2 to their corresponding parity-check digits in the received word in order to obtain the syndrome.

Block codes

Binary Hamming code

- If, for example, an error has hit the digit k_3 the syndrome digits in positions c_2 and c_3 will be 1 and will indicate failure, while in position c_1 no failure is indicated because c_1 does not check k_3.
- The situation is represented as :

$$(c_3, c_2, c_1) = (1, 1, 0),$$

  which corresponds to the column for k_3 on the list considered.
- The error has thus been located and can then be corrected. Obviously, this procedure can be applied to any value of n.

Block codes

Binary Hamming code

- Hamming codes are special in the sense that no other class of non-trivial codes can be so easily decoded and also because they are perfect, as defined next.

Definition

An (n, k, d) error-correcting code over $\mathrm{GF}(q)$, which corrects t errors, is defined as perfect if and only if

$$\sum_{i=0}^{t} (q-1)^i C_n^i = q^{n-k}.$$

Block codes

Perfect codes

- With the exception of Hamming codes, the binary $(23, 12, 7)$ Golay code and the $(11, 6, 5)$ ternary Golay code, there are no other nontrivial linear perfect codes.
- Nonlinear single error-correcting codes, with parameters identical to Hamming codes, were introduced by Vasilev in 1962.

Exercise

Consider the $(7, 3, 4)$ binary linear code having the following expressions for computing the redundant digits, also called parity-check digits.

$$\begin{aligned} c_1 &= k_1 \oplus k_2, \quad c_2 = k_2 \oplus k_3, \\ c_3 &= k_1 \oplus k_3, \quad c_4 = k_1 \oplus k_2 \oplus k_3. \end{aligned}$$

Each block containing three information digits is encoded into a seven digit codeword. Determine the set of codewords for this code.

Exercise

Solution

Employing $c_1 = k_1 \oplus k_2$, $c_2 = k_2 \oplus k_3$, $c_3 = k_1 \oplus k_3$, $c_4 = k_1 \oplus k_2 \oplus k_3$ the following set of codewords results.

MESSAGES				CODEWORDS						
0	0	0		0	0	0	0	0	0	0
0	0	1		0	0	1	0	1	1	1
0	1	0		0	1	0	1	1	0	1
0	1	1		0	1	1	1	0	1	0
1	0	0	$\Rightarrow$	1	0	0	1	0	1	1
1	0	1		1	0	1	1	1	0	0
1	1	0		1	1	0	0	1	1	0
1	1	1		1	1	1	0	0	0	1
k_1	k_2	k_3		k_1	k_2	k_3	c_1	c_2	c_3	c_4

Exercise

Write the generator matrix and the parity-check matrix for the code in the previous exercise, both in reduced echelon form.

Exercise

Solution

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ k_1 & k_2 & k_3 & c_1 & c_2 & c_3 & c_4 \end{bmatrix}.$$

Therefore, it follows that

$$\mathbf{g} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}, \quad \mathbf{h} = \mathbf{g}^T = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Exercise

Solution

The code parity-check matrix $\mathbf{H}$ takes the following form:

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ k_1 & k_2 & k_3 & c_1 & c_2 & c_3 & c_4 \end{bmatrix}$$

Block codes

Low-density parity-check codes

- In 1993 the coding community was surprised with the discovery of turbo codes, more than 40 years after Shannon's capacity theorem, referred to by many as Shannon's *promised land*.
- Turbo codes were the first capacity approaching practical codes.
- Not long after the discovery of turbo codes, their strongest competitors called low-density parity-check (LDPC) codes were rediscovered.

Block codes

Low-density parity-check codes

- LDPC codes have proved to perform better than turbo codes in many applications.
- LDPC codes are linear block codes discovered by Gallager in 1960, which have a decoding complexity that increases linearly with block length.
- At the time of their discovery there was no computational means for their implementation in practice nor to perform computer simulations.

Block codes

Low-density parity-check codes

- Some 20 years later a graphical representation of LDPC codes was introduced (Tanner, 1981) which paved the way to their rediscovery, accompanied by further theoretical advances.
- It was shown that long LDPC codes with iterative decoding achieve a performance, in terms of error rate, very close to the Shannon capacity.

Block codes

Low-density parity-check codes

- LDPC codes have the following advantages with respect to turbo codes.

  1) Do not require a long interleaver in order to achieve low error rates.

  2) Achieve lower block error rates and their error floor occurs at lower bit error rates, for a decoder complexity comparable to that of turbo codes.

Block codes

Low-density parity-check codes

- LDPC codes are defined by their parity-check matrix $\mathbf{H}$.
- Let ρ and γ denote positive integers, where ρ is small in comparison with the code block length and γ is small in comparison with the number of rows in $\mathbf{H}$.

Definition

A binary LDPC code is defined as the set of codewords that satisfy a parity-check matrix $\mathbf{H}$, where $\mathbf{H}$ has ρ 1's per row and γ 1's per column. The number of 1's in common between any two columns in $\mathbf{H}$, denoted by λ, is at most 1, i.e., $\lambda \leq 1$.

Block codes

Low-density parity-check codes

- After their rediscovery by MacKay and Neal (1996) a number of good LDPC codes were constructed by computer search, which meant that such codes lacked in mathematical structure and consequently had more complex encoding than naturally systematic LDPC codes.
- The construction of systematic algebraic LDPC codes based on finite geometries was introduced by Lin and others.

Cyclic Codes

Among the codes in the class of block codes cyclic codes are the most important from the point of view of practical engineering applications.

Cyclic codes are used in communication protocols, in music CDs, in magnetic recording, etc.. This is due to their structure being based on discrete mathematics, which allows a considerable simplification in the implementation of encoders and decoders.

The formal treatment of cyclic codes is done in terms of polynomial rings, with polynomial coefficients belonging to a Galois field $\mathrm{GF}(q)$, modulo $x^n - 1$, where n denotes the block length.

Definition

A block code is called a cyclic code whenever a cyclic shift, applied to any of its codewords, produces a codeword in the same code, i.e., if $\mathbf{v} = (v_0, v_1, v_2, \ldots, v_{n-1})$ is a codeword then

$$\mathbf{v}^i = (v_{n-i}, v_{n-i+1}, \ldots, v_0, v_1, \ldots, v_{n-i-1})$$

obtained by shifting $\mathbf{v}$ cyclically by i places to the right, is also a codeword in the same code, where the indices in v are reduced modulo n.

An n-tuple $\mathbf{v}$ can be represented by a polynomial of degree at most $n-1$ as follows:

$$v(x) = v_0 + v_1 x + v_2 x^2 + \cdots + v_{n-1} x^{n-1}.$$

Matrix representation

Using properties of finite fields it can be shown that all the codewords of a (n, k, d) cyclic code are multiples of a well defined polynomial $g(x)$, of degree $n - k$, and conversely that all polynomials of degree at most $n - 1$ which are divisible by $g(x)$ are codewords of this code.

The polynomial $g(x)$ is called the code *generator polynomial* and is a factor of $x^n - 1$.

Each codeword of a cyclic code is a multiple of the code generator polynomial $g(x)$. It follows that the polynomials $g(x),\ xg(x),\ x^2g(x), \ldots,\ x^{k-1}g(x)$ are codewords. We also note that such codewords in particular are linearly independent.

The codewords $g(x),\ xg(x),\ x^2g(x), \ldots,\ x^{k-1}g(x)$ (linearly independent) can be used to construct a generator matrix $\mathbf{G}$ for the cyclic code which has $g(x)$ as its generator polynomial, as follows.

$$\mathbf{G} = \begin{bmatrix} x^{k-1}g(x) \\ \vdots \\ x^2g(x) \\ xg(x) \\ g(x) \end{bmatrix},$$

where we assume that each row of $\mathbf{G}$ contains n elements, consisting of the coefficients of the corresponding row polynomial and the remaining empty positions are filled with zeros.

For encoding purposes, the cyclic shift property of cyclic codes allows a sequential implementation of the $\mathbf{G}$ matrix which is presented next.

Encoder with $n-k$ stages of shift-register

This encoding procedure is based on the property that each codeword in a cyclic code is a multiple of the code generator polynomial $g(x)$. The k information digits can be represented by a polynomial $I(x)$, of degree at most $k-1$.

Multiplying the polynomial $I(x)$ by x^{n-k} we obtain $I(x)x^{n-k}$, which is a polynomial of degree at most $n-1$ which does not contain nonzero terms of degree lower than $n-k$.

Dividing $I(x)x^{n-k}$ by $g(x)$ we obtain:

$$I(x)x^{n-k} = Q(x)g(x) + R(x).$$

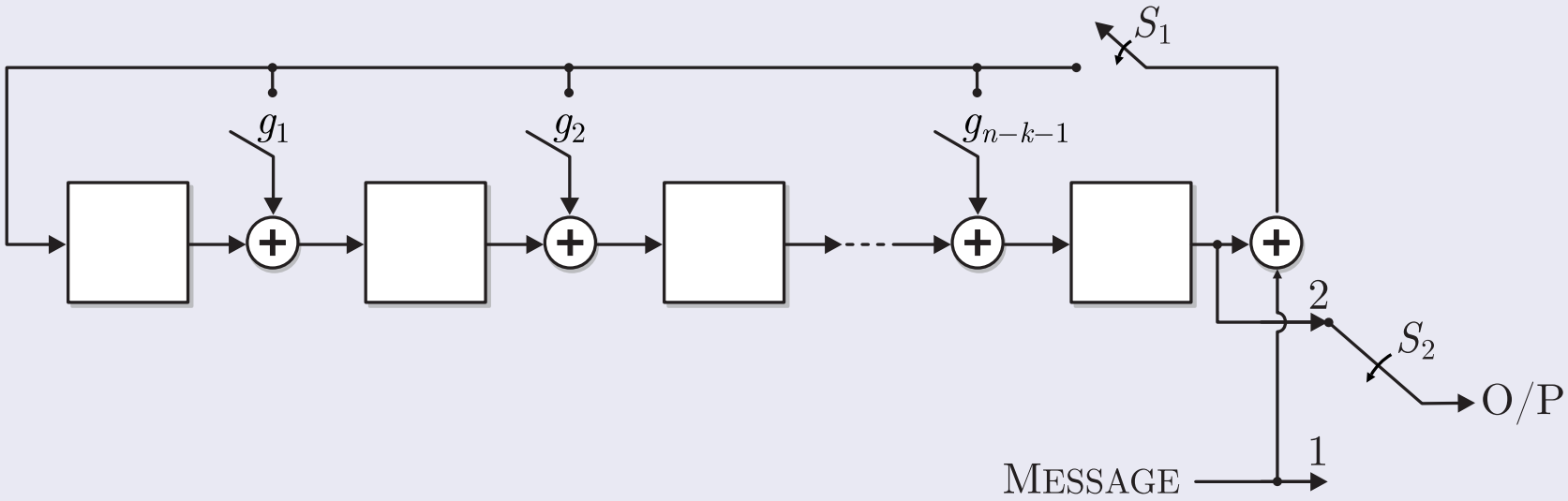
Polynomials $Q(x)$ and $R(x)$ are, respectively, the quotient polynomial and the remainder polynomial. $R(x)$ has degree lower than $g(x)$, i.e., $R(x)$ has degree at most $n-k-1$.

If $R(x)$ is subtracted from $I(x)x^{n-k}$, the result is a multiple of $g(x)$, i.e., the result is a codeword.

$R(x)$ represents the parity-check digits and has got no terms overlapping with $I(x)x^{n-k}$, as follows from our earlier considerations.

Let $g(x) = x^{n-k} + g_{n-k-1}x^{n-k-1} + \cdots + g_1 x + 1$. The circuit shown next employs $n-k$ stages of a shift-register and pre-multiplies the information polynomial $I(x)$ by x^{n-k}.

Encoder for a binary cyclic code



Another sequential encoding procedure exists for cyclic codes based on the polynomial $h(x) = (x^n - 1)/g(x)$, which employs k stages of shift-register. We chose not to present this procedure here, however it can be easily found in the coding literature.

Cyclic Hamming codes

The Hamming codes seen have a cyclic representation. Cyclic Hamming codes have a primitive polynomial $p(x)$ of degree m as their generator polynomial, and have the following parameters:

$$n = 2^m - 1,\ k = 2^m - m - 1,\ d = 3.$$

Cyclic Hamming codes are easily decodable by a Megitt decoder, or by an error-trapping decoder, which are described later.

Because Hamming codes are perfect codes, very often they appear in the literature in most varied applications, as for example their codewords being used as protocol sequences for the collision channel without feedback.

Maximum-length-sequence codes

The maximum period possible for a q-ary sequence generated by a shift-register of m stages, employing linear feedback, is $q^m - 1$.

We now look at binary maximum-length-sequence (m-sequence) codes, i.e., we consider $q = 2$.

Maximum-length-sequence codes

The m-sequence codes are cyclic, are dual codes of Hamming codes, and have the following parameters:

$$n = 2^m - 1,\ k = m,\ d = 2^{m-1},\ \text{for } m \geq 2.$$

The generator polynomial of an m-sequence code has the form $g(x) = (x^n - 1)/p(x)$, where $p(x)$ denotes a degree m primitive polynomial.

Maximum-length-sequence codes

The dictionary of an m-sequence code has an all-zeros codeword, and n non-zero codewords which result from n cyclic shifts of a non-zero codeword. It follows that all non-zero codewords have the same Hamming weight.

The m-sequence codes are also called equidistant codes or simplex codes.

Maximum-length-sequence codes

The m-sequence codes are completely orthogonalizable in one-step and as a consequence they are easily decodable by majority logic.

The non-zero codewords of an m-sequence code have many applications, including direct sequence spread spectrum, radar and location techniques.

Bose-Chaudhuri-Hocquenghem codes

Bose-Chaudhuri-Hocquenghem codes (BCH codes) were discovered independently and described in papers by Bose and Chaudhury (1960) and Hocquenghem (1959).

The BCH codes are cyclic codes and represent one of the most important classes of block codes having algebraic decoding algorithms.

Bose-Chaudhuri-Hocquenghem codes

For any two given positive integers $m,\ t$ there is a BCH code with the following parameters:

$$n = q^m - 1,\ n - k \leq mt,\ d \geq 2t + 1.$$

The BCH codes can be seen as a generalization of Hamming codes, capable of correcting multiple errors in a codeword. One convenient manner of defining BCH codes is by specifying the roots of the generator polynomial.

Bose-Chaudhuri-Hocquenghem codes

Definition

A primitive BCH code over $\mathrm{GF}(q)$, capable of correcting t errors, having block length $n = q^m - 1$, has as roots of its generator polynomial $g(x)$, α^{h_0}, α^{h_0+1}, $\ldots$, α^{h_0+2t-1}, for any integer h_0, where α denotes a primitive element of $\mathrm{GF}(q^m)$.

It follows that the generator polynomial $g(x)$ of a BCH code can be written as the least common multiple (LCM) of the minimal polynomials.

$$g(x) = \mathrm{LCM}\{m_0(x), m_1(x), ..., m_{2t-1}(x)\},$$

where $m_i(x)$ denotes the minimal polynomial of α^{h_0+i}, $0 \leq i \leq 2t-1$.

Bose-Chaudhuri-Hocquenghem codes

When α is not a primitive element $\mathrm{GF}(q^m)$ the resulting codes are called non-primitive BCH codes.

It follows that the respective block length is given by the multiplicative order of α. BCH codes with $h_0 = 1$ are called *narrow sense BCH codes*.

An alternative definition for BCH codes can be given in terms of the finite field Fourier transform of the generator polynomial $g(x)$.

Bose-Chaudhuri-Hocquenghem codes

The roots $\alpha^{h_0+i},\ 0 \leq i \leq 2t-1$, of $g(x)$ correspond to the zero components in the spectrum $G(z)$, in positions $h_0+i,\ 0 \leq i \leq 2t-1$.

Definition

A primitive BCH code over $\mathrm{GF}(q)$, capable of correcting t errors, having block length $n = q^m - 1$, is the set of all codewords over $\mathrm{GF}(q)$ whose spectrum is zero in $2t$ consecutive components $h_0+i,\ 0 \leq i \leq 2t-1$.

The $2t$ consecutive roots of $g(x)$ or, equivalently, the $2t$ null spectral components of $G(z)$ guarantee a minimum distance $\delta = 2t+1$, called the designed distance of the code, as shown next in a theorem known as the *BCH bound theorem*.

Bose-Chaudhuri-Hocquenghem codes

Theorem

Let n be a divisor of $q^m - 1$, for some positive integer m. If any non-zero vector $\mathbf{v}$ in $\mathrm{GF}(q)^n$ has a vector spectrum $\mathbf{V}$ with $d-1$ consecutive null components, $V_j = 0,\ j = h_0, h_0+1, \ldots, h_0+d-2$, then $\mathbf{v}$ has at least d non-zero components.

Reed-Solomon codes

Non-binary BCH codes

Reed-Solomon (RS) codes are non-binary BCH codes, with $m = h_0 = 1$, defined by the parameters:

$$n = q - 1,\ n - k = 2t,\ d = 2t + 1.$$

Generator polynomial

Given an element α primitive in $\mathrm{GF}(q)$, the generator polynomial of an RS code has the following form:

$$g(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^3)...(x - \alpha^{2t}).$$

Reed-Solomon codes

Binary mapping of RS codes

Many practical applications employ binary digits, therefore RS codes with $q = 2^r$ are a natural choice, having each 2^r-ary symbol represented by r binary digits.

Maximum distance separable codes

Since their minimum distance is equal to $n - k + 1$, RS codes constitute a class of *maximum distance separable* codes.

Reed-Solomon codes

Correction of random and phased bursts of errors

When mapped to binary, an RS code defined over $\mathrm{GF}(2^r)$ becomes a binary code of block length nr which is capable of correcting both random errors and burst errors.

Serial concatenation

Very often RS codes are employed as outer codes in serially concatenated coding schemes.

Golay codes

Code parameters

The Golay codes have the following parameters $n = 23,\ k = 12,\ d = 7$, with a binary alphabet, and $n = 11,\ k = 6,\ d = 5$, with a ternary alphabet.

Perfect codes with $t > 1$

The Golay codes are the only existent perfect codes with $t > 1$.

The binary $(23, 12, 7)$ Golay code

Let α be a primitive element of $\mathrm{GF}(2^{11})$ and notice that $2^{11} - 1 = 89 \times 23$.

It follows that $\beta = \alpha^{89}$ is a non-primitive element of order 23.

The $(23, 12, 7)$ binary cyclic Golay code is specified by the generator polynomial $g(x)$ having β as a root.

The binary $(23, 12, 7)$ Golay code

Roots of the minimal polynomial of β

The roots of the minimal polynomial of β are:

$$\beta, \beta^2, \beta^4, \beta^8, \beta^{16}, \beta^9, \beta^{18}, \beta^{13}, \beta^3, \beta^6, \beta^{12},$$

where the fact that $\beta^{23} = 1$ has been used.

Factoring of $x^{23} + 1$ over $\mathrm{GF}(2)$

The factoring of $x^{23} + 1$ over $\mathrm{GF}(2)$ produces $x^{23} + 1 =$
$(x+1)(x^{11}+x^{10}+x^6+x^5+x^4+x^2+1)(x^{11}+x^9+x^7+x^6+x^5+x+1)$.

The binary $(23, 12, 7)$ Golay code

Degree 11 factors of $x^{23} + 1$

Depending on the choice of the primitive polynomial that generates $\mathrm{GF}(2^{11})$, one of the two degree 11 factors of $x^{23} + 1$ will have β, β^2, β^3 and β^4 as the longest string of consecutive roots while the other degree 11 factor will have β^{19}, β^{20}, β^{21} and β^{22} as the longest string of consecutive roots.

Any of these two degree 11 factors of $x^{23} + 1$ can be the code generator polynomial.

The binary $(23, 12, 7)$ Golay code

BCH bound

Notice that by the BCH bound this code has a designed distance $\delta = 5$, while the code true minimum distance is 7.

The extended $(24, 12, 8)$ binary Golay code

The extended $(24, 12, 8)$ binary Golay code is obtained from the $(23, 12, 7)$ Golay code by appending to each codeword an overall parity-check digit, i.e., by appending a digit that makes even the total number of 1's in a codeword.

The ternary $(11, 6, 5)$ Golay code

Non-primitive BCH code over $\mathrm{GF}(3)$

The $(11, 6, 5)$ ternary Golay code can also be seen as a non-primitive BCH code over $\mathrm{GF}(3)$.

Roots of $x^{11} - 1$

If β is a primitive element in $\mathrm{GF}(3^5)$ then $(\beta^{22})^{11} = \beta^{242} = 1$.

Therefore, by considering $\alpha = \beta^{22}$, it follows that the powers $\alpha^i = (\beta^{22})^i$, $0 \leq i \leq 10$, are the roots of $x^{11} - 1$.

The ternary $(11, 6, 5)$ Golay code

Cyclotomic classes

The roots of $x^{11} - 1$ split into cyclotomic classes as $\{1\}$, $\{\alpha, \alpha^3, \alpha^9, \alpha^5, \alpha^4\}$, $\{\alpha^2, \alpha^6, \alpha^7, \alpha^{10}, \alpha^8\}$.

Irreducible factorization of $x^{11} - 1$ over $\mathrm{GF}(3)$

The binomial $x^{11} - 1$ factors into irreducible polynomials over $\mathrm{GF}(3)$ as

$$x^{11} - 1 = (x - 1)(x^5 + x^4 + 2x^3 + x^2 + 2)(x^5 + 2x^3 + x^2 + 2x + 2),$$

where the roots of $x^5 + x^4 + 2x^3 + x^2 + 2$ are $\{\alpha, \alpha^3, \alpha^9, \alpha^5, \alpha^4\}$ and the roots of $x^5 + 2x^3 + x^2 + 2x + 2$ are $\{\alpha^2, \alpha^6, \alpha^7, \alpha^{10}, \alpha^8\}$.

The ternary $(11, 6, 5)$ Golay code

Ternary cyclic $(11, 6, 5)$ Golay code

Consider the ternary cyclic $(11, 6, 5)$ Golay code with generator polynomial $g(x) = x^5 + x^4 + 2x^3 + x^2 + 2$.

BCH bound

The set of roots of $g(x)$ contains a maximum of three consecutive roots, namely α^3, α^4 and α^5, and thus by the BCH bound it guarantees a minimum Hamming distance of 4 but not 5 as required for a double-error correcting code.

The ternary $(11, 6, 5)$ Golay code

We can also consider the $(11, 6, 5)$ ternary Golay code generated by $x^5 + 2x^3 + x^2 + 2x + 2$, having α^6, α^7 and α^8 as its longest string of consecutive roots of the generator polynomial.

Extended $(12, 6, 6)$ ternary Golay code

The extended $(12, 6, 6)$ ternary Golay code is obtained from the $(11, 6, 5)$ Golay code by appending to each codeword an overall parity-check digit, i.e., by appending a digit that makes equal to zero the modulo 3 sum of the digits in a codeword.

Reed-Muller codes

Reed-Muller (RM) codes are binary codes which are equivalent to cyclic codes with an overall parity-check digit attached.

Hadamard product of vectors of the same length

The vector whose components are the product of the corresponding components of the factors.

Example

The $\mathbf{abc}$ Hadamard product of vectors $\mathbf{a} = (a_1, a_2, \ldots, a_n)$, $\mathbf{b} = (b_1, b_2, \ldots, b_n)$ and $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ is given by:

$$\mathbf{abc} = (a_1\, b_1\, c_1, a_2\, b_2\, c_2, ..., a_n\, b_n\, c_n).$$

Reed-Muller codes

Let $\mathbf{v_0}$ be the vector having 2^m components, all of which are equal to 1. Let $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_m}$ be vectors forming the rows of a matrix whose 2^m columns are all the distinct binary m-tuples.

Definition

The RM code of order r is defined by the generator matrix whose rows are the vectors $\mathbf{v_0}, \mathbf{v_1}, \ldots, \mathbf{v_m}$ and their respective Hadamard products two at a time, three at a time, ..., r at a time. For any positive integer m, the RM code of order r has the following parameters:

$$n = 2^m, \; k = \sum_{i=0}^{r} C_m^i, \; d = 2^{m-r}.$$

Reed-Muller codes

The RM codes are a large class of codes but have an error correction power in general lower than that of BCH codes of equivalent rate.

Two important aspects characterize RM codes.

1. They are easily decodable by majority logic.
2. They are a sub-class of codes constructed from the Euclidean geometry.

Quadratic residue codes

Quadratic residues

Before defining quadratic residue codes it is necessary to introduce *quadratic residues*.

Definition

An integer r is a quadratic residue of a prime number p, if and only if there exists an integer s such that $s^2 \equiv r \mod p$.

Quadratic residue codes

It can be shown that if $n = 8m \pm 1$ is a prime number then 2 is a quadratic residue of n. In this case $x^n + 1$ can be factored as $(x+1)\,G_r(x)\,G_{\overline{r}}(x)$, where:

$$G_r(x) = \prod_{r \in R_0} (x + \alpha^r) \quad \text{and} \quad G_{\overline{r}}(x) = \prod_{\overline{r} \in \overline{R_0}} (x + \alpha^{\overline{r}}),$$

where α denotes an element of multiplicative order n in an extension field of $\mathrm{GF}(2)$, R_0 denotes the set of quadratic residues modulo n and $\overline{R_0}$ denotes the set of quadratic non-residues modulo n.

The cyclic codes with generator polynomials $G_r(x)$, $(1+x)\,G_r(x)$, $G_{\overline{r}}(x)$ and $(x+1)\,G_{\overline{r}}(x)$ are called *quadratic residue codes*.

Quadratic residue codes

An efficient way of decoding quadratic residue codes is by means of permutations. Permutation decoders are usually more complex than algebraic BCH decoders.

However, the minimum distance of some quadratic residue codes of moderate length is greater than the minimum distance of BCH codes of comparable block length.

Alternant codes

The class of alternant codes is of great importance for including, as particular cases, the BCH codes, Goppa codes, Srivastava codes and Chien-Choy codes.

Alternant codes are constructed by a simple modification of the parity-check matrix of a BCH code.

Alternant codes

A BCH code of block length n and designed distance δ over $\mathrm{GF}(q)$ has a parity-check matrix $\mathbf{H} = [h_{ij}]$, where $h_{ij} = \alpha^{ij}$, $1 \leq i \leq \delta - 1, 0 \leq j \leq n - 1$, and $\alpha \in \mathrm{GF}(q^m)$ is a primitive nth root of unity.

Alternant codes are obtained by considering $h_{ij} = x_j^{i-1} y_j$, where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is a vector with distinct components, belonging to $\mathrm{GF}(q^m)$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ is a vector with non-zero components, also belonging to $\mathrm{GF}(q^m)$.

Exercise

Consider the $(2^m - 1, 2^m - m - 2, 3)$ binary cyclic Hamming code generated by $p(x)$, a primitive polynomial of degree m. Show that this code is capable of correcting any pattern containing at most two erasures.

Solution: By changing the values of at most two positions in a codeword of a code with minimum distance 3 cannot produce another codeword. Therefore by trial and error, at most four binary patterns need to be tested as replacements for two erasures, which are corrected in this manner.

Exercise

Construct a table with the elements of $\mathrm{GF}(8)$ expressed as powers of a primitive element α which is a root of $x^3 + x + 1$.

Solution:

Exponential form	Polynomial form
0	0
1	1
α	α
α^2	α^2
α^3	$\alpha + 1$
α^4	$\alpha^2 + \alpha$
α^5	$\alpha^2 + \alpha + 1$
α^6	$\alpha^2 + 1$

Exercise

Construct a table with the elements of $\mathrm{GF}(16)$ expressed as powers of a primitive element α which is a root of $x^4 + x + 1$.

Solution:

Exp. form	Poly. form	Exp. form	Poly. form
0	0	α^7	$\alpha^3 + \alpha + 1$
1	1	α^8	$\alpha^2 + 1$
α	α	α^9	$\alpha^3 + \alpha$
α^2	α^2	α^{10}	$\alpha^2 + \alpha + 1$
α^3	α^3	α^{11}	$\alpha^3 + \alpha^2 + \alpha$
α^4	$\alpha + 1$	α^{12}	$\alpha^3 + \alpha^2 + \alpha + 1$
α^5	$\alpha^2 + \alpha$	α^{13}	$\alpha^3 + \alpha^2 + 1$
α^6	$\alpha^3 + \alpha^2$	α^{14}	$\alpha^3 + 1$

Exercise

Let α be a root in $\mathrm{GF}(8)$ of the binary primitive polynomial $p(x) = x^3 + x + 1$. Determine the generator polynomial $g(x)$ of an RS code in $\mathrm{GF}(8)$ with minimum distance 5, such that $g(\alpha) = 0$.

Solution: The generator polynomial $g(x)$ is required to have four consecutive roots, where α is one of them. Using the roots $\alpha, \alpha^2, \alpha^3$ and α^4, and using a table of elements of $\mathrm{GF}(8)$ to simplify the result we obtain $g(x) = x^4 + \alpha^3 x^3 + x^2 + \alpha x + \alpha^3$. Verify that $\alpha^4, \alpha^5, \alpha^6$ and α, although not all consecutive roots, offer another solution to this problem.

Exercise

Determine the generator polynomial $g(x)$ of the $(15, 5, 7)$ binary BCH code, having roots α^i, $1 \leq i \leq 6$.

Solution: Let $g(x) = m_1(x)m_3(x)m_5(x)$ denote the generator polynomial. Using a table of elements of $\mathrm{GF}(16)$ we obtain $m_1(x) = x^4 + x + 1$, $m_3(x) = x^4 + x^3 + x^2 + x + 1$, and $m_5(x) = x^2 + x + 1$. Simplifying we obtain $g(x) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1$.

Decoding cyclic codes

The decoding procedures for linear block codes are also applicable to cyclic codes.

The algebraic properties associated with the cyclic structure allow important simplifications when implementing a decoder for a cyclic code, both for calculating the syndrome and for correcting errors.

Decoding cyclic codes

The syndrome computation consists in dividing by the generator polynomial $g(x)$ the polynomial representing the word received from the channel.

The remainder of this division is the syndrome, denoted by $s(x)$.

- If $s(x) = 0$, the received word is accepted as being a codeword.
- If $s(x) \neq 0$, declare that errors have occurred.

Decoding cyclic codes

Error detection

It follows that a circuit to detect errors with a cyclic code is rather simple.

Error correction

The problem of locating error positions in a received word for correction in general requires more elaborate techniques in order to be implemented in practice.

Berlekamp-Massey algorithm and the Euclidean algorithm

So far, the most important algebraic decoding techniques for cyclic codes are those based on the Berlekamp-Massey algorithm and on the Euclidean algorithm.

In the sequel we give a brief presentation of the more relevant decoding procedures for cyclic codes.

Meggitt decoder

The Meggitt decoder employs a circuit to identify those syndromes that correspond to error patterns containing an error in the highest order position of the received word, i.e., an error at position x^{n-1}.

When the digit at position x^{n-1} is being delivered to the sink it can be altered or not, depending on the decision determined by the circuit that identifies errors in that position.

Meggitt decoder

By cyclically shifting the received word, digit by digit with a shift-register, all digits will necessarily occupy position x^{n-1} after a certain number of shifts, and will be examined by the circuit that identifies errors in that position.

The decision for choosing a Meggitt decoder depends on the complexity of the circuit that identifies errors at position x^{n-1} of the received word.

A straightforward way of implementing a Meggitt decoder is by the use of programmable memories.

Error-trapping decoder

The decoding algorithm known as *error trapping* operates by cyclically shifting the syndrome shift-register, bit by bit, loaded with the syndrome of the received word, and computing its Hamming weight at each step.

If a syndrome of Hamming weight at most t (number of correctable errors) is detected, then the corresponding length $n - k$ segment of the error vector, containing all the errors, coincides with this shifted syndrome.

Error-trapping decoder

Once the error pattern is identified, or trapped, error correction is immediate.

However, a situation may occur where, after n cyclic shifts, the Hamming weight of the syndrome was never t or less.

In this case the decoder declares the occurrence of an error pattern that spreads over a length of cyclically consecutive positions greater than $n - k$.

Error-trapping decoder

This decoding procedure is more appropriate for use with low rate codes, since for a code with block length n, having k information digits and capable of correcting t errors per block, the efficient application of error trapping requires the condition

$$n/k = 1/R > t$$

to be satisfied.

Information set decoding

In a linear (n, k, d) code any set of k positions that can be independently specified in a codeword constitutes an *information set*.

As a consequence, the symbols in an information set define a codeword.

If an information set in a received n-tuple contains no errors then it is possible to reconstitute the transmitted codeword.

Information set decoding

The decoding algorithm based on information sets consists of the following steps:

1) Construct several information sets for the given code.
2) Form various estimates of the transmitted codeword, by decoding the received word using each of the information sets obtained in the previous step.
3) Compare the received word with the estimates obtained in the previous step and decide for the codeword nearest to the received word.

Threshold decoding

Majority logic decoding

Threshold decoding is also known as *majority logic decoding*. A great number of applications of this technique concentrates on cyclic codes.

Before describing threshold decoding it is necessary to introduce the concept of *parity-check sums* or simply *parity sums*.

Threshold decoding

Syndrome

The syndrome is represented by a vector having $n-k$ coordinates, $\mathbf{s} = (s_0, s_1, \ldots, s_{n-k-1})$, where, for a given (n, k, d) linear code with parity-check matrix $\mathbf{H} = [h_{ij}]$ and an error vector $(e_0, e_1, \ldots, e_{n-1})$ we have the relation:

$$\mathbf{s} = \mathbf{e}\mathbf{H}^T,$$

where each syndrome component $s_j,\ 0 \leq j \leq n-k-1$, is given by

$$s_j = \sum_{i=0}^{n-1} e_i h_{ij}.$$

Threshold decoding

Parity-check sum

The linear combination of syndrome digits $A = \sum_{i=0}^{n-k-1} a_i s_i$, where a_i is either 0 or 1, with the help of $s_j = \sum_{i=0}^{n-1} e_i h_{ij}$ can be written as

$$A = \sum_{i=0}^{n-1} b_i e_i, \tag{3}$$

where b_i is either 1 or 0.

An error in position e_i is said to be checked by A if the corresponding coefficient b_i in (3) is 1. Expression (3) is called a *parity-check sum*.

Orthogonal set

Definition

Given a set of J parity-check sums $A_1, A_2, \ldots, A_J$, such that a position e_l in the error vector is checked by all parity sums and all other positions $e_i,\ i \neq l$, in the error vector are checked at most once, then this set is said to be orthogonal on position e_l.

Threshold decoding

Assuming the occurrence of $t \leq \lfloor J/2 \rfloor$ errors in the received word, the threshold decoding procedure is based on the following reasoning:

i) $e_l = 1$. This means that the remaining $t-1$ errors will affect at most $t-1 \leq \lfloor J/2 \rfloor - 1$ of the J parity sums, thus leaving at least $J - \lfloor J/2 \rfloor + 1 = \lceil J/2 \rceil + 1$ parity sums equal to 1.

ii) $e_l = 0$. In this case the t errors will affect at most $\lfloor J/2 \rfloor$ parity sums, i.e., half of the parity sums in the worst case situation.

Threshold decoding

The threshold decoding rule consists in making $e_l = 1$, i.e., to declare the presence of an error in this position, whenever the majority of the parity sums are equal to 1.

Otherwise, make $e_l = 0$, i.e., in case the number of parity sums equal to 1 coincides with the number of parity sums equal to 0, or if the majority of the parity sums are equal to 0.

Threshold decoding

For a cyclic code, after decoding a given position in a codeword a cyclic shift is applied to that codeword and the same set of J parity-check sums are used to decode position e_{l-1}, and so on in a similar manner for decoding the remaining codeword positions until the complete codeword is decoded.

Codes for which $J = d - 1$ are said to be completely orthogonalizable.

Threshold decoding

However, it is not always possible to obtain directly J parity sums orthogonal on a given position of a codeword.

For various classes of codes orthogonal parity sums are obtained in L steps, where at each step one uses parity sums orthogonal on a sum of codeword positions.

Exercise

Consider the binary maximum-length sequence $(7, 3, 4)$ code generated by $g(x) = x^4 + x^3 + x^2 + 1$. Determine three parity-check sums, orthogonal on position e_6.

Solution: The code parity-check matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ e_6 & e_5 & e_4 & e_3 & e_2 & e_1 & e_0 \end{bmatrix}$$

Exercise

$$\begin{array}{lllll} A_1 & = & s_3 & = & e_6 + e_4 + e_3 \\ A_2 & = & s_1 & = & e_6 + e_5 + e_1 \\ A_3 & = & s_2 + s_0 & = & e_6 + e_2 + e_0. \end{array}$$

The three parity-check sums, A_1, A_2 and A_3, orthogonal on position e_6, are obtained.

Since $J = 3 = d - 1$, we conclude that this code is completely orthogonalizable in one-step, and by threshold decoding it corrects one error per codeword.

Exercise

Consider the binary cyclic $(7, 4, 3)$ Hamming code generated by $g(x) = x^3 + x + 1$. This code is the dual of the $(7, 3, 4)$ code of the previous Exercise and is threshold decodable in two-steps. We describe a two-step decoding algorithm for this code, with the help of the code $\mathbf{H}$ matrix.

Solution: The corresponding parity-check matrix $\mathbf{H}$ is given by

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

Exercise

Using the $\mathbf{H}$ matrix we obtain the following parity sums.

$$\begin{aligned} A_1 &= s_2 &&= e_6 + e_5 + e_4 + e_2 \\ A_2 &= s_2 + s_1 &&= e_6 + e_3 + e_2 + e_1 \\ A_3 &= s_0 &&= e_6 + e_5 + e_3 + e_0. \end{aligned}$$

Notice that A_1 and A_2 are orthogonal on the sum $e_6 + e_2$, and that A_1 and A_3 are orthogonal on the sum $e_6 + e_5$.

Exercise

Use the sums A_1 and A_2 to estimate the sum $e_6 + e_2$, and use the sums A_1 and A_3 to estimate $e_6 + e_5$. This concludes the first decoding step.

Use the two estimated sums, denoted by A_4 and A_5, respectively, i.e.,

$$\begin{aligned} A_4 &= e_6 + e_2 \\ A_5 &= e_6 + e_5, \end{aligned}$$

which are orthogonal on e_6, constitute the second decoding step and complete the orthogonalization process for this code.

Algebraic decoding

In general, for a given (n, k, d) code the decoding process has always a complexity higher than the corresponding encoding process.

From a practical point of view the best code is chosen subject to a specified budget.

This financial constraint can force the choice of suboptimum codes, however having a decoder which is amenable to practical implementation.

Algebraic decoding

Fortunately, for some classes of algebraic codes decoding algorithms were developed which are computationally efficient.

The problem of decoding algebraic codes consists in solving a system of nonlinear equations, whose direct solution in general is not obvious.

Berlekamp-Massey time domain decoding

An important algebraic decoding algorithm for BCH codes was published by Berlekamp in 1968.

Analyzing Berlekamp's algorithm, Massey (1969) showed that it provided a general solution to the problem of synthesizing the shortest linear feedback shift-register capable of generating a prescribed finite sequence of digits.

Since then this algorithm is known as the Berlekamp-Massey (BM) algorithm.

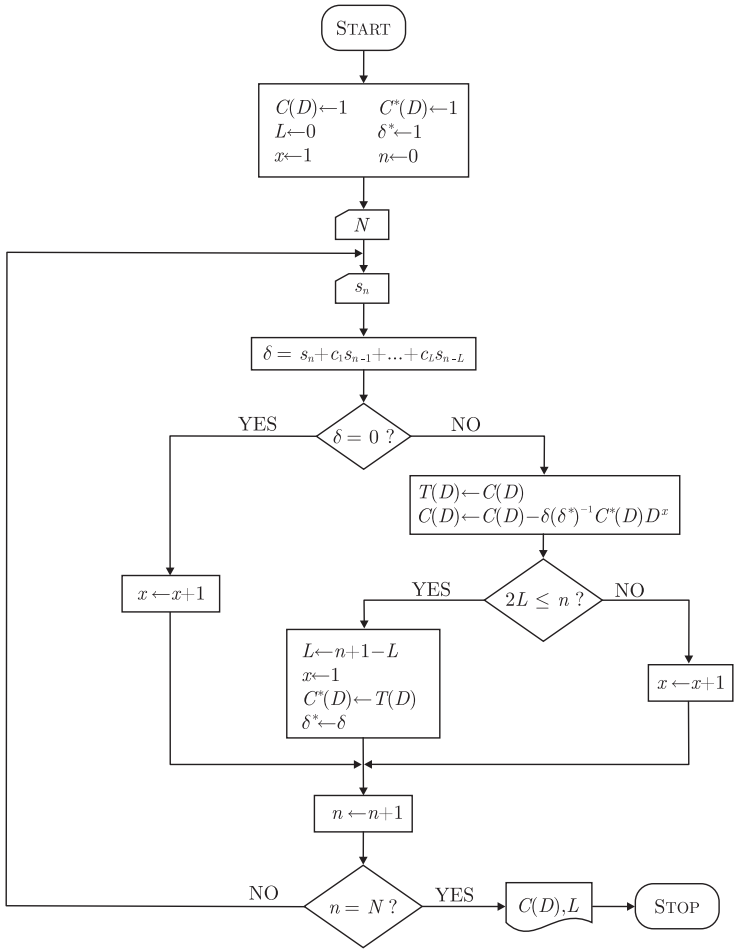
Berlekamp-Massey time domain decoding

The BM algorithm is widely applicable for decoding algebraic codes, including RS codes and BCH codes.

For binary BCH codes, there is no need to calculate error magnitude values, since in $\mathrm{GF}(2)$ it is sufficient to determine the positions of the errors to perform error correction.

For non-binary BCH codes, including RS codes, both error location and error magnitudes have to be determined to perform error correction.

Flow chart of the Berlekamp-Massey algorithm



Let (n, k, d) denote an algebraic code (BCH or RS) with generator polynomial $g(x)$ having coefficients in some finite field $\mathrm{GF}(q)$ and having roots $\alpha, \alpha^2, \ldots, \alpha^{2t}$.

Let $c(x) = \sum_{i=0}^{n-1} c_i x^i$ denote a codeword polynomial, let $e(x) = \sum_{i=0}^{n-1} e_i x^i$ denote the error polynomial with coefficients in $\mathrm{GF}(q)$ and let $r(x) = c(x) + e(x)$, with addition over $\mathrm{GF}(q)$, denote the received n-tuple in polynomial form.

The approach used for algebraic decoding contains the following steps.

1) Compute the first $2t$ coefficients $s_0, s_1, \ldots, s_{2t-1}$ of the syndrome polynomial $s(x) = s_0 + s_1 x + \cdots + s_{2t-1} x^{2t-1} + \cdots$, where $s_0 = r(\alpha), s_1 = r(\alpha^2), \ldots, s_{2t-1} = r(\alpha^{2t})$.
2) Use the sequence $s_0, s_1, \ldots, s_{2t-1}$ as input to the BM algorithm and compute the error-locator polynomial $\sigma(x)$, of degree τ, $\tau \leq t$, where

$$\sigma(x) = 1 + \sigma_1 x + \sigma_2 x^2 + \cdots + \sigma_\tau x^\tau.$$

3) Find the roots of $\sigma(x)$, denoted by $\beta_1, \beta_2, \ldots, \beta_\tau$, whose multiplicative inverses give the error locations.
4) Compute the error magnitudes in case of non-binary codes.

Berlekamp (1968) introduced a procedure for computing error magnitudes for non-binary cyclic codes and defined the polynomial

$$\begin{aligned} Z(x) &= 1 + (s_0 + \sigma_1)x + (s_1 + \sigma_1 s_0 + \sigma_2)x^2 + \cdots \\ &\quad \cdots + (s_{\tau-1} + \sigma_1 s_{\tau-2} + \sigma_2 s_{\tau-3} + \cdots + \sigma_\tau)x^\tau. \end{aligned} \tag{4}$$

Error magnitudes at positions $\beta_i^{-1},\ 1 \leq i \leq \tau$, are calculated as

$$e_i = \frac{Z(\beta_i)}{\prod_{j=1, j\neq i}^{\tau}(1 - \beta_j^{-1}\beta_i)}. \tag{5}$$

In the block diagram of the BM algorithm $C(D)$ plays the role of $\sigma(x)$.

By applying the BM algorithm to a received word an estimate of the error pattern is obtained, which has minimum Hamming weight and satisfies all the syndrome equations.

Whether or not such an estimate will be the true error pattern that occurred on the channel will depend on its Hamming weight τ in comparison to t.

Decoding succeeds whenever the condition $\tau \leq t$ is satisfied.

The Euclidean algorithm

The Euclidean algorithm is a technique which allows the calculation of the greatest common divisor of two integers, or of two polynomials.

Our main interest is to solve the key equation for decoding cyclic codes, and not to calculate the greatest common divisor.

The Euclidean algorithm

Beginning with two polynomials, $a(z)$ and $b(z)$, the Euclidean algorithm employs the following relationship:

$$f_i(z)a(z) + g_i(z)b(z) = r_i(z),$$

where, for any one of the polynomials $f_i(z)$, $g_i(z)$ or $r_i(z)$ replacing $h_i(z)$, the following recurrence relation is applied:

$$h_i(z) = h_{i-2}(z) - q_i(z)h_{i-1}(z),$$

subject to the following initial conditions.

The Euclidean algorithm

$$\begin{aligned} f_{-1}(z) &= g_0(z) &&= 1 \\ f_0(z) &= g_{-1}(z) &&= 0 \\ r_{-1}(z) &= a(z) \\ r_0(z) &= b(z). \end{aligned}$$

The Euclidean algorithm

The polynomial $q_i(z)$ is given by the integer part with non-negative exponents of the quotient

$$r_{i-2}(z)/r_{i-1}(z).$$

In order to solve the key equation we consider

$$a(z) = z^{2t} \text{ and } b(z) = S(z),$$

and apply the Euclidean algorithm, stopping when the degree of $r_i(z)$ is less than t. We then take

$$L(z) = g_i(z).$$

Exercise

BM decoding of a BCH code

Consider the $(15, 7, 5)$ binary BCH code, with $g(x) = m_1(x)m_3(x)$ $= (x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1)$ as generator polynomial and decode the received polynomial $r(x) = x + x^9$ with the BM algorithm.

Solution: The syndrome coefficients for the received polynomial $r(x) = x + x^9$ are calculated using a table of elements of $\mathrm{GF}(16)$ as

$$\begin{aligned} s_0 &= r(\alpha) &&= \alpha + \alpha^9 &&= \alpha^3 \\ s_1 &= r(\alpha^2) &&= \alpha^2 + \alpha^{18} &&= \alpha^6 \\ s_2 &= r(\alpha^3) &&= \alpha^3 + \alpha^{27} &&= \alpha^{10} \\ s_3 &= r(\alpha^4) &&= \alpha^4 + \alpha^{36} &&= \alpha^{12}. \end{aligned}$$

Exercise

Evolution of the BM algorithm for the input sequence $(s_0, s_1, s_2, s_3) = (\alpha^3, \alpha^6, \alpha^{10}, \alpha^{12})$.

n	s_n	δ	$T(D)$	$C(D) = \sigma(X)$	L	$C^*(D)$	δ^*	x
0	α^3	α^3	–	1	0	1	1	1
1	α^6	0	1	$1 + \alpha^3 D$	1	1	α^3	1
2	α^{10}	α^{13}	1	$1 + \alpha^3 D$	1	1	α^3	2
3	α^{12}	0	$1 + \alpha^3 D$	$1 + \alpha^3 D + \alpha^{10} D^2$	2	$1 + \alpha^3 D$	α^{13}	1

Exercise

The previous table is used to compute the error-locator polynomial with the BM algorithm for the input sequence $s_0, s_1, s_2, s_3 = \alpha^3, \alpha^6, \alpha^{10}, \alpha^{12}$.

$C(D) = 1 + \alpha^3 D + \alpha^{10} D^2$ is the error-locator polynomial, or $\sigma(x) = 1 + \alpha^3 x + \alpha^{10} x^2$, the roots of which are found by an exhaustive search to be $\beta_1 = \alpha^6$ and $\beta_2 = \alpha^{14}$.

Exercise

The error positions are the exponents of α in the representation of β_1^{-1} and β_2^{-1}, *i.e.*, $\beta_1^{-1} = \alpha^{-6} = \alpha^9$ and $\beta_2^{-1} = \alpha^{-14} = \alpha$.

Two errors are thus located, at positions x and x^9, respectively.

After the operation of error correction is completed, the decoded word is the all-zero codeword, *i.e.*, $c(x) = 0$.

Exercise

BM decoding of an RS code

Consider the $(7, 5, 3)$ RS code over $\mathrm{GF}(8)$, with generator polynomial $g(x) = (x - \alpha)(x - \alpha^2)$ and decode the received polynomial $r(x) = \alpha^2 x^3$ using the BM algorithm.

Solution: The syndrome coefficients for the received polynomial $r(x) = \alpha^2 x^3$ are calculated using a table of elements of $\mathrm{GF}(8)$ as

$$\begin{aligned} s_0 &= r(\alpha) &&= \alpha^5 \\ s_1 &= r(\alpha^2) &&= \alpha^8 = \alpha. \end{aligned}$$

Exercise

Evolution of the BM algorithm for the input sequence $(s_0, s_1) = (\alpha^5, \alpha)$.

n	s_n	δ	$T(D)$	$C(D) = \sigma(X)$	L	$C^*(D)$	δ^*	x
0	α^5	α^5	$-$	1	0	1	1	1
1	α	1	1	$1 + \alpha^5 D$	1	1	α^5	1
			$1 + \alpha^5 D$	$1 + \alpha^3 D$	1	$1 + \alpha^5 D$	1	1

This table is used to compute the error-location polynomial with the BM algorithm for the input sequence $s_0, s_1 = \alpha^5, \alpha$.

Exercise

The polynomial $C(D) = 1 + \alpha^3 D$ is the error-locator polynomial, and since $C(D) = \sigma(x)$ we write

$$\sigma(x) = 1 + \alpha^3 x,$$

whose root is found by an exhaustive search to be $\beta = \alpha^4$.

The error position is the exponent of α in the representation of β^{-1}, *i.e.*, $\beta^{-1} = \alpha^{-4} = \alpha^3$. Therefore, an error occurring at x^3 has been found, i.e., $e(x) = e_3 x^3$.

Exercise

In case of just a single error occurring, the error magnitude is easily found as follows.

$$s_0 = e(\alpha) = \alpha^5 = e_3\alpha^3,$$

and thus $e_3 = \alpha^2$.

Therefore, $e(x) = \alpha^2 x^3$ and $c(x) = r(x) - e(x) = 0$, i.e., the decoded polynomial is the all-zero polynomial.

It is clear that to correct single errors it is not necessary to resort to (4) and (5).

Exercise

BM decoding of an RS code

Consider the $(15, 11, 5)$ RS code over $\mathrm{GF}(16)$, with generator polynomial $g(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4)$ and decode the received polynomial $r(x) = \alpha^2 x + \alpha^3 x^9$ with the BM algorithm.

Solution: The syndrome coefficients for the received polynomial $r(x) = \alpha^2 x + \alpha^3 x^9$ are calculated using a table of elements of $\mathrm{GF}(16)$. as

$$\begin{aligned} s_0 &= r(\alpha) &= \alpha^3 + \alpha^{12} &= \alpha^{10} \\ s_1 &= r(\alpha^2) &= \alpha^4 + \alpha^6 &= \alpha^{12} \\ s_2 &= r(\alpha^3) &= \alpha^5 + 1 &= \alpha^{10} \\ s_3 &= r(\alpha^4) &= \alpha^6 + \alpha^9 &= \alpha^5. \end{aligned}$$

Exercise

Evolution of the BM algorithm for the input sequence $(s_0, s_1, s_2, s_3) = (\alpha^{10}, \alpha^{12}, \alpha^{10}, \alpha^5)$, where $a(x) = 1 + \alpha^2 D + \alpha D^2$.

n	s_n	δ	$T(D)$	$C(D)$	L	$C^*(D)$	δ^*	x
0	α^{10}	α^{10}	–	1	0	1	1	1
1	α^{12}	α^{14}	1	$1 + \alpha^{10} D$	1	1	α^{10}	1
2	α^{10}	α^{11}	$1 + \alpha^{10} D$	$1 + \alpha^2 D$	1	1	α^{10}	2
3	α^5	α^2	$1 + \alpha^2 D$	$1 + \alpha^2 D + \alpha D^2$	2	$1 + \alpha^2 D$	α^{11}	1
			$a(x)$	$1 + \alpha^3 D + \alpha^{10} D^2$	2	$a(x)$	α^{11}	2

This table is used to compute the error-locator polynomial with the BM algorithm for the input sequence $s_0, s_1, s_2, s_3 = \alpha^{10}, \alpha^{12}, \alpha^{10}, \alpha^5$.

Exercise

The polynomial $C(D) = 1 + \alpha^3 D + \alpha^{10} D^2$ is the error-locator polynomial, and since $C(D) = \sigma(x)$ we write

$$\sigma(x) = 1 + \alpha^3 x + \alpha^{10} x^2 = (1 + \alpha x)(1 + \alpha^9 x),$$

the roots of which are found by an exhaustive search to be $\beta_1 = \alpha^{14}$ and $\beta_2 = \alpha^6$.

The error positions are the exponents of α in the representation of β_1^{-1} and β_2^{-1}, *i.e.*, $\beta_1^{-1} = \alpha^{-14} = \alpha$ and $\beta_2^{-1} = \alpha^{-6} = \alpha^9$.

Exercise

Two errors are thus located, at positions x and x^9, respectively. The error magnitudes are found with the help of (4) and (5) as follows.

$$\begin{aligned} Z(x) &= 1 + (s_0 + \sigma_1)x + (s_1 + \sigma_1 s_0 + \sigma_2)x^2 \\ &= 1 + (\alpha^{10} + \alpha^3)x + (\alpha^{12} + \alpha^3\alpha^{10} + \alpha^{10})x^2 \\ &= 1 + \alpha^{12}x + \alpha^8 x^2. \end{aligned}$$

Exercise

Then

$$
\begin{aligned}
e_1 &= \frac{Z(\beta_1)}{(1+\beta_2^{-1}\beta_1)} = \frac{1+\alpha^{12}\alpha^{14}+\alpha^8\alpha^{13}}{(1+\alpha^9\alpha^{14})} = \frac{\alpha^4}{\alpha^2} = \alpha^2 \\
e_2 &= \frac{Z(\beta_2)}{(1+\beta_1^{-1}\beta_2)} = \frac{1+\alpha^{12}\alpha^6+\alpha^8\alpha^{12}}{(1+\alpha\alpha^6)} = \frac{\alpha^{12}}{\alpha^9} = \alpha^3.
\end{aligned}
$$

The error polynomial is thus $\alpha^2 x + \alpha^3 x^9$. After the operation of error correction is completed, the decoded polynomial is the all-zero polynomial, *i.e.*, $c(x) = 0$.

Finite Field Fourier Transform

Definition

Let $\mathbf{v} = (v_0, v_1, \ldots, v_i, \ldots, v_{n-1})$ be an n-tuple with coefficients in $\mathrm{GF}(q)$, where n divides $q^m - 1$ for some positive integer m, and let α be an element of multiplicative order n in $\mathrm{GF}(q^m)$. The n-tuple $\mathbf{V} = (V_0, V_1, \ldots, V_j, \ldots, V_{n-1})$ defined over $\mathrm{GF}(q^m)$, whose components are given by:

$$V_j = \sum_{i=0}^{n-1} \alpha^{ij} v_i, \quad j = 0, 1, \ldots, n-1,$$

is called the finite field Fourier transform of $\mathbf{v}$.

Finite Field Fourier Transform

Vectors $\mathbf{v}$ and $\mathbf{V}$ constitute a Fourier transform pair, usually represented as:

$$\mathbf{v} \Longleftrightarrow \mathbf{V}$$

or

$$\{v_i\} \longleftrightarrow \{V_j\}.$$

Indices i and j are referred to as time and frequency, respectively. Some important properties of the FFFT which have great utility in algebraic coding are presented next.

Finite Field Fourier Transform

PROPERTY 1 - Over $\mathrm{GF}(q)$, a finite field of characteristic p, the components $v_i,\ 0 \leq i \leq n-1$, of a vector $\mathbf{v}$ are related to the components $V_j,\ 0 \leq j \leq n-1$, of its finite field Fourier transform $\mathbf{V}$ through the expressions:

$$V_j = \sum_{i=0}^{n-1} \alpha^{ij}\, v_i$$

$$v_i = \frac{1}{(n \mod p)} \sum_{j=0}^{n-1} \alpha^{-ij}\, V_j .$$

Finite Field Fourier Transform

PROPERTY 2 - Given a Fourier transform pair $\{v_i\} \leftrightarrow \{V_j\}$ and constants c, i_0 and k, then:

$$
\begin{array}{lllll}
a) & c\{v_i\} & \longleftrightarrow & c\{V_j\} & \text{(Linearity)} \\
b) & \{v_{i-i_0}\} & \longleftrightarrow & \{V_j \alpha^{ji_0}\} & \text{(Time shift)} \\
c) & \{v_{ki}\} & \longleftrightarrow & \{V_{j/k}\} & \text{(Scaling)}(k,n) = 1.
\end{array}
$$

Finite Field Fourier Transform

PROPERTY 3 - Given two Fourier transform pairs

$$\{f_i\} \longleftrightarrow \{F_j\}$$

and

$$\{g_i\} \longleftrightarrow \{G_j\},$$

then:

$$\{f_i\, g_i\} \longleftrightarrow \{(1/n) F_k\, G_{j-k}\} \text{ (Frequency domain convolution).}$$

Finite Field Fourier Transform

Vectors $\mathbf{v}$ and $\mathbf{V}$ are also usually represented by polynomials as follows.

$$\begin{aligned} v(x) &= v_{n-1}x^{n-1} + v_{n-2}x^{n-2} + \cdots + v_1 x + v_0 \\ V(z) &= V_{n-1}z^{n-1} + V_{n-2}z^{n-2} + \cdots + V_1 z + V_0. \end{aligned}$$

It follows from this polynomial representation of $\mathbf{v}$ and $\mathbf{V}$ that

$$\begin{aligned} V_j &= \sum_{i=0}^{n-1} \alpha^{ij} v_i = v(\alpha^j) \\ v_i &= (1/n) \sum_{j=0}^{n-1} \alpha^{-ij} V_j = (1/n) V(\alpha^{-i}). \end{aligned}$$

Finite Field Fourier Transform

The roots of a polynomial in one domain and the components of the corresponding FFFT pair in the other domain are related as follows.

$v(x)$ has a root α^j, i.e., $v(\alpha^j) = 0$, if and only if $V_j = 0$. Conversely, $V(z)$ has a root α^{-i}, i.e., $V(\alpha^{-i}) = 0$, if and only if $v_i = 0$.

Euclidean frequency domain decoding

The decoding procedure known as *algebraic decoding in the frequency domain* is applicable to any cyclic code, being however more efficient for BCH codes.

The received vector $\mathbf{r}$ is assumed to result from the sum $\mathbf{v} + \mathbf{e}$ over $\mathrm{GF}(q)$ of a codeword $\mathbf{v}$ and an error vector $\mathbf{e}$, i.e., $\mathbf{r} = \mathbf{v} + \mathbf{e}$.

The steps required for using this decoding procedure are described next.

Euclidean frequency domain decoding

- **Fourier transform of the received vector**

  The Fourier transform $\mathbf{R}$ of the received vector $\mathbf{r}$ is computed. Due to linearity $\mathbf{R}$ can be expressed as a function of the corresponding Fourier transforms of $\mathbf{v}$ and $\mathbf{e}$, i.e., as $\mathbf{R} = \mathbf{V} + \mathbf{E}$.

- **Syndrome calculation**

$$S(z) = \sum_{j=0}^{2t-1} S_j z^j,$$

$$S_j = R_{j+h_0} = \sum_{i=0}^{n-1} r_i \alpha^{i(j+h_0)}, j = 0, 1, \ldots, 2t-1.$$

Euclidean frequency domain decoding

- **Error locator polynomial calculation**

  This step is also known as the *solution of the key equation*. The error locator polynomial, $L(z)$, has the form:

$$L(z) = \sum_{k=0}^{\nu-1} L_z z^k = \prod_{k=1}^{\nu} (1 - z\alpha^{i_k}),$$

  where $\nu \leq t$ and $i_1, i_2, \ldots, i_\nu$, correspond to the error locations.

The Euclidean algorithm is applied with $a(z) = z^{2t}$ and $b(z) = S(z)$, stopping when the degree of $r_i(z)$ becomes less than t. We take $L(z) = g_i(z)$.

Euclidean frequency domain decoding

- **Error vector Fourier transform calculation**

  The error vector $\mathbf{e}$ has its Fourier transform denoted by $\mathbf{E}$, which has the following polynomial representation

$$E(z) = \sum_{j=0}^{n-1} E_j z^j .$$

  Since the code is assumed to have $2t$ consecutive roots, it follows that the coefficients $E_j, j = h_0, h_0 + 1, \ldots, h_0 + 2t - 1$, are known.

Euclidean frequency domain decoding

The remaining unknown coefficients are calculated by means of the following recursive formula

$$\sum_{k=0}^{n-1} L_k E_{j-k} = 0,$$

which can be slightly simplified to

$$E_j = \sum_{k=0}^{\nu-1} L_k E_{j-k},$$

because $L_0 = 1$ and $L(z)$ has degree $\nu - 1 \leq t - 1$.

Euclidean frequency domain decoding

- **Error correction**

  The error vector is obtained as the inverse Fourier transform of vector $\mathbf{E}$, obtained in the previous step.

- Finally, the estimated transmitted codeword is given by

$$\mathbf{v} = \mathbf{r} - \mathbf{e}.$$

Exercise

Frequency domain decoding

Consider the binary $(7, 4, 3)$ BCH code having α, α^2 and α^4, as roots of the generator polynomial, where α denotes a primitive element of $\mathrm{GF}(2^3)$. Decode the received word $r(x) = x^6 + x^3$ using the Euclidean frequency domain decoder.

Solution: The received word $r(x)$ in vector form is expressed as

$$\mathbf{r} = (1, 0, 0, 1, 0, 0, 0).$$

Exercise

The Fourier transform of $r(x)$

From $r(x)$ we compute R and obtain

$$R = (\alpha^2, \alpha^4, \alpha^2, \alpha, \alpha, \alpha^4, 0),$$

where

$$R_j = \sum_{i=0}^{6} r_i \alpha^{ij} = \alpha^{6j} + \alpha^{3j}.$$

Syndrome calculation

For the syndrome we obtain $S_j = R_{j+1}, j = 0, 1$, which gives:

$$S_0 = R_1 = \alpha^4, S_1 = R_2 = \alpha,$$

and therefore, $S(z) = \alpha z + \alpha^4$.

Exercise

Error locator calculation

We find L(z) by means of the Euclidean algorithm.

i	$g_i(z)$	$r_i(z)$	$q_i(z)$
-1	0	z^2	$-$
0	1	$\alpha z + \alpha^4$	$-$
1	$\alpha^6 z + \alpha^2$	α^6	$\alpha^6 z + \alpha^2$

therefore, $L(z) = \alpha^6 z + \alpha^2$.

Error vector Fourier transform calculation

$$\sum_{k=0}^{\nu-1} L_k E_{j-k} = 0$$

$$L_0 E_j + L_1 E_{j-1} = 0.$$

Exercise

We know that $E_j = S_{j-1},\ j = 1, 2$. It thus follows that $E_1 = \alpha^4$ and $E_2 = \alpha$. From $L(z) = \alpha^6 z + \alpha^2$ we extract $L_0 = \alpha^2$ and $L_1 = \alpha^6$, which are then applied to the recursion $L_0 E_j + L_1 E_{j-1} = 0$ to produce

$$E_j = \alpha^4 E_{j-1}.$$

We use the recursion $E_j = \alpha^4 E_{j-1}$ to compute the remaining unknown values of $\mathbf{E}$ and obtain:

$$\mathbf{E} = (\alpha^3, \alpha^6, \alpha^2, \alpha^5, \alpha, \alpha^4, 1).$$

Exercise

Error correction

The inverse Fourier transform of $\mathbf{E}$ is the following:

$$\mathbf{e} = (0, 0, 1, 0, 0, 0, 0).$$

Therefore, we obtain $e(x) = x^4$, and the corrected word is given by:

$$v(x) = r(x) - e(x), \text{ i.e., } \quad v(x) = x^6 + x^4 + x^3.$$

Soft-decision Decoding

Probabilistic decoding

Probabilistic decoding algorithms are used in soft-decision decoding.

A soft-decision decoder employs received symbol reliability information supplied by the demodulator when deciding which codeword was transmitted.

Typically soft-decision decoding leads to a gain of at least 2.0 dB with respect to hard-decision decoding.

Decoding LDPC Codes

Efficient decoding of LDPC codes relies on the sum-product algorithm (SPA), which is a symbol-by-symbol soft-in soft-out iterative decoding algorithm.

The received symbols are fed to the decoder which employs iterations based on the code parity-check matrix $\mathbf{H}$ to improve the reliability of the decoded symbols.

Decoding LDPC Codes

After each iteration the available symbol reliability values are used to make hard decisions and to output a decoded binary n-tuple $\mathbf{z}$.

If $\mathbf{zH}^T = \mathbf{0}$ then $\mathbf{z}$ is a codeword and decoding stops.

If $\mathbf{zH}^T \neq \mathbf{0}$ then $\mathbf{z}$ is not a codeword and another iteration is performed by the decoder.

Decoding LDPC Codes

A stop condition is defined by specifying a maximum number of decoder iterations.

A decoding failure occurs if the maximum number of iterations is reached and the decoder does not find a codeword.

Decoding LDPC Codes

Let $\mathbf{v} = (v_1, v_2, \ldots, v_i, \ldots, v_n)$ denote a codeword and let $\mathbf{y}$ denote a received n-tuple with real valued coordinates.

The SPA is implemented by computing the marginal probabilities $P(v_i|\mathbf{y})$, for $1 \leq i \leq n$.

This SHORT COURSE is now concluded.

THANK YOU!