CODING THEORY
a first course

Henk C.A. van Tilborg

Contents

Preface

As the title of this book already suggests, this manuscript is intended to be a textbook suitable for a first course in coding theory. It is based on a course that is taught for several years at the Eindhoven University of Technology. The students that follow this course are mostly in the third or fourth year of their undergraduate program. Typically, half of them are computer science students, a third of them study mathematics and the remainder are students in electrical engineering or information technology.

All these students are familiar with linear algebra and have at least a rudimentary knowledge of probability theory. More importantly, it is assumed here that they are familiar with the theory of finite fields and with elementary number theory. Clearly the latter is not the case at many universities. It is for this reason that a large appendix has been added (Appendix A), containing all the necessary prerequisites with regard to finite field theory and elementary number theory.

All chapters contain exercises that we urge the students to solve. Working at these problems seems to be the only way to master this field. As a service to the student that is stranded with a problem or to give a student a chance to look at a (possibly different) solution, all problems are completely worked out in Appendix C.

The main part of this manuscript was written at the University of Pretoria in the summer of 1991. I gladly acknowledge the hospitality that Gideon Kühn and Walter Penzhorn extended to me during my stay there.

I would like to express my gratitude to Patrick Bours, Martin van Dijk, Tor Helleseth, Christoph Kirfel, Dirk Kleima, Jack van Lint, Paul van der Moolen, Karel Post, Hans Sterk, René Struik, Hans van Tilburg, Rob Versseput, Evert van de Vrie, Øyvind Ytrehus and many of the students for all the corrections and suggestions that they have given me.

A special word of thanks I owe to Iwan Duursma who made it possible to include at a very late stage a section on algebraic-geometry codes (he presented me a first draft for Section 4.6). Finally, I am indebted to Anneliese Vermeulen-Adolfs for her instrumental help in making this manuscript publisher-ready.

Eindhoven   Henk van Tilborg
the Netherlands
April 3, 1993

Chapter 1

A communication system

1.1   Introduction

Communicating information from one person to another is of course an activity that is as old as mankind. The (mathematical) theory of the underlying principles is not so old. It started in 1948, when C.E. Shannon gave a formal description of a communication system and, at the same time, also introduced a beautiful theory about the concept of information, including a good measure for the amount of information in a message.

In the context of this book, there will always be two parties involved in the transmission of information. The *sender* of the message(s) (also called the *source*) and the *receiver* (also called the *sink*). In some applications the sender will write information on a medium (like a floppy disc) and the receiver will read it out later. In other applications, the sender will actually transmit the information (for instance by satellite or telephone line) to the receiver. Either way, the receiver will not always receive the same information as was sent originally, simply because the medium is not always perfect. This medium will be discussed in the next section. In Section 1.3 we will discuss Shannon's answer to the problem of transmission errors.

1.2   The channel

The medium over which information is sent, together with its characteristics, is called the *channel*. These characteristics consist of an input alphabet X, an output alphabet Y, and a transition probability function P.

Unless explicitly stated otherwise, it will always be assumed that successive transmissions have the same transition probability function and are independent of each other. In particular, the channel is assumed to be memoryless.
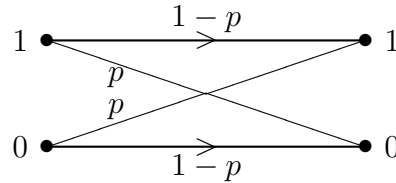
$$1-p$$

$$1 \qquad\qquad 1$$

$$p$$

$$p$$

$$0 \qquad\qquad 0$$

$$1-p$$

Figure 1.1: The binary symmetric channel

Definition 1.2.1 *A channel* $(X, Y; P)$ *consists of an alphabet* X *of input symbols, an alphabet* Y *of output symbols, and for each* x *in* X *and* y *in* Y *the conditional probability* $p(y|x)$ *that symbol* y *is received, when symbol* x *was transmitted (this probability is independent of previous and later transmissions).*

Of course $\sum_{y \in Y} p(y|x) = 1$ for all x in X.

The channel that we shall be studying the most will be the *Binary Symmetric Channel*, shortened to *BSC*. It is depicted in Figure 1.1.

Definition 1.2.2 *The Binary Symmetric Channel is the channel* $(X, Y; P)$ *with both* X *and* Y *equal to* $\{0, 1\}$ *and* P *given by* $p(1|0) = p(0|1) = p$ *and* $p(0|0) = p(1|1) = 1 - p$ *for some* $0 \leq p \leq 1$.

So, with probability $1 - p$ a transmitted symbol will be received correctly and with probability p it will be received incorrectly. In the latter case, one says that an *error* has occurred.

Of course, if $p > \frac{1}{2}$ the receiver gets a more reliable channel by inverting the received symbols. For this reason we shall always assume that $0 \leq p \leq \frac{1}{2}$.

The BSC gives a fair description of the channel in many applications. A straightforward generalization is the *q-ary symmetric channel*. It is defined by $X = Y$, both of cardinality q, and the transition probabilities $p(y|x) = 1 - p$, if $x = y$, and $p(y|x) = p/(q-1)$, if $x \neq y$.

Another type of channel is the *Gaussian channel*, given by $X = \{-1, 1\}$, $Y = \mathbb{R}$ and the probability density function $p(y|x)$ which is the Gaussian distribution with x as mean and with some given variance, depending on the reliability of the channel, i.e.

$$p(y|x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(y-x)^2/2\sigma^2}.$$

If the actual channel is Gaussian, one can reduce it to the BSC, by replacing every $y \geq 0$ by 1 and every $y < 0$ by 0, and by writing 0 instead of the the input symbol -1. By doing this, one loses information about the reliability of a received symbol. Indeed, the received symbol $y = 1.2$ is much more likely to come from the transmitted symbol 1, than the received symbol $y = 0.01$. By reducing the Gaussian channel to the BSC, one throws away information about the transmitted symbol.

In some applications the channel is inherently of the BSC-type and not a reduction of the Gaussian channel.

Similarly, by dividing $\mathbb{R}$ into three appropriate regions, $(-\infty, -a]$, $(-a, a)$ and $[a, \infty)$, $a \geq 0$, one obtains the following channel.

Definition 1.2.3 *The* *binary* *symmetric* error and erasure channel *is a channel* $(X, Y; P)$ *with* $X = \{0, 1\}$, $Y = \{0, *, 1\}$, *and* P *given by* $p(0|0) = p(1|1) = 1 - p' - p''$, $p(1|0) = p(0|1) = p'$ *and* $p(*|0) = p(*|1) = p''$ *for some non-negative* p' *and* p'' *with* $0 \leq p' + p'' \leq 1$.

The $*$ symbol means that there is too much ambiguity about the transmitted symbol. One speaks of an *erasure*.

In the above channels, the errors and erasures in transmitted symbols occur independently of each other. There are however applications where this is not the case. If the errors tend to occur in clusters we speak of a *burst-channel*. Such a cluster of errors is called a *burst*. A more formal definition will be given in Chapter 5. Even a mixture of the above error types can occur: random errors and bursts.

Unless explicitly stated otherwise, we will always assume the channel to be the BSC.

1.3   Shannon theory and codes

If one wants to transmit a 1 over the BSC that has error probability p, one can increase the reliability of the transmission by repeating the transmission a few times, say altogether five times. Similarly one can send five 0's if a single 0 needs to be transmitted. The receiver can use a simple majority vote on the received sequence y_1, y_2, y_3, y_4, y_5 to decide what the most likely transmitted symbol is. For instance, if 1,1,0,0,1 is received, the most likely transmitted sequence is of course 1,1,1,1,1.

With this system, it is still possible that the receiver makes an error, namely if three or more errors have occurred in a transmitted 5-tuple. If at most two errors occurred during the transmission, the receiver will make the correct estimate of the transmitted information. The probability of correct transmission of the information is given by the probability that no error occurred, plus the probability that exactly one of the five coordinates is in error, plus the probability that exactly two of the five coordinates are in error, so it is given by

$$(1 - p)^5 + \binom{5}{1} p^1 (1 - p)^4 + \binom{5}{2} p^2 (1 - p)^3.$$

For $p = 0.01$, this probablity is 0.999986 as opposed to 0.99 when only one symbol was transmitted. Of course, the price that has been paid for this dramatic increase in reliability is the transmission of five bits instead of just one!
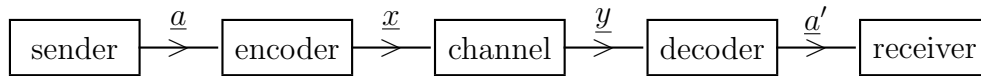
Figure 1.2: A communication system

Transmitting more symbols than is strictly necessary to convey the message is called adding *redundancy* to the message. Regular languages know the same phenomenon. The fact that one immediately sees the obvious misprints in a word, like 'lacomotivf', means that the word contains more letters than are strictly necessary to convey the message. The redundant letters enable us to correct the misspelling of the received word.

Definition 1.3.1 *An* encoder *is a mapping (or algorithm) that transforms each sequence of symbols from the message alphabet A to a sequence of symbols from the input alphabet X of the channel, in such a way that redundancy is added (to protect it better against channel errors).*

A complete decoder *is an algorithm that transforms the received sequence of symbols of the output alphabet Y of the channel into a message stream over A. If a decoder sometimes fails to do this (because it is not able to do so or to avoid messages that are too unreliable) one speaks of an* incomplete *decoder.*

The channel, encoder, and decoder, together with the sender and receiver, form a so-called *communication system*. See Figure 1.2, where $\underline{a}$ denotes the message stream, $\underline{x}$ a sequence of letters in X, $\underline{y}$ a sequence of letters in Y and $\underline{a}'$ a message stream.

A decoder that always finds the most likely (in terms of the channel probabilities) transmitted message stream $\underline{a}'$, given the received sequence $\underline{y}$, is called a *maximum likelihood* decoder. The decoder that was described above for the encoder that repeated each information bit five times is an example of a maximum likelihood decoder.

In Section 1.2 we have seen how the reduction of the Gaussian channel to the BSC throws away (valuable) information. A decoder that uses the output of this (reduced) BSC as its input is called a *hard decision* decoding algorithm, while it is called a *soft decision* decoding algorithm otherwise.

A final distinction is the one between encoders (and decoders) with and without memory.

Definition 1.3.2 *If the encoder maps k-tuples of symbols from the message alphabet A in a one-to-one way to n-tuples of symbols from the input alphabet X of the channel (independent of the other input k-tuples), the resulting set of $|A|^k$ output n-tuples is called a* block code. *For the elements of a block code one uses the name* codeword.

Definition 1.3.3 *If the encoder maps k-tuples of symbols from the message alphabet A to n-tuples of symbols from the input alphabet X of the channel in a way that also depends on the last m input k-tuples, where m is some fixed parameter, the resulting*

sequence of output n-tuples is called a convolutional code. *These output sequences are also named* codewords.

For a long stream of message symbols, one of course has to break it up into segments of length k each before they can be handled by the encoder.

Convolutional codes will be discussed in Chapter 6. Block codes form the main topic of this book. They will be extensively discussed in Chapters 2–5. Note that the code discussed before, in which each message symbol is repeated four times, is an example of a block code with $k = 1$ and $n = 5$.

Let M denote the size of A and let q denote the size of X. If all k-tuples of M-ary symbols are equally likely, one needs $\lceil k \log_2 M \rceil$ bits to denote one of them. This may not be so obvious in general, but when M is some power of 2, this is a straightforward observation. Similarly for each of q^n equally likely n-tuples of q-ary symbols one needs $n \log_2 q$ bits. Let the *information rate R* denote the amount of information that an input symbol of the channel contains. It follows that

$$R = \frac{k \log_2 M}{n \log_2 q}. \tag{1.1}$$

If $q = 2$, this reduces to $R = \frac{k \log_2 M}{n}$.

If $q = M$, equation (1.1) simplifies to $R = k/n$. The interpretation of the information rate in this case corresponds with the intuitive interpretation: if k information symbols are mapped into n channel input symbols from the same alphabet, the information density in these channel input symbols is k/n.

By using block and convolutional codes, the sender wants to get information to the receiver in a more reliable way than without using the codes. By repeating each information symbol sufficiently many times, one can achieve this and obtain a reliability arbitrarily close to 1. However, the price that one pays is the inefficient use of the channel: the rate of this sequence of codes tends to zero!

What Shannon was able to prove in 1948 (see: Shannon, C.E., *A mathematical theory of communication,* Bell Syst. Tech. J., *27,* pp. 379-423, 623-656, 1948) is that, as long as the rate R is smaller than some quantity $\mathcal{C}$, one can (for sufficiently long block lengths n) find encodings at rate R, such that the probability of incorrect decoding (when using maximum likelihood decoding) can be made arbitrarily small, while this is not possible for rates above that quantity! This result forms the foundation of the whole theory of error-correcting codes.

Definition 1.3.4 *The* entropy function $h(p)$ *is defined for* $0 \leq p \leq 1$ *by*

$$h(p) \quad = \quad \begin{cases} -p \log_2 p - (1 - p) \log_2(1 - p), & \text{if } 0 < p < 1, \\ 0, & \text{if } p = 0 \text{ or } 1. \end{cases}$$

$$\tag{1.2}$$

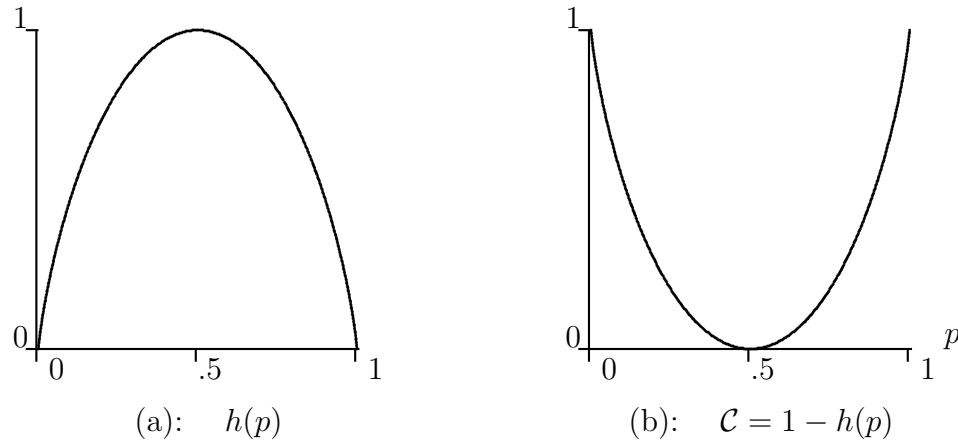(a): $h(p)$ (b): $\mathcal{C} = 1 - h(p)$

Figure 1.3: The entropy function $h(p)$ and the capacity $\mathcal{C}$ of the BSC

Although it will not be further discussed here, Shannon's information theory makes it possible to interpret $h(p)$ as the uncertainty that the receiver of a specific binary symbol (transmitted over the BSC with error probability p) still has about the actually transmitted symbol. In other words, $1 - h(p)$ is the amount of information that the received symbol carries about the transmitted symbol.

Theorem 1.3.5 (Shannon) *Consider the BSC with error probability p and let $\mathcal{C} = 1 - h(p)$. Then, for each rate R with $R < \mathcal{C}$, an infinite sequence of encodings E_l exists, where E_l maps binary k_l-tuples to binary n_l-tuples with $k_l = \lceil Rn_l \rceil$ (so E_l has rate $> R$), such that the corresponding maximum-likelihood decoding algorithms have a probability of incorrect decoding that goes exponentially fast to 0 as a function of n_l, for $l \to \infty$.*

For rates R greater than $\mathcal{C}$, no encodings can be made with error probabilities tending to zero.

One should realize that a decoding algorithm for an infinite class of codes that does not always yield the most likely transmitted sequence may still have a negligible probablity of incorrect decoding, when the length of these codes tends to infinity.

The quantity $\mathcal{C}$ in Theorem 1.3.5 is called the *capacity* of the channel. The entropy function and the capacity function are depicted in Figure 1.3.

As one can see in Figure 1.3 (b), the BSC with $p = 1/2$ cannot be used to transmit any information. The receiver may as well write down his own sequence of symbols instead of listening to the channel. At the other extreme, $p = 0$ and $p = 1$ imply that information can be sent at rate 1.

It is the ultimate goal of coding theory to find (families of) codes that approach the capacity of the BSC and that have efficient decoding algorithms.

1.4   Problems

1.4.1 Suppose that four messages are encoded into 000000, 001111, 110011 and 111100. These four messages are transmitted with equal probablity over a BSC with error probability p. If one receives a sequence different from the four sequences above, one knows that errors have been made during the transmission.

What is the probability that errors have been made during the transmission and that the receiver will not find out?

1.4.2 Suppose that either $(-1, -1, -1)$ or $(+1, +1, +1)$ is transmitted (each with probability 1/2) over the Gaussian channel defined by the density function

$$p(y|x) = \frac{1}{\sqrt{2\pi}} e^{-(y-x)^2/2}.$$

What is the most likely transmitted sequence when the word $(-1, +0.01, +0.01)$ is received?

What is the answer to this question if a maximum likelihood, hard decision decoding algorithm has been applied?

Chapter 2

Linear block codes

2.1 Block codes

In this chapter, block codes for the q-ary symmetric channel will be introduced. Let n be fixed and let the input and output symbols of the channel belong to an alphabet Q of cardinality q. The set of Q-ary n-tuples is denoted by Q^n.

A distance metric on Q^n that reflects the properties of the q-ary symmetric channel very well, is the following.

Definition 2.1.1 *The* Hamming distance $d(\mathbf{x}, \mathbf{y})$ *between* $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ *and* $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ *in* Q^n *is given by*

$$d(\mathbf{x}, \mathbf{y}) = |\{1 \le i \le n \mid x_i \ne y_i\}|. \tag{2.1}$$

In words: $d(\mathbf{x}, \mathbf{y})$ is the number of coordinates, where $\mathbf{x}$ and $\mathbf{y}$ differ. It follows from the properties of the q-ary symmetric channel that the more $\mathbf{x}$ and $\mathbf{y}$ differ, the more unlikely one will be received if the other was transmitted.

It is very simple to verify that $d(\mathbf{x}, \mathbf{x}) = 0$, $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ and that the triangle inequality holds: $d(\mathbf{x}, \mathbf{z}) \le d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$ for any $\mathbf{x}, \mathbf{y}$ and $\mathbf{z}$ in Q^n. So the Hamming distance is indeed a distance function.

A q-ary *block code* C of *length* n is any nonempty subset of Q^n. The elements of C are called *codewords*. If $|C| = 1$, the code is called *trivial*. Quite often one simply speaks of a "code" instead of a "block code".

To maximize the error-protection, one needs codewords to have sufficient mutual distance.

Definition 2.1.2 *The* minimum distance d *of a non-trivial code* C *is given by*

$$d = \min\{d(\mathbf{x}, \mathbf{y}) \mid \mathbf{x} \in C, \mathbf{y} \in C, \mathbf{x} \ne \mathbf{y}\}. \tag{2.2}$$

The error-correcting capability e *is defined by*

$$e = \left\lfloor \frac{d-1}{2} \right\rfloor. \tag{2.3}$$

The reason for the name error-correcting capability is quite obvious. If d is the minimum distance of a code C and if during the transmission of the codeword $\mathbf{c}$ over the channel at most e errors have been made, the received word $\mathbf{r}$ will still be closer to $\mathbf{c}$ than to any other codeword. So a maximum likelihood decoding algorithm applied to $\mathbf{r}$ will result in $\mathbf{c}$.

However, codes can also be used for *error-detection*. Let t be some integer, $t \leq e$. Then it follows from the definition of d that no word in Q^n can be at distance at most t from one codeword, while at the same time being at distance up to $d - t - 1$ from some other codeword. This means that one can correct up to t errors, but also detect if more than t have occurred, as long as not more than $d - t - 1$ errors have occurred.

Similarly, a code C with minimum distance d can also be used for the simultaneous correction of errors and erasures. Let e and f be fixed such that $2e + f < d$. A received word $\mathbf{r}$ with at most f erasures cannot have distance $\leq e$ from two different codewords $\mathbf{c}_1$ and $\mathbf{c}_2$ at the other $n - f$ coordinates, because that would imply that $d(\mathbf{c}_1, \mathbf{c}_2) \leq 2e + f < d$. It follows that C is e-error, f-*erasure-correcting*.

A different interpretation of Definition 2.1.2 is that *spheres* of radius e around the codewords are disjoint. Let $B_r(\mathbf{x})$ denote the sphere of radius r around $\mathbf{x}$, defined by $\{\mathbf{y} \in Q^n \mid d(\mathbf{y}, \mathbf{x}) \leq r\}$. To determine the cardinality of $B_r(\mathbf{x})$ we first want to find the number of words at distance i to $\mathbf{x}$. To find these words, one needs to choose exactly i of the n coordinates of $\mathbf{x}$ and replace each by one of the other $q - 1$ alphabet symbols. So there are $\binom{n}{i}(q-1)^i$ words at distance i from $\mathbf{x}$ and thus

$$|B_r(\mathbf{x})| = \sum_{i=0}^{r} \binom{n}{i}(q-1)^i. \tag{2.4}$$

Since all spheres with radius e around the $|C|$ codewords are disjoint and there are only q^n distinct words in Q^n, we have proved the following theorem.

Theorem 2.1.3 (Hamming bound) *Let C be an e-error-correcting code. Then*

$$|C| \sum_{i=0}^{e} \binom{n}{i}(q-1)^i \leq q^n. \tag{2.5}$$

Let $d(\mathbf{x}, C)$ denote the distance from $\mathbf{x}$ to the code C, so $d(\mathbf{x}, C) = \min\{d(\mathbf{x}, \mathbf{c}) \mid \mathbf{c} \in C\}$. The next notion tells us how far a received word can possibly be removed from a code.

Definition 2.1.4 *The* covering radius ρ *of a code C is given by*

$$\rho = \max\{d(\mathbf{x}, C) \mid \mathbf{x} \in Q^n\}. \tag{2.6}$$

Since every word $\mathbf{x}$ in Q^n is at distance at most ρ to some codeword, say $\mathbf{c}$, it is also inside at least one of the spheres of radius ρ around the codewords (namely $B_\rho(\mathbf{c})$). So these spheres together cover Q^n. This proves the following theorem.

Theorem 2.1.5 *Let C be a code with covering radius ρ then*

$$|C| \sum_{i=0}^{\rho} \binom{n}{i} (q-1)^i \geq q^n. \tag{2.7}$$

It follows from (2.5) and (2.7) that $e \leq \rho$ for every code. Equality turns out to be possible!

Definition 2.1.6 *An e-error-correcting code C with covering radius ρ is called* perfect *if*

$$e = \rho.$$

A different way of saying that an e-error-correcting code C is perfect is "the spheres with radius e around the codewords cover Q^n" or "every element in Q^n is at distance at most e from a unique codeword".

Equations (2.5) and (2.7) imply the following theorem.

Theorem 2.1.7 (Sphere packing bound) *Let C be an e-error-correcting code. Then C is perfect if and only if*

$$|C| \sum_{i=0}^{e} \binom{n}{i} (q-1)^i = q^n. \tag{2.8}$$

We have already seen a code that is perfect: the binary code of length 5 with just the two codewords $(0,0,0,0,0)$ and $(1,1,1,1,1)$. It is a special example of what is called the q-ary *repetition code* of length n:

$$C = \{(\overbrace{c,c,\ldots,c}^{n}) \mid c \in Q\}.$$

This code has minimum distance $d = n$. So, for odd n the code has $e = (n-1)/2$. In the binary case one can take $Q = \{0,1\}$ and gets $C = \{\mathbf{0},\mathbf{1}\}$, where $\mathbf{0} = (0,0,\ldots,0)$ and $\mathbf{1} = (1,1,\ldots,1)$. It follows that a word with at most $(n-1)/2$ coordinates equal to 1, is in $B_{(n-1)/2}(\mathbf{0})$, but not in $B_{(n-1)/2}(\mathbf{1})$, while a word with at least $(n+1)/2$ coordinates equal to 1, is in $B_{(n-1)/2}(\mathbf{1})$, but not in $B_{(n-1)/2}(\mathbf{0})$. This means that $\rho = (n-1)/2 = e$ and that the binary repetition code of odd length is perfect.

Instead of stating that C is a code of length n, minimum distance d and cardinality M, we shall just say that C is a (n, M, d) code.

Clearly if one applies the same coordinate permutation to all the codewords of C one obtains a new code C' that has the same parameters as C. The same is true if for each coordinate one allows a permutation of the symbols in Q. Codes that can be obtained

from each other in this way are called *equivalent*. From a coding theoretic point of view they are the same.

The *rate* of a q-ary code C of length n is defined by

$$R = \frac{\log_q |C|}{n}. \tag{2.9}$$

This definition coincides with (1.1), if one really maps q-ary k-tuples into q-ary n-tuples (then $R = k/n$), but it also coincides with (1.1) in general. Indeed, if the encoder in (1.3.1) has an input alphabet of size $|C|$ and maps each input symbol onto a unique codeword (so $k = 1$ and $M = |C|$), equation (1.1) reduces to (2.9).

Before we end this section, two more bounds will be given.

Theorem 2.1.8 (Singleton bound) *Let C be a q-ary (n, M, d) code. Then*

$$M \leq q^{n-d+1}. \tag{2.10}$$

Proof: Erase in every codeword the last $d - 1$ coordinates. Because all codewords originally have distance at least d, the new words will still all be distinct. Their length is $n - d + 1$. However, there are only q^{n-d+1} distinct words of length $n - d + 1$ over an alphabet of size q.

$\square$

Codes with parameters (n, q^{n-d+1}, d) (i.e. for which the Singleton bound holds with equality) are called *maximum-distance-separable codes* or simply *MDS codes*.

Now that we have seen several bounds that give an upper bound on the size of a code in terms of n and d, it is good to know that also lower bounds exist.

Theorem 2.1.9 (Gilbert-Varshamov bound) *There exist q-ary (n, M, d) codes satisfying*

$$M \geq \frac{q^n}{\sum_{i=0}^{d-1} \binom{n}{i}(q-1)^i}. \tag{2.11}$$

Proof: As long as a q-ary (n, M, d) code C satisfies $M < \frac{q^n}{\sum_{i=0}^{d-1} \binom{n}{i}(q-1)^i}$ the spheres with radius $d - 1$ around the words in C will not cover the set of all q-ary n-tuples, so one can add a word to C that has distance at least d to all elements of C.

$\square$

2.2 Linear codes

If one assumes the alphabet Q and the code C to have some internal structure, one can construct and analyze codes much more easily than in the absence of such structures.

From now on Q will always have the structure of the Galois field $GF(q)$, the finite field with q elements (for the reader who is not familiar with finite fields Appendix A is included). As a consequence $q = p^m$ for some prime p. Very often q will simply be 2.

Now that $Q = GF(q)$, we can associate the set of words Q^n with an n-dimensional vector space over $GF(q)$. It will be denoted by $V_n(q)$. For $V_n(2)$ we shall simply write V_n. The elements in $V_n(q)$ are of course vectors, but will occasionally still be called words. The symbols denoting vectors will be underlined.

If a codeword $\underline{c}$ has been transmitted but the vector $\underline{r}$ is received, the error pattern, caused by the channel, is the vector $\underline{e}$ with

$$\underline{r} = \underline{c} + \underline{e}. \tag{2.12}$$

The addition in (2.12) denotes the vector addition in $V_n(q)$. The number of errors that occurred during the transmission is the number of non-zero entries in $\underline{e}$.

Definition 2.2.1 *The* Hamming weight $w(\underline{x})$ *of a vector $\underline{x}$ is the number of non-zero coordinates in $\underline{x}$. So*

$$w(\underline{x}) = d(\underline{x}, \underline{0}).$$

Now that Q^n has the structure of the vector space $V_n(q)$, we can define the most important general class of codes.

Definition 2.2.2 *A* linear code C *of length n is any linear subspace of $V_n(q)$.*

If C has dimension k and minimum distance d, one says that C is an $[n, k, d]$ code.

Note that a q-ary (n, M, d) code C has cardinality M, while a q-ary $[n, k, d]$ code C is linear and has cardinality q^k. The parameter d in the notations (n, M, d) and $[n, k, d]$ is sometimes omitted.

To determine the minimum distance of an unstructured (n, M, d) code C one has to compute the distance between all $\binom{M}{2}$ pairs of codewords. For linear codes without further structure, the next theorem will prove that this complexity is only $M - 1$. This is the first bonus of the special structure of linear codes.

Theorem 2.2.3 *The minimum distance of a linear code C is equal to the minimum non-zero weight in C.*

Proof: Since C is linear, with $\underline{x}$ and $\underline{y}$ in C also $\underline{x} - \underline{y}$ is in C. The theorem now follows from the two observations:

$$d(\underline{x}, \underline{y}) = d(\underline{x} - \underline{y}, \underline{0}) = w(\underline{x} - \underline{y}),$$
$$w(\underline{x}) = d(\underline{x}, \underline{0}),$$

which state that the distance between any two distinct codewords is equal to the weight of some non-zero codeword and vice versa.

□

So, to determine the minimum distance of a linear code, one never needs to do more than to find the lowest weight of all $M - 1$ non-zero codewords.

There are two standard ways of describing a k-dimensional linear subspace: one by means of k independent basis vectors, the other uses $n - k$ linearly independent equations. Both techniques turn out to be quite powerful in this context.

Definition 2.2.4 *A generator matrix G of an $[n, k, d]$ code C is a $k \times n$ matrix, of which the k rows form a basis of C. One says "the rows of G generate C".*

It follows that

$$C = \{\underline{a}G \mid \underline{a} \in V_k(q)\} \tag{2.13}$$

The codeword $\underline{c} = \underline{a}G$ in (2.13) is the result of the encoding algorithm "multiplication by G" applied to the so-called *information vector* or *message vector $\underline{a}$*.

Example 2.2.5 A generator matrix of the q-ary $[n, 1, n]$ repetition code is given by

$$G = (1 \ 1 \ \cdots\cdots \ 1).$$

Example 2.2.6 The binary *even weight code* is defined as the set of all words of even weight. It is a linear code with parameters $[n, n - 1, 2]$. A generator matrix of the even weight code is given by

$$G = \begin{pmatrix} 1 & 0 & 0 & \cdots & \cdots & 0 & 1 \\ 0 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 0 & 0 & 1 & \cdots & \cdots & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & & \vdots & \vdots \\ \vdots & \vdots & \vdots & & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \cdots & 1 & 1 \end{pmatrix}.$$

Examples 2.2.5 and 2.2.6 are also examples of what is called a generator matrix in *standard form*, i.e. a generator matrix of the form $(I_k \ P)$, where I_k is the $k \times k$ identity matrix. If G is in standard form, the first k coordinates of a codeword $\underline{a}G$ produce the information vector $\underline{a}$ itself! For this reason these coordinates are called *information symbols*. The last $n - k$ coordinates are added to the k information symbols to make error-correction possible.

Since the generator matrix of a linear code has full row rank, it is quite obvious that any linear code is equivalent to a linear code that does have a generator matrix in standard form. The quantity $r = n - k$ is called the *redundancy* of the code.

Non-linear codes of cardinality q^k sometimes also have the property that on the first k coordinates all q^k possible (information) sequences occur. These codes are called *systematic* on the first k coordinates. It follows from the proof of the Singleton bound

(Theorem 2.1.8) that MDS codes are systematic on every k-tuple of coordinates. Also the converse holds (see Problem 2.5.10).

The second way of describing linear codes is as follows.

Definition 2.2.7 *A* parity check matrix H *of an* $[n, k, d]$ *code C is an* $(n-k) \times n$ *matrix, satisfying*

$$\underline{c} \in C \Longleftrightarrow H\underline{c}^T = \underline{0}^T. \tag{2.14}$$

In other words C is the null space (solution space) of the $n - k$ linearly independent equations $H\underline{x}^T = \underline{0}^T$.

If G is in standard form $(I_k \quad P)$, one can take $H = (-P^T \quad I_{n-k})$. This is so, because this H has rank $n - k$ and satisfies $GH^T = -P + P = O_{k,n-k}$, the all-zero matrix of size $k \times (n - k)$. From this standard form of H it is clear that the last $n - k$ coordinates of a codeword are uniquely determined by the initial k coordinates and the parity check equations. For this reason the last $n - k$ symbols are called the *parity check symbols* of C.

Let $(\underline{x}, \underline{y})$ denote the regular inner product $\sum_{i=1}^n x_i y_i$ in $V_n(q)$. We shall say that two vectors are *orthogonal* to each other if they have inner product zero. A word of warning is in place: in $V_n(q)$ a word can be orthogonal to itself without being $\underline{0}$. For instance, in $V_4(3)$ the vector $(1, 0, 2, 1)$ is orthogonal to itself! In V_n every even-weight vector is orthogonal to itself. In particular, it follows that a set of mutually orthogonal vectors do not have to be linearly independent.

Definition 2.2.8 *The* dual code $C^\perp$ *of an* $[n, k, d]$ *code C is defined by*

$$C^\perp = \{\underline{x} \in V_n(q) \mid (\underline{x}, \underline{c}) = 0 \text{ for all } \underline{c} \in C\}. \tag{2.15}$$

It is quite clear that $C^\perp$ is a linear subspace of dimension $n - k$. So $C^\perp$ is an $[n, n-k, d^\perp]$ code, where $d^\perp$ denotes the minimum distance of $C^\perp$. Also, because $GH^T = O$, it is straightforward to check that $C^\perp$ has as its generator matrix the parity check matrix H of C and as its parity check matrix the generator matrix G of C. Another easy observation is that the dual code of the dual of C is C itself: $(C^\perp)^\perp = C$.

We have already seen that a non-zero word can be orthogonal to itself. It is also possible that a non-zero vector can be in C and in $C^\perp$ at the same time. There exist, as a matter of fact, codes C that are completely contained in their dual $C^\perp$. Such codes are called *self-orthogonal*. If $C = C^\perp$, the code is called *self-dual*.

Examples 2.2.5 and **2.2.6** (continued)

Over $GF(2)$ the two matrices in Examples 2.2.5 and 2.2.6 are orthogonal to each other. It follows that in $V_n(2)$ the repetition code and the even weight code are duals of each other.

We now come to the second important advantage of having the extra structure of being linear available. Up to now, the only way to decode a received word was to compare it

with all codewords and find the closest. This technique has complexity $|C|$ and thus for q-ary $[n, k, d]$ codes complexity q^k.

Definition 2.2.9 *Let C be a q-ary $[n, k, d]$ code with parity check matrix H. The syndrome $\underline{s}$ of a vector $\underline{x}$ in $V_n(q)$ is the vector in $V_{n-k}(q)$ defined by $\underline{s} = H\underline{x}^T$.*

Note that a syndrome $\underline{s}$ is a column vector, while the vectors related to transmitted or received words are row vectors.

If the syndrome of a received word $\underline{r}$ is $\underline{0}$, then $\underline{r}$ is a codeword and most likely no error has occurred during the transmission.

In general, if the codeword $\underline{c}$ has been transmitted and the word $\underline{r} = \underline{c} + \underline{e}$ has been received, where $\underline{e}$ is the error pattern, one has

$$H\underline{r}^T = H(\underline{c} + \underline{e})^T = H\underline{c}^T + H\underline{e}^T = H\underline{e}^T. \tag{2.16}$$

So, the syndrome of $\underline{r}$ is completely determined by that of $\underline{e}$. The real decoding problem is how to find the codeword $\underline{c}$ that is closest to the received word $\underline{r}$, in other words to find a vector $\underline{e}$ of minimal weight, such that $\underline{r} - \underline{e}$ is in C.

Now let $\underline{s}$ in $V_{n-k}(q)$ be the syndrome of a received word $\underline{r}$. Then not just $\underline{r}$ is a solution of $\underline{s} = H\underline{x}^T$, but all vectors $\underline{r} + \underline{c}$ with $\underline{c}$ in C form the solution space of this system of linear equations. The set $\{\underline{r} + \underline{c} \mid \underline{c} \text{ in } C\}$ forms a *coset* of C in the additive group $V_n(q)$. We need to find a vector $\underline{e}$ of minimal weight in this coset. This minimum weight word in the coset is called the *coset leader* of the coset. As we shall see later, this coset leader does not have to be unique. However, if C is e-error-correcting, each word of weight at most e will be the unique coset leader of a unique coset. Indeed if two distinct words of weight at most e would lie in the same coset, their difference would be a codeword of weight at most $2e$, a contradiction with the minimum distance of C, which is $2e + 1$ or $2e + 2$.

Once $\underline{e}$ has been found, the codeword $\underline{c} = \underline{r} - \underline{e}$ is a good maximum-likelihood estimate of the transmitted codeword, in the sense that no other codeword has a higher probability of being transmitted. That $\underline{r} - \underline{e}$ is indeed a codeword, follows from the fact that both vectors have the same syndrome, so their difference has syndrome $\underline{0}$, i.e. is in C. It follows from the above that the next algorithm is indeed a maximum-likelihood decoding algorithm for each linear code.

Algorithm 2.2.10 (Syndrome decoding) *Let $\underline{r}$ be the received vector.*

1. *Compute the syndrome $\underline{s} = H\underline{r}^T$ of the received vector $\underline{r}$.*

2. *Find the coset leader $\underline{e}$ of the coset with syndrome $\underline{s}$.*

3. *Decode $\underline{r}$ into $\underline{c} = \underline{r} - \underline{e}$.*

Once one has made a table of all syndromes with a corresponding coset leader, Algorithm 2.2.10 yields a maximum-likelihood decoding algorithm with complexity q^{n-k}. So for $k > n/2$, this algorithm is faster than the brute-force approach of comparing the received word with all q^k codewords. In subsequent chapters we shall meet codes with decoding algorithms that do not have an exponentially-fast growing complexity.

Example 2.2.11 Consider the binary $[6, 3, 3]$ code C with parity check matrix

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

That the minimum distance of C is indeed 3, can (because the parameters are small in this case) be checked quite easily by hand (and of course by Theorem 2.2.3). Indeed, C contains four codewords of weight 3 and three of weight 4.

A much simpler way to find the minimum distance of C is the observation that any two distinct columns of H are linearly independent. So a non-zero word can only have syndrome $\underline{0}$ if its weight is at least three.

Now, seven syndromes turn out to have a unique coset leader. For instance, the syndrome $\underline{s} = (1, 0, 1)^T$ is equal to the fifth column in H, so it has $(0, 0, 0, 0, 1, 0)$ as unique coset leader.

However, to write syndrome $\underline{s} = (1, 1, 1)^T$ as linear combination of columns of H one needs two columns. There are three possible ways to do this, giving rise to the three coset leaders: $(1, 0, 0, 1, 0, 0)$, $(0, 1, 0, 0, 1, 0)$ or $(0, 0, 1, 0, 0, 1)$. We can take any one of these, say the first.

If the error pattern is indeed one of these eight coset leaders, one will find the transmitted codeword back. This is the case for each single-error pattern, but also for $(1, 0, 0, 1, 0, 0)$. So, syndrome decoding of C, when used over the BSC with error probability p, yields the following probability of correct decoding:

$$(1 - p)^6 + 6p(1 - p)^5 + p^2(1 - p)^4.$$

Note that in the expansion of this expression no linear term occurs. In the uncoded case, i.e. if the information vector (a_1, a_2, a_3) is equal to the transmitted vector, the probablity of correct "decoding" is $(1 - p)^3$. It follows that for small values of p and here in fact for $0 < p < 1/2$, the $[6, 3, 3]$ code will give a better performance.

To conclude this section, we present the "linear" version of Theorems 2.1.8 and 2.1.9. The proof of the first one is immediate.

Theorem 2.1.8 (Singleton bound) (continued)

Let C be a q-ary $[n, k, d]$ code. Then

$$k \leq n - d + 1. \tag{2.17}$$

Theorem 2.1.9 (Gilbert-Varshamov bound) (continued)

Let k be the smallest integer satisfying

$$q^k \geq \frac{q^n}{\sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i}. \tag{2.18}$$

Then a q-ary $[n, k, d]$ code does exist.

Proof: If one word, say $\underline{u}$, has distance at least d to all words in a linear code C with minimum distance d, then the linear code

$$\{\alpha \underline{u} + \underline{c} \mid \alpha \text{ in } GF(q), \underline{c} \text{ in } C\}$$

properly contains C and still has minimum distance at least d. Indeed, $d(\alpha' \underline{u} + \underline{c}', \alpha'' \underline{u} + \underline{c}'') = d(\underline{c}', \underline{c}'') \geq d$, if $\alpha' = \alpha''$ and $\underline{c}', \underline{c}''$ are distinct codewords in C. If $\alpha' \neq \alpha''$, one has $d(\alpha' \underline{u} + \underline{c}', \alpha'' \underline{u} + \underline{c}'') = d(\alpha''' \underline{u}, \underline{c}''') = d(\underline{u}, \underline{c}'''/\alpha''') \geq d$, by assumption. Here $\alpha''' = \alpha' - \alpha'' \neq 0$ and $\underline{c}''' = \underline{c}' - \underline{c}'' \in C$.

Starting with $C = \{\underline{0}\}$ apply the above argument repeatedly until (2.18) is met.

$\square$

2.3 The MacWilliams relations

When studying properties of a linear code, it is often important to know how many codewords have a certain weight (in particular this is true for the weights close to d).

Definition 2.3.1 *Let C be a code. Then the* weight enumerator $A(z)$ *of C is given by*

$$A(z) = \sum_{i=0}^{n} A_i z^i = \sum_{\underline{c} \in C} z^{w(\underline{c})}. \tag{2.19}$$

So, A_i, $0 \le i \le n$, counts the number of code words of weight i in C.

For instance, the q-ary repetition code of length n has weight enumerator $1 + (q-1)z^n$. The code in Example 2.2.11 has weight enumerator $1 + 4z^3 + 3z^4$.

In 1963, F.J. MacWilliams showed that the weight enumerators of a linear code C and of its dual code $C^{\perp}$ are related by a rather simple formula. This relation will turn out to be a very powerful tool.

We need a definition and a lemma first.

Definition 2.3.2 *A* character χ *of $GF(q)$ is a mapping of $GF(q)$ to the set of complex numbers with absolute value 1, satisfying*

$$\chi(\alpha + \beta) = \chi(\alpha)\chi(\beta) \text{ for all } \alpha \text{ and } \beta \text{ in } GF(q). \tag{2.20}$$

The principal *character maps every field element to 1.*

It follows from $\chi(0) = \chi(0+0) = \chi^2(0)$ that $\chi(0) = 1$.

An example of a non-principal character χ of $GF(p) = \{0, 1, \ldots, p-1\}$ is given by $\chi(a) = \exp^{a2\pi i/p}$. It is also quite easy to find a non-principal character of $GF(q)$. For instance, if $GF(q)$ has characteristic p and ω is a complex, primitive p-th root of unity, the function $\chi(\alpha) = \omega^{A(\alpha)}$, where A is any non-trivial linear mapping from $GF(q)$ to $GF(p)$, will define a non-principal character of $GF(q)$. When representing $GF(q)$, $q = p^m$, as an m-dimensional vector space over $GF(p)$ the projection on the first coordinate already gives such a mapping. Also the *trace*-function Tr, defined by $Tr(x) = x + x^p + \ldots x^{p^{m-1}}$ is such a mapping (see also Problem A.6.19). The fact however is that below we do need to make an explicit choice for χ.

Lemma 2.3.3 *Let χ be a character of $GF(q)$. Then*

$$\sum_{\alpha \in GF(q)} \chi(\alpha) = \begin{cases} q, & \text{if } \chi \text{ is the principal character,} \\ 0, & \text{otherwise.} \end{cases} \tag{2.21}$$

Proof: For the principal character the assertion is trivial. For a non-principal character, take β in $GF(q)$ such that $\chi(\beta) \neq 1$. Then

$$\chi(\beta) \sum_{\alpha \in GF(q)} \chi(\alpha) = \sum_{\alpha \in GF(q)} \chi(\alpha + \beta) = \sum_{\alpha \in GF(q)} \chi(\alpha).$$

This equation can be rewritten as $(1 - \chi(\beta)) \sum_{\alpha \in GF(q)} \chi(\alpha) = 0$. Since $\chi(\beta) \neq 1$ the assertion follows.

$\square$

Theorem 2.3.4 (MacWilliams) *Let $A(z)$ be the weight enumerator of a q-ary, linear code C and let $B(z)$ be the weight enumerator of the dual code $C^{\perp}$. Then*

$$
\begin{aligned}
B(z) &= \frac{1}{|C|}(1 + (q-1)z)^n A\left(\frac{1-z}{1+(q-1)z}\right) = \\
&= \frac{1}{|C|} \sum_{i=0}^{n} A_i (1-z)^i (1+(q-1)z)^{n-i}.
\end{aligned}
\tag{2.22}
$$

Proof: Let χ be a non-principal character of $GF(q)$. We shall evaluate the expression

$$\sum_{\underline{c} \in C} \sum_{\underline{u} \in V_n(q)} \chi((\underline{c}, \underline{u})) z^{w(\underline{u})}.
\tag{2.23}$$

in two different ways, obtaining in this way the left hand and the right hand sides in (2.22).

Changing the order of summation in (2.23) yields

$$\sum_{\underline{u} \in V_n(q)} z^{w(\underline{u})} \sum_{\underline{c} \in C} \chi((\underline{c}, \underline{u})).
\tag{2.24}$$

The inner sum in (2.24) is equal to $|C|$, when $\underline{u}$ is an element in $C^{\perp}$, because all inner products $(\underline{c}, \underline{u})$ are zero in this case. However, if $\underline{u}$ is not an element in $C^{\perp}$, the inner products $(\underline{c}, \underline{u})$ take on each value in $GF(q)$ equally often (by the linearity of the inner product). It follows from Lemma 2.3.3 that the inner sum in (2.24) is now equal to 0. So (2.24), and thus also (2.23), is equal to

$$\sum_{\underline{u} \in C^{\perp}} z^{w(\underline{u})}|C| = |C|B(z),
\tag{2.25}$$

by Definition 2.3.1 applied to $C^{\perp}$.

Now we shall evaluate (2.23) in a different way.
The inner sum of (2.23) $\sum_{\underline{u} \in V_n(q)} \chi((\underline{c}, \underline{u})) z^{w(\underline{u})}$ is equal to

$$\sum_{(u_1, u_2, \ldots, u_n) \in V_n(q)} \chi(c_1 u_1 + c_2 u_2 + \cdots + c_n u_n) z^{w((u_1, u_2, \cdots, u_n))} =$$

$$\sum_{u_1 \in V_1(q)} \cdots \sum_{u_n \in V_1(q)} \chi(c_1 u_1) \cdots \chi(c_n u_n) z^{w(u_1)} \cdots z^{w(u_n)} =$$

$$\prod_{i=1}^{n} \sum_{u_i \in V_1(q)} \chi(c_i u_i) z^{w(u_i)}. \tag{2.26}$$

The inner sum in this last expression is equal to $1 + (q-1)z$ if $c_i = 0$. If $c_i \neq 0$ the inner sum is equal to

$$1 + z \sum_{\alpha \neq 0} \chi(\alpha) = 1 - z\chi(0) = 1 - z,$$

by Lemma 2.3.3.

So, the inner sum in (2.23) is equal to $(1-z)^{w(\mathcal{C})}(1 + (q-1)z)^{n-w(\mathcal{C})}$ and thus (by Definition 2.3.1) Equation (2.23) can be rewritten as

$$\sum_{i=0}^{n} A_i (1-z)^i (1 + (q-1)z)^{n-i}.$$

Setting this equal to (2.25) proves the theorem.

$\square$

Instead of finding the weight enumerator of a $[n, k, d]$ code with $k > n/2$ directly, it is often easier to find the weight enumerator of its dual code and then apply the MacWilliams relation.

Examples 2.2.5 and **2.2.6** (continued)

The weight enumerator of the repetition code is $1 + (q-1)z^n$. So its dual has weight enumerator

$$\frac{1}{q} \left\{ (1 + (q-1)z)^n + (q-1)(1-z)^n \right\}.$$

In particular in the binary case, it follows that the even weight code has weight enumerator

$$\frac{1}{2} \left\{ (1+z)^n + (1-z)^n \right\} = \sum_{i \text{ even}} \binom{n}{i} z^i.$$

Of course, the fact that the even weight code does not contain odd weight words and does contain all even weight words just is the definition of this code.

A more interesting example of the MacWilliams relations will be given in the next chapter.

The weight enumerator $A(z)$ of a binary linear code C is very helpful when studying the probability Pr_e that a maximum likelihood decoder makes a decoding error, i.e. the probability that, while one codeword has been transmitted, the received word is in fact closer to another codeword.

By the linearity of C we may assume that $\underline{0}$ was the transmitted codeword. Let $\underline{c}$ be a non-zero codeword of weight w. Let $Pr_e(\underline{c})$ denote the probability that the received vector is closer to $\underline{c}$ than to $\underline{0}$, though $\underline{0}$ was transmitted. Then

$$Pr_e(\underline{c}) = \sum_{i \geq \lceil w/2 \rceil} \binom{w}{i} p^i (1-p)^{w-i}.$$

Now the probablity Pr_e of incorrectly decoding can be bounded above as follows

$$Pr_e \leq \sum_{\underline{c} \in C, \underline{c} \neq \underline{0}} Pr_e(\underline{c}).$$

It follows from

$$
\begin{aligned}
\sum_{i \geq \lceil w/2 \rceil} \binom{w}{i} p^i (1-p)^{w-i} &\leq p^{w/2}(1-p)^{w/2} \sum_{i \geq \lceil w/2 \rceil} \binom{w}{i} \leq \\
&\leq p^{w/2}(1-p)^{w/2} 2^w = \\
&= \left(2\sqrt{p(1-p)}\right)^w
\end{aligned}
$$

that

$$Pr_e \leq \sum_{w>0} A_w \left(2\sqrt{p(1-p)}\right)^w,$$

where A_w denotes the number of codewords of weight w in C.

This proves the following theorem.

Theorem 2.3.5 *The probablity Pr_e that a maximum likelihood decoding algorithm decodes a received word incorrectly, when a codeword from a linear code with weight enumerator $A(z)$ has been transmitted, satisfies*

$$Pr_e \leq A\left(2\sqrt{p(1-p)}\right) - 1. \tag{2.27}$$

2.4 Linear unequal error protection codes

There are applications where one wants to protect some data bits better than others. For instance, an error in the sign or in the most significant bit in the binary representation of a number is much more serious than in the least significant bit.

It will turn out that the extra protection that one can give to some of the information bits, when using a linear code C, depends very much on the particular generator matrix G that has been chosen.

Definition 2.4.1 *The* separation vector $\underline{s}(G) = (s(G)_1, s(G)_2, \ldots,$
$s(G)_k)$ *of a generator matrix G of a k-dimensional linear code C in $V_n(q)$ is defined by*

$$s(G)_i = \min\{w(\underline{a}G) \mid \underline{a} \in V_k(q), \ a_i \neq 0\}, \qquad 1 \leq i \leq k. \tag{2.28}$$

It follows from this definition that two information vectors that differ in the i-th coordinate give rise to codewords that differ in at least $s(G)_i$ coordinates. Indeed, if $\underline{a}$ and $\underline{b}$ are elements in $V_k(q)$ such that $(a_i - b_i) \neq 0$, then $(\underline{a} - \underline{b})G$ will have weight at least $s(G)_i$. It follows that $\underline{a}G$ and $\underline{b}G$ will have distance at least $s(G)_i$. This observation proves the following theorem.

Theorem 2.4.2 *Let $\underline{s}(G)$ be the separation vector of a generator matrix G of a k-dimensional linear code C in $V_n(q)$. Then complete maximum likelihood decoding of a received word $\underline{r} = \underline{a}G + \underline{e}$ will yield the correct i-th information symbol a_i, $1 \leq i \leq k$, if the error pattern $\underline{e}$ has weight at most $\lfloor (s(G)_i - 1)/2 \rfloor$.*

Obviously, the minimum distance d of a linear code C is equal to the minimum of the $s(G)_i$'s, i.e. $d = \min_{1 \leq i \leq k} s(G)_i$.

A linear code that has a generator matrix G such that not all the coordinates in its separation vector are equal is called a *linear unequal error protection code*. This will be abbreviated to: a LUEP code.

Example 2.4.3 The matrix

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

generates a binary $[4, 2, 2]$ code with separation vector $(3, 2)$. So a_1 can still be determined correctly, if a single error has been made.

By permuting the rows of the generator matrix (together with the same permutation applied to the coordinates of the message vector $\underline{a}$) one can obtain a generator matrix G of the same code with the property:

$$s(G)_1 \geq s(G)_2 \geq \cdots \geq s(G)_k.$$

From now on, we shall always assume that the separation vector is ordered in this way.

Different generator matrices of the same code may have separation vectors that cannot be compared, e.g. (7,5,3) versus (6,5,5). Fortunately, every linear code C has a generator matrix with the property that its separation vector is coordinatewise greater than or equal to the separation vector of any other generator matrix of C.

Definition 2.4.4 *A generator matrix G of a linear code C is called* optimal *if for every generator matrix of C each coordinate of its separation vector is smaller than or equal to the corresponding coordinate of $\underline{s}(G)$.*

Clearly, if a code C has several optimal generator matrices, they will all have the same separation vector, called the *separation vector* of the code.

Of course different codes of the same length and dimension can exist that have incomparable separation vectors. For instance the matrices

$$G_1 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \quad \text{and}$$

$$G_2 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

generate 2-dimensional codes in V_7 with optimal separation vectors $(6, 2)$ resp. $(5, 4)$.

To prove that each linear code has an optimal generator matrix, we need some notation.

Let C be a k-dimensional code in $V_n(q)$ and let G be a generator matrix of C. The i-th row of G will be denoted by g_i, $1 \leq i \leq k$, and the set of rows of G by $R(G)$. Further $C[w]$ is defined as the set of codewords in C of weight at most w, so $C[w] = \{c \in C \mid w(c) \leq w\}$. Let $< C[w] >$ be the linear span of the vectors in $C[w]$. Clearly, $< C[w] >$ is a linear subcode of C. Let $R(G)[w]$ be the smallest subset of $R(G)$ with a linear span containing $< C[w] >$, i.e.

$$R(G)[w] = \cap \{X \subset R(G) \mid \ < C[w] > \ \subset \ < X >\}.$$

Lemma 2.4.5 *The generator matrix G of a k-dimensional code in $V_n(q)$ satisfies*

$$s(G)_i \leq w \qquad \Leftrightarrow \qquad g_i \in R(G)[w], \tag{2.29}$$

for all $0 \leq w \leq n$ and $1 \leq i \leq k$.

Proof:
$\Rightarrow$: If $g_i \notin R(G)[w]$ then $C[w] \subset \ < R(G) \setminus \{g_i\} >$ and hence $s(G)_i > w$.
$\Leftarrow$: If $g_i \in R(G)[w]$ then $C[w] \not\subset \ < R(G) \setminus \{g_i\} >$. So,

$$C[w] \cap (C \setminus \ < R(G) \setminus \{g_i\} >) \neq \emptyset.$$

In other words, a codeword c of weight at most w exists such that $\alpha_i \neq 0$ in $c = \sum_{i=1}^{k} \alpha_i g_i$. So $s(G)_i \leq w$.

$\square$

Lemma 2.4.6 *The generator matrix G of a k-dimensional code in $V_n(q)$ is optimal if and only if $< C[w] > \ = \ < R(G)[w] >$ for each $0 \leq w \leq n$.*

Proof:
$\Rightarrow$: By the definition of $< R(G)[w] >$ one trivially has $< C[w] > \ \subset \ < R(G)[w] >$. Let $< C[w] >$ have rank l and let G' be any generator matrix of C, whose last l rows generate $< C[w] >$. It follows that $s(G')_i > w$, $1 \leq i \leq k - l$. By the optimality of G we may conclude that $s(G)_i > w$, $1 \leq i \leq k - l$, and thus that $< C[w] > \ \supset \ < g_{k-l+1}, \cdots, g_k >$.

It follows that $<C[w]>=<R(G)[w]>$.

$\Leftarrow$: Consider any other generator matrix G' of C and let i be minimal such that $s(G')_i > s(G)_i$ (if no such i exists, there is nothing to prove). Put $w = s(G')_i - 1$. From $s(G')_1 \geq \ldots \geq s(G')_i > w$, it follows that $C[w] \subset <\underline{g}'_{i+1}, \ldots, \underline{g}'_k>$ and thus that $<C[w]> \subset <\underline{g}'_{i+1}, \ldots, \underline{g}'_k>$.

On the other hand, from $s(G)_k \leq \ldots \leq s(G)_i \leq s(G')_i - 1 = w$ and Lemma 2.4.5 it follows that $\underline{g}_i, \ldots, \underline{g}_k \in R(G)[w]$. Combining these results we get the following contradiction:

$\underline{g}_i, \ldots, \underline{g}_k \subset <R(G)[w]> =$
$<C[w]> \subset <\underline{g}'_{i+1}, \ldots, \underline{g}'_k>$.

$\square$

The proof of Lemma 2.4.6 also tells us how to find an optimal generator matrix of a linear code. Consider with the set of lowest weight codewords. Take any basis of the linear span of these vectors and fill the bottom part of the generator matrix with these basis vectors. Now take the set of lowest weight codewords, that are not in the span of the rows of G yet, and look at the linear span of them and the current rows of G. Extend the previous basis to a basis of this new linear span. And so on.

This algorithm proves the the following theorem.

Theorem 2.4.7 *Any linear code has an optimal generator matrix.*

If the lowest weight codewords in a linear code generate the whole code, one obviously does not have a LUEP code.

Example 2.4.8 Consider the binary $[8, 3, 3]$ code generated by the vectors (01010111), (00101111) and (11100000). The lowest weight codewords have weight 3. Their linear span has dimension 2; it is generated by the lower two rows in the matrix G^{opt} below. The lowest weight not occurring in the linear span of these rows is 5. The weight five vectors together with the rows in G^{opt} that are already filled in generate a 3-dimensional space (the whole code). So the previous basis can be extended with one row to generate this new linear span (the whole code). This gives the following optimal generator matrix

$$G^{opt} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix},$$

with separation vector $(5, 3, 3)$.

In the literature the interested reader can find extensive tables of optimal LUEP codes.

2.5 Problems

2.5.1 Consider the vector space $\{0, 1\}^6$ with Hamming distance. What is the volume of a sphere with radius 1?

Is it possible to find 9 codewords at mutual distance at least 3?

2.5.2 Consider the BSC with error probability p. Four messages are encoded with words in $\{0,1\}^6$. Maximize the minimum distance of such a code. Show that such a code is unique and equivalent to a linear code. Give the weights of all the coset leaders of this linear code. What is the probability that a transmitted codeword from this code will be correctly decoded by a maximum likelihood decoding algorithm.

2.5.3 Construct a $(4, 9, 3)$ code over the alphabet $\{0, 1, 2\}$.

2.5.4 Consider a binary channel, that has a probability of $p = 0.9$ that a transmitted symbol is received correctly and a probability of $q = 0.1$ of producing an erasure (so a $*$ is received). Construct a length-5 code of maximum cardinality that decodes any received codeword with at most one erasure correctly.

What is the probability that a codeword transmitted over this channel is decoded correctly?

2.5.5 Suppose that all the rows in the generator matrix G of a binary linear code C have even weight. Prove that all codewords in C have even weight.

2.5.6 Let C be the binary $[9, 5]$ code with parity check matrix

$$
H = \begin{pmatrix}
0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1
\end{pmatrix}.
$$

Find the coset leader(s) of the cosets containing the following words:

a)   1 1 1 1 0 1 0 0 0
b)   1 1 0 1 0 1 0 1 1
c)   0 1 0 0 1 0 0 1 0

2.5.7 Let the generator matrix G of a binary, selfdual code C have the property that each row has a weight that is a multiple of 4.

Prove that $w(\underline{c}) \equiv 0 \pmod 4$ for every codeword $\underline{c}$ in C.

2.5.8 Let C be a binary, selfdual $[24, 12, 8]$ code in which all words have a weight divisible by 4. Prove that the all-one vector is in the code C.

Determine the weight enumerator of C.

2.5.9 Let H be the parity check matrix of a q-ary $[n, k, d]$ code C with covering radius ρ.

Prove that d is determined by the two properties: 1) each $(d-1)$-tuple of columns of H is linearly independent and 2) H contains at least one d-tuple of linearly dependent columns.

Prove that ρ is determined by the two properties: 1) each vector in $V_{n-k}(q)$ can be written as a linear combination of some ρ-tuple of columns of H and 2) there exists at least one vector in $V_{n-k}(q)$ that cannot be written as linear combination of any $\rho - 1$ columns of H.

2.5.10 Let C be an $(n, q^k, d = n - k + 1)$ code over $GF(q)$, i.e. C is a MDS code. Prove that C is systematic on every k-tuple of coordinates.

How many codewords of weight d does C have (as a function of n, k and q), if the all-zero word is in C.

2.5.11 Show what kind of results the Gilbert, Singleton, and Hamming Bound give when applied to $q = 2$, $n = 16$ and $d = 3, 5, 7, 9$.

Put all the results in the following table to compare them.

$n = 16$	$d = 3$	$d = 5$	$d = 7$	$d = 9$
Gilbert				
Gilbert for linear codes				
Singleton				
Hamming				

2.5.12 Let H be a binary $m \times n$ matrix, constructed by taking each odd weight vector of length m exactly once as column.

Give an example of such a matrix for $m = 5$.

Determine n as function of m.

Let C be the binary, linear code with H as parity check matrix. Determine words at distance 0, 1 and 2 from C for your example.

Determine for $m \geq 3$ the dimension, the minimum distance and the covering radius of C.

How many cosets of C have a unique coset leader and what is the weight of this coset leader?

The other cosets have more coset leaders. How many and what is their weight?

2.5.13 Let the parity check matrix H have as columns all vectors of length 5 (each exactly once) that do not start with two zeros. So

$$H = \begin{pmatrix} 00000000111111111111111 \\ 11111111000000011111111 \\ 00001111000011110000111 \\ 00110011001100110011 \\ 01010101010101010101 \end{pmatrix}.$$

Let C be the code with H as parity check matrix. Decode the following three words

$$00001010000101000010$$
$$10001000100010000001$$
$$01010010011000001010000.$$

How many cosets of C have a unique coset leader and what is the weight of this coset leader?

The other cosets have more coset leaders. How many and what is their weight?

This code is used over a BSC with error probability p. What is the probability of correctly decoding a received word?

What is the covering radius ρ of C.

Let $\underline{x}$ have distance 1 to C. To how many codewords does $\underline{x}$ have distance 2?

Let $\underline{x}$ have distance 2 to C. To how many codewords does $\underline{x}$ have distance 2?

2.5.14 Let C be a binary, linear, perfect, 1 error-correcting code of length $n = 2^m - 1$, $m \geq 2$. Define

$$C^{sh} := \{(c_2, c_3, \ldots, c_n) \mid (0, c_2, \ldots, c_n) \in C\}.$$

Determine the dimension and the parameters e, d and ρ of C^{sh}.

For each $\underline{x} \in V_{n-1}(2)$ define

$$B(\underline{x}, i) := \mid \{\underline{c} \in C^{sh} \mid d(\underline{x}, \underline{c}) = i\} \mid. \tag{1}$$

Derive an upper bound on $B(\underline{x}, 2)$, if $d(\underline{x}, C^{sh}) = 1$. (hint: translate the whole vector space over $\underline{x}$ to get $\underline{x}$ in the origin; write down the words at distance 2 from $\underline{x}$.)

Derive an upper bound on $B(\underline{x}, 2)$, if $d(\underline{x}, C^{sh}) = 2$.

Derive an upper bound on

$$\sum_{\underline{x}, d(\underline{x}, C^{sh}) \geq 1} (B(\underline{x}, 1) + B(\underline{x}, 2)). \tag{2}$$

Compute this sum exactly by substituting (1) into (2), followed by changing the order of summation.

Compare the two answers. What is your conclusion for the upper bounds on $B(\underline{x}, 2)$?

2.5.15 Let C be the binary code generated by

$$
G = \begin{pmatrix}
000000011111111 & 1001 \\
000111100001111 & 0101 \\
011001100110011 & 0011 \\
101010101010101 & 0000
\end{pmatrix}.
$$

Prove that C is a LUEP code. Determine its separation vector.

Give an optimal generator matrix of C.

Chapter 3

Some code constructions

3.1 Making codes from other codes

In this section, a number of techniques to construct other codes from a given code will be discussed.

Definition 3.1.1 *Let C be a q-ary (n, M, d) code. The* extended *code C^{ext} of C is defined by*

$$C^{ext} = \{(c_1, c_2, \ldots, c_n, -\sum_{i=1}^{n} c_i) \mid \underline{c} \text{ in } C\} \tag{3.1}$$

So, the words in C^{ext} are obtained from those of C by adding an overall parity check symbol.

Clearly, in the binary case, all words in C^{ext} will have an even weight. This shows that if C is a binary $(n, M, 2e + 1)$ code, C^{ext} will be a binary $(n + 1, M, 2e + 2)$ code.

If G and H are the generator matrix, resp. parity check matrix of an $[n, k, d]$ code C, the generator matrix G^{ext} and parity check matrix H^{ext} of the extended code C^{ext} are given by:

$$G^{ext} = \left(\begin{array}{c|c} G & \begin{array}{c} -\sum_{j=1}^{n} g_{1j} \\ -\sum_{j=1}^{n} g_{2j} \\ \vdots \\ -\sum_{j=1}^{n} g_{kj} \end{array} \end{array} \right). \tag{3.2}$$

and

$$H^{ext} = \left(\begin{array}{ccccc|c} 1 & 1 & \cdots & 1 & 1 \\ \hline & & & & 0 \\ & & H & & \vdots \\ & & & & 0 \end{array} \right). \tag{3.3}$$

Two ways of making a code one symbol shorter are given in the following definition:

Definition 3.1.2 *By* puncturing *a code on a specific coordinate, one simply deletes that coordinate in all codewords.*

By shortening *a code on a specific coordinate, say the j-th, with respect to symbol α, one takes only those codewords that have an α on coordinate j and then deletes that j-th coordinate.*

Let C be a q-ary (n, M, d) code. Then on each coordinate at least one symbol will occur at least $\lceil M/q \rceil$ times. This proves the only non-trivial part in the next theorem.

Theorem 3.1.3 *Let C be a q-ary (n, M, d) code with $d \geq 2$. Then by puncturing a coordinate one obtains an $(n - 1, M, \geq d - 1)$ code.*

If C is a q-ary (n, M, d) code, one can shorten on any coordinate with respect to the most frequently occurring symbol on that coordinate to obtain an $(n - 1, \geq \lceil M/q \rceil, \geq d)$ code.

The next technique of constructing a code from another one, will be restricted to binary linear codes.

Let, without loss of generality, the top row, called $\underline{c}$, of the generator matrix G of a binary $[n, k, d]$ code C, have weight d and let all the ones of $\underline{c}$ be permuted to the front, i.e.

$$
G = \left(\begin{array}{cccc|cccc}
1 & 1 & \cdots & 1 & 0 & 0 & \cdots\cdots & 0 \\
\hline
& & G^1 & & & & G^2 &
\end{array} \right) \quad \begin{array}{l} \underline{c} \text{ of weight } d \end{array} \quad .
$$

Then the code generated by the restriction of the rows of G to the last $n - d$ coordinates, i.e. the code generated by G^2, is called the *residual code* C^{res} of C with respect to $\underline{c}$.

Theorem 3.1.4 *Let C be a binary $[n, k, d]$ code and let $\underline{c}$ be a codeword of weight d in C. Then the residual code C^{res} of C with respect to $\underline{c}$ has parameters $[n - d, k - 1, \geq \lceil d/2 \rceil]$.*

Proof: Consider any non-trivial linear combination of the last $k - 1$ rows of G and write it as $(\underline{a}^1, \underline{a}^2)$ corresponding to G^1 and G^2. Clearly

$$
w(\underline{a}^1) + w(\underline{a}^2) \geq d,
$$

but also, because of the distance to the top row,

$$
(d - w(\underline{a}^1)) + w(\underline{a}^2) \geq d.
$$

Adding these two equations gives $w(\underline{a}^2) \geq \lceil d/2 \rceil$. So any non-trivial linear combination of the rows of G^2 has weight at least $\lceil d/2 \rceil$. It follows that G^2 has indeed rank $k - 1$ and that the minimum distance of the code generated by G^2, i.e. C^{res}, is at least $\lceil d/2 \rceil$.

$\square$

As an example, consider the question of the existence of a binary $[13, 6, 5]$ code. If it existed, the residual code with respect to any minimum weight codeword would have parameters $[8, 5, \geq 3]$. This however is impossible by the Hamming bound (Theorem 2.1.3). So, no binary $[13, 6, 5]$ code exists. It is however possible to construct a non-linear $(13, 2^6, 5)$ code, but that construction will not be given here.

The above technique can easily be generalized to other fields than $GF(2)$. One can also define the residual code of C with respect to a codeword of weight more than d. Both generalizations are left as an exercise to the reader.

The following corollary is a direct consequence of Theorem 3.1.4 and the fact that $n' \geq d'$ in a $[n', 1, d']$ code.

Corollary 3.1.5 (The Griesmer bound) *Let C be an $[n, k, d]$ binary linear code. Then*

$$n \geq \sum_{i=0}^{k-1} \lceil d/2^i \rceil. \tag{3.4}$$

The next construction will be needed in Section 3.3.

Theorem 3.1.6 ($(\underline{u}, \underline{u}+\underline{v})$ construction) *Let C_1 be a binary (n, M_1, d_1) code and C_2 a binary (n, M_2, d_2) code. Then the code C defined by*

$$C = \{(\underline{u}, \underline{u} + \underline{v}) \mid \underline{u} \text{ in } C_1, \; \underline{v} \text{ in } C_2\} \tag{3.5}$$

has parameters $(2n, M_1 M_2, d)$ with $d = \min\{2d_1, d_2\}$.

If C_1 and C_2 are both linear, then so is C.

Proof: The length and cardinality of C are obvious. So the only thing left to check is the minimum distance of C.

Let $(\underline{u}_1, \underline{u}_1 + \underline{v}_1)$ and $(\underline{u}_2, \underline{u}_2 + \underline{v}_2)$ be two distinct words in C. If $\underline{v}_1 = \underline{v}_2$, the distance between these two codewords is obviously equal to twice the distance between $\underline{u}_1$ and $\underline{u}_2$, so the distance is at least $2d_1$.

If $\underline{v}_1 \neq \underline{v}_2$, then on each of the at least d_2 coordinates where $\underline{v}_1$ and $\underline{v}_2$ differ, either also $\underline{u}_1$ and $\underline{u}_2$ differ or $\underline{u}_1 + \underline{v}_1$ and $\underline{u}_2 + \underline{v}_2$ differ. So in this case $(\underline{u}_1, \underline{u}_1 + \underline{v}_1)$ and $(\underline{u}_2, \underline{u}_2 + \underline{v}_2)$ have distance at least d_2.

The proof of the linearity of C in case that both C_1 and C_2 are linear is straightforward.

$\square$

The next construction also makes use of two codes, but now over different fields.

Let C_1 be a q-ary (n_1, M_1, d_1) code and let C_2 be a M_1-ary (n_2, M_2, d_2) code. Note that the alphabet size of code C_2 is equal to M_1, the cardinality of C_1. So, one can replace each element in this alphabet of size M_1 by a unique codeword in C_1. Replacing each coordinate in the codewords of C_2 by the corresponding vector in C_1 results in a q-ary

code that is called the *concatenated code* of the *inner code C_1* and the *outer code C_2*. The following theorem is now obvious.

Theorem 3.1.7 (Concatenated code) *Let C_1 be a q-ary (n_1, M_1, d_1) code and C_2 be a M_1-ary (n_2, M_2, d_2) code. Then the concatenated code of the inner code C_1 and outer code C_2 is a q-ary code with parameters $(n_1 n_2, M_2, d_1 d_2)$.*

3.2 The Hamming codes

Equation (2.14) can be interpreted as follows: a word $\underline{c}$ is a codeword in a linear code C if and only if (abbreviated in the sequel to "iff") the coordinates of $\underline{c}$ give a dependency relation between the the columns of the parity check matrix H of C.

In particular the minimum distance d of a linear code will satisfy $d \geq 2$ iff H does not contain the all-zero column. Similarly, $d \geq 3$ iff H does not contain two columns that are linearly dependent (the dual of such a code is sometimes called a *projective code*). In general, C will have minimum distance $\geq d$ iff each $d - 1$-tuple of columns in H is linearly independent. If at least one d-tuple of columns is linearly dependent, the minimum distance will be exactly equal to d.

In view of the above, we now know that the length of a q-ary $[n, k, 3]$ code is upper bounded by the maximum number of pairwise linearly independent vectors in $V_r(q)$, where $r = n - k$ is the redundancy of C. Now $V_r(q)$ has $q^r - 1$ non-zero vectors. They can be divided into $(q^r - 1)/(q - 1)$ groups of size $q - 1$, each consisting of a non-zero vector in $V_r(q)$ together with all its non-zero scalar multiples. The extreme case that $n = (q^r - 1)/(q - 1)$ leads to the following definition.

Definition 3.2.1 (Hamming code) *The q-ary Hamming code of length $n = (q^r - 1)/(q - 1)$ and redundancy r is defined by the parity check matrix that consists of (a maximum set of) columns that are all pairwise linearly independent.*

It is denoted by $\mathcal{H}_r(q)$. The minimum distance of $\mathcal{H}_r(q)$ is equal to 3.

The reason for writing "the" Hamming code instead of "a" Hamming code simply is that all q-ary Hamming codes of the same length are equivalent to each other.

Example 3.2.2 The binary $[7, 4, 3]$ Hamming code $\mathcal{H}_3(2)$ has parity check matrix

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

The columns are the binary representation of the numbers $1, 2, \ldots, 7$.

Example 3.2.3 The ternary $[13, 10, 3]$ Hamming code $\mathcal{H}_3(3)$ has parity check matrix

$$H = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \end{pmatrix}.$$

The examples above give a systematic way of making the parity check matrix of a Hamming code: Take only the columns, whose first non-zero entry, when reading from the top to the bottom, is a 1.

Theorem 3.2.4 *The q-ary $[n = (q^r - 1)/(q-1), k = n - r, 3]$ Hamming code is perfect.*

Proof: The volume of a sphere of radius 1 around a codeword is $1 + n(q-1) = q^r = q^{n-k}$. From

$$|C|q^{n-k} = q^k q^{n-k} = q^n$$

we have that equality holds in the Hamming bound (Theorem 2.1.3). So the covering radius of the Hamming code is also 1. The statement now follows from Definition 2.1.6.

$\square$

Decoding the Hamming code is extremely simple. Let $\underline{r}$ be a received word. Compute its syndrome $\underline{s} = H\underline{r}^T$. Definition 3.2.1 implies that $\underline{s}$ is the scalar multiple of some column of the parity check matrix, say α times the j-th column. Now simply subtract α from the j-th coordinate to find the closest codeword. Indeed, the vector $(r_1, \ldots, r_{j-1}, r_j - \alpha, r_{j+1}, \ldots, r_n)$ has syndrome $\underline{0}$.

The fact that a code is perfect implies that its weight enumerator is uniquely determined by its parameters. We shall demonstrate this for the binary Hamming code.

Consider one of the $\binom{n}{w}$ words of weight w in V_n, say $\underline{x}$. It is either a codeword in the $[n = 2^r - 1, n - r, 3]$ Hamming code, or it lies at distance 1 from a unique codeword, say from $\underline{c}$. In the latter case, $\underline{c}$ has either weight $w + 1$ and $\underline{x}$ can be obtained from $\underline{c}$ by replacing one of its $w + 1$ one-coordinates into a 0, or $\underline{c}$ has weight $w - 1$ and $\underline{x}$ can be obtained from $\underline{c}$ by replacing one of its $n - (w - 1)$ zero-coordinates into a 1. Since each of these coordinate changes yield different words of weight w (otherwise $d = 3$ cannot hold), we have proved that the weight enumerator $A(z)$ of the binary Hamming code of length $n = 2^r - 1$ satisfies the recurrence relation

$$\binom{n}{w} = A_w + (w+1)A_{w+1} + (n - w + 1)A_{w-1}, \quad 0 \leq w \leq n. \tag{3.6}$$

With $A_0 = 1$ and $A_1 = A_2 = 0$, one can now easily determine the whole weight enumerator recursively. For instance $w = 2$ yields $\binom{n}{2} = 3A_3$, i.e. $A_3 = n(n-1)/6$.

With the standard technique of solving recurrence relations one can find a closed expression for $A(z)$. It is however much easier to derive this with the MacWilliams relations.

The dual code of a Hamming code is called the *Simplex code*. In Examples 3.2.2 and 3.2.3 one can see that all rows of the parity check matrices (thus of the generator matrices of the corresponding Simplex codes) have weight q^{r-1}. This turns out to hold for all non-zero codewords in the Simplex code.

Theorem 3.2.5 *All non-zero codewords in the Simplex code of length $(q^r - 1)/(q - 1)$ have weight q^{r-1}, so the Simplex code has weight enumerator*

$$A(z) = 1 + (q^r - 1)z^{q^{r-1}}. \tag{3.7}$$

Proof: Suppose that the Simplex code has a codeword of weight w with $w > q^{r-1}$, say $\underline{c}$. Without loss of generality we may assume that the first w coordinates of $\underline{c}$ are non-zero and that they all are equal to 1 (otherwise consider an equivalent Simplex code, which will have the same structure).

Use $\underline{c}$ as top row of a new generator matrix of this code. The first w $(> q^{r-1})$ columns of this generator matrix all start with a 1 and are then followed by $r - 1$ other coordinates. However there are only q^{r-1} different q-ary $(r - 1)$-tuples, so at least two of the first $w > q^{r-1}$ columns are identical to each other. Since this generator matrix is also the parity check matrix of the Hamming code, we have a contradiction with Definition 3.2.1.

If the Simplex code of length $(q^r - 1)/(q - 1)$ has a codeword of weight w with $w < q^{r-1}$, a similar contradiction can be obtained by considering the columns where this codeword has its zero-coordinates.

$\square$

The next theorem now immediately follows from the MacWilliams relations (Theorem 2.3.4).

Theorem 3.2.6 *The weight enumerator of the q-ary Hamming code of length $n = (q^r - 1)/(q - 1)$ is given by $A(z) =$*

$$\tfrac{1}{q^r} \left\{ (1 + (q - 1)z)^n + (q^r - 1)(1 - z)^{q^{r-1}}(1 + (q - 1)z)^{n - q^{r-1}} \right\}. \tag{3.8}$$

3.3 Reed-Muller codes

The final class of linear codes in this chapter dates back to a paper by D.E. Muller in 1954. In the same year I.S. Reed proposed a decoding algorithm for this code. Although this class of codes can be generalized to other fields, we shall only discuss the binary case.

In this section we consider binary polynomials $f = f(x_1, x_2, \ldots, x_m)$ in m variables, where m is some fixed integer. Since only binary values will be substituted in f and $x^2 = x$ for $x = 0$ and 1, each variable will only occur to the power at most 1. Of course terms like $x_2 x_4 x_5$ can occur. If f contains a term which is the product of r variables but no term which is a product of $\geq r + 1$ variables, f is said to have *degree r*. Clearly $0 \leq r \leq m$.

Any polynomial f of degree $\leq r$ can be written as follows:

$$f(x_1, x_2, \ldots, x_m) = \sum_{l=0}^{r} \sum_{1 \leq i_1 < i_2 < \cdots < i_l \leq m} a_{i_1 i_2 \cdots i_l} x_{i_1} x_{i_2} \cdots x_{i_l}, \tag{3.9}$$

where $a_\emptyset$ denotes the constant term.

For example, $1 + x_1 + x_3 + x_1 x_2 + x_2 x_3 x_4$ is a polynomial of degree 3.

Now, for the m variables $x_1, x_2, \ldots, x_m$ one can substitute all the $n = 2^m$ points of V_m to obtain all function values. For this purpose, we shall choose the lexicographical ordering of the n points of V_m, where the first coordinate is the most significant. The successive points of V_m are named $\underline{u}_0, \underline{u}_1, \ldots, \underline{u}_{n-1}$.

For example, when $m = 4$, one gets in this way:

$\underline{u}_0$	$\underline{u}_1$	$\underline{u}_2$	$\underline{u}_3$	$\underline{u}_4$	$\underline{u}_5$	$\underline{u}_6$	$\underline{u}_7$	$\underline{u}_8$	$\underline{u}_9$	$\underline{u}_{10}$	$\underline{u}_{11}$	$\underline{u}_{12}$	$\underline{u}_{13}$	$\underline{u}_{14}$	$\underline{u}_{15}$
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Definition 3.3.1 *The r-th order* Reed-Muller code $\mathcal{RM}(r, m)$ *of length* $n = 2^m$ *is defined by*

$$\mathcal{RM}(r, m) = \{(f(\underline{u}_0), f(\underline{u}_1), \ldots, f(\underline{u}_{n-1})) \mid degree(f) \leq r\}. \tag{3.10}$$

The vector $\underline{c} = (f(\underline{u}_0), f(\underline{u}_1), \ldots, f(\underline{u}_{n-1}))$ is often called the *characteristic vector* of f.

Clearly $\mathcal{RM}(0, m) \subset \mathcal{RM}(1, m) \subset \cdots \subset \mathcal{RM}(m, m)$ and all these codes are linear. Also it follows from the definition that $\mathcal{RM}(0, m)$ is the repetition code of length $n = 2^m$.

Example 3.3.2 The polynomial $f(x_1, x_2, x_3, x_4) = 1 + x_1 + x_3 + x_1 x_2 + x_2 x_3 x_4$ gives rise to the following codeword in $\mathcal{RM}(3, 4)$:

$$1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1 \ 0 \ 1$$

Consider a vector $\underline{b}$ in V_m. The polynomial $(x_1 - b_1 + 1)(x_2 - b_2 + 1) \cdots (x_m - b_m + 1)$ has degree m, is equal to 1 for $\underline{x} = \underline{b}$ and is zero otherwise. It follows that each weight-one vector in V_n is an element of $\mathcal{RM}(m, m)$. By the linearity of the Reed-Muller codes, we may conclude that each vector in V_n lies in some Reed-Muller code. However there are exactly 2^n vectors in V_n, just as there are $2^{1 + \binom{m}{1} + \binom{m}{2} + \cdots + \binom{m}{m}} = 2^{2^m} = 2^n$ polynomials of the form (3.9). It follows that each polynomial $f(x_1, x_2, \ldots, x_m)$ gives rise to a unique word of length $n = 2^m$ and the other way around.

Now that we have a 1-1 correspondence between vectors of length $n = 2^m$ and polynomials $f(x_1, x_2, \ldots, x_m)$, we shall identify these two and say that f is in $\mathcal{RM}(r, m)$,

when f has degree less than or equal to r. Similarly we shall say that two polynomials are orthogonal to each other, when we really mean that their characteristic vectors are orthogonal to each other.

The reasoning above also proves that the dimension of $\mathcal{RM}(r, m)$ is equal to the number of choices of the coefficients in (3.9), i.e. $1 + \binom{m}{1} + \cdots + \binom{m}{r}$.

Theorem 3.3.3 *The r-th Reed-Muller code $\mathcal{RM}(r, m)$ of length $n = 2^m$ has parameters* $[n, \sum_{l=0}^{r} \binom{m}{l}, 2^{m-r}]$.

Proof: It remains to show that $\mathcal{RM}(r, m)$ has minimum distance 2^{m-r}. That it cannot be more follows from the fact that the polynomial $x_1 x_2 \cdots x_r$ is in $\mathcal{RM}(r, m)$ and that its function value is 1 iff $x_1 = x_2 = \ldots = x_r = 1$ (independent of values of $x_{r+1}, \ldots, x_m$), so its weight is 2^{m-r}.

That the minimum distance is at least 2^{m-r} will follow with an induction argument from Theorem 3.1.6. For $m = 1$, the statement is trivial.

By splitting the terms in a polynomial $f(x_1, x_2, \ldots, x_m)$ in $\mathcal{RM}(r, m)$ into those that do contain x_1 and those that do not, one can write f as $p(x_2, x_3, \ldots, x_m) + x_1 q(x_2, x_3, \ldots, x_m)$, with p in $\mathcal{RM}(r, m - 1)$ and q in $\mathcal{RM}(r - 1, m - 1)$. Also each choice of p in $\mathcal{RM}(r, m - 1)$ and q in $\mathcal{RM}(r - 1, m - 1)$ gives rise to a unique f in $\mathcal{RM}(r, m)$.

Now p corresponds to a codeword $\underline{u}$ in $\mathcal{RM}(r, m-1)$, which by the induction hypothesis has minimum distance $d_1 = 2^{m-r-1}$. Similarly, q corresponds to a codeword $\underline{v}$ in $\mathcal{RM}(r - 1, m - 1)$, with minimum distance $d_1 = 2^{m-r}$. Moreover $p + x_1 q$ has characteristic vector $(\underline{u}, \underline{u} + \underline{v})$.

It follows from the above that $\mathcal{RM}(r, m)$ is the result of the $(\underline{u}, \underline{u} + \underline{v})$ construction in Theorem 3.1.6 applied to $C_1 = \mathcal{RM}(r, m - 1)$ and $C_2 = \mathcal{RM}(r - 1, m - 1)$. The same theorem now says that $\mathcal{RM}(r, m)$ has minimum distance greater than or equal to $\min\{2d_1, d_2\}$
$= \min\{2 \cdot 2^{m-r-1}, 2^{m-r}\} = 2^{m-r}$.

$\square$

In Table 3.1 one can find the characteristic vectors of all possible terms in a polynomial in x_1, x_2, x_3 and x_4. They are listed in order of non-decreasing degree. As a result the top row generates $\mathcal{RM}(0, 4)$, the top $1 + \binom{4}{1}$ rows generate $\mathcal{RM}(1, 4)$, etc. Note that row $x_1 x_2$ is just the coordinate-wise product of rows x_1 and x_2. The same holds for the other higher degree terms. This implies in particular, that the innerproduct of two monomials is just the weight of their product. Since $f(\underline{b}) = g(\underline{b}) = 1$ if and only if $(fg)(\underline{b}) = 1$, the same is true for all polynomials: the innerproduct of two polynomials f and g is equal to the weight of their product fg.

The generator matrix G of $\mathcal{RM}(1, m)$ has as rows the characteristic vectors of

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
x_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
x_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
x_3	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
x_4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
x_1x_2	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
x_1x_3	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1
x_1x_4	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1
x_2x_3	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1
x_2x_4	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1
x_3x_4	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1
$x_1x_2x_3$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
$x_1x_2x_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
$x_1x_3x_4$	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
$x_2x_3x_4$	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
$x_1x_2x_3x_4$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Table 3.1: The monomials in x_1, x_2, x_3, x_4 generate V_{16}.

$1, x_1, x_2, \ldots, x_m$. So the first row is $\underline{1}$. The columns of G are, apart from an initial 1, the vectors $\underline{u}_0, \underline{u}_1, \ldots, \underline{u}_{n-1}$. If one leaves out the first row and column in G one gets the generator matrix of the Simplex code. Equivalently, $\mathcal{RM}(1, m)$ is the dual code of the extended Hamming code.

Since each monomial of degree less than m has even weight, and since they span $\mathcal{RM}(m-1, m)$, it follows that all codewords in $\mathcal{RM}(m-1, m)$ have even weight. The dimension of $\mathcal{RM}(m-1, m)$ is equal to $\sum_{l=0}^{m-1} \binom{m}{l} = 2^m - 1 = n - 1$. We conclude that $\mathcal{RM}(m-1, m)$ is the even weight code and thus that $\mathcal{RM}(m-1, m)$ is the dual of $\mathcal{RM}(0, m)$.

It turns out, in general, to be quite easy to determine the dual code of a Reed-Muller code.

Theorem 3.3.4 *The dual code of $\mathcal{RM}(r, m)$ is $\mathcal{RM}(m - r - 1, m)$.*

Proof: Since $n - \sum_{l=0}^{r} \binom{m}{l} = \sum_{l=0}^{m} \binom{m}{l} - \sum_{l=0}^{r} \binom{m}{l} = \sum_{l=r+1}^{m} \binom{m}{l} = \sum_{l=0}^{m-r-1} \binom{m}{l}$, we have that $\mathcal{RM}(m - r - 1, m)$ has the same cardinality as the dual code of $\mathcal{RM}(r, m)$.

The statement now follows from the fact that each element f in $\mathcal{RM}(m - r - 1, m)$ is orthogonal to each element g in $\mathcal{RM}(r, m)$. Indeed, f has degree at most r and g each vector has degree at most $m - r - 1$, so their product is in $\mathcal{RM}(m - 1, m)$. But $\mathcal{RM}(m - 1, m)$ is the even weight code, so fg has even weight. This implies that f and g are orthogonal to each other.

$\square$

Since we already know that $\mathcal{RM}(1, m)$ is the dual code of the extended Hamming code,

we now also know that $\mathcal{RM}(m-2, m)$ is equal to the extended Hamming code.

The next property of Reed-Muller codes shows that they have even more internal structure. In Chapter 2, two codes that could be obtained from each other by a coordinate permutation and by alphabet permutations for each coordinate were called equivalent. To preserve linearity, for linear codes we do not allow all alphabet permutations but just those that leave the finite field structure intact. For $q = 2$ this means that we only allow coordinate permutations.

It turns out that some coordinate permutations combined with alphabet/field permutations, when applied to a code, yield the very same code. This means that the set of codewords is mapped in a 1-1 way to the set of codewords. Such permutations are called *automorphisms* of a code. The automorphisms of a code C form a group denoted by $Aut(C)$.

To describe a class of automorphisms of the Reed-Muller code of length $n = 2^m$ it will prove to be useful to index the coordinates of the codewords $\underline{c} = (c_0, c_1, \ldots, c_{n-1})$ with the vectors $\underline{u}_0, \underline{u}_1, \ldots, \underline{u}_{n-1}$ in V_m. An invertible, affine transformation of V_m i.e. a mapping of the form $\underline{x} \to \underline{x}A + \underline{b}$, where A is a non-singular $m \times m$ matrix and $\underline{b}$ is in V_m, obviously is a 1-1 mapping of V_m to V_m. So, an invertible, affine transformation of V_m applied to the coordinates of the codewords of a Reed-Muller code yields an equivalent code.

For instance, when $m = 4$ ($n = 2^4 = 16$) the coordinate transformation applied to $\underline{c} = (c_0, c_1, \ldots, c_{15})$ given by

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \to \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

maps $(0, 0, 0, 0)$ to $(1, 0, 0, 1)$ so c_0 will appear at coordinate 9. Also $(1, 0, 0, 0)$ will be mapped to $(0, 1, 0, 0)$ and thus c_8 will appear at coordinate 4. Etc.

The nice thing is that applying these invertible, affine transformations to a Reed-Muller code will not just result in an equivalent code, but in the original code itself!

Theorem 3.3.5 *The automorphism group $Aut(\mathcal{RM}(r, m))$ of $\mathcal{RM}(r, m)$ contains the group of invertible, affine transformations of V_m.*

Proof: Applying an invertible, affine transformations to a codeword f of degree at most r, amounts to considering the word $f(\underline{x}A + \underline{b})$. Clearly $f(\underline{x}A + \underline{b})$ also has degree at most r, since each variable x_i in f is simply replaced by another linear expression (to be precise by: $a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{im}x_m + b_i$). So $f(\underline{x}A + \underline{b})$ is also a codeword in $\mathcal{RM}(r, m)$. Since $\underline{x} \to \underline{x}A + \underline{b}$ defined a permutation of the coordinates, it follows that it is indeed an automorphism of $\mathcal{RM}(r, m)$.

$\square$

The above theorem is a very important tool in the study of the weight structure of Reed-Muller codes, especially those of order 2. For instance the codeword

$$f(x_1, x_2, x_3, x_4) = x_1x_2 + x_1x_3 + x_1x_4 + x_2x_3 + x_2 + x_3 + 1$$

in $\mathcal{RM}(2,4)$ can be rewritten as

$$(x_1 + x_3 + 1)(x_2 + x_3 + x_4) + x_3x_4 + x_3 + x_4 + 1,$$

so it is equivalent under the invertible, affine transformation

$$
\begin{aligned}
x_1 + x_3 + 1 &\rightarrow x_1 \\
x_2 + x_3 + x_4 &\rightarrow x_2 \\
x_3 &\rightarrow x_3 \\
x_4 &\rightarrow x_4
\end{aligned}
$$

to

$$x_1x_2 + x_3x_4 + x_3 + x_4 + 1.$$

This polynomial in turn can be rewritten as $x_1x_2 + (x_3 + 1)(x_4 + 1)$, so it is equivalent to $x_1x_2 + x_3x_4$. Now, $x_1x_2 + x_3x_4$ is equal to 1 iff one of the terms is 1 and the other 0. It follows that $x_1x_2 + x_3x_4$ and thus f has weight $2 \cdot 1 \cdot 3 = 6$.

We end this section by describing a decoding algorithm for the Reed-Muller code. Since $\mathcal{RM}(r, m)$ has minimum distance 2^{m-r}, we want to correct t errors for all $t < 2^{m-r-1}$.

Consider the term $x_{m-r+1}x_{m-r+2}\cdots x_m$ in a codeword f in $\mathcal{RM}(r, m)$ (see (3.9)). Clearly it has innerproduct 1 with each of the 2^{m-r} words $(x_1 + u_1)(x_2 + u_2)\cdots(x_{m-r} + u_{m-r})$. Indeed, there is only coordinate where both terms are equal to one, namely $(u_1 + 1, u_2 + 1, \ldots, u_{m-r} + 1, 1, 1, \ldots, 1)$. Also, all other terms in f are orthogonal to $(x_1 + u_1)(x_2 + u_2)\cdots(x_{m-r} + u_{m-r})$, because their product (which will miss at least one of the variables $x_i, m - r + 1 \leq i \leq m$) is in $RM(m - 1, m)$ and thus has even weight.

It follows that the coefficient $a_{m-r+1, m-r+2, \cdots, m}$ of $x_{m-r+1}x_{m-r+2}\cdots x_m$ in a codeword f in $\mathcal{RM}(r, m)$ is equal to the innerproduct of f with each of the 2^{m-r} words $(x_1 + u_1)(x_2 + u_2)\cdots(x_{m-r} + u_{m-r})$.

Let $\underline{c}$ be the characteristic vector of f. What we have shown above translates into the 2^{m-r} equations:

$$
\begin{aligned}
a_{m-r+1, m-r+2, \cdots, m} &= c_0 &&+ c_1 &&+ \cdots + c_{2^r - 1}. \\
a_{m-r+1, m-r+2, \cdots, m} &= c_{2^r} &&+ c_{2^r+1} &&+ \cdots + c_{2.2^r - 1}. \\
&\vdots &&\vdots &&\qquad\vdots \\
a_{m-r+1, m-r+2, \cdots, m} &= c_{2^m - 2^r} &&+ c_{2^m - 2^r + 1} &&+ \cdots + c_{2^{m-r}.2^r - 1}.
\end{aligned}
$$

$$(3.11)$$

For instance, the choice of $u_1 = u_2 = \cdots = u_{m-r} = 1$ will give the equation that $a_{m-r+1, m-r+2, \cdots, m}$ is equal to the modulo 2 sum of those coordinates of $\underline{c}$ where $x_1 = x_2 = \cdots = x_{m-r} = 0$ i.e. coordinates $0, 1, \ldots, 2^r - 1$.

So, all the right hand sides in (3.11) will yield the same value, which in fact is equal to $a_{m-r+1,m-r+2,\cdots,m}$.

Now, suppose that $\underline{r} = \underline{c} + \underline{e}$ is a received word, with $\underline{c}$ in $\mathcal{RM}(r,m)$ and suppose that $\underline{e}$ has weight t, i.e. that t errors have been made. Substitution of the coordinates of $\underline{r}$ in the right hand sides of (3.11) will yield 2^{m-r} estimates for $a_{m-r+1,m-r+2,\cdots,m}$ of which at most t are incorrect and at least $2^{m-r} - t$ are correct, since no coordinate occurs more than once. Hence, if $t < 2^{m-r-1}$ the majority of the equations in (3.11) will yield the right value of $a_{m-r+1,m-r+2,\cdots,m}$.

To find a similar set of equations for the other coefficients of the degree-r terms in a codeword, we need to apply the appropriate code automorphism to (3.11). For instance to find $a_{m-r,m-r+2,\cdots,m}$ the transformation $x_{m-r} \leftrightarrow x_{m-r+1}$ will yield the right 2^{m-r} equations.

If the coefficients of all the terms of degree r in f have been found, the first step of the decoding algorithm is done. Indeed the decoding problem is now reduced to the decoding of $\mathcal{RM}(r-1,m)$

The above decoding algorithm is an example of a so-called *multi-step, majority-logic* decoding algorithm. The term "majority-logic" reflects the fact that a majority rule has been applied in the decoding process, just as "multi-step" reflects the fact that this procedure had to be applied several times.

3.4   Problems

3.4.1 Construct a $[8,4,4]$ binary code with the $(\underline{u}, \underline{u} + \underline{v})$-construction.

3.4.2 Generalize Theorem 3.1.4 to residual codes with respect to codewords of weight w, $w \geq d$.

3.4.3 Generalize Theorem 3.1.4 to q-ary codes.

3.4.4 Use Theorem 3.1.4 to prove the non-existence of a binary $[101, 7, 50]$ code.

3.4.5 Let G be a generator matrix of a binary, linear $[n,k,d]$ code C. Let the covering radius ρ of C satisfy $\rho < d$ and let $\underline{x}$ be a word at distance ρ to C. Determine the parameters $[N,K,D]$ of the code C^* generated by

$$\overset{\longleftarrow\quad d-\rho\quad \longrightarrow}{\left(\begin{array}{c|c} 0 & G \\ \hline 1 \;\cdots\cdots\; 1 & \underline{x} \end{array}\right)}.$$

3.4.6 Let C^* be an $[n + s, k + 1, d]$ code with generator matrix G^*. Suppose that some non-zero column $\underline{s}$ in G^* occurs s times with $s < d$.

Construct an $[n, k, d]$ code with covering radius at least $d - s$.

3.4.7 Let $\underline{x}$ be a codeword of weight w in the dual code of a binary $[n, k, d]$ code C. Shorten C with respect to the symbol 0 on all the coordinates where $\underline{x}$ is 1.

Prove that one obtains an $[n - w, \geq k - w + 1, \geq d]$ code.

3.4.8 Give the weight enumerator $A(z)$ of the binary Hamming code of length 31. Compute $\frac{A_i}{\binom{31}{i}/2^5}$ for $i = 5, 10$ and 15.

3.4.9 Derive a recurrence relation for the weight enumerator of a binary $(23, 2^{12}, 7)$ code and a ternary $(11, 3^6, 5)$ code containing the all-zero vector (hint: first show that these codes are perfect).

3.4.10 Use the preceding problem to argue that puncturing a binary evenweight $[24, 12, 8]$ code on each of its coordinates results in $[23, 12, 7]$ codes which all have the same weight enumerators.

Prove that this property implies that in this punctured code the number of codewords of weight $2i - 1$ and that of weight $2i$ are related, $0 \leq i \leq 12$. What is this relation?

Derive the weight enumerator of the $[23, 12, 7]$ code, obtained by puncturing a $[24, 12, 8]$ code with weight enumerator $1 + 759z^8 + 2576z^{12} + 759z^{16} + z^{24}$.

3.4.11 Suppose that a $(90, 2^{78}, 5)$ binary code C exists. Prove that C has to be perfect.

Derive a recurrence relation for the weight enumerator $A(z)$ of C.

Without loss of generality one may assume that C contains $\underline{0}$. Why? Determine $A_0, A_1, \ldots, A_7$. Why should these numbers be integers and what is your conclusion about the existence of C?

3.4.12 Prove that the Boolean function

$$\sum_{1 \leq i < j \leq m} a_{ij} x_i x_j + \sum_{1 \leq i \leq m} b_i x_i + c$$

can be mapped into the the Boolean function

$$x_1 x_2 + \sum_{3 \leq i < j \leq m} a'_{ij} x_i x_j + \sum_{3 \leq i \leq m} b'_i x_i + c' \qquad (*)$$

by an invertible affine transformation $\underline{x} \rightarrow U\underline{x} + \underline{u}$ (i.e. U is invertible) as long as at least one of the a_{ij}'s is not equal to 0.

Which standard forms can be obtained for the elements of $\mathcal{RM}(2, m)$ if one applies $(*)$ repeatedly?

Which weights do occur in $\mathcal{RM}(2,5)$?

Show that $\mathcal{RM}(2,5)$ is a selfdual code. Describe an easy way to find the weight enumerator of $\mathcal{RM}(2,5)$ (the actual calculations do not have to be made).

3.4.13 Let $\underline{c} = a_\emptyset + \sum_{1 \le i \le 5} a_i x_i + \sum_{1 \le i < j \le 5} a_{i,j} x_i x_j$ be a codeword in $\mathcal{RM}(2,5)$ and let $\underline{r} = \underline{c} + \underline{e}$ with $w(\underline{e}) \le 3$.

Write down 8 parity check equations that can be used (by a majority decoding technique) to determine

$$a_{1,3}.$$

Write down 16 parity check equations that can be used to determine

$$a_3$$

in $\underline{c}' = \underline{c} - \sum_{1 \le i < j \le 5} a_{i,j} x_i x_j$.

Write down 32 parity check equations that can be used to determine

$$a_\emptyset$$

in $\underline{c}'' = \underline{c}' - \sum_{1 \le i \le 5} a_i x_i$.

Chapter 4

Cyclic codes and Goppa codes

4.1 Introduction; equivalent descriptions

In Theorem 3.3.5 it was proved that certain coordinate permutations do map the code onto itself. By assuming such internal structure, it may be easier to find good codes. In this chapter we shall study linear codes that are invariant under a cyclic shift.

This cyclic structure makes these codes quite easy to implement with simple logical circuits.

Definition 4.1.1 *A code C is called* cyclic *if it is linear and for each $(c_0, c_1, c_2, \ldots, c_{n-1})$ in C also $(c_{n-1}, c_0, c_1, \ldots, c_{n-2})$ is in C.*

The Hamming code of length seven in Example 3.2.2 is equivalent to the code C with parity check matrix:

$$H = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}. \tag{4.1}$$

One can easily check this by observing that the columns of the paritycheck matrices of both codes are a permutation of each other.

It is also a small effort to check that the non-zero codewords in $C^\perp$ consist of the seven cyclic shifts of the top row in H.

It follows that if some vector is orthogonal to all words in $C^\perp$, so are all its cyclic shifts. So also C is a cyclic code. Indeed it consists of $\underline{0}$, $\underline{1}$ and all the cyclic shifts of the vectors $(\ 1\ \ 1\ \ 0\ \ 1\ \ 0\ \ 0\ \ 0\ )$ and $(\ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0\ )$.

The reader may already have noticed that the coordinates in Definition 4.1.1 are numbered from 0 to $n-1$. The reason is that powerful algebraic tools will be available, once

we have identified the words in $V_n(q)$ with q-ary polynomials over $GF(q)$ in the following way:

$$(c_0, c_1, c_2, \ldots, c_{n-1}) \quad \leftrightarrow \quad c_0 + c_1 x + \ldots + c_{n-1} x^{n-1}. \tag{4.2}$$

So, instead of writing $\underline{c}$ is in C, we shall often write $c(x)$ is in C.

Notice that multiplying $c(x)$ with x almost gives the polynomial corresponding to the cyclic shift of $\underline{c}$. To really get this cyclic shift, one has to reduce $xc(x)$ modulo $x^n - 1$, i.e. replace $xc(x)$ by its remainder after division by $x^n - 1$. Indeed:

$$
\begin{aligned}
xc(x) &\equiv x(c_0 + c_1 x + \ldots + c_{n-1} x^{n-1}) \equiv \\
&\equiv c_0 x + c_1 x^2 + \ldots + c_{n-2} x^{n-1} + c_{n-1} x^n \equiv \\
&\equiv c_{n-1} + c_0 x + c_1 x^2 + \ldots + c_{n-2} x^{n-1} \pmod{x^n - 1}.
\end{aligned}
$$

So, instead of considering the set of all q-ary polynomials in x, usually denoted by $GF(q)[x]$, we only work with the set of the residues of these polynomials modulo $x^n - 1$. This new set is denoted by $GF(q)[x]/(x^n - 1)$, to suggest the division by $x^n - 1$.

We conclude that a cyclic shift in $V_n(q)$ corresponds to a multiplication by x in $GF(q)[x]/(x^n - 1)$. Of course, there is no reason to limit us to multiplications by x. Multiplication with any polynomial is now possible; just divide the product by $x^n - 1$ and take the remainder. The mathematical terminology for this is: $GF(q)[x]/(x^n - 1)$ is a ring.

Since a cyclic code is linear by definition, with $c(x)$ in a cyclic code C, not only $xc(x)$ is in C, but also $x^2 c(x), x^3 c(x)$, etc., and all their linear combinations as well. In short, each multiple $u(x)c(x) \pmod{x^n - 1}$ is in C.

The next theorem shows that a cyclic code in $V_n(q)$ can be described by a single polynomial in $GF(q)[x]/(x^n - 1)$.

Theorem 4.1.2 *Let C be a cyclic code in $V_n(q)$. Then there exists a unique monic polynomial $g(x)$ over $GF(q)$ dividing $x^n - 1$ with the property*

$$c(x) \text{ is in } C \quad \textit{iff} \quad g(x) \text{ divides } c(x). \tag{4.3}$$

The polynomial $g(x)$ is called the generator polynomial *of C.*

Proof: Let $g(x)$ be the monic polynomial of lowest degree among the non-zero elements in C. We have already seen that all polynomial-multiples $u(x)g(x)$ of $g(x)$ with degree($u(x)$) + degree($g(x)$) $< n$ are also in C.

It remains to show that any codeword $c(x)$ in C, and thus also $x^n - 1$ which is 0 modulo $x^n - 1$, is a multiple of this $g(x)$.

Divide $c(x)$ by $g(x)$, so write $c(x) = u(x)g(x) + r(x)$, where degree($r(x)$) < degree($g(x)$). With $g(x)$ in C, also $u(x)g(x)$ is in C. Since $c(x)$ is in C too, the linearity of C implies

that also $r(x)$ is in the code C. But $g(x)$ is the monic polynomial of lowest degree among all the non-zero elements in C. We conclude that $r(x) = 0$ and thus that $c(x) = u(x)g(x)$.

$\square$

In $GF(q)[x]/(x^n - 1)$ there are more polynomials with property (4.3). See, for instance, Problem 4.6.8. Only $g(x)$ has the additional property that it divides $x^n - 1$.

The factorization of $x^7 - 1$ over $GF(2)$ into irreducible factors is given by

$$x^7 - 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1).$$

The cyclic $[7, 4, 3]$ Hamming code, at the beginning of this section has generator polynomial $x^3 + x + 1$.

The cyclic $[7, 6, 2]$ even weight code has generator polynomial $x + 1$.

The cyclic $[7, 1, 7]$ repetition code has generator polynomial $(x^3 + x + 1)(x^3 + x^2 + 1) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$.

Since $x^7 - 1$ has three irreducible factors over $GF(2)$ there are 2^3 distinct, binary cyclic codes of length 7. Some of these are equivalent.

Theorem 4.1.3 *Let C be a k-dimensional, cyclic code in $V_n(q)$ with generator polynomial $g(x)$. Then the degree of $g(x)$ is equal to $n - k$.*

Write $g(x) = g_0 + g_1 x + \ldots + g_{n-k} x^{n-k}$. Then a generator matrix G for C is given by:

$$G = \begin{pmatrix} g_0 & g_1 & \cdots & \cdots & g_{n-k} & 0 & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & \cdots & \cdots & g_{n-k} & 0 & \cdots & 0 \\ 0 & 0 & g_0 & g_1 & \cdots & \cdots & g_{n-k} & \cdots & 0 \\ \vdots & & \ddots & \ddots & & & & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & g_0 & g_1 & \cdots & \cdots & g_{n-k} \end{pmatrix}. \tag{4.4}$$

Proof: Let the degree of $g(x)$ be l. Then the $(n - l) \times n$ matrix G^* with as rows the codewords $x^i g(x)$, $0 \le i < n - l$, has rank $n - l$, because $g(x)$ is monic.

Since each codeword must be divisible by $g(x)$, it can be written as $u(x)g(x)$. But each codeword also has degree less than n, so it can be written as $u(x)g(x)$ with degree $u(x)$ less than $n - l$. In other words, each codeword is a linear combination of the rows of G^*. Since C has dimension k, we conclude that $k = n - l$ and that $G^* = G$.

$\square$

To encode information sequence $a(x) = a_0 + a_1 x + \ldots + a_{k-1} x^{k-1}$ one can of course multiply G by $(a_0, a_1, \ldots, a_{k-1})$, but a faster way is by means of the alternative generator matrix $G' = (P I_{n-k})$, that can easily be obtained from G by elementary row operations. Compute $a(x) x^{n-k}$ and divide the result by $g(x)$. Let $r(x)$ be the remainder (of degree less than $n - k$), so $r(x) \equiv a(x) x^{n-k} \pmod{g(x)}$.) Then

$\underline{c} = (-r_0, -r_1, \cdots, -r_{n-k-1}, a_0, a_1, \cdots, a_{k-1})$ is the encoding of $a(x)$. Indeed, $c(x) = -r(x) + a(x)x^{n-k} \equiv -r(x) + r(x) \equiv 0 \pmod{g(x)}$ and this $c(x)$ obviously is the encoding of $a(x)$ by means of G'.

From Theorem 4.1.2 we know that the generator polynomial $g(x)$ of a cyclic code C in $V_n(q)$ must divide $x^n - 1$. Let $h(x)$ be the quotient, so let $x^n - 1 = g(x)h(x)$. This polynomial $h(x)$ gives an alternative way to describe cyclic codes.

Theorem 4.1.4 *Let C be a cyclic code in $V_n(q)$ with generator polynomial $g(x)$. Let $h(x) = (x^n - 1)/g(x)$. Then, in $GF(q)[x]/(x^n - 1)$,*

$$c(x) \text{ is in } C \quad \text{iff} \quad c(x)h(x) = 0. \tag{4.5}$$

The polynomial $h(x)$ is called the parity check polynomial *of C.*

Proof: $\Rightarrow$ If $c(x)$ is in C it can be written as $a(x)g(x)$. So $c(x)h(x) = a(x)g(x)h(x) = a(x)(x^n - 1) = 0$ in $GF(q)[x]/(x^n - 1)$.

$\Leftarrow$ If $c(x)h(x) = 0$ in $GF(q)[x]/(x^n-1)$ one has that x^n-1 divides $c(x)h(x)$, i.e. $g(x)h(x)$ divides $c(x)h(x)$ and thus that $g(x)$ divides $c(x)$.

$\square$

Since $x^n - 1 \equiv 0 \pmod{x^n - 1}$, comparing the coefficient of x^l, $0 \le l < n$, in both sides of $g(x)h(x) \equiv 0 \pmod{x^n - 1}$ will give the following equation:

$$\sum_{i=0}^{n-1} g_i h_{l-i} = 0,$$

where the indices have to be taken modulo n. In this summation, the terms $0 \le i \le l$ give the coefficient of x^l in $g(x)h(x)$ and the terms $l + 1 \le i < n$ give the coefficient of x^{l+n} in $g(x)h(x)$, which reduces to x^l modulo $x^n - 1$.

It follows (by taking $l = i + j$) that each shift $(g_i, g_{i+1}, \ldots, g_{n-1}, g_0, \ldots, g_{i-1}), 0 \le i < n$, of $g(x)$ is orthogonal to each vector $(h_j, h_{j-1}, \ldots, h_0, h_{n-1} \ldots, h_{j+1}), 0 \le j < n$. Let C have dimension k, so $g(x)$ has degree $n - k$ and $h(x)$ has degree k. Then, the above shows that C has the following parity check matrix:

$$H = \begin{pmatrix} 0 & \cdots & 0 & 0 & h_k & \cdots & \cdots & h_1 & h_0 \\ 0 & \cdots & 0 & h_k & \cdots & \cdots & h_1 & h_0 & 0 \\ \vdots & & \cdot & & & \cdot & \cdot & & \vdots \\ \vdots & \cdot & & & \cdot & \cdot & & & \vdots \\ h_k & \cdots & \cdots & h_1 & h_0 & 0 & \cdots & 0 & 0 \end{pmatrix}. \tag{4.6}$$

The matrix H above also generates a cyclic code. The generator polynomial of this code (which is $C^\perp$) is $\frac{1}{h_0}(h_k + h_{k-1}x + \cdots + h_1 x^{k-1} + h_0 x^k) = \frac{1}{h_0} x^k h(1/x)$.

We conclude that the dual code of a cyclic code in $V_n(q)$, with generator polynomial $g(x)$, is also cyclic with generator polynomial $x^k h(1/x)$, where $x^n - 1 = g(x)h(x)$. This

dual code is equivalent to the cyclic code generated by $h(x)$ (just take all columns in reverse order).

To be able to say something about the minimum distance of cyclic codes, it will be necessary to consider an extension field of $GF(q)$ in which $x^n - 1$ factors completely into linear factors.

Consider for instance $GF(8)$. It can be viewed as $GF(2)[x]/(x^3 + x + 1)$, i.e. the set of binary polynomials in x reduced modulo the irreducible polynomial $x^3 + x + 1$. Equivalently, let α be a zero of $x^3 + x + 1$, so $\alpha^3 = 1 + \alpha$. Then $GF(8)$ has the binary polynomials of degree at most 2 in α as its elements. For the multiplication, one should use the relation $\alpha^3 = 1 + \alpha$ to reduce the degree to less than 3. For example $(1 + \alpha^2)^2 = 1 + \alpha^4 = \alpha(\alpha^3 + \alpha + 1) + \alpha^2 + \alpha + 1 = \alpha^2 + \alpha + 1$. As a matter of fact α is, in this example, a generator of the multiplicative group of $GF(2)[x]/(x^3 + x + 1)$. That is why α is called a *primitive element* of $GF(8)$ and why $x^3 + x + 1$ is called a *primitive polynomial*.

Now note that the columns in matrix (4.1) are exactly the elements $1, \alpha, \ldots, \alpha^6$, written as polynomials of degree at most 2 in α. So, the binary, cyclic Hamming code of length 7 can be described in terms of the non-binary (!) parity check matrix

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \end{pmatrix} \tag{4.7}$$

i.e. as the code $\{\underline{c} \text{ in } C \mid H\underline{c}^T = \underline{0}^T\}$. This cyclic code has generator polynomial $g(x) = 1 + x + x^3$, which factors into $(x - \alpha)(x - \alpha^2)(x - \alpha^4)$ over $GF(2^3)$, and parity check polynomial $h(x) = (1 + x)(1 + x^2 + x^3)$, which factors into $(x - 1)(x - \alpha^3)(x - \alpha^5)(x - \alpha^6)$ over $GF(2^3)$.

This follows from the factorization of $x^7 - 1$ in $GF(2)[x]$

$$x^7 - 1 = (1 + x)(1 + x + x^3)(1 + x^2 + x^3)$$

and the factorization of $x^7 - 1$ over $GF(2^3)$

$$x^7 - 1 = (x - 1)(x - \alpha) \cdots (x - \alpha^6).$$

So, one has

$$\frac{x^7 - 1}{1 + x + x^3} = (1 + x)(1 + x^2 + x^3),$$

as well as

$$\frac{x^7 - 1}{(x - \alpha)(x - \alpha^2)(x - \alpha^4)} = (x - 1)(x - \alpha^3)(x - \alpha^5)(x - \alpha^6).$$

In general

$$x^{q^m - 1} - 1 = \prod_{\xi \in GF(q^m),\ \xi \neq 0} (x - \xi).$$

So, one can get a complete factorization of $x^n - 1$ into linear factors over $GF(q^m)$, if $n \mid (q^m - 1)$, or equivalently $q^m \equiv 1 \pmod{n}$. Indeed, let $n \mid (q^m - 1)$ and let ω be a primitive

CHAPTER 4. CYCLIC CODES AND GOPPA CODES

element in $GF(q^m)$. Then the n different powers of $\alpha = \omega^{(q^m-1)/n}$ will all be a zero of $x^n - 1$ and thus

$$x^n - 1 = \prod_{i=0}^{n-1} (x - \alpha^i).$$

Since α is a zero of $x^n - 1$ and also generates the other zeros of $x^n - 1$, it is called a *primitive n-th root of unity.*

For the existence of an integer m such that $q^m \equiv 1 \pmod{n}$ it is necessary and sufficient to assume that the gcd of q and n is equal to 1. From now on, we shall always assume that. In particular, binary cyclic codes will always have odd length.

It also follows from the above that the generator polynomial $g(x)$ of a cyclic code C in $V_n(q)$ can be written as

$$g(x) = \prod_{i \in I} (x - \alpha^i), \tag{4.8}$$

where I is a subset of $\{0, 1, \ldots, n-1\}$, called the *defining set* of C with respect to α. The parity check polynomial $h(x)$ has the complement of I in $\{0, 1, \ldots, n-1\}$ as defining set.

In the notation of above, let $f(x)$ be an irreducible q-ary polynomial dividing $x^n - 1$ and let α^i be a zero of $f(x)$. From the theory of finite fields we know, that also $(\alpha^i)^q = \alpha^{iq}$ is a zero of $f(x)$ and by induction also $\alpha^{iq^2}, \ldots, \alpha^{iq^{m-1}}$ (not all these elements have to be different). Of course these exponents can be reduced modulo n, since $\alpha^n = 1$. The elements α^{iq^j} are called the *conjugates* of α^i. The set $\{iq^j \mid j = 0, 1, \ldots\}$ consisting of the exponents modulo n of these conjugates is called the *cyclotomic coset* C_i of i modulo n.

The set of all conjugates of a zero α^i of an irreducible polynomial $f(x)$ gives the complete factorization of $f(x)$ into linear factors:

$$f(x) = \prod_{l \in C_i} (x - \alpha^l).$$

This polynomial $f(x)$ is called the *minimal polynomial* of α^i and is often denoted by $m_i(x)$.

What we have shown above is that a generator polynomial of a cyclic code is the product of some minimal polynomials and that the defining set of a cyclic code is the union of the corresponding cyclotomic cosets. A necessary and sufficient condition for this is that the defining set I has the property

$$i \in I \quad \Rightarrow qi \in I,$$

where qi of course has to be reduced modulo n.

Example 4.1.5 Let $q = 3$ and $n = 11$. To find the smallest extension field of $GF(3)$ that contains the 11-th roots of unity, one has to determine (the smallest) m with $11 | (q^m - 1)$.

One obtains $m = 5$. So $x^{11} - 1 = \prod_{i=0}^{n-1}(x - \alpha^i)$, where $\alpha = \omega^{(3^5-1)/11}$ for some (each) primitive element ω in $GF(3^5)$.

There are three cyclotomic cosets. The first is $\mathcal{C}_0 = 0$, giving rise to the ternary polynomial $m_0(x) = x - 1$. The other two are

$$\mathcal{C}_1 = \{1, 3, 9, 5, 4\}$$

and

$$\mathcal{C}_{-1} = \{2, 6, 7, 10, 8\}.$$

They give rise to the irreducible, ternary polynomials

$$\begin{aligned} m_1(x) &= (x - \alpha)(x - \alpha^3)(x - \alpha^9)(x - \alpha^5)(x - \alpha^4) = \\ &= x^5 + x^4 - x^3 + x^2 - 1 \end{aligned}$$

and

$$\begin{aligned} m_{-1}(x) &= (x - \alpha^2)(x - \alpha^6)(x - \alpha^7)(x - \alpha^{10})(x - \alpha^8) = \\ &= x^5 - x^3 + x^2 - x - 1 \end{aligned}$$

(or the other way around depending on the choice of α).

The code generated by $m_1(x)$ (or by $m_{-1}(x)$) has dimension $k = 6$.

The code in Example 4.1.5 will be studied further in Section 4.3.

In view of the above, it is not necessary to give the whole defining set I of a cyclic code, but just one element from each cyclotomic coset in the defining set. The other elements (corresponding to the conjugates of these elements) have to be there anyhow. For instance, it is enough to say that the code in Example 4.1.5 generated by $m_1(x)$ has defining set $\{1\}$.

We summarize the preceding theory in the following theorem.

Theorem 4.1.6 *Consider $V_n(q)$ with $\gcd(q, n) = 1$. Let m satisfy $q^m \equiv 1 \pmod{n}$ and let ω be a primitive element in $GF(q^m)$. Then $\alpha = \omega^{(q^m-1)/n}$ is a primitive n-th root of unity.*

Let $I = \{i_1, i_2, \ldots, i_l\}$ be a subset of $\{0, 1, \ldots, n-1\}$. Then I is the defining set of a q-ary, cyclic code C of length n and each cyclic code can be defined by such sets.

Let $m_i(x)$ be the minimal polynomial of α^i. Then C can be described in the following ways:

$$C = \{c(x) \mid m_i(x) \text{ divides } c(x) \text{ for all } i \text{ in } I\}, \tag{4.9}$$

$$C = \{c(x) \mid c(\alpha^i) = 0 \text{ for all } i \text{ in } I\}, \tag{4.10}$$

$$C = \{\underline{c} \in V_n(q) \mid H\underline{c}^T = \underline{0}^T\}, \tag{4.11}$$

where

$$H = \begin{pmatrix} 1 & \alpha^{i_1} & \alpha^{2i_1} & \cdots & \cdots & \alpha^{(n-1)i_1} \\ 1 & \alpha^{i_2} & \alpha^{2i_2} & \cdots & \cdots & \alpha^{(n-1)i_2} \\ \vdots & \vdots & & & & \vdots \\ 1 & \alpha^{i_l} & \alpha^{2i_l} & \cdots & \cdots & \alpha^{(n-1)i_l} \end{pmatrix}.$$

One example of this theorem we have already seen. The binary Hamming code of length 7 with parity check matrix (4.1) is a cyclic code with generator polynomial $m_1(x) = 1 + x + x^3$ which has α (and its conjugates) as zero. An equivalent parity check matrix is given in (4.7).

4.2 BCH codes and Reed-Solomon codes

In (4.7) we have seen that the binary Hamming code of length 7 is (equivalent to) a cyclic code. This turns out to be true for all binary Hamming codes (to extend this result to other fields one has to be a little bit more careful).

Theorem 4.2.1 *Let α be a primitive element in $GF(2^m)$. Then the binary, cyclic code C of length $n = 2^m - 1$ with defining set $\{1\}$ has parameters $[n = 2^m - 1, n - m, 3]$, so it is a binary Hamming code.*

Proof: Theorem 4.1.6 gives the $1 \times n$ parity check matrix (in $GF(2^m)$) of the code with defining set $\{1\}$. Since α is a primitive element of $GF(2^m)$ (and thus a primitive n-th root of unity) all the columns in H, when written as binary vectors, are distinct non-zero vectors in V_m.

This proves that C is the binary Hamming code of length $2^m - 1$.

$\square$

The defining set $\{1\}$ in Theorem 4.2.1 gives a binary code that is 1-error-correcting, so one may want to try $\{1, 2\}$ as defining set for a 2-error-correcting, binary code. This does not work however, because $c(\alpha) = 0$ implies (for binary $c(x)$) that $c(\alpha^2) = 0$, and thus this equation will not give any additional relation.

Theorem 4.2.2 *Let α be a primitive element in $GF(2^m)$. The binary cyclic code C of length $n = 2^m - 1$ with defining set $\{1, 3\}$ is 2-error correcting.*

Instead of giving a proof that the code C in Theorem 4.2.2 has minimum distance 5, we shall give an algorithm for decoding up to 2 errors.

The parity check matrix of C is given by:

$$H = \begin{pmatrix} 1 & \alpha^1 & \alpha^2 & \cdots & \cdots & \alpha^{(n-1)} \\ 1 & \alpha^3 & \alpha^{2 \cdot 3} & \cdots & \cdots & \alpha^{(n-1) \cdot 3} \end{pmatrix}$$

Let $\underline{c}$ be the transmitted word and $\underline{r}$ the received word. So $\underline{r} = \underline{c} + \underline{e}$, where $\underline{e}$ has weight at most 2. The syndrome of $\underline{r}$ consists of the elements s_1, s_3 in $GF(2^m)$, defined by

$$s_1 = r(\alpha) = c(\alpha) + e(\alpha) = e(\alpha),$$

$$s_3 = r(\alpha^3) = c(\alpha^3) + e(\alpha^3) = e(\alpha^3),$$

where we have used the conditions on $\underline{c}$ in $C : c(\alpha) = c(\alpha^3) = 0$.

There are three cases to be considered:

Case 0: $w(\underline{e}) = 0$.

In this case $\underline{r} = \underline{c}$ and no errors have been made. Further $s_1 = s_3 = 0$.

Case 1: $w(\underline{e}) = 1$.

Let the single error be at coordinate i, so $e(x) = x^i$. One has $s_1 = e(\alpha) = \alpha^i$ and $s_3 = e(\alpha^3) = \alpha^{3i}$. In particular, $s_3 = s_1^3$ and $s_1 \neq 0$.

Case 2: $w(\underline{e}) = 2$.

Let the two errors be at coordinates i and j, so $e(x) = x^i + x^j$, $i \neq j$. One has $s_1 = e(\alpha) = \alpha^i + \alpha^j$ and $s_3 = e(\alpha^3) = \alpha^{3i} + \alpha^{3j}$.

Note that $s_3 + s_1^3 = \alpha^{3i} + \alpha^{3j} + (\alpha^i + \alpha^j)^3 = \alpha^{2i}\alpha^j + \alpha^i\alpha^{2j} = \alpha^{i+j}(\alpha^i + \alpha^j) = \alpha^{i+j}s_1$. Since $i \neq j$, in this case $s_3 \neq s_1^3$.

It also follows that

$$x^2 + s_1 x + \frac{s_3 + s_1^3}{s_1} = x^2 + (\alpha^i + \alpha^j)x + \alpha^{i+j} = (x - \alpha^i)(x - \alpha^j).$$

So the zeros of $x^2 + s_1 x + \frac{s_3 + s_1^3}{s_1}$ will give the locations of the errors.

Together, these three cases prove that the following algorithm will decode up to two errors.

Algorithm 4.2.3 *Consider the binary, cyclic 2-error-correcting code of length $n = 2^m - 1$ of Theorem 4.2.2. Let $\underline{r}$ be a received word, which is at distance at most 2 from a codeword.*

Then the closest codeword to $\underline{r}$ can be found in the following way:

- *Determine the syndrome $s_1 = r(\alpha)$ and $s_3 = r(\alpha^3)$.*

- *There are three mutually excluding cases:*

    1. *If $s_1 = s_3 = 0$, the received word $\underline{r}$ is a codeword.*
    2. *If $s_3 = s_1^3 \neq 0$, a single error has been made at coordinate i, determined by $s_1 = \alpha^i$.*

3. If $s_3 \neq s_1^3$, two errors have been made. Their coordinates i and j are determined by the zeros α^i and α^j of $z^2 + s_1 z + \frac{s_3 + s_1^3}{s_1}$.

We shall now describe a very general class of cyclic codes with certain guaranteed minimum distance properties. They are named after R.C. Bose, D.K. Ray-Chaudhuri and A. Hocquenghem (the first two found this class independently of the last).

Definition 4.2.4 *Let α be a primitive n-th root of unity in an extension field of $GF(q)$ and let I be the defining set of a q-ary cyclic code C of length n.*

If I contains $d_{BCH} - 1$ consecutive integers (taken modulo n), one says that C is a BCH code of designed distance d_{BCH}.

If I contains $\{1, 2, \ldots, d_{BCH} - 1\}$ as a subset, the code C will be called a narrow-sense BCH code.

If $n = q^m - 1$, the code C is called a primitive BCH code (in this case α itself is a primitive field element).

The justification of the notation d_{BCH} will be given in the following theorem.

Theorem 4.2.5 (BCH bound) *The minimum distance d of a BCH code with designed distance d_{BCH} satisfies*

$$d \geq d_{BCH}. \tag{4.12}$$

Proof: We shall only prove the theorem for the special case of narrow-sense codes. The general case will follow from a theorem in the next section.

Take any non-zero codeword $c(x)$ in C and compute the polynomial

$$C(X) = \sum_{i=1}^{n} c(\alpha^i) X^{n-i}.$$

The polynomial $C(X)$ is called the *Mattson-Solomon polynomial* of $c(x)$. Because $c(\alpha^i) = 0$ for $1 \leq i \leq d_{BCH} - 1$ by assumption, we conclude that $C(X)$ is a polynomial in X of degree at most $n - d_{BCH}$, so it has at most $n - d_{BCH}$ distinct zeros.

Now, for $0 \leq l \leq n - 1$,

$$
\begin{aligned}
C(\alpha^l) &= \sum_{i=1}^{n} c(\alpha^i) \alpha^{-il} = \sum_{i=1}^{n} \sum_{j=0}^{n-1} c_j \alpha^{ij} \alpha^{-il} = \\
&= \sum_{j=0}^{n-1} \sum_{i=1}^{n} c_j \alpha^{i(j-l)} = nc_l,
\end{aligned}
$$

because $1 + \alpha^u + \alpha^{2u} + \ldots + \alpha^{(\frac{n}{u}-1)u}$ for $1 \leq u < n$ is equal to $\frac{\alpha^n - 1}{\alpha^u - 1}$ which is zero. It follows that at most $n - d_{BCH}$ coordinates c_l can be zero and thus that $c(x)$ has weight at least d_{BCH}.

$\square$

How to decode up to $e_{BCH} = \lfloor (d_{BCH} - 1)/2 \rfloor$ errors is an entirely different problem. This will be discussed in a more general setting in Section 4.5.

Example 4.2.6 In order to construct a binary, narrow-sense BCH code of length 15 with designed distance 7, one has to take a defining set including $\{1, 2, \ldots, 6\}$. Clearly, $GF(2^4)$ is the smallest extension field of $GF(2)$ containing primitive 15-th roots of unity.

The relevant cyclotomic cosets in $GF(2^4)$ are $\{1, 2, 4, 8\}$, $\{3, 6, 12, 9\}$ and $\{5, 10\}$, with corresponding irreducible polynomials $m_1(x)$, $m_3(x)$ and $m_5(x)$ of degree 4, 4, and 2 respectively.

So, the binary cyclic code of length 15 generated by the 10-th degree polynomial $g(x) = m_1(x)m_3(x)m_5(x)$ has parameters $[15, 5, \geq 7]$.

Example 4.2.7 Consider the binary code of length 23 with defining set $\{1\}$ and let α be a primitive 23-rd root of unity. One can find α in $GF(2^{11})$.

The cyclotomic coset of α^1 is given by $\{1, 2, 4, 8, 16, 9, 18, 13, 3, 6, 12\}$ of cardinality 11. In particular it contains 1, 2, 3 and 4. By the BCH bound this code has parameters $[23, 12, \geq 5]$.

The actual minimum distance of the code in Example 4.2.7 will be determined in Section 4.3.

The following special subclass of BCH codes turns out to be very important in many applications.

Definition 4.2.8 *A* Reed-Solomon code *is a narrow-sense q-ary BCH code of length* $n = q - 1$.

The smallest extension field of $GF(q)$ containing n-th roots of unity with $n = q - 1$ is of course $GF(q)$ itself. Let α be a primitive element of $GF(q)$. The generator polynomial $g(x)$ of the Reed-Solomon code with designed minimum distance d_{BCH} is given by $\prod_{i=1}^{d_{BCH}-1}(x - \alpha^i)$, which has degree $d_{BCH} - 1$. So this code has parameters $[n = q - 1, n - (d_{BCH} - 1), \geq d_{BCH}]$. It follows however from the linear version of the Singleton bound at the end of Section 2.2 that equality must hold, i.e. the actual minimum distance is equal to d_{BCH}. A different way to see this is to observe that $g(x)$ has degree $d_{BCH} - 1$ and thus weight at most d_{BCH}. It follows that the minimum distance of this Reed-Solomon code is also bounded above by d_{BCH} and thus it must be equal to d_{BCH}.

So, Reed-Solomon codes are MDS. Shortening a MDS code again gives a MDS code. Thus, we have proved the first statement in the next theorem.

Theorem 4.2.9 *The (shortened) Reed-Solomon codes are MDS. Also, the extended Reed-Solomon code is MDS.*

Proof: The extended code is defined by (3.1.1). A codeword $c(x)$ of weight d cannot satisfy $c(1) = 0$, because then it would have d consecutive zeros and by the BCH bound it would have weight at least $d + 1$. So $c(1) \neq 0$ and thus the extension of a weight d codeword will have weight $d + 1$.

$\square$

4.3 The Golay codes

In this section we want to further analyze two codes that we have already seen before. The first one is the binary $[23, 12, \geq 5]$ code C described in Example 4.2.7.

By looking at the cyclotomic coset containing α one sees that $x^{23} - 1 = (x - 1)m_1(x)m_{-1}(x)$ and that $m_1(x) = x^{11}m_{-1}(1/x)$.

Now consider a codeword $c(x)$ in C of even weight w, say $c(x) = x^{i_1} + x^{i_2} + \cdots + x^{i_w}$. It follows that

$$c(x) \equiv 0 \ (\text{mod} \ (x - 1)m_1(x))$$

and, for the same reason, that

$$x^{23}c(1/x) \equiv \sum_{u=1}^{w} x^{-i_u} \equiv 0 \ (\text{mod} \ (x - 1)m_{-1}(x)).$$

Since $x^{23} - 1 = (x - 1)m_1(x)m_{-1}(x)$, it follows that $c(x)x^{23}c(1/x) \equiv 0 \ (\text{mod} \ x^{23} - 1)$ and thus, because w is even,

$$0 \equiv \sum_{u=1}^{w} x^{i_u} \sum_{v=1}^{w} x^{-i_v} \equiv \sum_{u \neq v, u, v = 1}^{w} x^{i_u - i_v} \ (\text{mod} \ x^{23} - 1).$$

Writing $\sum_{i=1}^{22} s_i x^i$ for this sum, we conclude that each s_i must be zero modulo 2. However, the sum also does not change under the mapping $x \to 1/x$ modulo $x^{23} - 1$, which shows that $s_i = s_{23-i}$.

The expression $\sum_{u \neq v, u, v = 1}^{w} x^{i_u - i_v}$ contains $w(w-1)$ terms. Whenever there is a cancelation of terms, say $i_u - i_v \equiv i_{u'} - i_{v'} \ (\text{mod} \ 23)$, also a second cancelation of terms occurs: $i_v - i_u \equiv i_{v'} - i_{u'} \ (\text{mod} \ 23)$. So terms disappear four at a time. It follows that $w(w-1) \equiv 0 \ (\text{mod} \ 4)$. A different way of saying the same is: $w(w - 1) \equiv \sum_{i=1}^{22} s_i \equiv \sum_{i=1}^{11} 2s_i \equiv 0 \ (\text{mod} \ 4)$, since each s_i is even.

Since $2(2 - 1) \equiv 2 \ (\text{mod} \ 4)$, we can conclude that no codewords in C occur of weight 2 mod 4.

The code C obviously contains the all-one vector because $\sum_{i=0}^{22} x^i = (x^{23} - 1)/(x - 1) = m_1(x)m_{-1}(x)$. Let $A(z)$ be the weight enumerator of C. From $A_i = A_{23-i}$ for all i, we now know: $A_5 = A_{18} = 0$, $A_6 = 0$, proving that C has minimum distance at least 7.

However, $2^{12} \sum_{i=0}^{3} \binom{23}{i} = 2^{23}$, so the minimum distance cannot be more. The code C is perfect! It is called the *binary Golay code*.

Theorem 4.3.1 *The binary, cyclic code of length 23 with defining set $\{1\}$ has parameters $[23, 12, 7]$, so it is a perfect code.*

Let us now study another code that we have seen before: the cyclic, ternary $[11, 6, \geq 4]$ code in Example 4.1.5 generated by $x^5 + x^4 - x^3 + x^2 - 1$ (or its reciprocal). The extended code C^{ext} has generator matrix G^{ext}:

$$\begin{pmatrix}
-1 & 0 & +1 & -1 & +1 & +1 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & -1 & 0 & +1 & -1 & +1 & +1 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & -1 & 0 & +1 & -1 & +1 & +1 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & -1 & 0 & +1 & -1 & +1 & +1 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & -1 & 0 & +1 & -1 & +1 & +1 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & +1 & -1 & +1 & +1 & -1
\end{pmatrix}.$$

One can easily check that the top row is orthogonal to itself and all other rows. So, each linear combination of rows of G^{ext} is orthogonal to each linear combination of rows. It follows that the extended code C^{ext} is selfdual. From $1^2 \equiv (-1)^2 \pmod 3$, it follows that each vector in C^{ext} has a weight divisible by 3. So it is a $[12, 6, \geq 6]$ code and the original cyclic code has parameters $[11, 6, \geq 5]$.

Again the Hamming bound gives: $3^6 \sum_{i=0}^{2} \binom{11}{i} 2^i = 3^{11}$, so the minimum distance can not be more than 5. Also this code is perfect! It is called the *ternary Golay code*.

Theorem 4.3.2 *The ternary, cyclic code of length 11 with defining set $\{1\}$ has parameters $[11, 6, 5]$, so it is a perfect code.*

We now know the following perfect codes in $V_n(q)$: the Golay codes, the Hamming codes, the binary repetition codes of odd length and the trivial codes of cardinality 1. It has been proved that there are no other parameter sets for which a perfect code can exist. Moreover, the Golay codes are *unique*, which means that any code with the same parameters as one of the Golay codes is in fact equivalent to it.

4.4 Goppa codes

To make it clear that the class of codes to be defined in this section is a generalization of the class of BCH codes, we shall first give an alternative description of the BCH codes.

To do this properly, first a notational matter has to be settled.

Let $a(x)$ and $G(x)$ be two polynomials over some field, that have no factor in common. By Euclid's Algorithm (see Algorithm 4.5.3), a polynomial $u(x)$ exists, such that $a(x)u(x) \equiv 1 \pmod{G(x)}$. It is quite customary to denote $u(x)$ by $a^{-1}(x)$ or $\frac{1}{a(x)}$, just like $\frac{1}{2} \equiv 4 \pmod 7$. For instance $\frac{1}{1-r} \equiv 1 + r + \cdots + r^{l-1} \equiv \frac{1-r^l}{1-r} \pmod{r^l}$.

Theorem 4.4.1 *The q-ary narrow sense, BCH code of length n with designed distance d_{BCH} is equivalent to the code*

$$\{\underline{c} \in V_n(q) \mid \sum_{i=0}^{n-1} \frac{c_i}{x - \alpha^i} \equiv 0 \;(\text{mod } x^{d_{BCH}-1})\}, \tag{4.13}$$

where α is an n-th root of unity in an extension field of $GF(q)$.

Proof: Clearly a necessary and sufficient condition for $\underline{c}$ to be in the BCH code is given by $S(x) = 0$, where $S(x) = \sum_{j=1}^{d_{BCH}-1} S_j x^{j-1}$ with $S_j = c(\alpha^j)$.

From

$$
\begin{aligned}
S(x) &\equiv \sum_{j=1}^{d_{BCH}-1} S_j x^{j-1} \equiv \\
&\equiv \sum_{j=1}^{d_{BCH}-1} \sum_{i=0}^{n-1} c_i \alpha^{ij} x^{j-1} \equiv \\
&\equiv \sum_{i=0}^{n-1} c_i \sum_{j=1}^{d_{BCH}-1} \alpha^{ij} x^{j-1} \equiv \\
&\equiv \sum_{i=0}^{n-1} c_i \alpha^i \frac{1 - \alpha^{i(d_{BCH}-1)} x^{d_{BCH}-1}}{1 - \alpha^i x} \equiv \\
&\equiv \sum_{i=0}^{n-1} c_i \alpha^i \frac{1}{1 - \alpha^i x} \equiv \\
&\equiv -\sum_{i=0}^{n-1} \frac{c_i}{x - \alpha^{-i}} \;(\text{mod } x^{d_{BCH}-1})
\end{aligned}
$$

it follows that the BCH code is equivalent to the code defined by (4.13). Indeed, one only needs to replace α by α^{-1}. □

Definition 4.4.2 *Let $L = \{\alpha_0, \alpha_1, \ldots, \alpha_{n-1}\}$ be a subset of $GF(q^m)$ of size n and let $G(x)$ be a polynomial of degree s over $GF(q^m)$ that is not zero in any of the elements α_i. Then, the* Goppa code $\Gamma(L, G)$ *is defined by*

$$\Gamma(L, G) = \{\underline{c} \in V_n(q) \mid \sum_{i=0}^{n-1} \frac{c_i}{x - \alpha_i} \equiv 0 \;(\text{mod } G(x))\}. \tag{4.14}$$

By taking $G(x) = x^{d_{BCH}-1}$ and $\alpha_i = \alpha^i, 0 \leq i < n$, relation (4.14) reduces to (4.13), so the class of Goppa codes contains the BCH codes as a subclass.

Quite clearly, Goppa codes are linear.

Whenever convenient, we shall say that the i-th coordinate of a Goppa code is at position α_i, $0 \leq i \leq n - 1$.

Theorem 4.4.3 *The Goppa code $\Gamma(L, G)$ of length n with $G(x)$ of degree s has parameters $[n, \geq n - ms, \geq s + 1]$.*

Proof:

i) We shall first show that $\Gamma(L, G)$ has minimum distance at least $s + 1$.

Let $\underline{c}$ be a codeword of weight w, $w > 0$ and let the non-zero coordinates of $\underline{c}$ be at positions $\{\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_w}\}$.

Write the summation in (4.14) as one fraction. Then, because the denominator has no factor in common with $G(x)$, condition (4.14) is equivalent to stating that the numerator in (4.14) is divisible by $G(x)$, i.e.

$$G(x) \text{ divides } \sum_{l=1}^{w(\underline{C})} c_{i_l} \prod_{1 \leq j \leq w, j \neq l} (x - \alpha_{i_j}).$$

However this numerator has degree at most $w - 1$. It follows that $w - 1 \geq s$.

ii) To prove the bound on the dimension of $\Gamma(L, G)$, we write each $1/(x - \alpha_i)$, $0 \leq i \leq n - 1$, as a polynomial $G_i(x) = \sum_{j=0}^{s-1} G_{ij} x^j$ modulo $G(x)$.

Condition (4.14) can now be written as

$$\sum_{i=0}^{n-1} c_i G_i(x) \equiv 0 \ (\text{mod } G(x))$$

or, alternatively, by considering the coefficients of x^j, $0 \leq j \leq s - 1$,

$$\sum_{i=0}^{n-1} c_i G_{ij} = 0 \quad \text{for } 0 \leq j \leq s - 1.$$

This means that $\Gamma(L, G)$ can be defined by s linear equations over $GF(q^m)$ and thus by $\leq ms$ linear equations over $GF(q)$. Hence, $\Gamma(L, G)$ has dimension at least $n - ms$.

$\square$

Example 4.4.4 *(to be continued)* Let α be the primitive element in $GF(2^4)$ satisfying $\alpha^4 + \alpha^3 + 1 = 0$. Consider the binary Goppa code $\Gamma(L, G)$ of length 12 with $G(x) = (x + \alpha)(x + \alpha^{14}) = x^2 + \alpha^8 x + 1$ and $L = \{\alpha^i \mid 2 \leq i \leq 13\}$.

The inverses $G_i(x) = \sum_{j=0}^{s-1} G_{ij} x^j$, $2 \leq i \leq 13$, of $(x - \alpha^i)$ modulo $G(x)$ are given by the columns $(G_{i0}, G_{i1})^T$ in the following parity check matrix of $\Gamma(L, G)$:

$$H = \begin{pmatrix} \alpha^9 & \alpha & \alpha^8 & \alpha^{13} & \alpha^7 & \alpha^5 & 0 & \alpha^9 & \alpha & \alpha^6 & \alpha^5 & \alpha^6 \\ \alpha^{14} & \alpha^3 & \alpha & \alpha^4 & \alpha^7 & \alpha^1 & \alpha^{14} & \alpha^4 & \alpha^{14} & \alpha^9 & \alpha^9 & \alpha^3 \end{pmatrix}.$$

To check the correctness of the first column of H, i.e. that $\dfrac{1}{x - \alpha^2} \equiv \alpha^9 + \alpha^{14} x$ (mod $x^2 + \alpha^8 x + 1$), note that

$$(x - \alpha^2)(\alpha^9 + \alpha^{14} x) \equiv \alpha^{11} + \alpha^7 x + \alpha^{14} x^2 \equiv$$

$$\equiv \alpha^{11} + \alpha^7 x + \alpha^{14}(1 + \alpha^8 x) \equiv 1 \pmod{x^2 + \alpha^8 x + 1}.$$

The above parity check matrix H can be written as a binary matrix by writing each power of α with respect to the basis $1, \alpha, \alpha^2, \alpha^3$. One obtains

$$H = \begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\
0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1
\end{pmatrix}.$$

It is not so difficult to check that H has rank 8, which makes this Goppa code a $[12, 4, \geq 3]$ code.

In the binary case, much better bounds on the minimum distance can be given than the bound in Theorem 4.4.3.

Theorem 4.4.5 *Let the defining polynomial $G(x)$ of the binary Goppa code $\Gamma(L, G)$ be a polynomial over $GF(2^m)$ of degree s that has no multiple zeros. Then, the Goppa code $\Gamma(L, G)$ will have minimum distance at least $2s + 1$.*

Proof: Write $L = \{\alpha_0, \alpha_1, \ldots, \alpha_{n-1}\}$ and let $\underline{c}$ be a codeword of weight w, $w \geq 0$. Define $f(x) = \prod_{i=0}^{n-1}(x - \alpha_i)^{c_i}$. It has degree w.

Now $f'(x)/f(x) = \sum_{i=0}^{n-1} c_i/(x - \alpha_i)$. So equation (4.14) is equivalent to $G(x)$ divides $f'(x)$, as $f(x)$ and $f'(x)$ have no common factors. But, because $q = 2$, $f'(x)$ only has terms with even exponents, say $f'(x) = f_1 + f_3 x^2 + \cdots + f_{2u+1}x^{2u}$, with $2u < w$. In $GF(2^m)$ this implies that $f'(x) = (g_1 + g_3 x + \cdots + g_{2u+1}x^u)^2$, with $g_i^2 = f_i$.

We conclude that $G(x)$ divides $(g(x))^2$, with $g(x)$ of degree at most u, where $2u < w$. Since $G(x)$ has no multiple zeros, one even has that $G(x)$ divides $g(x)$. Hence $s \leq u$ and thus $w > 2u \geq 2s$.

$\square$

Example 4.4.4 (continued) Since the Goppa polynomial $G(x)$ in Example 4.4.4 is a polynomial over $GF(2^4)$ without repeated zeros and is of degree 2, we may now conclude that the code $\Gamma(L, G)$ in that example is a $[12, 4, \geq 5]$ code.

4.5   A decoding algorithm

Goppa codes, BCH codes and Reed-Solomon codes can all be decoded efficiently by the same decoding technique (see Algorithm 4.5.8). It makes use of Euclid's Algorithm (see

Algorithm 4.5.3). We shall only describe how errors can be corrected, but the reader should know that this decoding algoritm can be adapted to perform error and erasure decoding.

We shall give the most general version: the decoding of Goppa codes. For the correct decoding of a received word, one needs to know two things: the location where the errors occurred and what their values are.

Let $G(x)$ be a polynomial over $GF(q^m)$ of degree s that has no multiple zeros and let $L = \{\alpha_0, \alpha_1, \dots, \alpha_{n-1}\}$ be a subset of $GF(q^m)$ of cardinality n. Let $\underline{c}$ in the Goppa code $\Gamma(L, G)$ be a transmitted codeword and let the vector $\underline{r} = \underline{c} + \underline{e}$ denote the received word.

Define the set B of *error locations* $B = \{\alpha_i \mid e_i \neq 0\}$ and for the β in B the corresponding *error value* $e_\beta = e_i$, where $\beta = \alpha_i$.

Definition 4.5.1 *The* error locator polynomial $\sigma(x)$ *and the* error evaluator polynomial $\omega(x)$ *of an error vector are defined by*

$$\sigma(x) = \prod_{\beta \in B} (x - \beta), \tag{4.15}$$

$$\omega(x) = \sum_{\beta \in B} e_\beta \prod_{\gamma \in B, \gamma \neq \beta} (x - \gamma). \tag{4.16}$$

The error locations are simply the zeros of $\sigma(x)$. The next theorem will give some properties of $\sigma(x)$ and $\omega(x)$. In particular, it will tell how to find the error values and also how $\sigma(x)$ and $\omega(x)$ are related to the syndrome $S(x)$ of the received word: $S(x) = \sum_{i=0}^{n-1} r_i/(x - \alpha_i) \equiv \sum_{\beta \in B} e_\beta/(x - \beta) \pmod{G(x)}$.

Theorem 4.5.2 *Let $\underline{e}$ be an error vector of weight t. Let $S(x)$ be its syndrome and let $\sigma(x)$ and $\omega(x)$ be its error locator, resp. error evaluator polynomial. Then*

$$degree(\sigma(x)) = t = |B|, \qquad degree(\omega(x)) \leq t - 1, \tag{4.17}$$

$$gcd(\sigma(x), \omega(x)) = 1, \tag{4.18}$$

$$e_\beta = \omega(\beta)/\sigma'(\beta), \quad \beta \in B, \tag{4.19}$$

$$\sigma(x)S(x) \equiv \omega(x) \pmod{G(x)}. \tag{4.20}$$

Equation (4.20) is called the key equation *for the decoding algorithm.*

Proof: Equations (4.17) and (4.18) follow directly from definitions (4.15) and (4.16).

To prove equation (4.19), take the derivative of $\sigma(x)$ to get $\sigma'(x) = \sum_{\beta \in B} \prod_{\gamma \in B, \gamma \neq \beta} (x - \gamma)$. Substitution of $\beta = \nu$, ν in B, yields $\sigma'(\nu) = \prod_{\gamma \in B, \gamma \neq \nu} (\nu - \gamma)$. Substitution of $\beta = \nu$ in $\omega(x)$ will now result in $e_\nu \sigma'(\nu)$.

Finally,

$$S(x)\sigma(x) \quad \equiv \quad \sum_{\beta \in B} \frac{e_\beta}{x - \beta} \prod_{\gamma \in B} (x - \gamma) \equiv$$

$$\equiv \sum_{\beta \in B} e_{\beta} \prod_{\gamma \in B, \gamma \neq \beta} (x - \gamma) \equiv$$
$$\equiv \omega(x) \pmod{G(x)}. \tag{4.21}$$

$\square$

Determining $\sigma(x)$ and $\omega(x)$ from (4.21) amounts to applying (the well-known) Euclid's Algorithm for polynomials to $G(x)$ and $S(x)$.

Algorithm 4.5.3 (Euclid's Algorithm) *Let $a(x)$ and $b(x)$ be two q-ary polynomials, where $degree(a(x)) \geq degree(b(x))$.*

Define the sequences of polynomials $s_i(x), u_i(x), v_i(x)$ and $q_i(x)$, where the degrees of $s_i(x)$ are strictly decreasing, recursively as follows.

$$\begin{array}{ll} s_0(x) = a(x), & s_1(x) = b(x), \\ u_0(x) = 1, & u_1(x) = 0, \\ v_0(x) = 0, & v_1(x) = 1, \\ i = 1. \end{array}$$

While $s_i(x) \neq 0$ do begin

$$\begin{array}{l} i := i + 1 \\ \text{Define } q_i(x) \text{ and } s_i(x) \text{ by the division} \\ s_{i-2}(x) = q_i(x)s_{i-1}(x) + s_i(x) \\ \text{with } degree(s_i(x)) < degree(s_{i-1}(x)). \\ \text{Define } u_i(x) \text{ and } v_i(x) \text{ by} \\ u_{i-2}(x) = q_i(x)u_{i-1}(x) + u_i(x), \\ v_{i-2}(x) = q_i(x)v_{i-1}(x) + v_i(x). \\ \text{end} \end{array}$$

n=i.

Then

$$\gcd(a(x), b(x)) = s_{n-1}(x) = u_{n-1}(x)a(x) + v_{n-1}(x)b(x). \tag{4.22}$$

Before we can derive the property of the above algorithm that will be essential for the decoding algorithm, we need three lemmas. Some readers may want to skip these lemmas with their proofs and move directly to Theorem 4.5.7.

Lemma 4.5.4 *The sequences of polynomials defined in Algorithm 4.5.3 satisfy the following equations:*

$$
\begin{array}{llll}
(i) & (-1)^{i+1}a(x) = v_i(x)s_{i-1}(x) - v_{i-1}(x)s_i(x), & 1 \le i \le n-1, \\
(ii) & (-1)^i b(x) = u_i(x)s_{i-1}(x) - u_{i-1}(x)s_i(x), & 1 \le i \le n-1, \\
(iii) & (-1)^i = u_i(x)v_{i-1}(x) - u_{i-1}(x)v_i(x), & 1 \le i \le n-1, \\
(iv) & degree(u_i(x)) + degree(s_{i-1}(x)) = degree(b(x)), & 2 \le i \le n-1, \\
(v) & degree(v_i(x)) + degree(s_{i-1}(x)) = degree(a(x)), & 1 \le i \le n-1, \\
(vi) & s_i(x) = u_i(x)a(x) + v_i(x)b(x), & 0 \le i \le n.
\end{array}
$$

Proof: Straightforward induction on i. $\qquad\qquad\square$

For our purposes, we are not interested in both the $u_i(x)$ and the $v_i(x)$ sequences, only the $v_i(x)$ sequence will do.

Lemma 4.5.5 *Let k and l be two nonnegative integers satisfying $k + l = degree(a(x)) - 1$ and $l \ge degree(\gcd(a(x), b(x)))$. Then there exists a unique integer i, $1 \le i \le n-1$, such that*

$$degree(v_i(x)) \le k, \tag{4.23}$$

$$degree(s_i(x)) \le l. \tag{4.24}$$

Proof: Since the degrees of the polynomials $s_i(x)$ are strictly decreasing, we can define a unique integer i by

$$\text{degree}(s_i(x)) \le l \le \text{degree}(s_{i-1}(x)) - 1. \tag{4.25}$$

From (v) it follows that

$$\text{degree}(v_i(x)) \le k \le \text{degree}(v_{i+1}(x)) - 1. \tag{4.26}$$

These two equations not only imply the existence of the integer i in the statement of the lemma, but also its uniqueness. Indeed equation (4.25) shows that no smaller value of i can be taken, while equation (4.26) shows that no larger value of i is permissable.

$\qquad\qquad\square$

Relation (vi) can be written as $v_i(x)b(x) \equiv s_i(x) \pmod{a(x)}$, while (v) (and the decreasing degrees of the $s_i(x)$ sequence) implies that $degree(v_i(x)) + degree(s_i(x)) < degree(a(x))$. The next theorem shows that polynomials $v(x)$ and $s(x)$ with these two properties are unique up to a common factor.

Lemma 4.5.6 *Let $s(x)$ and $v(x)$ be two non-zero polynomials satisfying*

$$v(x)b(x) \equiv s(x) \pmod{a(x)}, \tag{4.27}$$

$$degree(v(x)) + degree(s(x)) < degree(a(x)) \tag{4.28}$$

Then there exists a unique integer $i, 1 \le i \le n$, and a polynomial $\lambda(x)$ such that

$$v(x) = \lambda(x)v_i(x) \tag{4.29}$$

$$s(x) = \lambda(x)s_i(x). \tag{4.30}$$

Proof: Let $l = \text{degree}(s(x))$ and $k = \text{degree}(a(x)) - \text{degree}(s(x)) - 1$. By (4.27) $\gcd(a(x), b(x))$ divides $s(x)$, so $l \geq \text{degree}(\gcd(a(x), b(x)))$. Let i be the unique integer defined in Lemma 4.5.5 for these values of k and l. From (4.25) it follows that $\text{degree}(s_{i-1}(x)) \geq l + 1 \geq \text{degree}(s_i(x)) + 1$. Similarly (4.26) implies that $\text{degree}(v_{i+1}(x)) \geq k + 1 \geq \text{degree}(v_i(x)) + 1$. We conclude that if there exists an integer i satisfying (4.29) and (4.30) it must be unique and equal to the integer defined by Lemma 4.5.5.

Equation (4.27) can be written as

$$s(x) = u(x)a(x) + v(x)b(x) \tag{4.31}$$

for some $u(x)$. Property (vi) in Lemma 4.5.4 has a similar form:

$$s_i(x) = u_i(x)a(x) + v_i(x)b(x) \tag{4.32}$$

Eliminating $b(x)$ from these two relations yields

$$s_i(x)v(x) \equiv s(x)v_i(x) \pmod{a(x)}.$$

However, both $s_i(x)$ and $s(x)$ have degree at most l and both $v(x)$ and $v_i(x)$ have degree at most k. Since $k + l < \text{degree}(a(x))$ we conclude that

$$s_i(x)v(x) = s(x)v_i(x).$$

Substituting this back in (4.31) and (4.32) results in

$$u_i(x)v(x) = u(x)v_i(x).$$

But $u_i(x)$ and $v_i(x)$ have gcd 1 by property (iii) in Lemma 4.5.4, so $u(x) = \lambda(x)u_i(x)$ and $v(x) = \lambda(x)v_i(x)$ for some polynomial $\lambda(x)$. Substituting these again in (4.31) and (4.32) yields $s(x) = \lambda(x)s_i(x)$.

$$\square$$

We are finally able to give the theorem on which the decoding algorithm for the Goppa codes is based.

Theorem 4.5.7 *Let $\sigma(x)$ and $\omega(x)$ be the error-locator and error-evaluator polynomials of an error pattern of weight at most $\lfloor s/2 \rfloor$, where s is the degree of the polynomial $G(x)$ of the Goppa code $\Gamma(L, G)$. Let $S(x)$ be the syndrome of the error pattern. Then*

$$\sigma(x) = \lambda v_i(x), \tag{4.33}$$

$$\omega(x) = \lambda s_i(x), \tag{4.34}$$

where $s_i(x)$ and $v_i(x)$ are obtained from applying Euclid's Algorithm to $a(x) = G(x)$ and $b(x) = S(x)$ until $\text{degree}(s_i(x)) < \lfloor s/2 \rfloor$ for the first time and where λ is chosen such that $\lambda v_i(x)$ is monic.

Proof: The polynomials $\sigma(x)$ and $\omega(x)$ satisfy the conditions of Lemma 4.5.6, so $\sigma(x) = \lambda(x)v_i(x)$ and $\omega(x) = \lambda(x)s_i(x)$. From (4.18) we conclude that $\lambda(x)$ must be a constant. This has to be chosen so as to make $\sigma(x)$ monic (see (4.15)).

$\square$

Now we can formulate the decoding algorithm for the class of Goppa codes (and its subclasses of BCH and Reed-Solomon codes).

So, let $\Gamma(L, G)$ be a q-ary Goppa code of length n with degree$(G(x)) = s$ and $L = \{\alpha_0, \alpha_1, \ldots, \alpha_{n-1}\}$ in $GF(q^m)$. Let $\underline{r}$ be a received vector, $\underline{r} = \underline{c} + \underline{e}$, with weight$(\underline{e}) \leq \lfloor s/2 \rfloor$ and $\underline{c}$ in $\Gamma(L, G)$. Then, by Theorem 4.5.7, the following algorithm applied to $\underline{r}$ will output $\underline{c}$.

Algorithm 4.5.8 (Decoding Goppa codes) *Write $\underline{r} = (r_0, r_1, \ldots, r_{n-1})$.*

1. *Compute the syndrome*

$$S(x) \equiv \sum_{i=0}^{n-1} \frac{r_i}{x - \alpha_i} \pmod{G(x)}.$$

2. *Apply Euclid's Algorithm to $a(x) = G(x)$ and $b(x) = S(x)$ until degree$(s_i(x)) < \lfloor s/2 \rfloor$ for the first time. Let ν be the leading coefficient of $v_i(x)$. Set $\sigma(x) = v_i(x)/\nu$ and $\omega(x) = s_i(x)/\nu$.*

3. *Find the set $B = \{\beta \text{ in } GF(q^m) \mid \sigma(\beta) = 0\}$ of error locations.*

4. *Determine the error values $e_\beta = \omega(\beta)/\sigma'(\beta)$ for all β in B.*

5. *Determine $\underline{e} = (e_0, e_1, \ldots, e_{n-1})$ from $e_i = e_\beta$ if β is in B and $\beta = \alpha_i$ and $e_i = 0$ otherwise.*

6. *Set $\underline{c} = \underline{r} - \underline{e}$.*

Remark 1: In Step 3 in the above algorithm one simply has to try the various α_i in L until degree$(\sigma(x))$ zeros have been found.

Remark 2: The division by ν in Step 2 is not necessary. Indeed, the zeros of $\sigma(x)$ are not changed by a scalar multiplication and for the computation of the error values (in Step 4) both numerator and denominator contain the same factor $\underline{\nu}$.

Remark 3: For the decoding of BCH and RS codes over an extension field of $GF(2)$ one can compute the syndrome directly from $S_j = r(\alpha^j), 1 \leq j \leq d_{BCH} - 1$. The error locations are now however given by the reciprocals of the elements in B. The reason for this lies in the equivalence of the BCH code and the Goppa code under the mapping $\alpha \to \alpha^{-1}$ (see the proof of Theorem 4.4.1).

Let $\Gamma(L, G)$ be a binary Goppa codes defined by an irreducible Goppa polynomial $G(x)$ of degree s over $GF(2^m)$. Then $\Gamma(L, G)$ is s error-correcting by Theorem 4.4.5. However, Algorithm 4.5.8 will only decode up to $\lfloor s/2 \rfloor$ errors, because the Euclidean Algorithm can not be applied directly. Note that in this case (4.16) reduces to $\omega(x) = \sum_{\beta \in B} \prod_{\gamma \in B, \gamma \neq \beta}(x - \gamma) = \sigma'(x)$. So the key equation (4.20) can now be rewritten as

$$\sigma(x)S(x) \equiv \sigma'(x) \; (\text{mod } G(x)). \tag{4.35}$$

By splitting off the squares and the non-squares is $\sigma(x)$ one can write

$$\sigma(x) = \alpha^2(x) + x\beta^2(x) \tag{4.36}$$

with

$$\text{degree}(\alpha(x)) \leq s/2 \quad \text{and} \quad \text{degree}(\beta(x)) \leq (s-1)/2. \tag{4.37}$$

Since $GF(2^m)$ has characteristic 2, it follows that

$$\sigma'(x) = \beta^2(x). \tag{4.38}$$

Because $G(x)$ is irreducible, one can use Euclid's Algorithm to find the multiplicative inverse $T(x)$ of $S(x)$ modulo $G(x)$. So $T(x)$ is determined by $S(x)T(x) \equiv 1 \; (\text{mod } G(x))$. Substituting (4.36) and (4.38) in (4.35) yields

$$(T(x) + x)\,\beta^2(x) \equiv \alpha^2(x) \; (\text{mod } G(x)). \tag{4.39}$$

If $T(x) = x$, one has that $\alpha(x) = 0$ and thus that $\sigma(x) = x\beta^2(x)$. Since $\sigma(x)$ has no repeated zeros, it follows that $\beta(x) = 1$ and that $\sigma(x) = x$.

In the case that $T(x) \neq x$, more work has to be done to find $\sigma(x)$. First we determine the uniquely defined polynomial $R(x)$ satisfying $R^2(x) \equiv T(x) + x \; (\text{mod } G(x))$. Indeed, this can be done since squaring polynomials modulo $G(x)$ is a linear mapping, because the characteristic of $GF(2^m)$ is 2. The inverse of this linear mapping applied to $T(x) + x$ yields $R(x)$ (store this inverse mapping when decoding more words).

After substitution of $T(x) + x \equiv R^2(x) \; (\text{mod } G(x))$ in (4.39), one obtains $R^2(x)\beta^2(x) \equiv \alpha^2(x) \; (\text{mod } G(x))$ and thus that

$$R(x)\beta(x) \equiv \alpha(x) \; (\text{mod } G(x)). \tag{4.40}$$

It follows that in exactly the same way as in Algorithm 4.5.8 (i.e. by means of Theorem 4.5.7) Euclid's Algorithm can be applied to $G(x)$ and $R(x)$ to determine the polynomials $\alpha(x)$ and $\beta(x)$, satisfying (4.37) and (4.40).

Putting $\sigma(x) = \alpha^2(x) + x\beta^2(x)$ one is now ready to proceed with Algorithm 4.5.8 (only Steps 3, 5 and 6 are needed, because $e_\beta = 1$ for $\beta \in B$).

It is in the above way that binary Goppa codes can be decoded.

4.6 Geometric Goppa codes

The Goppa codes that are discussed in Section 4.4 are now known to belong to a much larger class of codes. It was Goppa himself who formulated the generalization. The codes in this larger class are all constructed with the help of algebraic curves and are therefore called *geometric Goppa codes*, *algebraic-geometry codes* or just *AG codes*. For a general treatment, we refer the reader to the literature. To construct a particular class of codes, we use:

Definition 4.6.1 *Let q be a power of a prime and let s and t be two positive integers such that $\gcd(s, q) = 1$ and $\gcd(s, t) = 1$. Let $f \in GF(q)[x]$ be a q-ary polynomial of degree t with no multiple zeros. Let the set $\mathcal{P}$ be defined by*

$$\mathcal{P} = \{(x, y) \mid y^s = f(x), \ x \in GF(q), \ y \in GF(q)\}. \tag{4.41}$$

The set $\mathcal{P}$ with its defining equation $y^s = f(x)$ is called an algebraic curve, *its elements are called* points. *Let n denote the cardinality of $\mathcal{P}$ and write $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$. For non-negative m, we define a vector space $L(m)$ of functions by the following linear span over $GF(q)$*

$$L(m) \ = \ \langle x^i y^j \mid is + jt \leq m, \ 0 \leq i, \ 0 \leq j < s \rangle. \tag{4.42}$$

The AG-code $C(\mathcal{P}, m)$ *is defined as the following set of q-ary vectors:*

$$C(\mathcal{P}, m) = \{(u(P_1), u(P_2), \ldots, u(P_n)) \mid u \in L(m)\}. \tag{4.43}$$

Notice that the choice of the parameters s and t and of the function f is suppressed in the notation $C(\mathcal{P}, m)$.

The first question that arises is: what are the parameters of AG-codes? It is clear that the length of an AG-code is given by the number of points in the set $\mathcal{P}$, i.e. by n. The dimension of an AG-code is related to the dimension of $L(m)$ and is treated in Corollary 4.6.5. The restriction $j < s$ in (4.42) can be made, because terms $x^i y^j$ with $j \geq s$ can be reduced by means of the relation $y^s = f(x)$. We shall give methods to estimate the minimum distance in Lemma 4.6.6. First, we shall discuss some examples.

Example 4.6.2 The first non-trivial choice for the numbers s and t in Definition 4.6.1 is to take $s = t = 1$. With $f(x) = x$, the set $\mathcal{P}$ consists of the q points (x, x), $x \in GF(q)$. The corresponding curve is called an *affine line*. In the definition of $L(m)$ we now have $j = 0$ and $L(m)$ is spanned by the monomials x^i, $0 \leq i \leq m$, i.e. $L(m)$ is the space of polynomials in x of degree at most m. Clearly, any polynomial in $L(m)$ has at most m zeros.

For $0 \leq m \leq q - 1$ this code $C(\mathcal{P}, m)$ has dimension $k = m + 1$ and any non-zero codeword has at most m coordinates equal to 0. Thus the parameters of this AG-code are $[q, m+1, q-m]$. In particular the code meets the Singleton bound (see also Problem 4.7.1). Note that for $m = q - 1$ the code has parameters $[q, q, 1]$ and we will clearly get no larger codes for $m > q - 1$.

Example 4.6.3 Consider the finite field $GF(4) = GF(2)[x]/(f(x))$ with $f(x) = x^2 + x + 1$ and let α be a zero of $f(x)$. Thus $GF(4) = \{0, 1, \alpha, \alpha^2\}$. Also, let $s = 3, t = 2$, where t is the degree of f. What are the points in $\mathcal{P}$ (see (4.41)), i.e. what (x, y) satisfy $y^3 = x^2 + x + 1$? Obviously we have the point $(\alpha, 0)$ and also $(\alpha^2, 0)$ is a point. If $y \neq 0$ then $y^3 = 1$ and only $x = 0$ or $x = 1$ do satisfy $y^3 = f(x)$. The complete set of points becomes

$$\mathcal{P} = \{(\alpha, 0), (\alpha^2, 0), (0, 1), (1, 1), (0, \alpha), (1, \alpha), (0, \alpha^2), (1, \alpha^2)\}$$

The vectorspaces $L(m)$, for the first few values of m are given by:

$$
\begin{aligned}
L(0) &= \langle 1 \rangle, \\
L(1) &= \langle 1 \rangle, \\
L(2) &= \langle 1, y \rangle, \\
L(3) &= \langle 1, y, x \rangle, \\
L(4) &= \langle 1, y, x, y^2 \rangle, \\
L(5) &= \langle 1, y, x, y^2, xy \rangle, \\
L(6) &= \langle 1, y, x, y^2, xy, x^2 \rangle, \\
L(7) &= \langle 1, y, x, y^2, xy, x^2, xy^2 \rangle.
\end{aligned}
$$

The pattern is by no means as simple as for the affine line in Example 4.6.2. The equation $y^3 = x^2 + x + 1$ corresponds to a so-called *elliptic curve*. For $m \geq 1$, the dimension of $L(m)$ increases by 1 when m increases by 1 (this is indeed true for larger values of m as follows from Corollary 4.6.5 below). Thus, for $1 \leq m \leq 7$, the code $C(\mathcal{P}, m)$ has dimension $k = m$. The dimension is one lower than in Example 4.6.2, because this time we have no monomial $x^i y^j$ with $is + jt = 3i + 2j = 1$.

For the minimum distance, note that the function $y \in L(2)$ has two zeros, the points: $(\alpha, 0)$ and $(\alpha^2, 0)$. The function $x \in L(3)$ has three zeros, the points: $(0, 1), (0, \alpha)$ and $(0, \alpha^2)$. In general the number of zeros of a function $f \in L(m)$ can be shown to be not larger than m (see Lemma 4.6.6). We conclude that the code $C(\mathcal{P}, m)$ in this example has parameters $[8, m, d \geq 8 - m]$, for $1 \leq m \leq 7$.

To obtain the dimension of $L(m)$ in general and to clarify the restriction $j < s$ in Definition 4.6.1, we need the following lemma.

Lemma 4.6.4 *Let s, t be positive integers with $\gcd(s, t) = 1$ and let r be any integer. Precisely one of the following two equations has a non-negative integer solution (i, j):*

$$i's + j't = st - s - t - r \tag{4.44}$$

$$i''s + j''t = r \tag{4.45}$$

Proof: Both equations have integer solutions, since $\gcd(s, t) = 1$. Let (i', j') and (i'', j'') be the unique solutions with $0 \leq j', j'' < s$ (any solution can be uniquely transformed in a solution satisfying this restriction by adding a proper multiple of the equation $ts - st = 0$ to it). In particular $0 \leq j' + j'' \leq 2s - 2$. Addition of the Equations (4.44) and (4.45) yields

$$(i' + i'')s + (j' + j'')t = st - s - t.$$

It also follows that s divides $(j' + j'' + 1)t$, hence s divides $(j' + j'' + 1)$ and by the inequality on $j' + j''$ above we must have $j' + j'' + 1 = s$. Thus $i' + i'' = -1$ and precisely one of i' and i'' is negative. The claim now follows.

$\square$

Corollary 4.6.5 *Let $g = (s - 1)(t - 1)/2$. Then, for $m \geq 2g - 1$,*

$$\text{dimension of } L(m) = m + 1 - g \tag{4.46}$$

Proof: Equation (4.45) has no non-negative solutions for $r < 0$ and, by Lemma 4.6.4, Equation (4.44) will then have a non-negative solution. Thus we can find non-negative solutions to $is + jt = m$ (see (4.42)) for $m > st - s - t = 2g - 1$ and the dimension of $L(m)$ will increase as a function of m for $m \geq 2g$. This increase is always by 1, because the relation $is + jt = m$ can not have two solutions with $0 \leq j < s$.

It remains to prove that the dimension of $L(2g - 1)$ is equal to g. But by Lemma 4.6.4, for every $0 \leq r \leq g - 1$, precisely one of Equations (4.44) and (4.45) has a solution (note that substitution of the values $0 \leq r \leq g - 1$ in (4.44) and (4.45) covers exactly the whole range of right hand sides in between 0 and $2g - 1$). The g solutions correspond to the g monomials $x^i y^j$ that form a basis for $L(2g - 1)$.

$\square$

Lemma 4.6.6 *Any non-zero function $u(x, y)$ in $L(m)$ has at most m zeros in $\mathcal{P}$.*

Proof: From Definition 4.6.1 we recall that $\mathcal{P}$ is s subset of $GF(q) \times GF(q)$ and that the points in $\mathcal{P}$ satisfy $y^s = f(x)$ with $f(x)$ of degree t and $\gcd(s, q) = \gcd(s, t) = 1$. It follows that an extension field of $GF(q)$ exists, say $GF(q^l)$, that contains a primitive s-th root of unity ζ. We shall show that even among the points in $\mathcal{P}' = \{(x, y) \mid y^s = f(x), \ x \in GF(q^l), \ y \in GF(q^l)\}$ the function $u(x, y)$ will have at most m zeros.

The points of $\mathcal{P}'$ can be divided in classes $\mathcal{P}'_x$, each class containing the points with a fixed x-coordinate. If x is a zero of $f(x)$, the class $\mathcal{P}'_x$ contains one element: $(x, 0)$. For x with $f(x) \neq 0$, the class $\mathcal{P}'_x$ is equal to $\{(x, y), (x, \zeta y), \ldots, (x, \zeta^{s-1} y)\}$ and contains s elements. Now, consider the classes in which the zeros of $u(x, y)$ are contained. We clearly find the same classes if we consider the zeros of the function:

$$u^*(x, y) = u(x, y) \cdot u(x, \zeta y) \cdots u(x, \zeta^{s-1} y). \tag{4.47}$$

69

By (4.42) one can write $u(x, y)$ as

$$u(x, y) = \sum_{0 \leq i, 0 \leq j < s, is + jt \leq m} u_{i,j} x^i y^j, \quad u_{i,j} \in GF(q).$$

If in the evaluation in (4.47) of the product a term

$$u_{i_0, j_0} x^{i_0} y^{j_0} \cdot u_{i_1, j_1} x^{i_1} (\zeta y)^{j_1} \cdots u_{i_{s-1}, j_{s-1}} x^{i_{s-1}} (\zeta^{s-1} y)^{j_{s-1}},$$

with $i_0 + i_1 + \ldots + i_{s-1} = i$ and $j_0 + j_1 + \ldots + j_{s-1} = j$, occurs then so does the term

$$u_{i_{s-1}, j_{s-1}} x^{i_{s-1}} y^{j_{s-1}} \cdot u_{i_0, j_0} x^{i_0} (\zeta y)^{j_0} \cdots u_{i_{s-2}, j_{s-2}} x^{i_{s-2}} (\zeta^{s-1} y)^{j_{s-2}},$$

which is just the previous term multiplied by $\zeta^{j_0 + j_1 + \ldots + j_{s-1}} = \zeta^j$. For the same reason, this term also occurs multiplied by each of $\zeta^{2j}, \ldots, \zeta^{(s-1)j}$. It follows from $1 + \zeta^j + \ldots + \zeta^{j(s-1)} = 0$, $1 \leq j \leq s - 1$, that these s terms cancel, unless j is a multiple of s, say $j = sj'$.

We conclude that $u^*(x, y)$ only consists of monomials of the form $x^{i'} y^{j's}$. With the substitution $y^s = f(x)$, we can eliminate y from the product (4.47) and write

$$u^*(x, y) = U(x, y^s) = U(x, f(x)) = V(x).$$

After substitution of $y^s = f(x)$ in U, the monomial $x^i (y^s)^{j'}$ is of degree $i + tj'$ in x. This degree satisfies

$$
\begin{aligned}
s(i + tj') \quad &= si + tj \\
&= (si_0 + tj_0) + (si_1 + tj_1) + \ldots + (si_{s-1} + tj_{s-1}) \\
&\leq m + m + \ldots + m = sm.
\end{aligned}
$$

Thus $V(x)$ is a polynomial in x of degree at most m and the zeros of $u^*(x, y)$ in $\mathcal{P}'$ can be grouped into at most m classes. A class $\mathcal{P}'_x$ with s elements contributes at most one zero of $u(x, y)$ (and at most one zero to each of the other $u(x, \zeta^k y)$, $1 \leq k \leq s - 1$). A class $\mathcal{P}'_x$ of the form $\{(x, 0)\}$ trivially contains at most one zero of $u(x, y)$. So, each of the at most m classes containing zeros of $u(x, y)$ contributes at most one such zero in $\mathcal{P}$.

$$\square$$

We can now give the parameters of AG-codes (see Definition 4.6.1).

Theorem 4.6.7 *Let $C(\mathcal{P}, m)$ be an $AG - code$ as defined in Definition 4.6.1. Let g be given by Corollary 4.6.5 and let $2g - 1 \leq m < n$. Then, $C(\mathcal{P}, m)$ has parameters $[n, m + 1 - g, n - m]$.*

Proof: The length of $C(\mathcal{P}, m)$ is simply equal to the cardinality of the set $\mathcal{P}$. Since $m < n$, Lemma 4.6.6 implies that the space $L(m)$ contains no non-zero function that is zero at all points in $\mathcal{P}$. In particular the space $L(m)$ and the code $C(\mathcal{P}, m)$ are isomorphic as vector spaces and they have the same dimension. With $m \geq 2g - 1$, the dimension now

follows from Corollary 4.6.5. The minimum distance follows immediately from Lemma 4.6.6.

$\square$

Comparison of the parameters of an AG-code with the Singleton bound (see Equation (2.17)) shows that for small values of g the AG-codes are almost optimal. Larger values of g will still yield good codes, provided that the curve $\mathcal{P}$ has many points.

Let $C^*(\mathcal{P}, m)$ denote the dual code of $C(\mathcal{P}, m)$. It is also called an AG-code. It can be described using the same set of points $\mathcal{P}$, but using a different vector space of functions. For our purposes it is enough to realize that for the description of $C^*(\mathcal{P}, m)$ (by means of a generator matrix and a parity check matrix) it is enough to have a description of $C(\mathcal{P}, m)$. We shall now derive a decoding procedure for the dual code $C^*(\mathcal{P}, m)$. From Theorem 4.6.7 we already know that $C^*(\mathcal{P}, m)$ is a q-ary code of length n and dimension $n - (m + 1 - g)$. Its minimum minimum distance will follow from the decoding procedure.

We need several lemmas. Let $\underline{r} = \underline{c} + \underline{e}$ be a received word, with $\underline{c}$ the transmitted codeword in $C^*(\mathcal{P}, m)$ and $\underline{e}$ the error vector. Let the points P_k, $1 \leq k \leq n$, in $\mathcal{P}$ have coordinates (x_k, y_k). It follows that $\sum_{k=1}^{n} c_k x_k^i y_k^j = 0$ for the pairs (i, j) with $is + jt \leq m$, $0 \leq i$, $0 \leq j < s$.

Definition 4.6.8 *Define the* degree *of a monomial* $\phi(x, y) = x^i y^j$ *by* $is + jt$ *and assume that the monomials* $\phi(x, y)$ *are put in order of increasing (see Corollary 4.6.5) degree. Thus, if the space* $L(m)$ *is of dimension* M, *we may write*

$$L(m) = \langle \phi_1 = 1, \phi_2, \ldots, \phi_M \rangle.$$

For any pair of integers (i, j), *we define the* 1st *order syndrome of the vector* $\underline{e}$ *as*

$$S_{i,j} = \sum_{k=1}^{n} e_k x_k^i y_k^j \tag{4.48}$$

and call $is + jt$ *the* degree *of* $S_{i,j}$.

For any pair of positive integers (u, v) *with* $\phi_u(x, y) = x^{i_1} y^{j_1}$ *and* $\phi_v(x, y) = x^{i_2} y^{j_2}$, *we define the* 2nd *order syndrome*

$$T_{u,v} = \sum_{k=1}^{n} e_k \phi_u(P_k) \phi_v(P_k) \tag{4.49}$$

and call $degree(\phi_u \phi_v) = (i_1 + i_2)s + (j_1 + j_2)t$ *the* degree *of* $T_{u,v}$.

It follows from this definition that for all $u, v \geq 1$

$$T_{u,v} = S_{i_1+i_2, j_1+j_2}. \tag{4.50}$$

Using Equation (4.50) one can find the $S_{i,j}$ from the $T_{u,v}$ and vice versa (take $v = 1$ to go from $T_{u,v}$ to $S_{i,j}$). Note that by (4.41) each $S_{i,j}$ with $j \geq s$ can be expressed in terms of the $S_{i,j}$'s with $j < s$.

If a 1st order syndrome $S_{i,j}$ of degree m exists it is unique if we choose it with $0 \leq i, 0 \leq j < s$. There can be several 2nd order syndromes of a given degree. Indeed, $T_{v,u}$ and $T_{u,v}$ are obviously equal to each other and there may be other pairs (u', v') with $\text{degree}(T_{u',v'}) = \text{degree}(T_{u,v})$. This fact will be exploited later.

Note that the 1st order syndrome $S_{i,j}$ of the error vector $\underline{e}$ and of the received vector $\underline{r}$ is the same for the pairs (i, j) with $x^i y^j \in L(m)$. So, from a received vector one can directly compute the following 1st and 2nd order syndromes

$$S_{i,j}, \quad \text{for } is + jt \leq m, \ 0 \leq i, \ 0 \leq j < s,$$
$$T_{u,v}, \quad \text{for } \phi_u \cdot \phi_v \in L(m).$$

These are the syndromes of degree at most m.

Lemma 4.6.9 *The error vector* $(e_1, e_2, \ldots, e_n)$ *is uniquely determined by the* 1st *order syndromes* $S_{i,j}$, *for* $i, j \in \{0, 1, \ldots, q - 2\}$. *It can be computed with an inverse* Fourier *transform.*

Proof: Let $1 \leq l \leq n$ and let the coordinates (x_l, y_l) of P_l be non-zero, say $x_l = \alpha^g, y_l = \alpha^h$. Since $\sum_{i=0}^{q-2} a^i = 0$ for $a \in GF(q) \setminus \{0, 1\}$, one has

$$\sum_{i=0}^{q-2}\sum_{j=0}^{q-2} S_{i,j} \cdot (\alpha^u)^i \cdot (\alpha^v)^j = \sum_{i=0}^{q-2}\sum_{j=0}^{q-2}\sum_{k=1}^{n} e_k x_k^i y_k^j (\alpha^u)^i (\alpha^v)^j =$$

$$= \sum_{k=1}^{n} e_k \left(\sum_{i=0}^{q-2} (x_k \alpha^u)^i \right) \left(\sum_{j=0}^{q-2} (y_k \alpha^v)^j \right)$$

$$= \begin{cases} 0 & \text{if } (u, v) \neq (-g, -h), \\ e_l & \text{if } (u, v) = (-g, -h). \end{cases}$$

We leave the case $x_l = 0$ or $y_l = 0$ as an exercise to the reader.

$\square$

It follows from this lemma that it is sufficient for the decoding of $C^*(\mathcal{P}, m)$ to determine the 1st order syndromes $S_{i,j}$ up to degree $(q-2)(s+t)$. We shall show how the syndromes $S_{i,j}$ with degree $is + jt$ in $\{m + 1, m + 2, \ldots, (q - 2)(s + t)\}$ can be determined one by one.

To obtain $S_{i,j}$ of degree $m + 1$ we consider all $T_{u,v}$ of degree $m + 1$, i.e. all $T_{u,v}$ with

$$\phi_u \cdot \phi_v \in L(m + 1) \setminus L(m). \tag{4.51}$$

To compute these $T_{u,v}$'s, the crucial step is to put them in an array.

Lemma 4.6.10 *Let the matrix* $\mathbf{T}^{(u,v)}$ *be defined as*

$$\mathbf{T}^{(u,v)} = (T_{i,j})_{1\leq i\leq u, 1\leq j\leq v}.$$

Then matrix $\mathbf{T}^{(u,v)}$ *has rank at most* $\tau = w_H(\underline{e})$.

Proof: Let $G^{(u)}$ and $G^{(v)}$ be generator matrices for codes $C(\mathcal{P}, u^*)$ of dimension u and $C(\mathcal{P}, v^*)$ of dimension v resp. (so $L(u^*) = \langle \phi_1, \phi_2, \ldots, \phi_u \rangle$, and $L(v^*) = \langle \phi_1, \phi_2, \ldots, \phi_v \rangle$). Let $\Delta(\underline{e})$ denote the matrix with $(e_1, e_2, \ldots, e_n)$ on the main diagonal and zeros elsewhere. We have by (4.49) that

$$\mathbf{T}^{(u,v)} = G^{(v)} \cdot \Delta(\underline{e}) \cdot G^{(u)T}, \tag{4.52}$$

and the rank of $\mathbf{T}^{(u,v)}$ is obviously at most equal to τ.

$\square$

We denote by $\mathbf{T}$ any sufficiently large array that contains each matrix $\mathbf{T}^{(u,v)}$ as submatrix in its upper-left corner. An upper bound on the size of $\mathbf{T}$ is $(q-2)(s+t) \times (q-2)(s+t)$.

Although the following lemma is stated for the submatrices of $\mathbf{T}$, it will be clear from the proof that it holds for the submatrices of any matrix.

Lemma 4.6.11 *Let the positive integers* u, v *satisfy*

$$rank\ \mathbf{T}^{(u-1,v-1)} = rank\ \mathbf{T}^{(u-1,v)} = rank\ \mathbf{T}^{(u,v-1)}. \tag{4.53}$$

Then, there exists a unique value for the entry $T_{u,v}$*, such that*

$$rank\ \mathbf{T}^{(u,v)} = rank\ \mathbf{T}^{(u-1,v-1)}. \tag{4.54}$$

Proof: It follows from rank $\mathbf{T}^{(u-1,v-1)}$ = rank $\mathbf{T}^{(u,v-1)}$ that coefficients α_a, $1 \leq a \leq u-1$, exist such that $T_{u,b} = \sum_{a=1}^{u-1} \alpha_a T_{a,b}$, $1 \leq b \leq v-1$. If $T_{u,v} \neq \sum_{a=1}^{u-1} \alpha_a T_{a,v}$, the rank of $\mathbf{T}^{(u,v)}$ will obviously be one more than the rank of $\mathbf{T}^{(u-1,v-1)}$. From this contradiction with (4.54) the unique choice of $T_{u,v}$ follows.

On the other hand, taking $T_{u,v} = \sum_{a=1}^{u-1} \alpha_a T_{a,v}$, we obtain rank $\mathbf{T}^{(u,v)}$ = rank $\mathbf{T}^{(u-1,v)}$ = rank $\mathbf{T}^{(u-1,v-1)}$.

$\square$

Definition 4.6.12 *A pair* (u, v) *that satisfies (4.53), but fails (4.54) is called a* discrepancy *of* $\mathbf{T}$.

A pair (u, v) *with* $T_{u,v}$ *satisfying (4.51) for which condition (4.53) holds is called a* candidate. *A candidate is called* correct *if also condition (4.54) is fulfilled. Otherwise it is called an* incorrect *candidate.*

It is easy to show (by elementary row and column operations) that no row or column of a matrix can contain two discrepancies. Another way to look at the discrepancies of $\mathbf{T}$ (or any other matrix) is given by the following lemma.

Lemma 4.6.13 *The rank of* $\mathbf{T}^{(u,v)}$ *equals the number of discrepancies among* (i,j) *with* $1 \leq i \leq u, 1 \leq j \leq v$.

A pair (u,v) *satisfies (4.53) if and only if there are no discrepancies among* (i,v), $i < u$ *and* (u,j), $j < v$.

Proof: Consider $\mathbf{T}^{(u,v)}$ (start this procedure with the empty submatrix $\mathbf{T}^{(0,0)}$). As long as rank $\mathbf{T}^{(u,v+1)}$ = rank $\mathbf{T}^{(u,v)}$, increase v by one (this argument also holds for rows). If rank $\mathbf{T}^{(u,v+1)}$ = rank $\mathbf{T}^{(u,v)} + 1$, there will be exactly one discrepancy among the positions $(i, v+1)$, $1 \leq i \leq u$. This is position $(1, v+1)$, if rank $\mathbf{T}^{(1,v)} = 0$ and rank $\mathbf{T}^{(1,v+1)} = 1$, and position $(i, v+1)$ where i is the smallest/unique value with rank $\mathbf{T}^{(i,v+1)}$ = rank $\mathbf{T}^{(i,v)} + 1$. Note that i is less than or equal to u.

The second statement now follows from the first.

$\square$

With the available information on the 2nd order syndromes it is possible to determine if the pair (u,v) is a candidate, but not if it is a correct or an incorrect candidate. On the assumption that a candidate is correct its value can be uniquely determined by Lemma 4.6.11.

In the decoding algorithm we shall consider all candidates (u,v) and simply assume that they are correct (which may not always be the case). Then we determine the syndromes $T_{u,v}$ for these candidates by means of Lemma 4.6.11 and for each we compute the corresponding $S_{i,j}$ of degree m (by (4.50)). Assuming that the number of errors τ is limited, the number of discrepancies will also be limited by Lemmas 4.6.10 and 4.6.13. Assuming that we can show that the number of correct candidates exceeds the number of incorrect candidates, we can recongnize the correct value of $S_{i,j}$ among all the computed $S_{i,j}$'s with an easy majority vote.

Lemma 4.6.14 *Let* $m \geq 4g - 2$. *The array* $\mathbf{T}$ *contains* $N := m + 2 - 2g$ *pairs* (u,v) *satisfying (4.51). No two of these pairs lie in the same row or column.*

Proof: Consider a pair (u,v) satisfying (4.51), i.e. $\phi_u \cdot \phi_v \in L(m+1) \setminus L(m)$. Clearly not both ϕ_u and ϕ_v can be in $L(2g-1)$, since then $\phi_u \cdot \phi_v \in L(4g-2) \subseteq L(m)$. Thus, by Lemma 4.6.5, we find g pairs (u,v) with $\phi_u \in L(2g-1)$ (for each of these u's there is a unique v with $\phi_u \cdot \phi_v \in L(m+1) \setminus L(m)$ because $m + 1 -$ degree $(\phi_u) \geq m + 1 - (2g-1) \geq 2g$) and, for the same reason, g pairs with $\phi_v \in L(2g-1)$. For $\phi_u, \phi_v \notin L(2g-1)$, we must have $\phi_u, \phi_v \in L(m+1-2g) \setminus L(2g-1)$. This gives another $m + 1 - 2g - (2g-1) = m + 2 - 4g$ pairs, yielding a total of $m + 2 - 2g$ pairs (u,v) satisfying (4.51).

The second assertion also follows trivially, because from degree$(T_{u,v})$ = degree$(T_{u',v'})$ it follows that $u = u'$ if and only if $v = v'$.

$\square$

Lemma 4.6.15 *Let D be the number of known discrepancies. Among the N pairs (u, v) satisfying (4.51) are at least $N - 2D$ candidates.*

Proof: From the second statement in Lemma 4.6.13 we know that each of the N pairs satisfying (4.51) is a candidate if it has no discrepancy in its row or column. By Lemma 4.6.14 no two of these N pairs occur in the same row or column. It follows that each of the D discrepancies prevents at most two pairs from being a candidate.

$\square$

Lemma 4.6.16 *Among all candidates no more than $\tau - D$ discrepancies can occur.*

Proof: The total number of discrepancies can not exceed the rank of the array $\mathbf{T}$ which is at most τ. Since D discrepancies are already known (at pairs (u, v) with $\phi_u \cdot \phi_v \in L(m)$), at most $\tau - D$ discrepancies can occur at pairs that are candidates.

$\square$

Lemma 4.6.17 *Let $m \geq 4g - 2$ (as in Lemma 4.6.14) and $2\tau < m + 2 - 2g$. Then the number of correct candidates T will exceed the number of incorrect candidates F.*

Proof: This follows from the inequality

$$
\begin{aligned}
T + F \quad &\geq \quad N - 2D = m + 2 - 2g - 2\tau + 2\tau - 2D \\
&> \quad 2\tau - 2D \geq 2F,
\end{aligned}
$$

where the first inequality follows from Lemma 4.6.15, the equality from Lemma 4.6.14, the second inequality from the condition on τ, and the last inequality from Lemma 4.6.16.

$\square$

We can now put all the steps of the decoding algorithm of AG-codes together.

Algorithm 4.6.18 (Decoding AG-codes) *Let $m \geq 4g - 2$ and let $\underline{r} = \underline{c} + \underline{e}$ be a received word, where $\underline{c} \in C^*(\mathcal{P}, m)$ and $\underline{e}$ is an error vector of weight $\tau \leq \lfloor (m+1-2g)/2 \rfloor$.*

1. *Compute the 1st order syndromes $S_{i,j}$ of $\underline{r}$ for all $is + jt \leq m$, $0 \leq i$, $0 \leq j < s$ by means of (4.48).*

2. *Compute the 2nd order syndromes $T_{u,v}$ for all (u, v) with $\phi_u \cdot \phi_v \in L(m)$ by means of (4.50).*

3. *For $m' = m + 1, m + 2, \ldots, (q - 2)(s + t)$ determine the candidates (u, v) and assume that they are all correct. For each candidate (u, v) compute $T_{u,v}$ by means of Lemma 4.6.11 and the corresponding $S_{i,j}$ of degree m' with (4.50). The correct value of $S_{i,j}$ is obtained by taking a majority decision over the obtained values. Use the correct value of $S_{i,j}$ to compute all $T_{u,v}$ of degree m' in $\mathbf{T}$.*

4. *Compute the error vector $\underline{e}$ from the 1st order syndromes $S_{i,j}$, $0 \le i,j \le q-2$, by means of Lemma 4.6.9.*

5. *Put $\underline{c} = \underline{r} - \underline{e}$.*

Theorem 4.6.19 *The code $C^*(\mathcal{P}, m)$, with $m \ge 4g-2$, has parameters $[n, n-m-1+g, \ge m+2-2g]$.*

Proof: Since $C^*(\mathcal{P}, m)$ is the dual code of the $[n, m+1-g, n-m]$ code $C(\mathcal{P}, m)$, it has length n and dimension $n - m - 1 + g$. The decoding algorithm above shows that the minimum distance is at least equal to $m + 2 - 2g$ for m odd. This can however also be shown to hold for m even.

□

Note that, just as as $C(\mathcal{P}, m)$, also $C^*(\mathcal{P}, m)$ satisfies the bound $k + d \ge n + 1 - g$ (compare this with the Singleton bound (Equation 2.17).

For large q we need to compute a large number of $S_{i,j}$'s before Lemma 4.6.17 can be applied. To obtain all the required syndromes, Step 3 in Algorithm 4.6.18 needs to be repeated many times. In this situation the following theorem is useful. It combines the procedure that we have presented with another procedure, for which we refer to the literature (see the remarks at the end of the section).

Theorem 4.6.20 *For a curve with $g \ge 1$, consider the matrix $T^{(\tau+g, \tau+g)}$. In general, not all entries can be computed with the syndromes of the received vector. We look for a linear relation among the known columns, i.e. for a solution of*

$$T^{(\tau+g, \tau+g)} \underline{a} = \underline{0},$$

where $a_i = 0$ if the i-th column of $T^{(\tau+g, \tau+g)}$ contains an unknown entry.

If no nonzero solution $\underline{a}$ exists, it suffices in Step 3 of Algorithm 4.6.18 to compute only the candidates (u, v) with $1 \le u, v \le \tau + g$. A majority decision among these will yield the syndrome of degree m'. If a nonzero solution $\underline{a}$ does exist, the algorithm can be terminated as follows.

Let $f = a_1 \phi_1 + a_2 \phi_2 + \ldots + a_{\tau+g} \phi_{\tau+g}$. Then $f(P_k) = 0$, for $e_k \ne 0$. Applying erasure decoding to the zeros of f yields the error vector as a unique solution.

We omit the full proof, but make some observations. Note that a nonzero solution $\underline{a}$ can be obtained as soon as all entries in the $\tau + 1$-th column are known. The entry $T_{\tau+g, \tau+1}$ is thus the syndrome of highest degree that needs to be known. With Corollary 4.6.5 it has degree at most $(\tau + g + g - 1) + (\tau + 1 + g - 1)$. For $2\tau < m + 2 - 2g$ the degree is bounded by $2\tau + 3g - 1 \le m + g$. Thus, indeed not all the required entries can be computed with the syndromes of the received vector, but it is only necessary to apply Step 3 in Algorithm 4.6.18 with $m' = m + 1, m + 2, \ldots, m + g$.

By then we know the first $\tau + 1$ columns of $T_{\tau+g,\tau+g}$ and we can obtain the function f. Erasure decoding will succeed if the number of zeros of f is not too large. Say f is a linear combination of $\phi, \phi_2, \ldots, \phi_l$ with $a_l \neq 0$. Thus, when this f has been found, all entries in $T^{(\tau+g,l)}$ are known. It also follows that we know all syndromes of degree up to degree($\phi_{\tau+g}\phi_l$). Erasure decoding is thus applied with respect to a code $C^*(\mathcal{P}, \hat{m})$ with $\hat{m} = \text{degree}(\phi_{\tau+g}\phi_l)$. By Theorem 4.6.7, this code has distance $\hat{d} \geq \text{degree}(\phi_{\tau+g}\phi_l) + 2 - 2g$. By Corollary 4.6.5, degree($\phi_{\tau+g}) \geq 2g$, so $d \geq \text{degree}(\phi_l) + 2$. By Lemma 4.6.6, the function f has at most degree(ϕ_l) zeros. Since this number is less than the distance $\hat{d}$, erasure decoding will yield the unique error vector.

Example 4.6.21 We consider an AG-code over $GF(16)$. In Definition 4.6.1 we take $y^5 = x^2 + x + 1$, so $s = 5$, $t = 2$ and $g = (s-1)(t-1)/2 = 2$. The set $\mathcal{P}$ contains 32 points. For $m = 13$ we obtain the code $C(\mathcal{P}, 13)$ with parameters $[32, 12, \geq 19]$ and the code $C^*(\mathcal{P}, 13)$ with parameters $[32, 20, \geq 11]$.

Let a received word for the code $C^*(\mathcal{P}, 13)$ contain five errors. Thus $\tau = w(\underline{e}) = 5$. Following Algorithm 4.6.18 and using Theorem 4.6.20, we arrive at the matrix $T^{(7,7)}$. For each entry the degree of $\phi_u\phi_v$ is given below:

$$\begin{pmatrix} 0 & 2 & 4 & 5 & 6 & 7 & 8 \\ 2 & 4 & 6 & 7 & 8 & 9 & 10 \\ 4 & 6 & 8 & 9 & 10 & 11 & 12 \\ 5 & 7 & 9 & 10 & 11 & 12 & 13 \\ 6 & 8 & 10 & 11 & 12 & 13 & 14 \\ 7 & 9 & 11 & 12 & 13 & 14 & 15 \\ 8 & 10 & 12 & 13 & 14 & 15 & 16 \end{pmatrix}.$$

Only the entries with degree 13 and less can be computed as 2nd order syndromes from the 1st order syndromes and the parity check matrix. We determine each of the unknown entries and start with 14. Among the known entries we look for discrepancies and for our example we assume the error pattern is such that they are divided as follows

$$\begin{pmatrix} 0 & * & 0 & 0 & 0 & 0 & 0 \\ * & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & * & 0 & 0 & 0 \\ 0 & 0 & * & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 14 \\ 0 & 0 & 0 & 0 & 0 & 14 & \# \\ 0 & 0 & 0 & 0 & 14 & \# & \# \end{pmatrix}.$$

The whole array contains five discrepancies (marked as $*$), so one discrepancy is missing and will occur at one of the unknown entries. The number of known discrepancies is

$D = 4$. How many entries are of degree 14? Our matrix contains three, but in the complete array there are four more in the top four rows and four more in the left four columns. A total of $N = 11$. This agrees with $N = m + 2 - 2g$ as in Lemma 4.6.14.

The next step is to determine the candidates: entries of degree 14 with no discrepancy in their row or column. The three displayed entries of degree 14 are the only candidates. Note that Lemma 4.6.15 tells us that there should be at least $N - 2D = 3$. Now we assume that none of the candidates is a discrepancy, i.e. that all of the matrices $T^{(5,7)}, T^{(6,6)}, T^{(7,5)}$ have rank four. Then the values of $T_{5,7}, T_{6,6}, T_{7,5}$ can be computed. Since we only miss one more discrepancy, at most one of the matrices $T^{(5,7)}, T^{(6,6)}, T^{(7,5)}$ has rank five and thus at most one of the values of $T_{5,7}, T_{6,6}, T_{7,5}$ is incorrect. We find at least two values among $T_{5,7}, T_{6,6}, T_{7,5}$ that agree and we take this to be the true value.

Next the entries of degree 15 can be determined and by then we can find a solution to $T^{(7,7)}\underline{a} = \underline{0}$, since the rank is five and the six known columns must be dependent.

Remarks: The generalization by V.D. Goppa that was mentioned at the beginning of this section appeared as "Codes on algebraic curves," *Soviet Math. Dokl.*, vol. 24, pp.170-172, 1981. There, Goppa defines the codes $C^*(\mathcal{P}, m)$ in terms of differentials on curves. The description of the codes $C(\mathcal{P}, m)$ in terms of functions appears in later papers. In a later paper, Tsfasman, Vlăduţ, and Zink showed that for $q \geq 49$, well chosen, infinite sequences of AG-codes exist that exceed the Gilbert-Varshamov lower bound on the size of block codes (see Theorem 2.1.9).

The use of a so-called error-locating function f in Theorem 4.6.20 is called the *basic algorithm* and was formulated in 1988 by the Danish mathematicians Justesen, Larsen, Havemose, Elbrønd Jensen and Høholdt. It was later shown by Skorobogatov and Vlăduţ that the basic algorithm applies to all AG–codes. Lemma 4.6.17 and Algorithm 4.6.18 are due to Feng and Rao. They considered a particular class of AG–codes. Duursma proved that their ideas are applicable to all AG–codes. He also showed that it suffices to consider the matrix $T^{(\tau+g,\tau+g)}$ by combining the two procedures in Theorem 4.6.20. Ehrhard suggested a different procedure that uses Theorem 4.6.20, but that avoids the determination of the unknown entries. All these results can be found in the IEEE Transactions on Information Theory.

4.7 Problems

4.7.1 Let α be a primitive element of $GF(q)$ and let $n = q - 1$. The q-ary code C of length n is defined by

$$\{(f(1), f(\alpha), \ldots, f(\alpha^{n-1})) \mid f \in GF(q)[x], \mathrm{degree}(f) < k\},$$

where $k \leq n$.

Show that C is a linear code and determine its dimension.

Show that C is a MDS code, i.e. meets the Singleton bound with equality.

Prove that C is a cyclic code.

4.7.2 Consider the factorization of $x^{11} - 1$ into irreducible factors over $GF(3)$.

How many of these factors are there and what are their degrees?

What is the smallest extension field of $GF(3)$ that contains all the zeros of $x^{11} - 1$?

Which field elements in this extension field are the zeros of the various irreducible factors of $x^{11} - 1$?

4.7.3 Let C be a binary, narrow-sense BCH code of length 93 and designed distance 13.

What are the zeros of the generator polynomial of C?

What is the dimension of C?

What does the BCH-bound state about the actual minimum distance?

4.7.4 Let C be a binary, narrow-sense BCH code of length 15 and designed distance 5.

Determine the generator polynomial of C.

4.7.5 Let α be a primitive 33-rd root of unity. What is the smallest extension field of $GF(2)$ containing α?

Let C be the cyclic code with defining set $\{0, 1\}$ with respect to α. What is the dimension of C?

What does the BCH bound say about the minimum distance of C?

Prove that this bound is tight for C.

4.7.6 Consider the binary, cyclic code of length 15 with parity check matrix

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^i & \cdots & \alpha^{14} \\ 1 & \alpha^3 & \alpha^6 & \cdots & \alpha^{3i} & \cdots & \alpha^{3 \cdot 14} \end{pmatrix},$$

where α is a zero of $x^4 + x + 1$ (see TableB.2).

What is the minimum distance of this code by the BCH-bound?

Let a received word have syndrome

$$\begin{pmatrix} \alpha^7 \\ \alpha^{14} \end{pmatrix}.$$

What is the most likely corresponding error pattern?

4.7.7 Let α be a primitive element in $GF(2^6)$. Let $C_{i,j}$ denote the binary code with parity check matrix

$$H_{i,j} = \begin{pmatrix} 1 & \alpha^i & \alpha^{2i} & \cdots & \alpha^{62i} \\ 1 & \alpha^j & \alpha^{2j} & \cdots & \alpha^{62j} \end{pmatrix},$$

i.e. $C_{i,j}$ has defining set $\{i, j\}$ with respect to α.

Prove that $C_{1,3}$ is equivalent to $C_{5,15}$ under a coordinate transformation, but not to $C_{3,9}$.

4.7.8 Let C be a q-ary cyclic code of length n (so $\gcd(q, n) = 1$) with generator polynomial $g(x)$ and parity check polynomial $h(x)$.

Since $\gcd(g(x), h(x)) = 1$ (why?), the extended version of Euclid's Algorithm yields polynomials $a(x)$ and $b(x)$ satisfying

$$a(x)g(x) + b(x)h(x) = 1.$$

Let $i(x) = a(x)g(x) = 1 - b(x)h(x)$. Prove that

a) $i(x)$ is a codeword,
b) $i(x)c(x) \equiv c(x) \pmod{x^n - 1}$ for each codeword $c(x)$ in C,
c) modulo $x^n - 1$, each codeword in C is a multiple of $i(x)$ and vice versa,
d) $i^2(x) \equiv i(x) \pmod{x^n - 1}$,
e) A polynomial satisfying a) and b) is unique modulo $x^n - 1$.

This element $i(x)$ is called the *idempotent* of C. It generates C.

4.7.9 Let C be the smallest cyclic code of length n containing a given codeword $a(x)$. Show that the generator polynomial of C is given by $g(x) = \gcd(a(x), x^n - 1)$.

4.7.10 Let C be the binary narrow-sense BCH code of length 31 with designed minimum distance 7, so its generator polynomial is $g(x) = m_1(x)m_3(x)m_5(x)$. Let α be a zero of $m_1(x) = x^5 + x^2 + 1$ (see TableB.3).

Of a received word $r(x)$ the syndrome is given by $s_1 = r(\alpha) = \alpha^{10}$, $s_3 = r(\alpha^3) = \alpha^{22}$ and $s_5 = r(\alpha^5) = \alpha^{25}$. Find the most likely error pattern in $\underline{r}$.

4.7.11 Let C be the 2^3-ary Reed-Solomon code of length 7 with minimum distance 5. Let $GF(2^3)$ be generated by α satisfying $\alpha^3 + \alpha + 1 = 0$ (see TableB.1).

Suppose the vector $\underline{r} = (\alpha^3, \alpha, 1, \alpha^2, 0, \alpha^3, 1)$ is received. Decode $\underline{r}$ with Euclid's Algorithm.

4.7.12 Compute the number g as defined in Corollary 4.6.5 for Example 4.6.2 and Example 4.6.3 and verify the claims in these examples about the dimension of the space $L(m)$.

4.7.13 Let $s = 5$ and $t = 4$. Give the dimension of the space $L(m)$, for each m in the range $\{0, 1, \ldots, 15\}$.

4.7.14 Consider the same code as in Example 4.6.21 but now with $m = 11$, so $d = 9$. For a received word the 1st order syndromes $S_{i,j}$ are computed and the corresponding part of $\mathbf{T}$ is given by

$$
\mathbf{T} =
$$

	v	1	2	3	4	5	6	7	8	9	10	11
u		0	2	4	5	6	7	8	9	10	11	12
1	0	0	1	0	0	0	1	1	0	0	0	
2	2	1	0	0	1	1	0	0	0			
3	4	0	0	1	0	0	0					
4	5	0	1	0	0	0						
5	6	0	1	0	0							
6	7	1	1	0								
7	8	1	0									
8	9	0	0									
9	10	0										
10	11	0										
11	12											

where the second row and column indices denote the degree of the corresponding ϕ_u and ϕ_v. (The fact that all the entries above have binary values can not be expected in general, but is done to keep the calculations manageable.)

1. Determine the known discrepancies and what is their number D?

2. Determine the pairs (u, v) with $T_{u,v}$ of degree 12. Which $S_{i,j}$ can be computed from these $T_{u,v}$'s?

3. Determine the candidates and compute the corresponding $T_{u,v}$'s under the assumption that they are correct.

4. Determine the corresponding estimates of $S_{i,j}$. What does the majority decision yield for $S_{i,j}$.

5. Determine the value of $T_{u,v}$ for all $T_{u,v}$ of degree 12.

4.7.15 Consider the codes $C(\mathcal{P}, m)$ in Example 4.6.3, for $1 \le m \le 8$. Show that for $1 \le i \le 8$

$$
\sum_{P \in \mathcal{P}} \phi_i(P) = 0.
$$

Use this to prove that for $1 \leq m \leq 7$,

$$C^*(\mathcal{P}, m) = C(\mathcal{P}, 8 - m).$$

Chapter 5

Burst correcting codes

5.1 Introduction; two bounds

In many applications, especially those where data are stored on a magnetic medium, errors tend to occur in clusters rather than completely independently of each other.

Definition 5.1.1 *In $V_n(q)$ a vector $\underline{e}$ is called a* burst *of length b if it has the form*

$$(0, 0, \ldots, 0, \overbrace{e_i, e_{i+1}, \ldots, e_{i+b-1}}^{b}, 0, 0, \ldots, 0), \tag{5.1}$$

where $e_i \neq 0$ and $e_{i+b-1} \neq 0$.

The segment $(e_i, e_{i+1}, \ldots, e_{i+b-1})$ is called the burst pattern. *Also, one says that this burst starts at coordinate i.*

Codes that can correct any burst of length up to b are called *b-burst-correcting.* In this chapter we shall only be interested in linear b-burst-correcting codes.

Lemma 5.1.2 *A linear code C is b-burst-correcting if and only if all bursts of length up to b have distinct syndromes.*

Proof: Clearly, if distinct bursts have distinct syndromes, one can correct them.

On the other hand, if bursts $\underline{b}_1$ and $\underline{b}_2$ of length at most b have the same syndrome, their difference $\underline{b}_1 - \underline{b}_2$ will be a codeword! The received word $\underline{b}_1$ can now be written as $\underline{c}_1 + \underline{b}_1$ with $\underline{c}_1 = \underline{0}$ in C, but also as $\underline{c}_2 + \underline{b}_2$ with $\underline{c}_2 = \underline{b}_1 - \underline{b}_2$ also in C. So C is not b-burst-correcting.

$\square$

Lemma 5.1.3 *Let C be a linear b-burst-correcting code with parity check matrix H. Then any two disjoint groups of b consecutive columns of H consist of $2b$ independent vectors.*

Proof: If not, there exists a non-trivial dependency of the $2b$ columns, corresponding to the two disjoint groups of b consecutive coordinates. This dependency can be represented by two distinct bursts of length b each, say $\underline{b}_1$ and $\underline{b}_2$.

The dependency now reads like $H(\underline{b}_1 + \underline{b}_2)^T = \underline{0}^T$. This implies that the bursts $\underline{b}_1$ and $-\underline{b}_2$ have the same syndrome. This contradicts Lemma 5.1.2.

$\square$

An immediate consequence of Lemma 5.1.3 is the following bound.

Theorem 5.1.4 (Reiger bound) *Let C be a q-ary, k-dimensional, b-burst-correcting code of length n. Then the redundancy $r = n - k$ satisfies*

$$r \geq 2b, \tag{5.2}$$

so $|C| \leq q^{n-2b}$.

In the sequel we shall often refer to the Reiger bound.

Many of the b-burst-correcting codes that will be constructed further on will be cyclic. It turns out that they are able to correct all cyclic shifts of bursts of length up to b, in particular those that start near the end and that end somewhere at the beginning of the word, i.e. error patterns of the form:

$$(\overbrace{e_0, \ldots, e_{b-(n-i)-1}}^{b-(n-i)}, 0, 0, \ldots \ldots, 0, 0, \overbrace{e_i, \ldots, e_{n-1}}^{n-i}), \tag{5.3}$$

where $i > n - b$, $e_i \neq 0$ and $e_{b-(n-i)-1} \neq 0$.

Error vectors of the form (5.3) are called *cyclic bursts* and cyclic codes that can correct all cyclic bursts of length up to b are called *cyclic b-burst-correcting*.

Again Lemma 5.1.2 can be applied to derive a bound.

Theorem 5.1.5 (Abramson bound) *Let C be a q-ary, cyclic b-burst-correcting code of length n. Then its redundancy r satisfies*

$$n \leq \frac{q^{r-b+1} - 1}{q - 1}. \tag{5.4}$$

Proof: Let us count the number of distinct non-zero cyclic bursts. Each one is defined by its starting coordinate (n possibilities), a non-zero entry there ($q-1$ choices) followed by $b - 1$ arbitrarily chosen coordinates. So their number is $n(q - 1)q^{b-1}$. Now all these vectors and also the $\underline{0}$-vector should have distinct syndromes. So

$$q^r \geq 1 + n(q - 1)q^{b-1}.$$

Dividing this by q^{b-1} yields

$$q^{r-b+1} \geq \frac{1}{q^{b-1}} + n(q - 1)$$

c_{11}	c_{12}	$\cdots$	$\cdots$	c_{1n}	codeword $\underline{c}^{(1)}$
c_{21}	c_{22}	$\cdots$	$\cdots$	c_{2n}	codeword $\underline{c}^{(2)}$
$\vdots$	$\vdots$			$\vdots$	$\vdots$
c_{t1}	c_{t2}	$\cdots$	$\cdots$	c_{tn}	codeword $\underline{c}^{(t)}$
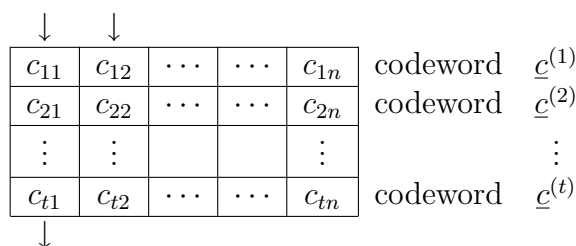
Figure 5.1: Interleaving C at depth t.

and thus

$$q^{r-b+1} - 1 \geq n(q-1).$$

$\square$

Cyclic, b-burst-correcting codes for which (5.4) holds with equality are called *optimum*. They will be discussed in Section 5.5.

5.2 Two standard techniques

In this section two very widely used constructions will be given of binary, b-burst-correcting codes. Both make use of error-correcting codes.

The first technique changes the order of the coordinates of several consecutive codewords in such a way that a burst is spread out over the various codewords.

Let C be a code of length n and let t be some positive integer. Consider all $t \times n$ matrices which have codewords in C as their rows. Read these matrices out columnwise from top to bottom starting with the leftmost column. The resulting code will be denoted by $Int_t(C)$; its length is tn. The process of constructing $Int_t(C)$ out of C is called *interleaving C* at *depth t*. In Figure 5.1 the interleaving technique is depicted for the t codewords $\underline{c}^{(i)} = (c_{i1}, c_{i2}, \ldots, c_{in})$, $1 \leq i \leq t$.

Quite obviously a burst of length b in the interleaved code will affect each row of the matrix (so each codeword $c^{(i)}$) in at most $\lceil b/t \rceil$ consecutive coordinates. This proves the following theorem.

Theorem 5.2.1 *Let C be a b-burst-correcting code in V_n and let $Int_t(C)$ be the interleaved code with depth t. Then $Int_t(C)$ is a bt-burst-correcting code of length tn.*

Sometimes one simply takes an e-error correcting code for C. In this way, one can also correct more bursts in $Int_t(C)$, say of lengths B_j, $1 \leq j \leq l$, as long as $\sum_{j=1}^{l} \lceil B_j/t \rceil \leq e$.

It turns out that interleaving a cyclic code of length n results in a code that is again cyclic.

Theorem 5.2.2 *Let C be a b-burst-correcting, cyclic code of length n and let $Int_t(C)$ be obtained from C by interleaving it at depth t. Then $Int_t(C)$ is a cyclic code of length tn.*

Proof: Let $g(x)$ be the generator matrix of C. We shall show that $g(x^t)$ is the generator matrix of $Int_t(C)$.

Take a codeword in $Int_t(C)$ and let the rows of the corresponding $t \times n$ matrix be $c^{(i)}(x) = a^{(i)}(x)g(x)$, $1 \le i \le t$.

The interleaved word $c(x)$ is given by

$$
\begin{aligned}
c(x) &= c_{11} + c_{21}x + \cdots + c_{t1}x^{t-1} + c_{12}x^t + \cdots + c_{t,n}x^{tn-1} \\
&= c^{(1)}(x^t) + xc^{(2)}(x^t) + \cdots + x^{t-1}c^{(t)}(x^t) \\
&= a^{(1)}(x^t)g(x^t) + xa^{(2)}(x^t)g(x^t) + \cdots + x^{t-1}a^{(t)}(x^t)g(x^t) \\
&= \left(a^{(1)}(x^t) + xa^{(2)}(x^t) + \cdots + x^{t-1}a^{(t)}(x^t) \right) g(x^t) \\
&= a(x)g(x^t),
\end{aligned}
$$

so it is indeed in the cyclic code of length tn generated by $g(x^t)$.

Clearly the reverse is also true: each codeword $a(x)g(x^t)$ in the code generated by $g(x^t)$ can be written as

$$
\left(a^{(1)}(x^t) + xa^{(2)}(x^t) + \cdots + x^{t-1}a^{(t)}(x^t) \right) g(x^t)
$$

and can thus be obtained by interleaving t codewords in C.

$\square$

The second technique of making b-burst-correcting codes is to start with an e-error-correcting code C of length n over $GF(2^m)$ and write each coordinate as a binary vector by using a binary basis of $GF(2^m)$. In this way one obtains a binary code of length nm with the same cardinality as C. A burst of length $(e-1)m+1$ will affect at most e (consecutive) m-tuples, so at most e coordinates of the code C. In other words, we have obtained a $(e-1)m+1$-burst-correcting code.

Most often a (shortened) Reed-Solomon code or an extended Reed-Solomon code is used for this purpose, because these codes meet the Singleton bound (see (2.17) for $b = (d-1)/2$ independent errors with equality. Note that they can correct more bursts of shorter lengths too. Also note that they meet the Reiger bound (see (5.2) for bursts of length $b = (d-1)/2$ with equality.

Note that this construction is a special case of the concatenated code construction (see Theorem 3.1.7). The inner code simply is the $[m, m, 1]$ code V_m.

The two constructions in this section may be combined to obtain codes that can correct even longer bursts.

5.3 Fire codes

In this section a class of cyclic, b-burst-correcting codes will be constructed by more algebraic methods.

The burst pattern and starting point of a cyclic burst $b(x)$ of length b can easily be denoted by $B(x)$ and i, $0 \leq i \leq n - 1$, satisfying

$$b(x) = B(x)x^i, \quad B(0) \neq 0, \quad \text{degree}(B(x)) \leq b - 1. \tag{5.5}$$

Let the *period* of a polynomial $f(x)$ with $f(0) \neq 0$ be defined as the smallest positive integer v for which $f(x)$ divides $x^v - 1$.

Definition 5.3.1 *The q-ary* Fire code *is a cyclic code in $V_n(q)$ with generator polynomial $g(x) = (x^{2b-1} - 1)f(x)$, where*

1. *$f(x)$ is an irreducible polynomial of degree m with $m \geq b$,*

2. *$f(x)$ does not divide $x^{2b-1} - 1$,*

3. *n is the smallest positive integer such that $g(x)$ divides $x^n - 1$.*

Note, that the second condition in Definition 5.3.1 is equivalent to

2'. $\gcd(f(x), x^{2b-1} - 1) = 1$,

because $f(x)$ is irreducible.

Lemma 5.3.2 *The Fire code, defined in Definition 5.3.1, has length $n = lcm[v, 2b - 1]$, where v is the period of $f(x)$.*

Proof: Let v be the period of $f(x)$.

It follows from 2'. that the statement $g(x)$ divides $x^n - 1$ is equivalent to the statement that both $f(x)$ and $x^{2b-1} - 1$ divide $x^n - 1$. This in turn is equivalent to saying that both v and $2b - 1$ must divide n. However, this is equivalent to saying that $lcm[v, 2b - 1]$ divides n. Since n was chosen to be minimal, the Lemma now follows.

$\square$

Theorem 5.3.3 *The Fire code, defined in Definition 5.3.1, is b-burst-correcting.*

We shall give a proof of this theorem by deriving a b-burst-correcting decoding algorithm for the Fire code.

Let $r(x)$ be a received word and assume that $r(x) = c(x) + b(x)$, where $c(x)$ is a codeword in the Fire code and $b(x)$ a non-zero burst of length at most b. Determine $s_1(x)$ and $s_2(x)$, defined by:

$$s_1(x) \equiv r(x) \equiv b(x) \pmod{x^{2b-1} - 1}, \deg ree(s_1(x)) < 2b - 1. \tag{5.6}$$

$$s_2(x) \equiv r(x) \equiv b(x) \pmod{f(x)}, \quad \text{degree}(s_2(x)) < m. \tag{5.7}$$

Writing $b(x) = x^i B(x)$ as in (5.5), it follows that

$$s_1(x) = x^{i'} B(x),$$

where $i \equiv i' \pmod{2b-1}$, $0 \leq i' < 2b - 1$ and the exponents have to be taken modulo $2b - 1$. The nice thing now is that i' and $B(x)$ uniquely follow from $s_1(x)$. Indeed, let l be the longest gap in $s_1(x)$, when the exponents are viewed cyclically modulo $2b - 1$ (a *gap* of length l is a sequence of l consecutive zero-coordinates, bordered on both sides by a non-zero element). Because $B(x)$ has length at most b, this l will be at least $(2b - 1) - b = b - 1$ but possibly more. However, there cannot be two distinct gaps of length at least $b - 1$, because $(b - 1) + 1 + (b - 1) + 1 = 2b$ which is more than the $2b - 1$ different exponents we are considering. Note, that this gap of length at least $b - 1$ will end at coordinate $i' - 1$ and that $B(x)$ is given by $x^{-i'} s_1(x) \pmod{x^{2b-1} - 1}$.

So, of the burst $b(x) = x^i B(x)$ that we want to determine, we know already $B(x)$ and $i' \equiv i \pmod{2b-1}$. Write $i = i' + j(2b - 1)$.

We know that

$$s_2(x) \equiv x^{i'+j(2b-1)} B(x) \pmod{f(x)}, \quad 0 \leq j < \frac{v}{\gcd(v, 2b-1)}.$$

The question that remains is: does this relation uniquely determine j modulo $v/\gcd(v, 2b-1)$? The answer is affirmative.

Indeed, by the irreducibility of $f(x)$ and since $B(x) \neq 0$, if $x^{i'+j(2b-1)} B(x) \equiv x^{i'+j'(2b-1)} B(x) \pmod{f(x)}$, then also $(x^{(j-j')(2b-1)} - 1) B(x)$ will be divisible by $f(x)$. Since $f(x)$ has degree at least b, one even has that $(x^{(j-j')(2b-1)} - 1)$ must be divisible by $f(x)$. So v divides $(j - j')(2b - 1)$. It follows that $j \equiv j' \pmod{v/\gcd(v, 2b-1)}$.

We conclude that j can be found by evaluating $x^{i'+j(2b-1)} B(x)$ modulo $f(x)$ for $j = 0, 1, \cdots, v - 1$ until it is equal to $s_2(x)$.

The above method of trying out $j = 0, j = 1$, and so on, can be replaced by a single calculation. Let α be a zero of $f(x)$, so α has order v. Since $B(x)$ has degree less than m and $f(x)$ is irreducible, it follows that $B(\alpha) \neq 0$. So, the value of j modulo $v/\gcd(v, 2b-1)$ can now directly be computed from

$$\alpha^{i'+j(2b-1)} B(\alpha) = s_2(\alpha), \tag{5.8}$$

Of course the above calculation has to be performed in $GF(q^m)$, an extension field of $GF(q)$ that contains the zeros of $f(x)$. Also, note that $(2b - 1)j$ is uniquely determined modulo $(2b-1)v/\gcd(v, 2b-1) = \text{lcm}[v, 2b-1] = n$. This proves that also the burst-start is uniquely determined from the syndrome. In other words, we have shown that the Fire code generated by $g(x) = (x^{2b-1} - 1)f(x)$ is indeed b-burst-correcting.

We summarize the above decoding algorithm.

Algorithm 5.3.4 (Decoding Fire codes) *Let $r(x)$ be the received word.*

1. *Compute the syndrome $s_1(x)$ and $s_2(x)$ from (5.6) and (5.7).*
   If $s_1(x) = 0$ and $s_2(x) = 0$, put $c(c) = r(x)$ and STOP.

2. *Determine the burst pattern $B(x)$ and the starting point i modulo $2b-1$, denoted by i', from the unique gap of length at least $b-1$ in $s_1(x) \pmod{x^{2b-1}-1}$.*

3. *Find $j \pmod{v/\gcd(v, 2b-1)}$ satisfying (5.8).*

4. *Put $c(x) = r(x) - x^{i'+j(2b-1)}B(x)$.*

Fire codes have redundancy $r = m + 2b - 1 \geq 3b - 1$. Compare this with the Reiger bound $r \geq 2b$.

Example 5.3.5 Consider the binary Fire code with $b = 3$, generated by $g(x) = (x^5 - 1)f(x)$ where $f(x)$ is the primitive polynomial $1 + x + x^3$. Since $f(x)$ has period 7, this Fire code has length $n = 5 \times 7 = 35$ and dimension $k = 35 - 5 - 3 = 27$.

Now, let $s_1(x) = 1 + x^3$ and $s_2(x) = 1 + x$ be the syndrome of a received word $r(x)$. Following Decoding Algorithm 5.3.4, we find the gap of length at least 2 in $s_1(x)$ at $0x + 0x^2$. So the burst starts at $i = 3 + j5$ and has pattern $1 + x^2$, because $x^3(1 + x^2) \equiv 1 + x^3 \pmod{x^5 - 1}$. To find j one has to solve

$$\alpha^{3+j5}(1 + \alpha^2) = 1 + \alpha.$$

Now $1 + \alpha^2 = \alpha^6$ and $1 + \alpha = \alpha^3$, so j follows from $3 + j5 + 6 \equiv 3 \pmod{7}$, which has $j = 3$ as solution.

We conclude that the burst in $r(x)$ is given by $x^{18}(1 + x^2)$.

Example 5.3.6 Consider the binary Fire code with $b = 4$, generated by $g(x) = (x^7 - 1)f(x)$ with $f(x) = 1 + x + x^2 + x^4 + x^6$. Note that $f(x)$ has period 21, so this Fire code has length $n = \text{lcm}[7, 21] = 21$.

Let $s_1(x) = x + x^3 + x^4$ and $s_2(x) = x^3 + x^5$ be the syndrome of a received word $r(x)$. Following Decoding Algorithm 5.3.4, we find (modulo $x^7 - 1$) the gap of length at least 3 in $s_1(x)$ at $0x^5 + 0x^6 + 0x^0$. So the burst starts at $i = 1 + j7$ and has pattern $1 + x^2 + x^3$.

Consider $GF(2^6) = GF(2)[x]/(p(x))$, where $p(x) = 1 + x + x^6$ is a primitive polynomial. Let ω be a zero of $p(x)$. One can easily check that $\alpha = \omega^3$ is a zero of $f(x)$.

To find j one has to solve

$$\alpha^{1+j7}(1 + \alpha^2 + \alpha^3) = \alpha^3 + \alpha^5.$$

Now $1 + \alpha^2 + \alpha^3 = 1 + \omega^6 + \omega^9 = \omega^{49}$ and $\alpha^3 + \alpha^5 = \omega^9 + \omega^{15} = \omega^{10}$, so j follows from

$$3(1 + j7) + 49 \equiv 10 \pmod{63},$$

which has $j \equiv 1 \pmod{3}$ as solution.

We conclude that the burst in $r(x)$ is given by $x^8(1 + x^2 + x^3)$.

5.4 Array codes

The interleaving technique in Theorem 5.2.1 shows that one can get burst-correcting codes by filling a matrix rowwise and reading it out columnwise. In this section we consider binary matrices with even-weight rows and columns and obtain a burst-correcting code by reading the entries out in a particular way.

Definition 5.4.1 *Let n_1 and n_2 be two positive integers with $n_2 \geq n_1$. The set of all binary $n_1 \times n_2$ matrices with even row and column sums is called the (n_1, n_2) array code $\mathcal{A}(n_1, n_2)$.*

Clearly, $\mathcal{A}(n_1, n_2)$ is a linear, even-weight code of length $n_1 n_2$ and dimension $(n_1-1)(n_2-1)$. For instance, the bits in the upper left $(n_1 - 1) \times (n_2 - 1)$ submatrix can be viewed as a set information symbols. The other bits can be uniquely determined from the even parity sums of the rows and columns.

It is also obvious that $\mathcal{A}(n_1, n_2)$ cannot contain a word of weight 2, but does contain weight 4 codewords (for instance fill the upper left 2×2 submatrix with ones) so $\mathcal{A}(n_1, n_2)$ is a $[n_1 n_2, (n_1 - 1)(n_2 - 1), 4]$ code.

In terms of redundancy, $\mathcal{A}(n_1, n_2)$ is a poor 1-error-correcting code, the decoding of a single error on the other hand is extremely simple. Indeed let $\underline{r}$ be a received vector and let R be the corresponding $n_1 \times n_2$ matrix. Let $R = C + E$, with C in $\mathcal{A}(n_1, n_2)$ and let E have a one at position (s, t), so $E_{st} = 1$, and zeros elsewhere.

Compute the horizontal syndrome $\underline{h}$ defined by $h_i = \sum_{1 \leq j \leq n_2} R_{ij} = \sum_{1 \leq j \leq n_2} E_{ij}$, $1 \leq i \leq n_1$ and the vertical syndrome $\underline{v}$ defined by $v_j = \sum_{1 \leq i \leq n_1} R_{ij} = \sum_{1 \leq i \leq n_1} E_{ij}$, $1 \leq j \leq n_2$.

It follows that (s, t) is uniquely determined from the syndrome

$$\underline{h} = (0, 0, \ldots, 0, \overset{s}{1}, 0, \ldots, 0)$$

and

$$\underline{v} = (0, 0, \ldots, 0, \overset{t}{1}, 0, \ldots, 0).$$

To study the burst-correcting capability of the array codes, one first needs to define a particular read out. Here, we shall only study the diagonalwise read out, starting at the top-left position, each diagonal followed by the adjacent one to the right, while the column indices have to be taken modulo n_2. This read out will be called the $+1$-read out. In Figure 5.2 the $+1$-read out is depicted for the $\mathcal{A}(5, 6)$ code.

Other read outs are of course possible. In particular the $+s$-read out, where each diagonal is followed by the diagonal s further to the right (and where $\gcd(s, n_2) = 1$), can be found in the literature.

So for an element in $\mathcal{A}(n_1, n_2)$ we now have two representations: The matrix $(A_{ij})_{1 \leq i \leq n_1, 1 \leq j \leq n_2}$ and the vector $\underline{a} = (a_0, a_1, \ldots, a_{n-1})$, with $n = n_1 n_2$.

0	5	10	15	20	25
26	1	6	11	16	21
22	27	2	7	12	17
18	23	28	3	8	13
14	19	24	29	4	9

Figure 5.2: The read out for $\mathcal{A}(5,6)$.

One can easily check that they are related in the following way

$$A_{ij} = a_{n_1(j-i)+i-1}, \tag{5.9}$$

where the indices of the coordinates of $\underline{a}$ have to be taken modulo n.

The array code $\mathcal{A}(n_1, n_2)$ with the $+1$-read is not able to correct all bursts of length up to n_1. To see this, consider the syndrome $\underline{v} = \underline{0}$ and $\underline{h} = (1, 1, 0, 0, \ldots, 0)$. The error pattern $E^{(j)}$ defined by $E_{1j} = E_{2j} = 1$ and zeros elsewhere will yield this syndrome for all $1 \leq j \leq n_2$. However, this error pattern is a burst of length n_1, starting (with a 1) at position $(2, j)$ and with burst pattern $(1, \overbrace{0, 0, \cdots, 0}^{n_1-2}, 1)$. For instance, ones at coordinates 1 and 5 in Figure 5.2 and zeros elsewhere corresponds to a burst of length 5. Its syndrome is identical to the burst of length 5 with its ones at coordinates 6 and 10.

Lemma 5.4.2 *Let $\mathcal{A}(n_1, n_2)$, $n_2 > n_1$, be the array code with the $+1$-read out. If $n_2 \leq 2n_1 - 4$, then $\mathcal{A}(n_1, n_2)$ is not able to correct all bursts of length up to $n_1 - 1$.*

Proof: It suffices to give two distinct bursts of length at most $n_1 - 1$ which have the same syndrome.

Consider the burst of length b, $2 \leq b \leq n_1 - 1$, starting at coordinate 0, so at position $(1, 1)$, and with pattern $(1, \overbrace{0, 0, \cdots, 0}^{b-2}, 1)$. So, its second 1 is at coordinate $b - 1$, i.e. at position (b, b). The syndrome of this burst can also be obtained by putting a one at position $(1, b)$ and a second 1 at position $(b, 1)$.

These two positions correspond by (5.9) to the coordinates $(1 - b)n_1 + b - 1 = n_1 n_2 - (b - 1)n_1 + (b - 1)$ and $(b - 1)n_1$. The question to be answered is: is there a value for b, $2 \leq b \leq n_1 - 1$, for which these two coordinates are in a burst of length at most $n_1 - 1$.

Now the distance between these two coordinates u and v is $\min\{v - u, u - v\}$, where the differences have to be computed modulo n. So we want to find b, $2 \leq b \leq n_1 - 1$, for which

$$n_1 n_2 - b + 1 \leq (2n_1 - 1)(b - 1) \leq n_1 n_2 + b - 1.$$

This reduces to

$$\frac{n_2 + 2}{2} \le b \le \frac{n_2 + 2}{2} + \frac{n_2}{2(n_1 - 1)}.$$

From $n_1 < n_2$, it follows that $n_2/(2(n_1 - 1)) \ge 1/2$, so $b = \lceil (n_2 + 2)/2 \rceil$ is a solution to these inequalities.

Moreover, $n_2 \le 2n_1 - 4$ implies that $(n_2 + 2)/2 \le n_1 - 1$, so $b = \lceil (n_2 + 2)/2 \rceil$ also satisfies the condition $b \le n_1 - 1$.

$\square$

Lemma 5.4.2 gives a necessary condition for the array code $\mathcal{A}(n_1, n_2)$, $n_2 > n_1$, to be $(n_1 - 1)$-burst-correcting. This condition turns out to be sufficient.

Theorem 5.4.3 *The array code $\mathcal{A}(n_1, n_2)$, $n_2 > n_1$, is $(n_1 - 1)$-burst-correcting if and only if $n_2 \ge 2n_1 - 3$.*

Instead of proving Theorem 5.4.3 directly we shall derive an extremely simple decoding algorithm. In view of Lemma 5.4.2 we may assume that $n_2 \ge 2n_1 - 3$.

Let $R = C + B$ be a received word, where C is a codeword and B be a burst of length at most $n_1 - 1$. Compute the syndromes $\underline{h}$ and $\underline{v}$ of R, which is the same as the syndromes of the burst B. We only need to discuss the case that $\underline{h}$ and $\underline{v}$ are not both equal to the all-zero vector.

It follows from the $+1$-read out that a burst of length n_1 will affect each row or column at most once. In other words, in the computation of the syndrome there will never be a cancelation of errors! Or, to say it differently, if $h_i = 1$, there is exactly one error in the i-th row and a row contains no errors if $h_i = 0$. The same holds for the columns. It follows from this observation that $\underline{h}$ and $\underline{v}$ have the same Hamming weight, say w.

Next, view the coordinates of $\underline{v}$ cyclicly. Since the burst B affects at most $n_1 - 1$ consecutive columns, $\underline{v}$ will contain a gap of zeros of length at least $n_2 - b \ge n_2 - n_1 + 1$. But there cannot be two gaps of length at least $n_2 - n_1 + 1$, because this would imply a total number of coordinates of at least $(n_2 - n_1 + 1) + 1 + (n_2 - n_1 + 1) + 1$, which is more than n_2 under the condition $n_2 \ge 2n_1 - 3$.

Let the coordinate immediately to the right of the gap be j_1 (so $v_{j_1} = 1$ and $v_{j_1 - l} = 0$ for $1 \le l \le (n_2 - n_1 + 1)$ and let the subsequent 1-coordinates in $\underline{v}$ have indices $j_2, j_3, \ldots, j_w$. Let $1 \le i_1 < i_2 < \cdots < i_w \le n_1$ denote the 1-coordinates in $\underline{h}$.

We shall now show that the errors in B are at positions (i_l, j_l), $1 \le l \le w$. It suffices to show that $E_{i_1, j_1} = 1$.

Assume the contrary, so assume that $E_{i_1, j_1} = 0$. It follows from $h_i = 0$ for $i < i_1$ that the error in column j_1 is at a position (u, j_1) with $u > i_1$. Similarly it follows from $v_{j_1 - l} = 0$ for $1 \le l \le (n_2 - n_1 + 1)$ that the error in row i_1 is at position (i_1, v) with

$j_1 < v \leq j_1 + n_1 - 2$, So, $E_{(i_1,j_1)} \neq 1$ implies that $E_{(u,j_1)} = 1$ for some $u \geq i_1$ and that $E_{(i_1,v)} = 1$ for some $j_1 \leq v \leq j_1 + n_1 - 2$.

The proof will be finished by showing that the positions (u, j_1) (i_1, v) under these conditions will never lie in the same burst of length at most $n_1 - 1$. The reason for this is that the distance between these positions is at least $n_1 - 1$. Because the coordinates are taken modulo $n_1 n_2$, one needs to compute the minimum of two distances.

By (5.9)), these distances are

$$|n_1 n_2 - \{(v - i_1)n_1 + i_1 - 1\} + \{(j_1 - u)n_1 + u - 1\}| =$$

$$|n_1 n_2 - (v - j_1)n_1 - (u - i_1)(n_1 - 1)|.$$

and

$$|(v - j_1)n_1 + (u - i_1)(n_1 - 1)|.$$

The second expression is clearly always greater than or equal to $n_1 - 1$. To prove the same inequality for the first expression, we use that the inequalities on u and v imply that $v - j_1 \leq n_1 - 2$ and $u - i_1 \leq n_1 - 1$. So the first distance between the two positions (u, j_1) and (i_1, v) is at least

$$n_1 n_2 - (n_1 - 2)n_1 - (n_1 - 1)^2,$$

which, by the inequality $n_2 \geq 2n_1 - 3$, is at least

$$n_1(2n_1 - 3) - (n_1 - 2)n_1 - (n_1 - 1)^2 = n_1 - 1.$$

So the two positions (u, j_1) and (i_1, v) with $u > i_1$ and $j_1 < v \leq j_1 + n_1 - 2$ have actual distance at least $n_1 - 1$, so they are not in the same burst of length at most $n_1 - 1$.

This proves the correctness of the following decoding algorithm.

Algorithm 5.4.4 (Decoding array codes) *Let $R(x)$ be the received word.*

1. *Compute the horizontal syndrome $\underline{h}$ defined by $h_i = \sum_{1 \leq j \leq n_2} R_{ij} = \sum_{1 \leq j \leq n_2} E_{ij}$, $1 \leq i \leq n_1$ and the vertical syndrome $\underline{v}$ defined by $v_j = \sum_{1 \leq i \leq n_1} R_{ij} = \sum_{1 \leq i \leq n_1} E_{ij}$, $1 \leq j \leq n_2$.*

2. *Let $1 \leq i_1 < i_2 < \cdots < i_w \leq n_1$ denote the 1-coordinates in $\underline{h}$.*

3. *Find the gap of length at least $n_2 - n_1 + 1$ in $\underline{v}$, where the coordinates have to be taken modulo n_2. Let the gap end at coordinate $j_1 - 1$.*

4. *Let $j_1 < j_2 < \cdots < j_w \leq j_1 + n_1 - 2$ denote the 1-coordinates in $\underline{v}$.*

5. *The burst $B(x)$ is one at the positions $(i_l, j_l), 1 \leq l \leq w$ and zero everywhere else.*

6. *$C(x) = R(x) + B(x)$.*

We shall demonstrate this algorithm in an example. Let $n_1 = 5, n_2 = 7$ and let the received word be given by

1	1	1	0	0	1	1
0	1	1	0	0	1	0
1	0	0	1	1	1	0
0	1	1	0	1	1	1
0	1	0	0	1	0	0

We follow Algorithm 5.4.4

1. $\underline{h} = (1, 1, 0, 1, 0)$ and $\underline{v} = (0, 0, 1, 1, 1, 0, 0)$.

2. $i_1 = 1, i_2 = 2, i_3 = 4$.

3. $\underline{v}$ has the gap of length at least 3 at coordinates $6, 7, 1$ and 2. So, $j_1 = 3$.

4. $j_1 = 3, j_2 = 4, j_3 = 5$.

5. The burst $B(x)$ is one at the positions $(1, 3), (2, 4), (4, 5)$.

So the burst is given by

0	0	1	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	1	0	0
0	0	0	0	0	0	0

This burst starts at position $(4, 5)$, which is coordinate 8 and has pattern $(1, 0, 1, 1)$.

5.5 Optimum, cyclic, b-burst-correcting codes

In this section we discuss the existence of optimum, cyclic binary b-burst-correcting codes, see Section 5.1. These codes have length $n = 2^m - 1$ and redundancy $r = m + b - 1$. From the Reiger bound it follows that $m \geq b + 1$.

Let us first discuss some small cases.

An optimum, cyclic, 1-burst-correcting code: $n = 2^m - 1, r = m, m \geq 2$.

Since $b = 1$, we are really looking for a 1-error-correcting code of length $n = 2^m - 1$. From Theorem 4.2.1 we know that this code has parity check matrix

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \cdots & \cdots & \alpha^{n-1} \end{pmatrix},$$

where α is a primitive element in $GF(2^m)$. Let $p(x)$ be the minimal polynomial of α. It follows that the code has as generator polynomial the primitive polynomial $p(x)$.

An optimum, cyclic, 2-burst-correcting code: $n = 2^m - 1, r = m + 1, m \geq 3$.

Take again a primitive polynomial $p(x)$ of degree m and consider the cyclic code with generator polynomial $g(x) = (1 + x)p(x)$. Clearly, this code has the right redundancy. It can correct every burst of length at most 2, as we shall now show. Let α be a zero of $p(x)$. The parity check matrix H of this code has the form

$$H = \begin{pmatrix} 1 & 1 & 1 & \cdots & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \cdots & \alpha^{n-1} \end{pmatrix}.$$

The syndrome of the burst $b(x)$ of length 1 starting at coordinate i (so $b(x) = x^i$) is simply $s_0 = b(1) = 1$, and $s_1 = b(\alpha) = \alpha^i$.

A burst $b(x)$ of length 2 starting at coordinate i must have the form $b(x) = x^i + x^{i+1}$, so it has syndrome $s_0 = b(1) = 0$, and $s_1 = b(\alpha) = \alpha^i + \alpha^{i+1} = \alpha^i(1 + \alpha)$. So we have the following decoding algorithm.

Algorithm 5.5.1 (Decoding an opt. cyclic 2-burst-corr. code) *Let $r(x)$ be the received word. Compute the syndrome $s_0 = r(1)$ and $s_1 = r(\alpha)$.*

- *If $s_0 = s_1 = 0$, put $b(x) = 0$.*

- *If $s_0 = 1$ and $s_1 = \alpha^i$, put $b(x) = x^i$.*

- *If $s_0 = 0$ and $s_1 = \alpha^j$, put $b(x) = x^i(1 + x)$, where i is defined by $\alpha^i(1 + \alpha) = \alpha^j$.*

Put $c(x) = r(x) + b(x)$.

Let us now look for an optimum, cyclic, 3-burst-correcting code; so $n = 2^m - 1, r = m + 2, m \geq 4$. In view of the preceding it is natural to try $g(x) = e(x)p(x)$ as generator polynomial, where $p(x)$ is a primitive polynomial of degree m and $e(x)$ has degree 2. Since $g(x)$ has to divide $x^n - 1$ it follows that $e(x)$ must be equal to $1 + x + x^2$.

On the other hand, $1 + x + x^2 = (x^3 - 1)/(x - 1)$ and $x^3 - 1$ divides $x^{2^m-1} - 1$ if and only if m is even!

There are however even more restrictions. Consider the bursts $b_1(x) = 1$ and $b_2(x) = (1+x)x^i$, $0 \leq i < n$. They need to have different syndromes, so their difference (or sum) should not be divisible by $(1 + x + x^2)p(x)$.

Now, $b_1(x) + b_2(x) = 1 + (1+x)x^i$, $1 \le i < n$ is divisible by $1 + x + x^2 = (x^3 - 1)/(x - 1)$ if and only if $i \equiv 1 \pmod 3$. Thus, to make the code 3-burst-correcting, we have the following necessary condition.

$$i \equiv 1 \pmod 3 \quad \Rightarrow \quad (1 + x)x^i \not\equiv 1 \pmod{p(x)}.$$

Since $p(x)$ is a primitive polynomial, one can write $1 + x \equiv x^a \pmod{p(x)}$ for some $1 \le a \le n - 1$. The condition above now reduces to

$$i \equiv 1 \pmod 3 \quad \Rightarrow \quad x^{a+i} \not\equiv 1 \pmod{p(x)}.$$

i.e.

$$i \equiv 1 \pmod 3 \quad \Rightarrow \quad a + i \not\equiv 0 \pmod{2^m - 1}.$$

Since $3 | (2^m - 1)$, one simply gets $a \not\equiv 2 \pmod 3$.

Of course we only checked possible problems with the burst patterns 1 and $1 + x$. One really has to do this for all pairs of burst patterns, i.e. for all pairs of polynomials of degree less than 3, that have constant term 1. It turns out that no further conditions do arise. This proves:

Theorem 5.5.2 *Let $p(x)$ be a primitive polynomial of degree m. Then $g(x) = (1 + x + x^2)p(x)$ generates an optimum, cyclic 3-burst-correcting code if and only if*

1. *m is even,*

2. *a, defined by $1 + x \equiv x^a \pmod{p(x)}$, is not congruent to 2 modulo 3.*

Example 5.5.3 The cyclic code of length 15 generated by $g(x) = (1+x+x^2)(1+x+x^4)$ is an optimum, 3-burst-correcting code, because $1 + x \equiv x^4 \pmod{1 + x + x^4}$ and $4 \not\equiv 2 \pmod 3$.

Questions that arise very naturally are:

- Is it necessary to start with $g(x) = e(x)p(x)$, where $p(x)$ is primitive of degree m?

- Are there always primitive polynomials $p(x)$ of even degree m for which $a \not\equiv 2 \pmod 3$, where $1 + x \equiv x^a \pmod{p(x)}$?

- How to deal with $b \ge 4$?

There are very satisfactory answers to all these questions and they will be given here. Unfortunatly, it leads too far to prove all these statements. The interested reader can find the proofs in *On the existence of optimum cyclic burst-correcting codes,* K.A.S. Abdel-Ghaffar, e.o., IEEE Trans. Inform. Theory, vol. IT-32, pp.768-775, Nov. 1986.

Theorem 5.5.4 (Elspas-Short) *If $g(x)$ generates an optimum, cyclic b-burst-correcting code of length $n = 2^m - 1$, then $g(x) = e(x)p(x)$ with*

1. $degree(e(x)) = b - 1$, $e(0) = 1$, $e(x)$ *has no multiple factors.*

2. $p(x)$ *is a primitive polynomial of degree* $m, m \geq b+1$, *where* m *is a multiple of* m_e, *defined as the smallest* m_e *with* $e(x)|(x^{2^{m_e}-1} - 1)$.

All the conditions in the above theorem follow from the simple fact that $g(x)|(x^n - 1)$ (and the Reiger bound), except for the statement that $g(x)$ contains a primitive factor of degree m.

Definition 5.5.5 *The* AES-conditions *(short for Abramson-Elspas-Short conditions) on the primitive polynomial* $p(x)$ *in Theorem 5.5.4 are all the conditions that arise from*

$$B_1(x) \equiv x^i B_2(x) \ (\mathrm{mod}\ e(x)) \ \Rightarrow \ B_1(x) \not\equiv x^i B_2(x) \ (\mathrm{mod}\ p(x)),$$

for all $0 \leq i < n$ *and for all burst patterns* $B_1(x)$ *and* $B_2(x)$ *of degree at most* $b-1$ *with* $B_1(x) \neq B_2(x)$ *and both unequal to* $e(x)$.

Theorem 5.5.6 *A polynomial* $g(x)$ *generates an optimum, cyclic b-burst-correcting code of length* $n = 2^m - 1$, *if and only if it can be factored into* $g(x) = e(x)p(x)$ *which satisfy the conditions in Theorem 5.5.4 and the AES-conditions.*

Theorem 5.5.7 *Let* $e(x)$ *be a square-free polynomial of degree* $b-1$ *with* $e(0) = 1$. *Then, for all* m *sufficiently large and divisible by* m_e, *a primitive polynomial* $p(x)$ *of degree* m *exists such that* $e(x)p(x)$ *generates an optimum, cyclic b-burst-correcting code of length* $n = 2^m - 1$.

For two special cases even stronger results hold.

Theorem 5.5.8 *For each even* $m, m \geq 4$, *a primitive polynomial* $p(x)$ *of degree* m *exists such that* $(1+x+x^2)p(x)$ *generates an optimum, cyclic 3-burst-correcting code (of length* $n = 2^m - 1$).

Theorem 5.5.9 *For each even* $m, m \geq 10$, *a primitive polynomial* $p(x)$ *of degree* m *exists such that* $(1 + x^3)p(x)$ *generates an optimum, cyclic 4-burst-correcting code (of length* $n = 2^m - 1$).

The smallest degree primitive polynomial $p(x)$ of degree m such that $(1+x)(1+x+x^3)p(x)$ generates an optimum, cyclic 5-burst-correcting code of length $n = 2^m - 1$, is

$$p(x) = 1 + x + x^2 + x^3 + x^5 + x^9 + x^{10} + x^{13} + x^{15}.$$

5.6 Problems

5.6.1 By writing the coordinates of the words of an e-error-correcting, 16-ary Reed-Solomon code as binary words of length 4, one obtains a b-burst-correcting binary code.

What should e be to get $b = 9$? And for $b = 10$?

5.6.2 Interleave the binary Fire code generated by $(x^5 - 1)(1 + x + x^3)$ at depth 5. What kind of burst correcting code does one obtain?

5.6.3 Consider the binary Fire code generated by $g(x) = (x^7 - 1)(x^4 + x + 1)$. What is the length of this Fire code and what is the length of the bursts that it can correct.

Let $r(x)$ be a received word and let $r(x)$ modulo $g(x)$ be given by $x + x^2 + x^4 + x^5 + x^8$. Decode $r(x)$.

Compare the redundancy of this code with the Reiger bound.

5.6.4 Consider the array code $\mathcal{A}(7, 10)$. Give two bursts of length at most 6 that have the same syndrome.

5.6.5 Decode the received word R which is equal to a codeword C from $\mathcal{A}(6, 9)$ plus a burst B of length at most 5, and where

$$R = \begin{array}{|c|c|c|c|c|c|c|c|c|}
\hline
1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
\hline
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
\hline
0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\
\hline
1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\
\hline
0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\
\hline
1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
\hline
\end{array}$$

5.6.6 The binary, cyclic code of length 15 generated by $g(x) = (1 + x + x^2)(1 + x + x^4)$ is optimum, 3-burst correcting by Example 5.5.3. Decode $r(x) = 1 + x^2 + x^4 + x^6 + x^{10} + x^{13} + x^{14}$.

5.6.7 Derive the AES-conditions for $e(x) = 1 + x^3$.

Chapter 6

Convolutional codes

6.1 An example

In this chapter we shall turn our attention away from block codes. Instead, we shall discuss convolutional codes, which were already mentioned in Definition 1.3.3.

In several important applications, convolutional codes have taken over the role that block codes played before. An important reason for this is, that soft decision decoding has more or less the same complexity as hard decision decoding for convolutional codes and that significant gains in performance can be obtained in this way.

We start with a simple example, see Figure 6.1.

Starting with the length-3 register filled with zeroes, an infinite binary information string $\{a_i\}_{i \geq 0}$ is fed into the encoder one bit at a time. The encoder will produce two output bits $c_i^{(1)}$ and $c_i^{(2)}$ per input bit a_i, but these will not only depend on a_i but also on a_{i-1}
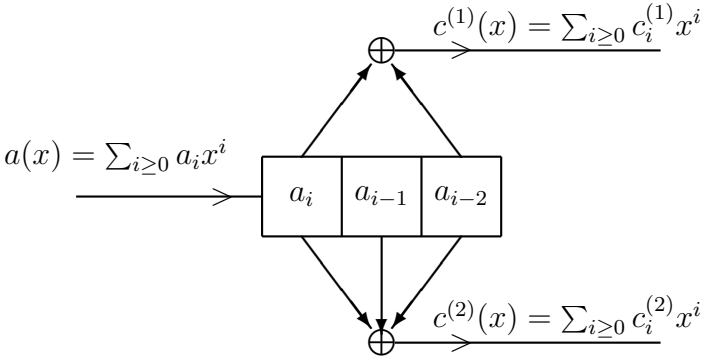
Figure 6.1: A rate 1/2 convolutional encoder.

and a_{i-2}, so indeed the encoder has some memory.

The defining relations for $c_i^{(1)}$ and $c_i^{(2)}$ are

$$
\begin{aligned}
c_i^{(1)} &= a_i + a_{i-2}, & (6.1)\\
c_i^{(2)} &= a_i + a_{i-1} + a_{i-2}. & (6.2)
\end{aligned}
$$

The two output sequences $\{c_i^{(j)}\}_{i\geq 0}$, $j = 1, 2$, are interleaved and sent over the channel. Upon reception, the received sequence is de-interleaved into the two sequences $\{c_i^{(j)}\}_{i\geq 0}$, $j = 1, 2$, from which (if no errors occurred) $\{a_i\}_{i\geq 0}$ can be easily obtained by

$$
(c_i^{(1)} + c_{i-1}^{(1)}) + c_{i-1}^{(2)} = a_i, \quad i \geq 0. \tag{6.3}
$$

How to decode errors in a received sequence will be the topic of Section 6.3.

To explain why these codes are called convolutional codes, we rewrite (6.1) as the convolutions

$$
\begin{aligned}
c^{(1)}(x) &= (1 + x^2)a(x), & (6.4)\\
c^{(2)}(x) &= (1 + x + x^2)a(x), & (6.5)
\end{aligned}
$$

where the sequences $\{a_i\}_{i\geq 0}$ and $\{c_i^{(j)}\}_{i\geq 0}$, $j = 1, 2$, are now represented by the power series $a(x) = \sum_{i\geq 0} a_i x^i$ and $c^{(j)}(x) = \sum_{i\geq 0} c_i^{(j)} x^i$, $j = 1, 2$.

Note that (6.3) can be written as

$$
(1 + x)c^{(1)}(x) + xc^{(2)}(x) = a(x). \tag{6.6}
$$

Equations (6.4) and (6.5) in turn can be written in matrix form

$$
a(x) \left(\begin{array}{cc} 1 + x + x^2 & 1 + x^2 \end{array} \right) = (c^{(1)}(x), c^{(2)}(x)).
$$

The code in Figure 6.1 is an example of a rate $1/2$ convolutional code because each information symbol is mapped into two output symbols. In the next section we shall discuss rate k/n convolutional codes in general. Also we shall define a distance metric on the set of outputsequences in order to measure the error-correcting capabilities of convolutional codes.

6.2 (n, k)-convolutional codes

Let $GF(2)[x]$ denote the set of power series in x over $GF(2)$.
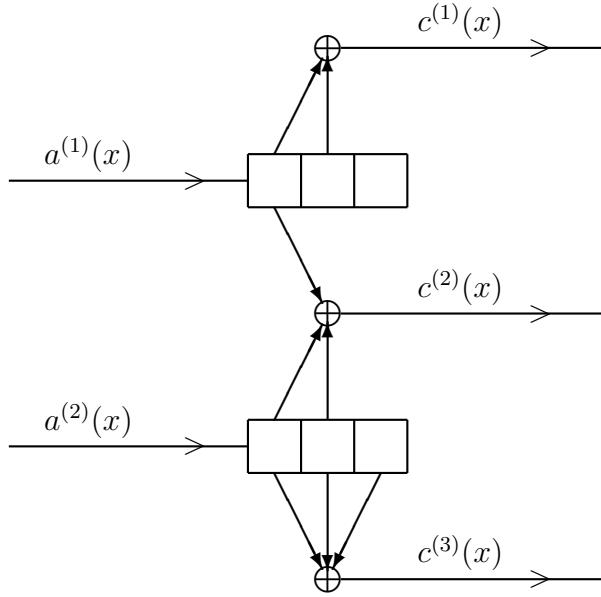
Figure 6.2: A $(3, 2)$-convolutional code of constraint length 2

Definition 6.2.1 *An (n, k)-convolutional code with constraint length M is any code C in $(GF(2)[x])^n$ with $n \geq k$ defined by*

$$\{(a^{(1)}(x), \ldots, a^{(k)}(x))G \mid a^{(i)}(x) \text{ in } GF(2)[x],\ 1 \leq i \leq k\}, \tag{6.7}$$

where G is a $k \times n$ matrix, called generator matrix, with as entries binary polynomials in x of which the highest degree is M.

The interleaved sequences $(c^{(1)}(x), c^{(2)}(x), \ldots, c^{(n)}(x))$ in C will again be called code-words.

In the literature one may see various other definitions of the constraint length of an encoder.

It follows from the above definition that a convolutional code is a linear code. Since k information bits are mapped into n output bits, one says that an (n, k)-convolutional code has *rate k/n*. In Figure 6.2 a $(3, 2)$-convolutional code is depicted with

$$G = \begin{pmatrix} 1 + x & 1 & 0 \\ 0 & 1 + x & 1 + x + x^2 \end{pmatrix}.$$

Note that the output $(c_i^{(1)}, c_i^{(2)}, \ldots, c_i^{(n)})$ at any time i depends on the current and last M inputs $(a_j^{(1)}, a_j^{(2)}, \ldots, a_j^{(k)})$, $j = i, i - 1, \ldots, i - M$. The last M inputs together form the *state* of the encoder (at moment i).

An (n, k)-convolutional code with generator matrix G is called *invertible* if a polynomial matrix H exists such that $GH^T = I_k$. It follows from (6.6) that the (2,1)-convolutional code of Figure 6.1 is invertible with

$$H = \begin{pmatrix} x & 1 + x \end{pmatrix}.$$

To determine in general if a convolutional code is invertible, we need to develop some theory.

Theorem 6.2.2 *Consider an (n, k)-convolutional code with generator matrix G. Let γ_i, $1 \leq i \leq k$, denote the gcd of all the $i \times i$ subdeterminants of G. Set $\gamma_0 = 1$ and let $\delta_i = \gamma_i / \gamma_{i-1}$, $1 \leq i \leq k$. Then*

$$G = A\Delta B, \tag{6.8}$$

where A is a $k \times k$ polynomial matrix with determinant 1, B is an $n \times n$ polynomial matrix with determinant 1, and Δ is a $k \times n$ matrix with $\Delta_{ii} = \delta_i$ and $\Delta_{ij} = 0$ otherwise.

The values δ_i, $1 \leq i \leq k$, are called the invariant factors *of the encoder G.*

Proof. All the row and column operations in this proof will be elementary operations, i.e. simple row or column permutations or adding a multiple of one row (column) to another row (column). So all these operations can be represented by a multiplication on the left with an invertible matrix or a multiplication on the right with an invertible matrix.

Using Euclid's Algorithm one can transform G by elementary row operations into:

$$G' = \begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ 0 & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & & \vdots \\ 0 & g_{k2} & \cdots & g_{kn} \end{pmatrix}. \tag{6.9}$$

If g_{11} does not divide the other elements in the first row, one can use the same technique applied to the columns of G' to obtain a matrix of the form

$$G'' = \begin{pmatrix} g_{11} & 0 & \cdots & 0 \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & & \vdots \\ g_{kn} & g_{k2} & \cdots & g_{kn} \end{pmatrix}, \tag{6.10}$$

with g_{11} of lower degree (it even divides the previous g_{11}). Again, if g_{11} does not divide the other elements in the first column, one can use elementary row operations to get a matrix of the form (6.9) with g_{11} of lower degree. Etc. This process stops with a matrix of the form (6.9) with g_{11} dividing all the other entries in the first row or with a matrix of the form (6.10) with g_{11} dividing all the other entries in the first column.

With elementary row or column operations we may simplify the generator matrix further to the the form:

$$G''' = \begin{pmatrix} g_{11} & \\ & G_1 \end{pmatrix} = \begin{pmatrix} g_{11} & 0 & \cdots & 0 \\ 0 & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & & \vdots \\ 0 & g_{k2} & \cdots & g_{kn} \end{pmatrix}. \tag{6.11}$$

If one of the elements g_{ij} with $i > 1$ and $j > 1$ is not divisible by g_{11}, we can add row i to the top row. With elementary row and column operations one again arrives at a matrix of the form (6.11) but with g_{11} of lower degree. Repeating this process, one arrives at a generator matrix of the form (6.11) that has the property that g_{11} divides all the elements in G_1. Clearly $g_{11} = \delta_1$.

Similarly, G_1 can be transformed into the form

$$\begin{pmatrix} \delta_2 & \\ & G_2 \end{pmatrix},$$

with $\delta_1 | \delta_2$ and all elements in G_2 divisible by δ_2. Repeating this process yields the form in (6.8).

From the form (6.8) and the property $\delta_1 | \delta_2 | \cdots | \delta_k$, if follows that the gcd γ_i of all $i \times i$ subdeterminants of Δ is equal to $\delta_1 \delta_2 \cdots \delta_i$. Since the matrices A and B in (6.8) are products of elementary row resp. column operations, the same holds for subdeterminants of G. So $\delta_i = \gamma_i / \gamma_{i-1}$.

$\square$

Since the matrices A and B in (6.8) are invertible, it follows from Theorem 6.2.2 that G is invertible if and only if Δ has a polynomial right inverse. This proves the following corollary.

Corollary 6.2.3 *The generator matrix of a convolutional code is invertible if and only if all its invariant factors are equal to 1.*

Write $\delta_k = x^u(1 + xv(x))$ and assume that $v(x) \neq 0$. Then the input $(0, 0, \ldots, \frac{1}{1+xv(x)})A^{-1}$ will result in x^u times the k-th row of B as output. This means that this input of infinite weight results in an output of finite weight (all terms in B are polynomials). This also means that a finite set of errors during the transmission can result in the decoding to an input that differs from the real input at infinitely many places. Such an encoder is called *catastrophic*. Clearly, one does not want to use catastrophic convolutional encoders.

On the other hand, assume that $\gamma_k = x^u$. Note that if $(c^{(1)}(x), c^{(2)}(x), \ldots, c^{(n)}(x))$ has finitely many non-zero terms, then so does $B^{-1}(c^{(1)}(x), c^{(2)}(x), \ldots, c^{(n)}(x))$. For the same reason $(a^{(1)}(x), a^{(2)}(x), \ldots, a^{(k)}(x))A$ cannot have finitely many non-zero terms, if $(a^{(1)}(x), a^{(2)}(x), \ldots, a^{(k)}(x))$ has infinitely many non-zero terms (A^{-1} has polynomials as entries). But $\gamma_k = x^u$ implies that all γ_i's (and consequently all δ_i's) are powers of x. Thus $(a^{(1)}(x), a^{(2)}(x), \ldots, a^{(k)}(x))A$ has infinitely many non-zero terms implies that $B^{-1}(c^{(1)}(x), c^{(2)}(x), \ldots, c^{(n)}(x))$ cannot have finitely many non-zero terms. It follows that the encoder cannot be catastrophic in this case.

The discussion above proves the following theorem.

Theorem 6.2.4 *The encoder of an (n, k)-convolutional code with invariant factors $\gamma_1 | \gamma_2 | \cdots | \gamma_k$ is catastrophic if and only if γ_k is not a power of x.*
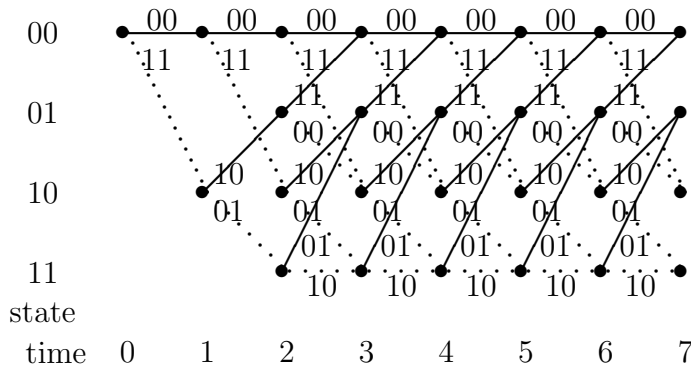
Figure 6.3: The trellis for the $(2,1)$-convolutional code.

Let us now take a new look at the encoder process. A way of describing it, that is different from the polynomial multiplication in Definition 6.2.1, is by means of a *trellis*, which is a graphical way of depicting the possible transitions from one state to another at succesive moments. Instead of giving a formal description, we shall discuss a typical example.

In Figure 6.3 one can find the trellis diagram of the $(2,1)$-convolutional code of Figure 6.1 for the times 0 up to 7. It is assumed that the initial state is the all zero state $(0,0)$.

If input a_0 is 0, the output of the encoder will be 0,0 and the next state of the encoder will again be $(0,0)$. If input a_0 is 1, the output will be 1,1 and the next state will be $(1,0)$. Etc.

In the diagram solid lines represent the state transitions corresponding to a 0-input, while the dotted lines represent state transitions corresponding to a 1-input. Next to these lines the output corresponding to that transition is given. Any path through this trellis diagram represents the genesis of a codeword.

The reader can easily check, for instance, that the input sequence 1, 0, 1, 1, 1, 0, 0 will result in the output sequence 11, 10, 00, 01, 10, 01, 11 and will end in state $(0,0)$.

For error correction one of course needs distinct output sequences to have sufficient distance.

Analogously to the Hamming distance, the *free distance* $d(\mathbf{u}, \mathbf{v})$ between two words $\mathbf{u} = \mathbf{u}(\mathbf{x}) = (u^{(1)}(x), u^{(2)}(x), \ldots, u^{(n)}(x))$ and $\mathbf{v} = \mathbf{v}(\mathbf{x}) = (v^{(1)}(x), v^{(2)}(x), \ldots, v^{(n)}(x))$ is defined by

$$d(\mathbf{u}, \mathbf{v}) = |\{(i,j) \mid u_j^{(i)} \neq v_j^{(i)}, \ 1 \leq i \leq n, \ j \geq 0\}|. \tag{6.12}$$

The *free distance* d_{free} of a convolutional code is the smallest distance between distinct output sequences. Because of the linearity of the code, d_{free} is also equal to the minimum
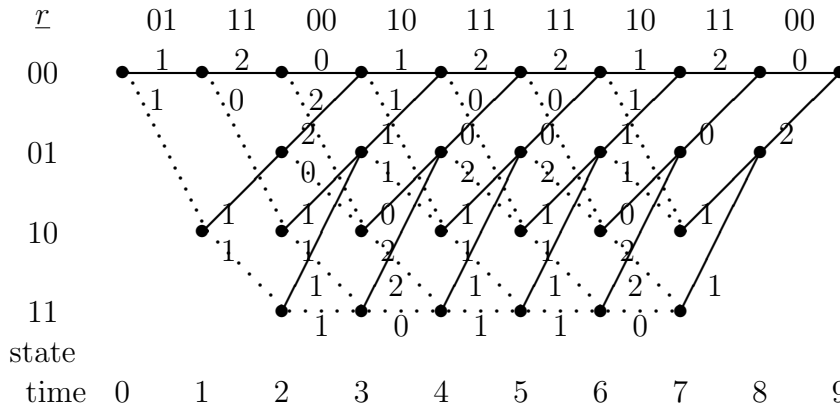
Figure 6.4: The distance between every edge and the corresponding output.

non-zero weight in the code.

To determine the free distance of a convolutional code, one has to determine the minimum weight of any path that leaves the zero-state and returns to it.

In the example of Figure 6.3 the free distance is 5. This value is the weight of the output sequence 11, 10, 11, ... coming from input sequence 1, 0, 0, ...

6.3 The Viterbi-decoding algorithm

For the decoding of a received sequence, the trellis description of the encoder will play a key role. Again we shall only consider the code of Figures 6.1 and 6.3. To make life easier, we shall assume that a codeword has been sent, that is the result of 7 information bits, followed by two zeroes.

Suppose that the received sequence $\underline{r}$ is

01, 11, 00, 10, 11, 11, 10, 11, 00.

It is given at the top of Figure 6.4.

It turns out not be necessary to compare the received sequence $\underline{r}$ with all possible length-9, output sequences starting and ending in (0,0).

We explain this in two steps. In Figure 6.4 each edge e has as *weight* $w(e)$ the distance between the corresponding output sequence (its value can be found in Figure 6.3) and the received sequence. For instance, the transition from state (0,0) at time 1 to state (0,0) at time 2 has output 00, which is at distance 2 from the received 11 during that transition.

For the next step we refer the reader to Figure 6.5
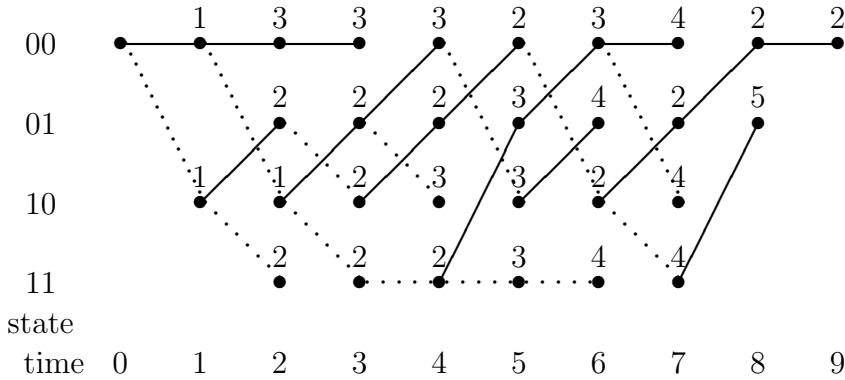
Figure 6.5: The lowest weight path to every state.

Quite clearly, for the states at time 1 and 2, there is a unique path leading to them. Each other state, say $\underline{s}$, has two incoming edges e_1 and e_2 coming from two states, say $\underline{s}_1$ and $\underline{s}_2$.

If the lowest weight path ending in state $\underline{s}_i$, $i = 1, 2$, has weight w_i then the lowest weight path ending in $\underline{s}$ will have weight

$$\min\{w_1 + w(e_1), w_2 + w(e_2)\}. \tag{6.13}$$

In this way, for all states the weight of the lowest weight path ending there can be computed recursively in a very simple way. In Figure 6.5, these values can be found. Also, in each state only one incoming edge, called *survivor* remains, namely one that minimizes (6.13). If both edges minimize (6.13), one of them will have to be designated as survivor.

The path that reaches state (0,0) at time 9 will clearly correspond to the codeword closest to the received sequence. From Figures 6.3 and 6.5 one gets as most likely codeword:

11, 10, 00, 10, 11, 11, 10, 11, 00.

Of course there is no need to perform steps 1 and 2 consecutively. They can be done simultaneously: when considering a state $\underline{s}$ one can compute the weights of the two incoming edges.

The decoding algorithm above is called the *Viterbi decoding algorithm*. When executing it, one needs to be able to reconstruct the lowest weight paths ending in each of the 4 states. The longer one waits, the longer these paths become (their length grows linearly in time). In general there are 2^{Mk} states. In most practical applications $k = 1$ and $M \leq 7$.

In our example we assumed that the transmitted codeword would end in the (0,0)-state at time 9. If that information is not available, one can set a parameter l and decide at
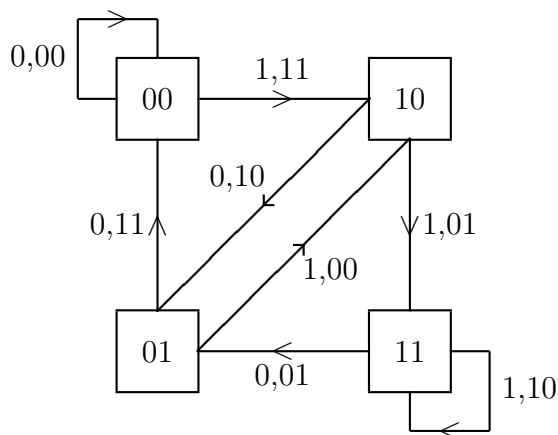
Figure 6.6: The state diagram of the (2,1)-convolutional encoder.

time $i + l$ what the most likely transition at time i was, $i \geq 0$. For instance, $l = 5$ in Figure 6.5 would result in the estimates $m_0 = 1, m_1 = 0, m_2 = 1$, etc.

In Section 1.2 the Gaussian channel was discussed. In this channel the probability density that a symbol r is received, while the symbol m was transmitted is given by $\frac{1}{\sqrt{2\pi}\sigma} \exp^{-(r-m)^2/2\sigma^2}$. So its logarithm is proportional to the squared Euclidean distance $|r - m|^2$ between r and m.

Fortunately, the Viterbi decoding algorithm is equally suitable for soft decision decoding! Indeed, instead of computing the Hamming distance between a possible transition output and the received sequence, one has to compute the squared Euclidean distance between these two. This calculation will affect the complexity of the decoding algorithm by some constant.

6.4 The fundamental path enumerator

In the previous section we found the free distance of the (2,1)- convolutional encoder of Figure 6.1 by looking at its trellis. For convolutional encoders with more states, one would like to have a more analytical method. To this end we need the equivalent of the weight enumerator of block codes. Again we shall use the (2,1)-convolutional encoder of Figure 6.1 to demonstrate these ideas.

In Figure 6.6 the four states of the encoder are depicted with their transitions. Along each edge the triple $m, c^{(1)}c^{(2)}$ denotes the input m and the corresponding output $c^{(1)}c^{(2)}$.

In Figure 6.7 the label $m, c^{(1)}c^{(2)}$ of each edge in the state diagram is replaced by the label $x^m y^{c^{(1)}+c^{(2)}} z$. So the exponent of x gives the Hamming weight of the input, the
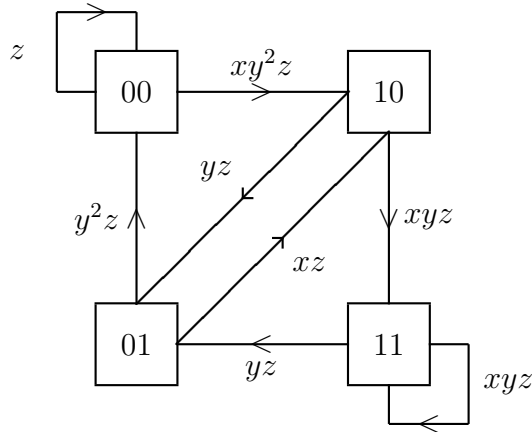
Figure 6.7: The labeled diagram of the (2,1)-convolutional encoder.

exponent of y gives the Hamming weight of the output and the exponent of z simply denotes the fact that the edge is a path of length 1.

We now can also give any path of positive length in the state diagram its own label: simply take the product of the labels of the edges along the path. From the label $\text{lb}(\mathcal{P}) = x^v y^w z^l$ of a path $\mathcal{P}$, we can conclude that $\mathcal{P}$ has length l, that v edges correspond to a 1-input (and $l - v$ edges correspond to a 0-input) and that the weight of the output sequence is w. The trivial path of length 0 has label 1.

Definition 6.4.1 *A* fundamental path *in the state diagram of a convolutional encoder is any path of positive length that starts in a state $\underline{s}$ and ends in state $\underline{0}$ without passing through $\underline{0}$ in between.*

To count these fundamental paths we need to define a generating function of these (and similar) paths.

Definition 6.4.2 *The* fundamental path enumerator $A_{\underline{s}}(x, y, z)$ *with respect to state $\underline{s}$ of a convolutional encoder is defined by*

$$A_{\underline{s}}(x, y, z) = \sum_{\substack{fundamental\ \mathcal{P}\ from\ \underline{s}\ to\ \underline{0}}} \text{lb}(\mathcal{P}). \tag{6.14}$$

Instead of $A_{\underline{0}}(x, y, z)$ one simply writes $A(x, y, z)$.

The (2,1)-convolutional encoder has the four fundamental path enumerators: $A_{00}(x, y, z)$, $A_{01}(x, y, z)$, $A_{11}(x, y, z)$ and $A_{10}(x, y, z)$. These four are strongly related!

Indeed, any fundamental path from 10 to 00 either

1. follows the edge with label yz, arrives in 01 and continues from there to $\underline{0}$ along any of the fundamental paths denoted by $A_{01}(x, y, z)$ or

2. follows the edge with label xyz, arrives in 11 and continues from there to $\underline{0}$ along any of the fundamental paths denoted by $A_{11}(x, y, z)$.

So we have the relation

$$A_{10}(x, y, z) = yzA_{01}(x, y, z) + xyzA_{11}(x, y, z).$$

In the same way, one can find the relations

$$
\begin{aligned}
A_{00}(x, y, z) &= xy^2zA_{10}(x, y, z), \\
A_{01}(x, y, z) &= y^2z + xzA_{10}(x, y, z), \\
A_{11}(x, y, z) &= yzA_{01}(x, y, z) + xyzA_{11}(x, y, z).
\end{aligned}
$$

This gives rise to the following matrix equation:

$$
\begin{pmatrix}
1 & 0 & -xy^2z & 0 \\
0 & 1 & -xz & 0 \\
0 & -yz & 1 & -xyz \\
0 & -yz & 0 & 1 - xyz
\end{pmatrix}
\begin{pmatrix}
A_{00}(x, y, z) \\
A_{01}(x, y, z) \\
A_{10}(x, y, z) \\
A_{11}(x, y, z)
\end{pmatrix}
=
\begin{pmatrix}
0 \\
y^2z \\
0 \\
0
\end{pmatrix}.
$$

$$(6.15)$$

From equation (6.15) it is now simple to determine $A_{0,0}(x, y, z)$ (use Kramer's Rule).

$$
A_{0,0}(x, y, z) = \frac{
\begin{vmatrix}
0 & 0 & -xy^2z & 0 \\
y^2z & 1 & -xz & 0 \\
0 & -yz & 1 & -xyz \\
0 & -yz & 0 & 1 - xyz
\end{vmatrix}
}{
\begin{vmatrix}
1 & 0 & -xy^2z & 0 \\
0 & 1 & -xz & 0 \\
0 & -yz & 1 & -xyz \\
0 & -yz & 0 & 1 - xyz
\end{vmatrix}
},
$$

and thus

$$A_{0,0}(x, y, z) = \frac{xy^5z^3}{1 - xyz - xyz^2} = \frac{xy^5z^3}{1 - xyz(1 + z)}. \tag{6.16}$$

From $\frac{xy^5z^3}{1-xyz(1+z)} = xy^5z^3\{1 + xyz(1 + z) + (xyz(1 + z))^2 + \cdots \}$ it follows that the fundamental path with lowest output weight (i.e. with the smallest exponent of y) has weight 5. In other words the free distance of this convolutional encoder is 5.

It also follows that there is only one fundamental path with output weight 5, since there is only one term with y^5, namely xy^5z^3, and its coefficient is 1. The exponent 1 of x

and the exponent 3 of z in xy^5z^3 tell that this fundamental path is caused by a single 1-input and that it has length 3 (so there were two 0-inputs).

Just like the weight enumerator of a cyclic code can be used to estimate the probablity of incorrect decoding (see the end of Section 2.3), the fundamental path enumerator $A(x, y, z)$ turns out to be very helpful when studying various error probabilities that one may want to evaluate. Here, we shall only give an upper bound on the so-called *first error probability* i.e. the probability that the decoder makes an error right at the beginning. By the linearity, this means that we are calculating the probability that starting in the zero state, only zeroes inputs have occurred, while the decoder finds one of the fundamental paths more likely.

Let $Pr^e(\mathcal{P}; \underline{0})$ denote the probability that fundamental path $\mathcal{P}$ is a more likely transmitted sequence, while in fact the all-zero sequence was transmitted.

The first error probablity Pr_1 is bounded above by

$$\sum_{\substack{\text{fundamental } \mathcal{P} \\ \text{from } \underline{0}}} Pr^e(\mathcal{P}; \underline{0}). \tag{6.17}$$

For the BSC with error probability p, the probability $Pr^e(\mathcal{P}; \underline{0})$ only depends on the Hamming weight w of the output sequence corresponding to $\mathcal{P}$:

$$Pr^e(\mathcal{P}; \underline{0}) = \sum_{k \geq \lceil w/2 \rceil} \binom{w}{k} p^k (1-p)^{w-k}.$$

From

$$\sum_{k \geq \lceil w/2 \rceil} \binom{w}{k} p^k (1-p)^{w-k} \leq p^{w/2}(1-p)^{w/2} \sum_{k \geq \lceil w/2 \rceil} \binom{w}{k} \leq$$
$$\leq p^{w/2}(1-p)^{w/2} 2^w =$$
$$= \left(2\sqrt{p(1-p)} \right)^w$$

it follows that

$$Pr_1 \leq \sum_{\substack{\text{fundamental} \\ \mathcal{P} \text{ from } \underline{0}}} (2\sqrt{p(1-p)})^{w(\mathcal{P})} = A(1, 2\sqrt{p(1-p)}, 1).$$

This proves the following theorem.

Theorem 6.4.3 *The first error probablity Pr_1 of a convolutional encoder with fundamental path enumerator $A(x, y, z)$ satisfies*
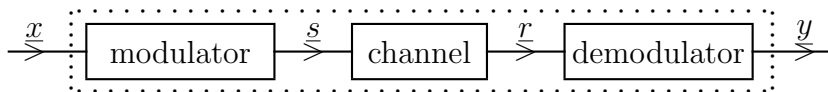
$$Pr_1 \leq A(1, 2\sqrt{p(1-p)}, 1). \tag{6.18}$$

Figure 6.8: The mathematical channel.

The first error probablity of the (2,1)-convolutional encoder of Figure 6.1 is thus bounded above by

$$\frac{32(p(1-p))^{2.5}}{1-4\sqrt{p(1-p)}}.$$

6.5 Combined coding and modulation

In Figure 1.2 a general communication system was depicted. The physical channel in Figure 1.2 is defined (see Definition 1.2.1) in a mathematical way by means of its transition probabilities. This description may be fine for most purposes, but is not always the full story.

In some applications, the mathematical channel consists of three components; see Figure 6.8.

The *modulator* transforms the input sequence $\underline{x}$ (coming from the encoder) into a sequence of waveforms s_i. The channel adds noise n_i to these waveforms. The *demodulator* transforms the sequence $\underline{r}$ of outputs of the channel back into a sequence $\underline{y}$ that will go to the decoder.

In the so-called *bandpass Gaussian channel* the waveforms $s_{ph}(t)$ that are physically transmitted over the channel at discrete time-intervals have the form

$$s_{ph}(t) = A(t)\cos(2\pi f_0 t + \phi(t)),$$

where f_0 is a (fixed) carrier frequency and where the amplitude $A(t)$ and phase $\phi(t)$ can be varied.

With $s_R(t) = A(t)\cos\phi(t)$ and $s_I(t) = A(t)\sin\phi(t)$ one can write $s_{ph}(t) = s_R(t)\cos(2\pi f_0 t) - s_I(t)\sin(2\pi f_0 t)$ and thus $s_{ph}(t)$ can be represented as the real part of the point in the complex plane:

$$s_{ph}(t) = \text{Re}\{s(t)e^{2\pi i f_0 t}\}, \tag{6.19}$$

where $s(t) = s_R(t) + is_I(t)$.

The complex representations of the waveforms are normally taken from a specific modulation set, called *constellation*. In the sequel M will denote the cardinality of this constellation.

a) $M = 2$ d) $M = 4$ g) $M = 8$

b) $M = 4$ e) $M = 8$ h) $M = 16$

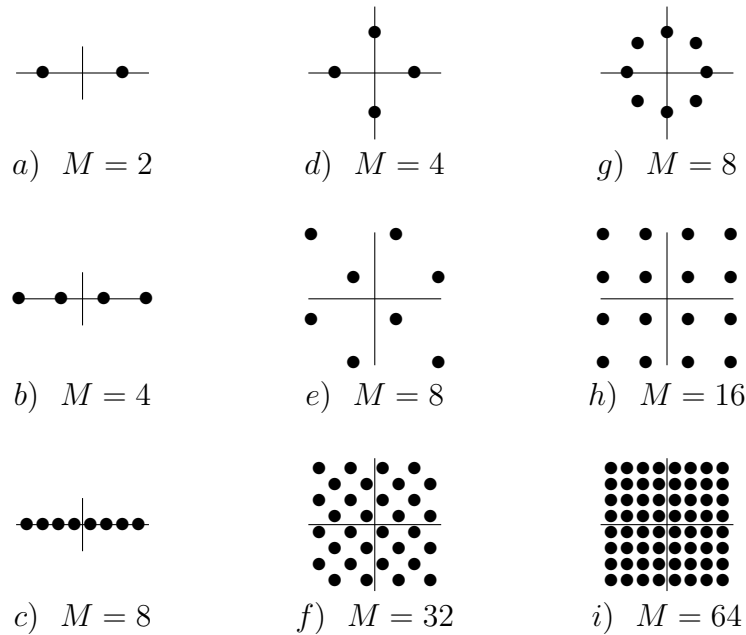c) $M = 8$ f) $M = 32$ i) $M = 64$

Figure 6.9: Some constellations.

The further these constellation points are apart (in the complex plane) the more likely it is that a received signal will be correctly demodulated. On the other hand, the energy it takes to transmit a signal $s(t)$ is proportional to $|s(t)|^2$. For this reason the constellation points are confined to a limited area around 0.

Typical constellations are depicted in Figure 6.9.

A received signal can also be represented by a complex point. The probability that signal r is received while s has been sent, is again given by the Gaussian distribution:

$$Pr(r|s) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-|r-s|^2/2\sigma^2},$$

so the logarithm of this probability is proportional to the squared Euclidean distance $|r - s|^2$ between r and s.

For a sequence $\underline{r}$ of received symbols, one has to find the most likely transmitted sequence $\underline{m}$ of transmitted symbols from the constellation, i.e. one has to maximize $\prod_i Pr(r_i|s_i)$ or, equivalently, minimize the squared Euclidean distance $\sum_i |r_i - s_i|^2$.

Now, the reason that we did not bother to discuss the modulation and demodulation aspects of the channel in all the previous sections simply is that the modulation was done in a straightforward way. For instance, with $M = 8$ in constellation e) in Figure 6.9 one takes 3 bits (coming from the encoder) at a time to determine which one of the $2^3 = 8$ constellation points was transmitted. Assuming that these 8 points have coordinates $\pm\frac{1}{2}$ and $\pm\frac{3}{2}$, the minimum squared Euclidean distance between two of these points is 2 (for

$$
\begin{array}{cccc}
a & b & a & b \\
c & d & c & d \\
a & b & a & b \\
c & d & c & d
\end{array}
$$

Figure 6.10: Partitioning constellation h) in four subconstellations

output	subconstellation
00	a
01	b
10	c
11	d

Figure 6.11: Choice of the subconstellation

instance $(\frac{1}{2}, \frac{3}{2})$ and $(\frac{3}{2}, \frac{1}{2})$ have squared Euclidean distance $1^2 + 1^2 = 2$).

Since about ten years, we know that coding can improve the performance of the modulation scheme. We shall demonstrate this technique with a typical example.

We know already that constellation e) in Figure 6.9 allows one to transmit 3 bits per signal point and also that these points have squared Euclidean distance at least two. However, we are going to allow the 16 points of constellation h) to be transmitted, but we still want to transmit three bits per point.

A way to do this is as follows. The 16 points of constellation h) are partitioned into four subconstellations, each of size 4, in the way depicted in Figure 6.10.

Now, one of the three input bits is fed into a (2,1)-convolutional encoder, say the one of Figure 6.1. The output of the encoder determines a specific subconstellation in a suitably chosen one-to-one way, say by the rule, depicted in Figure 6.11:

The other two input bits determine which of the four points in the chosen subconstellation will actually be transmitted.

Two different points in the same subconstellation have squared Euclidean distance at least four, instead of just two. Two points from different subconstellations can have a squared Euclidean distance of just one. However the (2,1) convolutional encoder will take care that for at least three transmissions the constellation points will be chosen from different subconstellations. The resulting squared Euclidean distance turns out to be even more than the 3×1 that we have now obtained.

Indeed, by the linearity of the convolutional code we may assume that one sequence of input bits of the (2,1) convolutional encoder consists of just zeroes, resulting in 00-outputs only, corresponding to points of the a-type subconstellation. The other input sequence will be one of the following two possibilities:

i) There is a single 1-input (all others are 0) in the second sequence. In this case the output sequence of the (2,1) convolutional encoder looks like:

$$\cdots, 00, 00, 11, 10, 11, 00, 00, \cdots,$$

corresponding to the sequence of subconstellations

$$\cdots, a \ , a \ , d \ , c \ , d \ , a \ , a \ , \cdots.$$

So, in this case the squared Euclidean distance between the two sequences of constellation points is at least $2 + 1 + 2 = 5$.

ii) There are more than one 1-inputs. The output sequence of the (2,1) convolutional encoder will differ in at least four steps from the output sequence coming from the all-zero input sequence. So, during at least four consecutive transmissions, points will be chosen from different constellations, resulting in a squared Euclidean distance of at least 4.

Taking the minimum of all possibilities, we conclude that this specific combined coding and modulation scheme has increased the minimum squared Euclidean distance from 2 to 4. In practical situations this already gives rise to a considerable increase in reliability.

6.6 Problems

6.6.1 Decode the received sequence

$$11, 11, 00, 11, 10, 01, 00, 11, 10$$

in Figure 6.3, if the transmitted codeword can be assumed to be the result of 7 information symbols followed by 2 zeroes.

6.6.2 Determine the invariant factors of the (5,3)-convolutional code with generator matrix

$$G = \begin{pmatrix} 1 & 1+x & 1+x^2 & 1 & 1+x+x^2 \\ 1+x & 1+x & x+x^3 & 0 & x+x^3 \\ x & x+x^2 & 1+x+x^2+x^3 & x & 1+x+x^3 \end{pmatrix}.$$

6.6.3 Let C be the (2,1)-convolutional code generated by

$$G = \begin{pmatrix} 1+x & 1+x^2 \end{pmatrix}.$$

Give an input sequence with infinitely many non-zero elements resulting in an output with finitely many non-zero elements.

Determine the path enumerator of this code.

6.6.4 Make the trellis for the (2,1)-convolutional code C generated by

$$G = \left(\; 1 + x + x^3 \quad 1 + x + x^2 + x^3 \; \right).$$

What is the free distance of C?

6.6.5 Assume that the points in Figure 6.10 have coordinates $\pm\frac{1}{2}$ and $\pm\frac{3}{2}$ and let the four points x_1, x_2, x_3, x_4 in each of the four subconstellations

$$x_1 \qquad x_2$$

$$x_3 \qquad x_4$$

be the image of two of the three information bits in the following way:

$$
\begin{array}{rcl}
00 & \to & x_1 \\
01 & \to & x_2 \\
11 & \to & x_3 \\
10 & \to & x_4
\end{array} \; .
$$

The choice of the particular subconstellation is made according to the rule in Figure 6.11 by the output of the convolutional code of Figure 6.1, which has the third information bit as input.

Decode the received sequence

$$\cdots, a_1, a_1, d_2, c_3, d_4, a_1, a_1, \cdots.$$

Appendix A

Finite fields

All readers are assumed to be familiar with $\mathcal{Q}$, $\mathcal{R}$, and $\mathcal{C}$, the sets of rational, real and complex numbers. These sets have nice properties with respect to addition and multiplication, properties that will be extensively discussed in this appendix. These three sets are examples of so-called fields.

In the context of coding theory fields of finite cardinality play a crucial role. This is so because most codes consist of vectors taken from a vector space over a finite field and, secondly, many good codes are constructed by algebraic methods. In this appendix an introduction will be given to the theory of finite fields. All possible results regarding finite fields that are needed in the chapters can be found in this appendix.

A.1 The integers

Let $\mathcal{N}$ denote the set of natural numbers and $\mathcal{Z}$ the set of integers. If an integer d *divides* an integer n, i.e. $n = k \cdot d$ for some $k \in \mathcal{Z}$, this will be denoted by $d \mid n$. If d does not divide n one writes $d \nmid n$.

An integer p, $p > 1$, is said to be prime, if it has no other positive divisors than 1 and p. It is already known since Euclid (300 B.C.) that there exist infinitely many prime numbers. With $p_1 = 2$, $p_2 = 3$, $p_3 = 5$, etc., a natural numbering of the primes is given.

Definition A.1.1 *The greatest common divisor of two integers a and b, denoted by $\gcd(a, b)$ or (a, b), is the, uniquely determined, greatest positive integer dividing both a and b, so*

$$\gcd(a, b) = \max\{d > 0 \ \mid \ (d \mid a) \wedge (d \mid b)\}. \tag{A.1}$$

Similarly the least common multiple of two integers a and b, denoted by $lcm[a, b]$ or $[a, b]$, is the, uniquely determined, least positive integer divisible by both a and b, so

$$lcm[a, b] = \min\{m > 0 \ \mid \ (a \mid m) \wedge (b \mid m)\}. \tag{A.2}$$

The next theorem follows directly from an algorithm that will be presented below.

Theorem A.1.2 *Let a and b be in $\mathcal{N}$. Then there exist integers u and v such that*

$$ua + vb = \gcd(a, b). \tag{A.3}$$

Let a and b be two positive integers with $b \geq a$. Clearly any divisor of a and b is a divisor of a and $b - a$ and vice versa. It follows that $\gcd(a, b) = \gcd(a, b - a)$ and for the same reason that $\gcd(a, b) = \gcd(r, a)$, where r is defined by usual "division with remainder", i.e. $b = q \cdot a + r$, $0 \leq r < a$. If $r = 0$, one may conclude that $\gcd(a, b) = a$, while if $r > 0$ one can continue in exactly the same way with a and r. This method gives an extremely fast way of computing the gcd of a and b. The next algorithm shows how to find u and v satisfying (A.3) in an efficient way.

Algorithm A.1.3 (Euclid's algorithm)

> *(initialize)* $s_0 = b$; $s_1 = a$;
>
> $\qquad u_0 = 0$; $u_1 = 1$;
>
> $\qquad v_0 = 1$; $v_1 = 0$;
>
> $\qquad n = 1$;
>
> *while* $s_n > 0$ *do*
>
> $\qquad begin\ n = n + 1$; $q_n = \lfloor s_{n-2}/s_{n-1} \rfloor$;
>
> $\qquad s_n = s_{n-2} - q_n s_{n-1}$;
>
> $\qquad$ *(so s_n is the remainder of s_{n-2} divided by s_{n-1})*
>
> $\qquad u_n = q_n u_{n-1} + u_{n-2}$;
>
> $\qquad v_n = q_n v_{n-1} + v_{n-2}$
>
> $\qquad end$;

$$u = (-1)^n u_{n-1}; \qquad v = (-1)^{n-1} v_{n-1}; \tag{A.4}$$

$$\gcd(a, b) = s_{n-1}. \tag{A.5}$$

Proof: Since the numbers s_n, $n \geq 1$, form a strictly decreasing sequence of nonnegative integers, the algorithm will terminate after at most b iterations (in fact it will terminate much faster; see Problems A.6.1 and A.6.2).

From the relation $s_n = s_{n-2} - q_n s_{n-1}$ in the algorithm and the argument given below Theorem A.1.2 it follows that

$$
\begin{aligned}
\gcd(a, b) &= \gcd(s_0, s_1) = \gcd(s_1, s_2) = \cdots = \\
&= \gcd(s_{n-1}, s_n) = \gcd(s_{n-1}, 0) = s_{n-1}.
\end{aligned}
$$

This proves (A.5). Next we want to prove that for all k, $0 \le k \le n$,

$$(-1)^{k-1} u_k a + (-1)^k v_k b = s_k. \tag{A.6}$$

For $k = 0$ and $k = 1$ (A.6) is true by the initialization values of u_0, u_1, v_0, and v_1. We proceed by induction. It follows from the relations given in the algorithm and from the induction hypothesis that

$$s_k = s_{k-2} - q_k s_{k-1} =$$

$$\{(-1)^{k-3} u_{k-2} a + (-1)^{k-2} v_{k-2} b\} - q_k \{(-1)^{k-2} u_{k-1} a + (-1)^{k-1} v_{k-1} b\} =$$

$$= (-1)^{k-1} \{u_{k-2} + q_k u_{k-1}\} a + (-1)^k \{v_{k-2} + q_k v_{k-1}\} b =$$

$$= (-1)^{k-1} u_k a + (-1)^k v_k b.$$

This proves (A.6) for all k, $0 \le k \le n$. Substitution of $k = n - 1$ in (A.6) yields

$$(-1)^n u_{n-1} a + (-1)^{n-1} v_{n-1} b = s_{n-1}. \tag{A.7}$$

Comparison of (A.7) with (A.3) proves the identities in (A.4).

$\square$

Theorem A.1.2 makes it possible to prove elementary properties of numbers.

Corollary A.1.4 *Let a, b, and d be integers such that d divides ab. Then $\gcd(d, a) = 1$ implies that $d \mid b$.*

Proof: Theorem A.1.2 applied to a and d shows the existence of integers u and v such that $1 = ua + vd$. Multiplication of this relation by b yields $b = u(ab) + (vb)d$. Since d divides both ab and d, it follows that d also divides $u(ab) + (vb)d$, i.e. b.

$\square$

It is now quite easy to prove (see Problem A.6.3) the following theorem.

Theorem A.1.5 (Fundamental Theorem of Number Theory)
Any positive integer has a unique factorization of the form

$$\prod_i p_i^{e_i}, \quad e_i \in \mathcal{N}. \tag{A.8}$$

Let $a = \prod_i p_i^{e_i}$, $e_i \in \mathcal{N}$, and $b = \prod_i p_i^{f_i}$, $f_i \in \mathcal{N}$. Then

$$\gcd(a, b) = \prod_i p_i^{\min\{e_i, f_i\}}, \tag{A.9}$$

$$\text{lcm}[a, b] = \prod_i p_i^{\max\{e_i, f_i\}}, \tag{A.10}$$

$$\gcd(a, b) \cdot \text{lcm}[a, b] = ab. \tag{A.11}$$

A special function on $\mathcal{N}$ that will be needed in Section A.5 will be defined next.

Definition A.1.6 (Euler's Totient Function) *The function $\phi(m)$ is defined as the number of integers less than or equal to m that are coprime with m, so*

$$\phi(m) = |\{1 \leq r \leq m \mid \gcd(r, m) = 1\}|. \tag{A.12}$$

Theorem A.1.7 *For all positive integers n*

$$\sum_{d|n} \phi(d) = n. \tag{A.13}$$

Proof: Let $d \mid n$. By writing $k = id$ one sees immediately that the number of elements k, $1 \leq k \leq n$, with $\gcd(k, n) = d$ is equal to the number of integers i with $1 \leq i \leq n/d$ and $\gcd(i, n/d) = 1$. So this number is $\phi(n/d)$. On the other hand, $\gcd(k, n)$ divides n for each integer k, $1 \leq k \leq n$. It follows that $\sum_{d|n} \phi(n/d) = \sum_{d|n} \sum_{1 \leq k \leq n, \gcd(k,n)=d} 1 = \sum_{1 \leq k \leq n} 1 = n$, which is equivalent to (A.13).

$\square$

Let p be a prime number. Since every integer r, $1 \leq r < p$, has gcd 1 with p, it follows that $\phi(p) = p - 1$. Also $\phi(p^e)$, p prime, can easily be evaluated. Indeed, since only the multiples of p have a nontrivial factor in common with p^e, one has:

$$\phi(p^e) = p^e - p^{e-1} = p^{e-1}(p-1) = p^e(1 - \frac{1}{p}). \tag{A.14}$$

If $m = p_1^{e_1} p_2^{e_2}$ with both p_1 and p_2 prime and both e_1 and e_2 positive, one can again easily evaluate $\phi(m)$. In this case m/p_1 integers in between 1 and m are a multiple of p_1, m/p_2 integers in between 1 and m are a multiple of p_2, but $m/p_1 p_2$ of these integers are a multiple of both p_1 and p_2 (namely the multiples of $p_1 p_2$). It follows that in this case

$$\phi(m) = \phi(p_1^{e_1} p_2^{e_2}) = m - \frac{m}{p_1} - \frac{m}{p_2} + \frac{m}{p_1 p_2} = m(1 - \frac{1}{p_1})(1 - \frac{1}{p_2}).$$

Generalizing this (by means of the Principle of Inclusion and Exclusion) one obtains in this way the following theorem.

Theorem A.1.8

$$\phi(m) = m \cdot \prod_{p \text{ prime, } p|m} (1 - \frac{1}{p}). \tag{A.15}$$

Two integers a and b are said to be *congruent* to each other *modulo m*, if their difference $b - a$ is divisible by m. This so-called *congruence relation* is denoted by:

$$a \equiv b \pmod{m}. \tag{A.16}$$

For instance $25 \equiv 3 \pmod{11}$ since $11|(25 - 3)$. In this terminology one also has $5 + 8 \equiv 2 \pmod{11}$ and $5 \times 9 \equiv 1 \pmod{11}$.

Lemma A.1.9 *Let $ka \equiv kb \pmod{m}$ and $\gcd(k, m) = 1$. Then $a \equiv b \pmod{m}$.*

Proof: Since $ka - kb = xm$, $x \in \mathcal{Z}$, and $\gcd(m, k) = 1$, it follows from Corollary A.1.4 that $m \mid (a - b)$, i.e. $a \equiv b \pmod{m}$.

$\square$

The simplest congruence, and one that one often has to solve, is the so-called *linear congruence relation*:

$$ax \equiv b \pmod{m}. \tag{A.17}$$

Theorem A.1.10 *The linear congruence relation $ax \equiv b \pmod{m}$ has a solution x for every $a, 1 \le a \le m - 1$ and every b if and only if m is a prime. If m is a prime, say $m = p$, this solution is unique modulo p.*

Proof: If m is composite, say $m = m_1 m_2$ with $1 < m_1 < m$, the congruence relation $m_1 x \equiv 1 \pmod{m}$ obviously does not have a solution, since both m and xm_1 are divisible by m_1, but 1 is not.

If m is a prime p, the sets $\{0, 1, \ldots, p-1\}$ and $\{0a, 1a, \ldots, (p-1)a\}$ are the same modulo p by Lemma A.1.9. In particular each element b in the first set can be written in a unique way as a product ax (modulo p) in the second set. It follows that the congruence relation $ax \equiv b \pmod{p}$ has a unique solution modulo p.

$\square$

The solution of $ax \equiv b \pmod{p}$, $1 \le a \le p - 1$, can easily be found with Euclid's Algorithm. Indeed, from $ua + vp = 1$ (see (A.3)), it follows that $ua \equiv 1 \pmod{p}$. So the solution x of the congruence relation $ax \equiv b \pmod{p}$ is given by $bu \pmod{p}$, since $a(bu) \equiv b(ua) \equiv b \pmod{p}$. One often writes a^{-1} for the unique element u satisfying $ua \equiv 1 \pmod{p}$.

Example A.1.11 To solve $17x \equiv 16 \pmod{37}$, one first applies Euclid's Algorithm to 17 and 37. One gets $(-13) \times 17 + 6 \times 37 = 1$. So $24 \times 17 \equiv (-13) \times 17 \equiv 1 \pmod{37}$.

It follows that $17^{-1} \equiv 24 \pmod{37}$ and that the solution of $17x \equiv 16 \pmod{37}$ is given by $x \equiv 17^{-1} \times 16 \equiv 24 \times 16 \equiv 14 \pmod{37}$.

A.2 Relations

The congruence relation defined in (A.16) is a special case of a concept that will be defined next.

Definition A.2.1 *Let S be any set and let U be a non-empty subset of $S \times S$. Then U defines a* r*elation $\sim$ on S by*

$$\forall_{s,t \in S} \ [s \sim t \ \textit{iff} \ (s, t) \in U]. \tag{A.18}$$

An equivalence relation is a relation with the additional properties:

reflexivity:	$\forall_{s \in S} \ [s \sim s]$	(E1)
symmetry:	$\forall_{s,t \in S} \ [(s \sim t) \Rightarrow (t \sim s)]$	(E2)
transitivity:	$\forall_{s,t,u \in S} \ [(s \sim t \wedge t \sim u) \Rightarrow (s \sim u)]$	(E3)

Let S be the set of straight lines in the (Euclidean) plane. Then "being parallel or equal" defines an equivalence relation. In Section A.1 we have seen another example. There, a relation (called congruence relation, see (A.16)) was defined on $S = \mathcal{Z}$ by $a \equiv b \pmod{m}$ iff $m \mid (a - b)$. The reader can easily check that this congruence relation indeed satisfies (E1), (E2) and (E3) and thus is an equivalence relation (see also Problem A.6.4).

An equivalence relation $\sim$ on a set S partitions S into so-called *equivalence classes* $< s >$ defined by

$$< s >= \{t \in S \ \mid \ t \sim s\}, \tag{A.19}$$

So $< s >$ consists of those elements in S that are equivalent to s. It is straightforward to check that for any equivalence class $< s >$ one has

i)	$\forall_{t,u \in <s>} \ [t \sim u],$
ii)	$\forall_{t \in <s>} \ \forall_{u \in S \backslash <s>} \ [\neg(t \sim u)].$

The equivalence classes of the congruence relation defined in (A.16) are called *congruence classes*. The set of congruence classes of the integers modulo m will be denoted by $\mathcal{Z}_m$. Its elements are $< 0 >, < 1 >, \ldots, < m - 1 >$ but will often simply be represented by the numbers $0, 1, \ldots, m - 1$.

A.3 Algebra

In this section sets will be discussed with operations defined on them that are similar to the $+$ and $\times$ defined on the reals.

Let S be a nonempty set. Any mapping from $S \times S$ into S is called an *operation* on S. The image of the pair (s, t) under this operation will, for the moment, be denoted by $s * t$. An operation $*$ is called *commutative* if

$$\forall_{s,t \in S} \ [s * t = t * s]. \tag{A.20}$$

An element e in S that satisfies

$$\forall_{s \in S} \ [s * e = e * s = s], \tag{A.21}$$

will be called a *unit-element* of $(S, *)$. If $(S, *)$ has a unit-element, it is be unique. Indeed, suppose that e and e' both satisfy (A.21). Then by (A.21) $e = e * e' = e'$.

Example A.3.1 Consider the operation $+$ (i.e. addition) defined on $\mathcal{Z}$. This operation is commutative and $(\mathcal{Z}, +)$ has 0 as unit-element.

Example A.3.2 Let S be the set of 2×2 integer matrices. Multiplication of matrices is not a commutative operation. Consider for instance

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \neq \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

This set S does however have a unit-element, namely

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

When applying the same operation repeatedly one often does not like the outcome to depend on the order in which the operations were executed. In other words one does not want to need brackets in expressions like $s * t * u$. An operation $*$ is called *associative*, if

$$\forall_{s,t,u \in S} \; [s * (t * u) = (s * t) * u]. \tag{A.22}$$

Example A.3.3 The operation $\wedge$ defined on the positive integers by $s \wedge t = s^t$ (so $\wedge$ stands for "raising to the power") is not associative as follows from

$$(2 \wedge 3) \wedge 2 = (2^3)^2 = 2^6 \neq 2^9 = 2^{(3^2)} = 2 \wedge (3 \wedge 2).$$

Definition A.3.4 *Let G be a nonempty set and $*$ an operation defined on G. Then the pair $(G, *)$ is called a *g*roup, if*

$$\text{operation } * \text{ is associative}, \tag{G1}$$
$$G \text{ contains a unit-element, say } e, \tag{G2}$$
$$\forall_{g \in G} \; \exists_{h \in G} \; [g * h = h * g = e]. \tag{G3}$$

The uniqueness of the element h in (G3) can be seen quite easily. Indeed, if h and h' both satisfy (G3), then $h = h * e = h * (g * h') = (h * g) * h' = e * h' = h'$. The element h satisfying (G3) is called the *inverse* of g, and will be denoted by g^{-1}.

A group $(G, *)$ is called *commutative*, if the operation $*$ is commutative. The reader easily checks that $(\mathcal{Z}, +)$ in Example A.3.1 is a commutative group. Other well-known examples of commutative groups are for instance $(\mathcal{Q}, +)$, $(\mathcal{Q} \setminus \{0\}, \cdot)$, $(\mathcal{R}, +)$, etc.

Example A.3.2 does not yield a group, because not all matrices have an inverse (e.g. the all-one matrix).

On the set $\mathcal{Z}_m$ defined at the end of Section A.2 one can define two operations (called addition and multiplication) that coincide with the modular arithmetic explained in (A.16):

$$< s > + < t > := < s + t >, \quad s, t \in \mathcal{Z}, \tag{A.23}$$
$$< s > \times < t > := < st >, \quad s, t \in \mathcal{Z}. \tag{A.24}$$

The proof of the next lemma is left as an exercise (Problem A.6.5).

Lemma A.3.5 *The set $\mathcal{Z}_m$ together with the addition $+$, as defined in (A.23), is a commutative group with $< 0 >$ as unit-element.*

Lemma A.3.6 *The set $\mathcal{Z}_m^* = \mathcal{Z}_m \setminus \{< 0 >\}$ together with the multiplication $\times$, as defined in (A.24), is a group if and only if m is a prime p. In this case $(\mathcal{Z}_p^*, \times)$ is commutative and has $< 1 >$ as unit-element.*

Proof: Clearly $(\mathcal{Z}_m^*, \times)$ satisfies properties (G1) and (G2). That it satisfies (G3) if and only if m is a prime is a direct consequence of Theorem A.1.10.

$\square$

If $(G, *)$ is a group and G contains a non-empty subset H such that $(H, *)$ is also a group one calls $(H, *)$ a *subgroup* of $(G, *)$. Because associativity already holds for the operation $*$, $(H, *)$ will be a subgroup of $(G, *)$ iff

the product of any two elements in H also lies in H,	(H1)
the unit-element e of $(G, *)$ lies in H,	(H2)
the inverse of an element of H also lies in H.	(H3)

It is left as an exercise (see Problem A.6.6) to check that (H2) and (H3) are equivalent to

$$\forall_{g,h\in H} \; [g * h^{-1} \in H]. \tag{H}$$

Let $m \in \mathcal{Z} \setminus \{0\}$. Then $(m\mathcal{Z}, +)$, where $m\mathcal{Z}$ is the set of all integer multiples of m, so $m\mathcal{Z} = \{mk \mid k \in \mathcal{Z}\}$, is a commutative subgroup of $(\mathcal{Z}, +)$, as one can easily check.

We shall now consider the situation where two operations are defined on a set. (The first will be denoted by $g + h$, the second by $g \times h$, $g \cdot h$ or just gh.)

Definition A.3.7 *The triple $(R, +, \times)$ is called a r*ing*, if*

$(R, +)$ is a commutative group	*(R1)*
(its unit-element will be denoted by 0),	
the operation $\times$ is associative,	*(R2)*
distributivity holds, i.e.	*(R3)*
$\forall_{r,s,t\in R} \; [r(s + t) = rs + rt \quad and \quad (r + s)t = rt + st].$	

The (additive) inverse of an element r in the group $(R, +)$ will be denoted by $-r$, just as $2r$ denotes $r + r$, $3r$ denotes $r + r + r$, etc. Note that 0 really behaves like a zero-element, because for every $r \in R$ one has that $0r = (r - r)r = r^2 - r^2 = 0$ and similarly that $r0 = 0$.

Suppose that the (multiplicative) operation $\times$ is commutative on R. Then the ring $(R, +, \times)$ will be called *commutative*. Examples of commutative rings are $(\mathcal{Z}, +, \cdot)$,

$(\mathcal{Q}, +, \cdot)$, $(\mathcal{R}, +, \cdot)$ and $(\mathcal{Z}_m, +, \cdot)$, $m \neq 0$. Also $m\mathcal{Z}$, the set of m-tuples, with the same addition and multiplication as in $(\mathcal{Z}, +, \cdot)$ forms a commutative ring.

Let $(R, +, \times)$ be a ring and $S \subset R$, such that $(S, +, \times)$ is also a ring. Then $(S, +, \times)$ is called a *subring*. Clearly $(\mathcal{Q}, +, \cdot)$ is a subring of $(\mathcal{R}, +, \cdot)$. The ring $(m\mathcal{Z}, +, \cdot)$ is a subring of $(\mathcal{Z}, +, \cdot)$.

Later the following special class of subrings will play an important role.

Definition A.3.8 *A subring $(I, +, \times)$ of a ring $(R, +, \times)$ is called an* i*deal, if*

$$\forall_{i \in I} \ \forall_{r \in R} \ [ir \in I \ \ and \ \ ri \in I]. \tag{I}$$

It is easy to check that an integer multiple of a multiple of m, $m \in \mathcal{Z} \setminus \{0\}$, is also a multiple of m.. From this observation it follows that $(m\mathcal{Z}, +, \cdot)$ forms an ideal in $(\mathcal{Z}, +, \cdot)$.

In a ring $(R, +, \times)$ the element 0 obviously can not have a multiplicative inverse, unless R consists of the single element 0. Indeed, suppose that $r0 = e$, for some $r \in R$. Then for each $x \in R$ one has that $x = xe = x(r0) = (xr)0 = 0$, i.e. $R = \{0\}$.

It is however quite possible that $(R^*, \times)$ with $R^* = R \setminus \{0\}$ has the structure of a group.

Definition A.3.9 *A triple $(F, +, \times)$ is called a* f*ield, if*

$$(F, +) \ \textit{is a commutative group} \tag{F1}$$
$$\textit{(its unit-element is denoted by 0)},$$
$$(F \setminus \{0\}, \times) \ \textit{is a group} \tag{F2}$$
$$\textit{(the multiplicative unit-element is denoted by e)},$$
$$\textit{distributivity holds}. \tag{F3}$$

The multiplicative inverse of an element a in a field will be denoted by a^{-1}.

Examples of fields are the rationals $(\mathcal{Q}, +, \times)$, the reals $(\mathcal{R}, +, \times)$ and the complex numbers $(\mathcal{C}, +, \times)$.

Unlike some rings, a field can not have so-called *zero-divisors*, i.e. elements a and b, both unequal to 0, whose product ab equals 0. Indeed, suppose that $ab = 0$ and $a \neq 0$. Then $b = e \times b = (a^{-1}a)b = a^{-1}(ab) = a^{-1}0 = 0$.

If a subring $(K, +, \times)$ of a field $(F, +, \times)$ also has the structure of a field, we shall call $(K, +, \times)$ a *subfield* of $(F, +, \times)$.

In the sequel we shall be interested in a so-called *finite* groups $(G, *)$, finite rings $(R, +, \times)$ and finite fields $(F, +, \times)$ of *order n*, meaning that G, resp. R and F are (finite) sets of cardinality n.

Analogously to rings we define a *commutative* field $(F, +, \cdot)$ to be a field, for which the multiplication is commutative. The following theorem will not be proved.

Theorem A.3.10 (Wedderburn) *Every finite field is commutative.*

In this appendix we shall study the structure of finite fields. It will turn out that finite fields of order q only exist when q is a prime power. Moreover these finite fields are essentially unique for a given size q. This justifies the widely accepted notation $\mathcal{F}_q$ or $GF(q)$ (where GF stands for Galois Field after the Frenchman Galois) for a finite field of order q. Examples of finite fields will follow in Section A.4.

Before we conclude this section, some additional properties of finite groups need to be derived.

Let $(G, \cdot)$ be a finite multiplicative group with unit-element e and let g be a nonzero element in G. Let g^2, g^3, etc., denote $g \times g$, $g \times g \times g$, etc. Also, let g^0 denote the unit-element e. Consider the sequence of elements e, g, $g^2, \ldots$ in G. Since G is finite, there exists a unique integer n such that the elements $e, g, \ldots, g^{n-1}$ are all different, while $g^n = g^j$ for some j, $0 \le j < n$. It follows that $g^{n+1} = g^{j+1}$, etc. If $j > 0$, it would also follow that $g^{n-1} = g^{j-1}$, contradicting the assumption on n. We conclude that $j = 0$, i.e. $g^n = e$. So the elements g^i, $0 \le i \le n - 1$, are all distinct and $g^n = e$.

It is now clear that the elements $e, g, \ldots, g^{n-1}$ form a subgroup H in G. Such a (sub)group H is called a *cyclic* (sub)group of order n. We say that the element g *generates* H and that g has order n.

Lemma A.3.11 *Let $(G, \cdot)$ be a group and g an element in G of order n. Then for all m*

$$g^m = e \text{ iff } n \text{ divides } m.$$

Proof: Write $m = qn + r$, $0 \le r < n$. Then $g^m = e$ iff $g^r = e$, i.e. iff $r = 0$, i.e. iff $n \mid m$.

$\square$

Lemma A.3.12 *Let $(G, \cdot)$ be a group and g an element in G of order n. Then element g^k, $k > 0$, has order $n/\gcd(k, n)$.*

Proof: Let m be the order of g^k. Since $k/\gcd(k, n)$ is integer, it follows that

$$(g^k)^{n/\gcd(k,n)} = (g^n)^{k/\gcd(k,n)} = (e)^{k/\gcd(k,n)} = e.$$

From Lemma A.3.11 we conclude that m divides $n/\gcd(k, n)$. To prove the converse, we observe that $(g^k)^m = e$. Lemma A.3.11 implies that n divides km. Hence $n/\gcd(k, n)$ divides m.

$\square$

It follows from Definition A.1.6 that exactly $\phi(n)$ powers of an element g of order n in G will have order n.

Corresponding to a subgroup $(H, \cdot)$ of a finite group $(G, \cdot)$ an equivalence relation $\sim$ on

G can be defined (see Problem A.6.8) by

$$a \sim b \quad \text{iff} \quad ab^{-1} \in H. \tag{A.25}$$

The equivalence classes are of the form $\{ha \mid h \in H\}$, as one can easily check. They all have the same cardinality as H. It follows that the number of equivalence classes is $\mid G \mid / \mid H \mid$. As a consequence $\mid H \mid$ divides $\mid G \mid$. This proves the following theorem.

Theorem A.3.13 *Let $(G, \cdot)$ be a finite group of order n. Then every subgroup $(H, \cdot)$ of $(G, \cdot)$ has an order dividing n. Also every element g, $g \neq e$ in G has an order dividing n.*

A.4 Constructions

Let $(R, +, \times)$ be a commutative ring with (multiplicative) unit-element e. Let $(S, +, \times)$ be an ideal in $(R, +, \times)$. The relation $\equiv$ on R will be defined by

$$\forall_{a,b \in R} \ [a \equiv b \ (\text{mod } R) \text{ iff } a - b \in S]. \tag{A.26}$$

It is a simple exercise to check that (A.26) defines an equivalence relation on R. With R/S (read: R modulo S) the set of equivalence classes will be denoted. As before $< a >$ will stand for the class containing a, so $< a >= \{a+s \mid s \in S\}$. Based on the operations in R, one defines in R/S

$$< a > + < b >:= < a + b >, \quad a, b \in R, \tag{A.27}$$

$$< a > \times < b >:= < ab >, \qquad a, b \in R, \tag{A.28}$$

It is necessary to verify that these definitions are independent of the particular choice of the elements a and b in the equivalence classes $< a >$ resp. $< b >$. Again it is very elementary to check the following theorem.

Theorem A.4.1 *Let $(R, +, \times)$ be a commutative ring with unit-element e and let $(S, +, \times)$ be an ideal in $(R, +, \times)$. With the above definitions $(R/S, +, \times)$ is a commutative ring with unit-element $< e >$.*

The ring $(R/S, +, \times)$, defined above, is called a *residue class ring*. It is instrumental in constructing finite fields.

Since $(m\mathcal{Z}, +, \cdot)$ is an ideal in the commutative ring $(\mathcal{Z}, +, \cdot)$, one can with Theorem A.4.1 also describe the ring $(\mathcal{Z}_m, +, \cdot)$ as the residue class ring $(\mathcal{Z}/m\mathcal{Z}, +, \cdot)$. This residue class ring is commutative and has $< 1 >$ as multiplicative unit-element.

Theorem A.4.2 *Let m be a positive integer. The ring $(\mathcal{Z}_m, +, \cdot)$ is a finite field with m elements if and only if m is prime.*

Proof: By Lemma A.3.5 the commutative ring $(\mathcal{Z}_m, +, \cdot)$ (with m elements) satisfies property (F1). Also (F3) holds trivially.

The statement now follows directly from Lemma A.3.6. $\qquad\qquad\square$

In the next section it will be shown that for p prime, $(\mathcal{Z}_p, +, \cdot)$ is essentially the only field with p elements.

Our next goal is to to construct finite fields of size q, where $q = p^m$, p prime.

Let $(F, +, \cdot)$ be a commutative field (not necessarily finite). Then $F[x]$ denotes the set of *polynomials* over F, i.e. the set of expressions

$$f(x) = \sum_{i=0}^{n} f_i x^i = f_0 + f_1 x + \cdots + f_n x^n, \tag{A.29}$$

for some n in $\mathcal{N}$, where the f_i, $0 \le i \le n$, lie in F and are called the *coefficients* of $f(x)$. Unless $f_0 = f_1 = \ldots = f_n = 0$ one can assume without loss of generality that $f_n \ne 0$. This integer n is called the *degree* of $f(x)$.

Addition and multiplication of polynomials is defined in the obvious way.

$$\sum_{i=0}^{n} f_i x^i + \sum_{i=0}^{n} g_i x^i = \sum_{i=0}^{n} (f_i + g_i) x^i. \tag{A.30}$$

$$\left(\sum_{i=0}^{m} f_i x^i \right) \left(\sum_{j=0}^{n} g_j x^j \right) = \sum_{k=0}^{m+n} \left(\sum_{i+j=k} f_i g_j \right) x^k. \tag{A.31}$$

It is now straightforward to verify the next theorem.

Theorem A.4.3 *Let $(F, +, \cdot)$ be a commutative field. Then $(F[x], +, \cdot)$ is a commutative ring with unit-element.*

What will be done now for $F[x]$ will be completely analogous to what has been done for $\mathcal{Z}$ in the previous sections. For that reason, not all arguments will be repeated.

So, in $F[x]$ the following notions can be defined: *divisibility*, *reducibility* (if a polynomial can be written as the product of two polynomials of lower degree), *irreducibility* (which is the analog of primality), the greatest common divisor gcd, the least common multiple lcm, a unique factorization theorem (the analog of Theorem A.1.5), Euclid's Algorithm, congruence relations, etc. Note that the gcd and lcm are now uniquely determined up to a multiplication by a constant.

In particular we have the following theorem and corollary (see also (A.3) and Theorem A.1.10).

Theorem A.4.4 *Let $a(x)$ and $b(x)$ be polynomials in $F[x]$. Then there exist polynomials $u(x)$ and $v(x)$ in $F[x]$, such that*

$$u(x)a(x) + v(x)b(x) = \gcd(a(x), b(x)). \tag{A.32}$$

Corollary A.4.5 *Let $a(x)$ and $f(x)$ be polynomials in $F[x]$ such that $a(x) \not\equiv 0$ (mod $f(x)$). The linear congruence relation $a(x)u(x) \equiv b(x)$ (mod $f(x)$) has a solution $u(x)$ for every $a(x)$, $a(x) \not\equiv 0$ (mod $f(x)$), and every $b(x)$ if and only if $f(x)$ is irreducible.*

If $f(x)$ is irreducible, this solution is unique modulo $f(x)$. It will be denoted by $(a(x))^{-1}b(x)$.

Let $s(x) \in F[x]$. It is easy to check that the set $\{a(x)s(x) \mid a(x) \in F[x]\}$ forms an ideal in the ring $(F[x], +, \cdot)$. We denote this ideal by $(s(x))$ and say that $s(x)$ *generates* the ideal $(s(x))$.

Conversely, let $(S, +, \cdot)$ be any ideal in $(F[x], +, \cdot)$, $S \neq F[x]$. Let $s(x)$ be a polynomial of lowest degree in S. Take any other polynomial $f(x)$ in S and write $f(x) = q(x)s(x)+r(x)$, degree$(r(x)) <$ degree$(s(x))$. With properties (I) and (R1), we then have that also $r(x)$ is an element of S. From our assumption on $s(x)$ we conclude that $r(x) = 0$ and thus that $s(x) \mid f(x)$.

It follows from the above discussion that any ideal in the ring $(F[x], +, \cdot)$ can be generated by a single element! A ring with this property is called a *principal ideal ring*.

From now on we shall restrict ourselves to finite fields.

Let $f(x) \in \mathcal{F}_p[x]$ of degree n. We shall say that f is a *p-ary* polynomial. Let $(f(x))$ be the ideal generated by $f(x)$. By Theorem A.4.3 we know that $(\mathcal{F}_p[x]/(f(x)), +, \cdot)$ is a commutative ring with unit-element $< 1 >$. It contains p^n elements.

Theorem A.4.6 *Let $(\mathcal{F}_p, +, \cdot)$ be a finite field with p elements. Let $f(x)$ be a polynomial of degree n over $\mathcal{F}_p$. Then the commutative ring $(\mathcal{F}_p[x]/(f(x)), +, \cdot)$ with p^n elements is a (finite) field if and only if $f(x)$ is irreducible over $\mathcal{F}_p$.*

Proof: The proof is analogous to that of Theorem A.4.2 and follows directly from Corollary A.4.5.

$\square$

Example A.4.7 Let $q = 2$. The field $\mathcal{F}_2$ consists of the elements 0 and 1. Let $f(x) = 1 + x^3 + x^4$. Then $f(x)$ does not have 0 or 1 as zeroes, so $f(x)$ does not contain a linear factor. Also $f(x)$ is not divisible by $x^2 + x + 1$, the only irreducible, binary polynomial of degree 2. We conclude that $f(x)$ is irreducible and that $(\mathcal{F}_2[x]/(f(x)), +, \cdot)$ is a finite field with $2^4 = 16$ elements. These sixteen elements can be represented by the sixteen binary polynomials of degree less than 4. The addition and multiplication, as defined by (A.30) and (A.31), have to be performed modulo $x^4 + x^3 + 1$. For instance

$$
\begin{aligned}
(1 + x + x^3)(1 + x^2) &\equiv 1 + x + x^2 + x^5 \equiv \\
&\equiv (x + 1)(x^4 + x^3 + 1) + x^2 + x^3 \equiv \\
&\equiv x^2 + x^3 \quad (\text{mod } x^3 + x + 1).
\end{aligned}
$$

Two questions that arise naturally at this moment are:

1. Does Theorem A.4.6 yield finite fields of size p^n for all primes p and integers n? In other words: does an irreducible, p-ary polynomial $f(x)$ of degree n exist for every prime number p and every n?

2. Do other finite fields exist?

The second question gets a negative answer in the next section. For the first question we shall state without proof an affirmative answer in the form of an explicit formula for the number of irreducible p-ary polynomials of degree n.

Let a be a nonzero element in $\mathcal{F}_q$ (at this moment we only know finite fields of size p with p prime). Clearly if $f(x)$ is irreducible over $\mathcal{F}_q$ then so is $af(x)$. Also the ideals in $\mathcal{F}_q[x]$ generated by $f(x)$ and by $af(x)$ are the same. So, in view of Theorem A.4.6 we shall only be interested in so-called *monic* polynomials of degree n, i.e. polynomials, whose leading coefficient (the coefficient of x^n) is equal to 1.

Before we can give an explicit expression for the number of monic, q-ary polynomials of degree n, we need to derive some more tools.

Definition A.4.8 *Let $n = \prod_{i=1}^{k} p_i^{e_i}$, where the p_i's are different primes and the e_i's positive, $1 \le i \le k$. Then the Möbius function $\mu(n)$ is defined by*

$$\mu(n) = \begin{cases} 1, & \text{if } n = 1, \\ 0, & \text{if } e_i \ge 2 \text{ for some } i, \ 1 \le i \le k, \\ (-1)^k, & \text{if } e_1 = e_2 = \cdots = e_k = 1. \end{cases}$$

In particular, $\mu(1) = 1$, $\mu(p) = -1$ and $\mu(p^i) = 0$, $i \ge 2$, for any prime p.

The Möbius function is defined in this way to satisfy the following property.

Theorem A.4.9 *Let n be a positive integer. Then*

$$\sum_{d|n} \mu(d) = \begin{cases} 1, & \text{if } n = 1, \\ 0, & \text{if } n > 1. \end{cases} \tag{A.33}$$

Proof: For $n = 1$ the assertion is trivial. For $n > 1$ we write, as above, $n = \prod_{i=1}^{k} p_i^{e_i}$, $e_i > 0$, $1 \le i \le k$. Then $k > 0$ and thus

$$\sum_{d|n} \mu(d) = \sum_{d|p_1 p_2 \cdots p_k} \mu(d) =$$

$$= 1 + \sum_{l=1}^{k} \sum_{1 \le i_1 < i_2 < \cdots < i_l \le k} \mu(p_{i_1} p_{i_2} \cdots p_{i_l}) =$$

$$\sum_{l=0}^{k} \binom{k}{l} (-1)^l = (1-1)^k = 0.$$

$\square$

Theorem A.4.10 *Let m and n be two positive integers such that $m \mid n$. Then*

$$\sum_{d,\ m|d|n} \mu(n/d) = \begin{cases} 1, & \text{if } m = n, \\ 0, & \text{otherwise.} \end{cases} \tag{A.34}$$

Proof: Let $n = n'm$. For each d, $m \mid d \mid n$, write $d = d'm$. Then $\sum_{d,\ m|d|n} \mu(n/d) = \sum_{d',d'|n'} \mu(n'/d')$, which by (A.33) is 1 for $n' = 1$, (i.e. $m = n$), and 0 for $n' > 1$.

$\square$

Theorem A.4.11 (Möbius Inversion Formula) *Let f be a function defined on $\mathcal{N}$ and let g be defined on $\mathcal{N}$ by*

$$g(n) = \sum_{d|n} f(d), \qquad n \in \mathcal{N}. \tag{A.35}$$

Then for all $n \in \mathcal{N}$

$$f(n) = \sum_{d|n} \mu(d)g(n/d) = \sum_{d|n} \mu(n/d)g(d). \tag{A.36}$$

Proof: By (A.35) and (A.34)

$$\sum_{d|n} \mu(n/d)g(d) = \sum_{d|n} \mu(n/d) \sum_{e|d} f(e) =$$

$$= \sum_{e|n} f(e) \sum_{d,\ e|d|n} \mu(n/d) = f(n).$$

$\square$

Definition A.4.12 *Let q be such that a finite field of cardinality q exists. Then we define*

$$I_q(n) = \# \ q\text{-ary, irreducible, monic polynomials of degree } n.$$

Without proof we state the following theorem:

Theorem A.4.13

$$\sum_{d|n} d I_q(d) = q^n \tag{A.37}$$

and thus by Theorem A.4.11

$$I_q(n) = \frac{1}{n} \sum_{d|n} \mu(d) q^{n/d}. \tag{A.38}$$

In particular

$$\frac{q^n}{n} \left\{ 1 - \frac{1}{q^{n/2-1}} \right\} \le I_q(n) \le \frac{q^n}{n} \left\{ 1 - \frac{1}{q^{n-1}} \right\}, \tag{A.39}$$

and

$$I_q(n) > 0. \tag{A.40}$$

An important property of the ring $F[x]$ of polynomials over a field F and one which will be needed in the next section is the following theorem.

Theorem A.4.14 *Any polynomial of degree n, $n > 0$, over a field F has at most n zeroes in F.*

Proof: For $n = 1$ the statement is trivial.

We proceed by induction on n. Let $u \in F$ be a zero of a polynomial $f(x)$ in $F[x]$ of degree n (if no such u exists, there is nothing to prove). Write $f(x) = (x-u)q(x)+r(x)$, $\mathrm{degree}(r(x)) < \mathrm{degree}(x-u) = 1$. It follows that $r(x)$ is a constant r. Substituting $x = u$ yields $r = 0$. So $f(x) = (x-u)q(x)$. Since $q(x)$ has degree $n-1$, it follows from the induction hypothesis that $q(x)$ will have at most $n-1$ zeroes in F. Because a field does not have zero-divisors, it follows from $f(x) = 0$ that $x = r$ or $q(x) = 0$. From this we deduce that $f(x)$ has at most n zeroes in F.

$\square$

Note that in Theorem A.4.14 the zeroes do not have to be distinct.

A.5 The structure of finite fields

It follows from Theorems A.4.2, A.4.6 and inequality (A.40), that finite fields $(\mathcal{F}_q, +, \times)$ exist for all prime powers q. We state this as a theorem.

Theorem A.5.1 *Let p be a prime and $q = p^m$, $m \geq 1$. Then a finite field of order q exists.*

Later in this section we shall see that every finite field can be described by the construction of Theorem A.4.6, but first we shall prove an extremely useful property of finite fields: their multiplicative group is cyclic.

Since the multiplicative group of $\mathcal{F}_q$ consists of $q-1$ elements, it follows from Theorem A.3.13 that every nonzero element in $\mathcal{F}_q$ has an order dividing $q-1$.

Definition A.5.2 *An element α in a finite field of order q is called an n-th root of unity if $\alpha^n = e$. An element α is called a primitive n-th root of unity if its order is exactly n. If α is a primitive $(q-1)$-st root of unity, then α is called a primitive element of $\mathcal{F}_q$.*

If one can show that $\mathcal{F}_q$ must contain a primitive element α, then the multiplicative group of $\mathcal{F}_q$ is cyclic and will be generated by α.

Theorem A.5.3 *Let $(\mathcal{F}_q, +, \times)$ be a finite field and let d be an integer dividing $q-1$. Then $\mathcal{F}_q$ contains exactly $\phi(d)$ elements of order d. In particular $(\mathcal{F}_q \setminus \{0\}, \times)$ is a cyclic group of order $q-1$, which contains $\phi(q-1)$ primitive elements.*

Proof: By Theorem A.3.13 every nonzero element in $\mathcal{F}_q$ has a multiplicative order d, dividing $q - 1$. On the other hand, suppose that $\mathcal{F}_q$ contains an element of order d, $d \mid (q - 1)$, say α. Then by Theorem A.4.14 (the zeroes of $x^d - 1$ are precisely $1, \alpha, \ldots, \alpha^{d-1}$) every d-th root of unity in $\mathcal{F}_q$ is a power of α. It follows from the remark below Lemma A.3.12 that $\mathcal{F}_q$ contains exactly $\phi(d)$ elements of order d, namely the elements α^i, with $0 \le i < d$ and $\gcd(i, d) = 1$.

Let $a(d)$ be the number of elements of order d in $\mathcal{F}_q$. Then the reasoning above implies that i) $a(d)$ is either 0 or equal to $\phi(d)$ and also that ii) $\sum_{d \mid q-1} a(d) = q - 1$. On the other hand, Theorem A.1.7 states that $\sum_{d \mid q-1} \phi(d) = q - 1$. It follows that $a(d) = \phi(d)$ for all $d \mid (q - 1)$.

$\square$

Corollary A.5.4 *Every element α in $\mathcal{F}_q$ satisfies*

$$\alpha^{q^n} = \alpha, \quad n \ge 1. \tag{A.41}$$

Proof: Equation (A.41) trivially holds for $\alpha = 0$. By Theorem A.3.13 or Theorem A.5.3 any nonzero element α in $\mathcal{F}_q$ has an order dividing $q - 1$. So it satisfies $\alpha^{q-1} = e$. Since $(q - 1) \mid (q^n - 1)$, it follows also that $\alpha^{q^n-1} = e$ and thus that $\alpha^{q^n} = \alpha$.

$\square$

Corollary A.5.5 *Let $\mathcal{F}_q$ be a finite field. Then*

$$x^q - x = \prod_{\alpha \in \mathcal{F}_q} (x - \alpha). \tag{A.42}$$

Proof: Every element α in $\mathcal{F}_q$ is a zero of $x^q - x$ by Corollary A.5.4. So the right hand side in (A.42) divides the left hand side. Because both sides in (A.42) are monic and of the same degree, equality holds.

$\square$

Example A.5.6 Consider the finite field $(\mathcal{F}_2[x]/(f(x)), +, \times)$, with $f(x) = x^4 + x^3 + x^2 + x + 1$. It contains $2^4 = 16$ elements, which can be represented by binary polynomials of degree < 4. The element x, representing the class $< x >$, is not a primitive element, since $x^5 \equiv (x + 1)f(x) + 1 \equiv 1 \pmod{f(x)}$. So x has order 5 instead of 15.

The element $1 + x$ is primitive, as one can verify in Table A.1, where the successive powers of $1 + x$ are written as polynomials in x of degree < 4. Multiplication of two field elements can be converted by means of Table A.1 into the much easier multiplication of two powers of the primitive 15-th root of unity $1 + x$ (the exponents only have to be added modulo 15). For instance

$$(1 + x + x^2 + x^3)(x + x^3) \equiv (1 + x)^3 (1 + x)^{14} \equiv (1 + x)^{17} \equiv$$

$$\equiv (1 + x)^2 \equiv 1 + x^2 \pmod{f(x)}.$$

	1	x	x^2	x^3
0	0	0	0	0
$(1+x)^0$	1	0	0	0
$(1+x)^1$	1	1	0	0
$(1+x)^2$	1	0	1	0
$(1+x)^3$	1	1	1	1
$(1+x)^4$	0	1	1	1
$(1+x)^5$	1	0	1	1
$(1+x)^6$	0	0	0	1
$(1+x)^7$	1	1	1	0
$(1+x)^8$	1	0	0	1
$(1+x)^9$	0	0	1	0
$(1+x)^{10}$	0	0	1	1
$(1+x)^{11}$	1	1	0	1
$(1+x)^{12}$	0	1	0	0
$(1+x)^{13}$	0	1	1	0
$(1+x)^{14}$	0	1	0	1

Table A.1: $(\mathcal{F}_2[x]/(x^4 + x^3 + x^2 + x + 1), +, \times)$, with primitive element $1 + x$.

The element $x + 1$ is a zero of the irreducible polynomial $y^4 + y^3 + 1$, as can be checked from $(1 + x)^4 + (1 + x)^3 + 1 \equiv (1 + x^4) + (1 + x + x^2 + x^3) + 1 \equiv 0 \pmod{f(x)}$. So in $(\mathcal{F}_2[x]/(g(x)), +, \cdot)$, with $g(x) = x^4 + x^3 + 1$, the element x is a primitive element. See Table A.2.

Tables A.1 and A.2 are both so-called *log tables* of $GF(16)$.

As a first step in proving that every finite field has a size that is the power of a prime, consider the successive sums of the unit-element e in $\mathcal{F}_q$: so e, $2e$, $3e$, etc. Since $\mathcal{F}_q$ is finite, not all these sums can be different. On the other hand, if $ie = je$, $i < j$, then also $(j - i)e = 0$. These observations justify the following definition.

Definition A.5.7 *The characteristic of a finite field $\mathcal{F}_q$ with unit-element e, is the smallest positive integer k such that $ke = 0$.*

Lemma A.5.8 *The characteristic of a finite field $\mathcal{F}_q$ is a prime.*

Proof: Suppose that the characteristic k is composite, i.e. $k = ij$, where $i > 1$ and $j > 1$. Then $0 = ke = (ie)(je)$, while $ie \neq 0$ and $je \neq 0$. So ie and je would be zero-divisors, contradicting that $\mathcal{F}_q$ is a field.

$\square$

Finite fields of the same size may be represented in different ways but still have the same intrinsic structure.

	1	x	x^2	x^3
0	0	0	0	0
1	1	0	0	0
x^1	0	1	0	0
x^2	0	0	1	0
x^3	0	0	0	1
x^4	1	0	0	1
x^5	1	1	0	1
x^6	1	1	1	1
x^7	1	1	1	0
x^8	0	1	1	1
x^9	1	0	1	0
x^{10}	0	1	0	1
x^{11}	1	0	1	1
x^{12}	1	1	0	0
x^{13}	0	1	1	0
x^{14}	0	0	1	1

Table A.2: $(\mathcal{F}_2[x]/(x^4 + x^3 + 1), +, \times)$, with primitive element x.

Definition A.5.9 *Two finite fields $(\mathcal{F}_q, +, \cdot)$ and $(\mathcal{F}_{q'}, \oplus, \odot)$ are said to be isomorphic, if there exists a one-to-one mapping ψ from $\mathcal{F}_q$ onto $\mathcal{F}_{q'}$ (so $q = q'$), such that for all a and b in $\mathcal{F}_q$*

$$\psi(\alpha + \beta) = \psi(\alpha) \oplus \psi(\beta)$$

$$\psi(\alpha \times \beta) = \psi(\alpha) \odot \psi(\beta).$$

In words: two fields are isomorphic if the addition and multiplication tables of both fields are the same, apart from a renaming of the fields elements.

Lemma A.5.10 *Let $(\mathcal{F}_q, +, \cdot)$ be a finite field of characteristic p. Then $(\mathcal{F}_q, +, \cdot)$ contains a subfield which is isomorphic to $(\mathcal{Z}_p, +, \cdot)$, the field of integers modulo p.*

Proof: Let e be the unit-element of $(\mathcal{F}_q, +, \cdot)$. The subset $\{ie \mid i = 0, 1, \ldots, p-1\}$ forms a subfield of $(\mathcal{F}_q, +, \cdot)$ which is isomorphic to $(\mathcal{Z}_p, +, \cdot)$ under the (obvious) isomorphism $\psi(ie) = < i >, \ 0 \leq i < p$.

□

In view of Lemma A.5.10 we can and shall from now on identify the above mentioned subfield of order p in a finite field (of characteristic p) with the field $\mathcal{Z}_p$. The subfield $\mathcal{F}_p$ is often called the *ground field* of $\mathcal{F}_q$. Conversely the field $\mathcal{F}_q$ is called an *extension field* of $\mathcal{F}_p$.

Theorem A.5.11 *Let $\mathcal{F}_q$ be a finite field of characteristic p. Then $q = p^m$, for some integer m, $m \geq 1$.*

Proof: Let m be the size of the smallest *basis* of $\mathcal{F}_q$ over $\mathcal{F}_p$, i.e. m is the smallest integer such that for suitably chosen elements α_i, $1 \leq i \leq m$, in $\mathcal{F}_q$ every element x in $\mathcal{F}_q$ can be written as

$$x = u_1\alpha_1 + u_2\alpha_2 + \cdots + u_m\alpha_m,$$

where $u_i \in \mathcal{F}_p$, $1 \leq i \leq m$.

Clearly $q \leq p^m$. On the other hand it follows from the minimality of m that different m-tuples $u_1, u_2, \ldots, u_m$ yield different field elements x. So $p^m \geq q$. We conclude that $q = p^m$.

$\square$

At this moment we know that a finite field $\mathcal{F}_q$ can only exist for prime powers q. Theorem A.5.1 states that $\mathcal{F}_q$ indeed does exist if q is a prime power. That all finite fields of the same size are in fact isomorphic to each other (and thus can be constructed by Theorem A.4.6) will be proved later.

Theorem A.5.12 *Let α be an element in a finite field $\mathcal{F}_q$ of characteristic p. Then over $\mathcal{F}_p$ (and any extension field of it)*

$$(x - \alpha)^p = x^p - \alpha^p. \tag{A.43}$$

Proof: Let $0 < i < p$. Then $\gcd(p, i!) = 1$, so

$$\binom{p}{i} \equiv \frac{p(p-1)\cdots(p-i+1)}{i(i-1)\cdots 2 \cdot 1} \equiv p\frac{(p-1)\cdots(p-i+1)}{i(i-1)\cdots 2 \cdot 1} \equiv 0 \;(\mathrm{mod}\; p).$$

And so with the binomial theorem, we have that

$$(x - \alpha)^p = x^p + (-\alpha)^p = x^p - \alpha^p,$$

where the last equality is obvious for odd p, while for $p = 2$ this equality follows from $+1 = -1$.

$\square$

Corollary A.5.13 *Let α_i, $1 \leq i \leq k$, be elements in a finite field $\mathcal{F}_q$ of characteristic p. Then for every n*

$$\left(\sum_{i=1}^{k}\alpha_i\right)^{p^n} = \sum_{i=1}^{k}\alpha_i^{p^n}.$$

Proof: For $k = 2, n = 1$ the statement follows again from the previous theorem (take $x = -\alpha_1, \alpha = \alpha_2$). Next use an induction argument on k and finally on n.

136

□

The following theorem often gives a powerful criterion to determine whether an element in a field $\mathcal{F}_q$ of characteristic p, actually lies in its ground field $\mathcal{F}_p$.

Theorem A.5.14 *Let $\mathcal{F}_q$ be a finite field of characteristic p. So $q = p^m$, $m > 0$, and $\mathcal{F}_q$ contains $\mathcal{F}_p$ as a subfield. Let α be an element in $\mathcal{F}_q$. Then α is an element of the subfield $\mathcal{F}_p$ if and only if α satisfies*

$$\alpha^p = \alpha. \tag{A.44}$$

Proof: The p elements in the subfield $\mathcal{F}_p$ satisfy (A.44) by Corollary A.5.4. On the other hand the polynomial $x^p - x$ has at most p zeroes in $\mathcal{F}_q$ by Theorem A.4.14, so it can not have any other zeroes.

□

Let α be an element in $\mathcal{F}_q$, a field of characteristic p, but $\alpha \notin \mathcal{F}_p$. Then $\alpha^p \neq \alpha$ by the previous theorem. Still there is a relation between α^p and α.

Theorem A.5.15 *Let α be an element in a finite field $\mathcal{F}_q$ of characteristic p. Let $f(x)$ be a polynomial over $\mathcal{F}_p$ such that $f(\alpha) = 0$. Then for all $n \in \mathcal{N}$*

$$f(\alpha^{p^n}) = 0. \tag{A.45}$$

Proof: Write $f(x) = \sum_{i=0}^m f_i x^i$, $f_i \in \mathcal{F}_p$. By Theorem A.5.14, $f_i^{p^n} = f_i, 0 \leq i \leq m$. Hence by Corollary A.5.13

$$0 = (f(\alpha))^{p^n} = (\sum_{i=0}^m f_i \alpha^i)^{p^n} = \sum_{i=0}^m (f_i \alpha^i)^{p^n} =$$

$$\sum_{i=0}^m f_i^{p^n} \alpha^{i \cdot p^n} = \sum_{i=0}^m f_i (\alpha^{p^n})^i = f(\alpha^{p^n}).$$

□

In the field $\mathcal{R}$ and its extension field $\mathcal{C}$ a similar phenomenon occurs. If $f(x)$ is a polynomial over the reals and $f(a) = 0$, $a \in \mathcal{C}$, then also $f(\bar{a}) = 0$, where $\bar{a}$ is the complex conjugate of a.

The following theorem states that the number of different elements α^{p^n}, $n = 0, 1, \ldots$, only depends on the numbers p and the order of α. Note that an element in a finite field of characteristic p will have an order dividing $p^m - 1$ for some value of m and thus this order will have no factor in common with p.

Theorem A.5.16 *Let α be an element of order n in a finite field of characteristic p. Let m be the multiplicative order of p modulo n, i.e. $p^m \equiv 1 \pmod{n}$, with $m \geq 1$ minimal. Then the m elements*

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

are all distinct and $\alpha^{p^m} = \alpha$.

Proof: By Lemma A.3.11 (twice) one has that $\alpha^{p^i} = \alpha^{p^j}$ if and only if $p^i \equiv p^j \pmod{n}$, i.e. iff $p^{i-j} \equiv 1 \pmod{n}$, i.e. iff $i \equiv j \pmod{m}$.

$\square$

The m elements $\alpha, \alpha^p, \alpha^{p^2}, \ldots, \alpha^{p^{m-1}}$ in Theorem A.5.16 are called the *conjugates* of α.

Example A.5.17 Consider $(\mathcal{F}_2[x]/(f(x)), +, \cdot)$ with $f(x) = x^4 + x^3 + x^2 + x + 1$ (see Example A.5.6). The element x has order 5. The multiplicative order of 2 modulo 5 is 4. So x, x^2, x^{2^2} and x^{2^3} are all different, while $x^{2^4} = x$. Indeed, $x^8 \equiv x^3 \pmod{f(x)}$ and $x^{16} \equiv x \pmod{f(x)}$.

Theorem A.5.18 *Let α be an element of order n in a finite field $\mathcal{F}_q$ of characteristic p. Let m be the multiplicative order of p modulo n. Then the q-ary polynomial*

$$m(x) = \prod_{i=0}^{m-1}(x - \alpha^{p^i}) \tag{A.46}$$

is in fact a polynomial with coefficients in $\mathcal{F}_p$, moreover $m(x)$ is irreducible over $\mathcal{F}_p$.

Proof: Clearly $m(x)$ is a polynomial over $\mathcal{F}_q$. Write $m(x) = \sum_{i=0}^{m} m_i x^i$. Then by Theorems A.5.12 and A.5.16

$$(m(x))^p = \prod_{i=0}^{m-1}(x - \alpha^{p^i})^p = \prod_{i=0}^{m-1}(x^p - \alpha^{p^{i+1}}) =$$

$$= \prod_{i=1}^{m}(x^p - \alpha^{p^i}) = \prod_{i=0}^{m-1}(x^p - \alpha^{p^i}) = m(x^p).$$

Hence

$$\sum_{i=0}^{m} m_i x^{pi} = m(x^p) = (m(x))^p = \left(\sum_{i=0}^{m} m_i x^i\right)^p = \sum_{i=0}^{m} m_i^p x^{pi}.$$

Comparing the coefficients of x^{pi} on both hands yields $m_i = m_i^p$. From Theorem A.5.14 we conclude that $m_i \in \mathcal{F}_p$, $0 \le i \le m$. So $m(x)$ is a polynomial in $\mathcal{F}_p[x]$. From Theorems A.5.15 and A.5.16 it follows that no polynomial in $\mathcal{F}_p[x]$ of degree less than m can have α as a zero. So $m(x)$ is irreducible in $\mathcal{F}_p[x]$.

$\square$

Corollary A.5.19 *Let α be an element of order n in a finite field of characteristic p. Let $m(x)$ be defined by (A.46) and let $f(x)$ be any p-ary polynomial that has α as zero. Then $f(x)$ is a multiple of $m(x)$.*

Proof: Combine Theorems A.5.15, A.5.16 and A.5.18.

$\square$

So $m(x)$ is the monic polynomial of lowest degree over $\mathcal{F}_p$, having α as a zero. For this reason the polynomial $m(x)$ is called the *minimal polynomial* of α. It has α and all the

conjugates of α as zeroes. The degree of the minimal polynomial $m(x)$ of an element α is often simply called the *degree* of α.

If $m(x)$ is the minimal polynomial of a primitive element, then $m(x)$ is called a *primitive polynomial*. Let α be a zero of a primitive polynomial $m(x)$. Often the minimal polynomial of a power α^i will be denoted by $m_i(x)$.

Example A.5.20 Consider $(\mathcal{F}_2[x]/(f(x)), +, \cdot)$ with $f(x)$ a fourth degree irreducible polynomial and let α be one of the primitive elements of this field.

Clearly $m_0(x) = x - 1$. Also $m_3(x)$ and $m_5(x)$ are easily determined. Indeed, $m_5(x)$ has as zeroes α^5 and its conjugates, so α^5 and α^{10}. Both these elements have order 3, and thus they are both zeroes of $(x^3 - 1)/(x - 1) = x^2 + x + 1$. We conclude that $m_5(x) = x^2 + x + 1$. For exactly the same reason the four primitive 5-th roots of unity $\alpha^3, \alpha^6, \alpha^9, \alpha^{12}$ form the zeroes of $m_3(x) = x^4 + x^3 + x^2 + x + 1$.

Since there are only two irreducible polynomials of degree four left, being $x^4 + x + 1$ and $x^4 + x^3 + 1$, it follows that $m_1(x) = x^4 + x + 1$ and $m_{-1}(x) = m_7(x) = x^4 + x^3 + 1$, or the other way around. Both apparently are primitive polynomials.

Let $m(x)$ be the minimal polynomial of an element α of degree m. It follows from Corollary A.5.19 that the p^m expressions $\sum_{i=0}^{m-1} f_i \alpha^i$, $f_i \in \mathcal{F}_p$, $0 \le i < m$, take on p^m different values. For these expressions addition and multiplication can be performed, just as in (A.30) and (A.31), where now $m(\alpha) = 0$ has to be used to reduce the degree of the outcome to a value less than m. It is quite easy to check that one obtains a field, that is isomorphic to $\mathcal{F}_p[x]/(m(x))$ and that clearly contains α.

If $m(x)$ is primitive, one has that $1, x, \ldots, x^{p^m-2}$ are all different modulo $m(x)$, just as the elements $1, \alpha, \ldots, \alpha^{p^m-2}$ are all different. See for instance Example A.5.6, where the primitive element $a = 1 + x$ has minimal polynomial $m(y) = 1 + y^3 + y^4$. Table A.2 shows the field $\mathcal{F}_2[y]/(m(y))$.

Lemma A.5.21 *Let $m(x)$ be an irreducible polynomial of degree m over a field with p elements and let n be a multiple of m. Then $m(x)$ divides $x^{p^n} - x$.*

Proof: Consider the residue class ring $\mathcal{F}_p[x]/(m(x))$. This ring is a field with $q = p^m$ elements by Theorem A.4.6. The field element $\alpha = < x >$ is a zero of $m(x)$, since $m(\alpha) = m(< x >) = < m(x) > = < 0 >$. It follows from Corollary A.5.4 that $\alpha = < x >$ is a zero of $x^{p^n} - x$, $n \ge 1$. By Corollary A.5.19 $m(x)$ divides $x^{p^n} - x$.

$\square$

Also the converse of Lemma A.5.21 is true.

Theorem A.5.22 *Over $\mathcal{F}_p[x]$ the polynomial $x^{p^n} - x$ is the product of all irreducible, monic, p-ary polynomials with a degree dividing n.*

Proof: Let $m \mid n$. By definition, there are $I_p(m)$ irreducible p-ary polynomials of degree m, all of which divide $x^{p^n} - x$ by Lemma A.5.21. The sum of their degrees is $m I_p(m)$.

But $\sum_{m|n} m I_p(m) = p^n$ by (A.37), which is the degree of $(x^{p^n} - x)$. It follows that the product of all irreducible, monic, p-ary polynomials of degree m, $m \mid n$, forms the complete factorization of $x^{p^n} - x$.

$\square$

Example A.5.23 $p = 2$, $n = 4$, $I_2(1) = 2$, $I_2(2) = 1$ and $I_2(4) = 3$ (see Section A.4). As follows from Theorem A.5.22, but can also be checked directly, one indeed has that

$$x^{16} - x =$$

$$= x(x + 1)(x^2 + x + 1)(x^4 + x + 1)(x^4 + x^3 + 1)(x^4 + x^3 + x^2 + x + 1).$$

Corollary A.5.24 *Let $f(x)$ be an irreducible polynomial in $\mathcal{F}_p[x]$ of degree m. Let $m \mid n$. Then a finite field with p^n elements contains m roots of $f(x)$.*

Proof: By Theorem A.5.22 $f(x)$ divides $x^q - x$, $q = p^n$. On the other hand $x^q - x = \prod_{\alpha \in \mathcal{F}_q}(x - \alpha)$, by (A.42).

$\square$

Theorem A.5.25 *Let p be a prime and m in $\mathcal{N}$. Then the finite field $\mathcal{F}_{p^m}$ is unique, up to isomorphism.*

Proof: Write $q = p^m$ and let $\mathcal{F}_q$ be any finite field of order q. Let $f(x)$ be any irreducible, p-ary polynomial of degree m. We shall show that $\mathcal{F}_q$ is isomorphic to $\mathcal{F}_p[x]/(f(x))$.

By Corollary A.5.24 $\mathcal{F}_q$ contains m zeroes of $f(x)$. Let α be one of these m zeroes. Since $f(x)$ is irreducible in $\mathcal{F}_p[x]$, we know that $1, \alpha, \ldots, \alpha^{m-1}$ are independent over $\mathcal{F}_p$. So any element in $\mathcal{F}_q$ can be uniquely written as $\sum_{i=0}^{m-1} u_i \alpha^i$, $u_i \in \mathcal{F}_p$, $0 \leq i \leq m - 1$.

The isomorphism $\mathcal{F}_q$ between and $\mathcal{F}_p[x]/(f(x))$ is given by $\psi(\sum_{i=0}^{m-1} u_i \alpha^i) = \sum_{i=0}^{m-1} u_i x^i$.

$\square$

Corollary A.5.26 *$\mathcal{F}_{p^m}$ is (isomorphic to) a subfield of $\mathcal{F}_{p^n}$ if and only if m divides n.*

Proof: The following assertions are all equivalent:

    i) $m \mid n$,

    ii) $(p^m - 1) \mid (p^n - 1)$,

    iii) $x^{p^m} - x$ divides $x^{p^n} - x$,

    iv) $\prod_{\alpha \in \mathcal{F}_{p^m}}(x - \alpha)$ divides $\prod_{\alpha \in \mathcal{F}_{p^n}}(x - \alpha)$,

    v) $\mathcal{F}_{p^m}$ is (isomorphic to) a subfield of $\mathcal{F}_{p^n}$. $\quad\square$

Example A.5.27 It follows from Corollary A.5.26 that $\mathcal{F}_{2^4}$ contains $\mathcal{F}_{2^2}$ as a subfield, but does not contain $\mathcal{F}_{2^3}$ as a subfield. From Table A.2 one can easily verify that the elements $0, 1, x^5$ and x^{10} form the subfield of order 2^2 in $\mathcal{F}_2[x]/(x^4 + x^3 + 1)$.

Most of the time in this appendix we have viewed $\mathcal{F}_q$, $q = p^m$ and p prime, as an extension field of $\mathcal{F}_p$. But all the concepts, defined in this appendix, can also be generalized to $\mathcal{F}_q[x]$. So one may want to count the number of irreducible polynomials of degree n in $\mathcal{F}_q[x]$ or discuss primitive polynomials over $\mathcal{F}_q$, etc. We leave it to the reader to verify that all the theorems in this appendix can indeed be generalized from $\mathcal{F}_p$ and $\mathcal{F}_{p^m}$ to $\mathcal{F}_q$ resp. $\mathcal{F}_{q^m}$, simply by replacing p by q and q by q^m.

Example A.5.28 The field $\mathcal{F}_{16}$ can be viewed as the residue class ring $\mathcal{F}_4[x]/\left(x^2 + x + \alpha\right)$, where α is an element in $\mathcal{F}_4$, satisfying $\alpha^2 + \alpha + 1 = 0$.

A.6 Problems

A.6.1 Let the sequence of *Fibonacci numbers* $\{F_n\}_{n=0}^{\infty}$ be defined recursively by $F_0 = 0$, $F_1 = 1$ and $F_n = F_{n-1} + F_{n-2}$, $n \geq 2$.

Show that for $n \geq 3$ Euclid's Algorithm (Algorithm A.1.3) applied to F_{n-1} and F_n determines their gcd in exactly $n - 2$ iterations.

A.6.2 Use an induction argument to show that the computation of the gcd of integers a and b, $b \geq a$, by Euclid's Algorithm involves at most $\lfloor \log_f b \rfloor$ iterations, where $f = (1 + \sqrt{5})/2$. (Hint: distinguish the cases $a \leq b/f$ and $b/f < a \leq b$.)

Note that $\lfloor \log_f F_n \rfloor = n - 2$.

A.6.3 Let p be a prime number dividing $a_1 a_2 \cdots a_k$, where $a_i \in \mathcal{Z}$, $1 \leq i \leq k$. Show that p divides at least one of the factors a_i, $1 \leq i \leq k$.

Prove Theorem A.1.5.

A.6.4 Describe the congruence relation, as defined in (A.16), as an equivalence relation (see Definition A.2.1), so give the sets S and U.

A.6.5 Give a formal proof of Lemma A.3.5.

A.6.6 Prove the equivalence of (H2) and (H3) with (H) in Section A.3.

A.6.7 What are the subgroups of $(\mathcal{Z}_{15}, +)$ and of $(\mathcal{Z}_{17}^*, \times)$?

What is the multiplicative order of the various elements in $(\mathcal{Z}_{17}^*, \times)$?

A.6.8 Prove that (A.25) indeed defines an equivalence relation and show that the equivalence classes are given by $\{ha \mid h \in H\}$, where $a \in G$.

A.6.9 Prove that the set $(1 + x^3)$ in $\mathcal{F}_2[x]$, consisting of all polynomial multiples of the polynomial $1 + x^3$, is an ideal in $(\mathcal{F}_2[x], +, \times)$.

Give a pair of zero-divisors in the residue class ring $(\mathcal{F}_2[x]/(1 + x^3), +, \times)$.

A.6.10 What is the product of the binary polynomials $1+x^2+x^4$ and $1+x+x^3$ modulo $1+x^2+x^5$?

A.6.11 Find all binary, irreducible polynomials of degree 2, 3, and 4 and check (A.38).

A.6.12 Use Euclid's Algorithm to find the multiplicative inverse of $1+x+x^3$ (mod $1+x^2+x^5$) over $\mathcal{F}_2$.

A.6.13 What are the multiplicative orders of x and of x^2+x^3 in $(\mathcal{F}_2[x]/(1+x+x^2+x^3+x^4),+,\times)$? What is the product of these two elements and what is its multiplicative order?

A.6.14 Make a log table of $GF(3)[x]/(2+2x+x^2)$ (hint: x is a primitive element).

A.6.15 Let α be a primitive element of $GF(2^6)$. Determine the zeroes of the minimal polynomials of α, α^3 and α^9.

A.6.16 What is the smallest extension field of $GF(3)$ containing a primitive 11-th root of unity α?

What is the degree of the minimal polynomial of α and what are the zeroes of this polynomial?

A.6.17 Which subfields are contained in $GF(625)$. Let α be a primitive element in $GF(625)$. Which powers of α constitute the various subfields of $GF(625)$.

A.6.18 What is the degree of a binary minimal polynomial of a primitive 17-th root of unity? How many such polynomials do exist? Prove that each is its own reciprocal. Determine them explicitly.

A.6.19 The *trace* mapping Tr is defined on $GF(p^m)$, p prime, by

$$Tr(x) = x + x^p + x^{p^2} + \cdots + x^{p^{m-1}}.$$

1. Prove that $Tr(x) \in GF(p)$, for every $x \in GF(p^m)$. So Tr is a mapping from $GF(p^m)$ to $GF(p)$.
2. Prove that Tr is a linear mapping from $GF(p^m)$ to $GF(p)$.
3. Prove that Tr takes on every value in $GF(p)$ equally often (hint: use Theorem A.4.14).

Appendix B

Tables of $GF(2^m)$, $m = 3, 4, 5, 6$

	1	α	α^2
1	1	0	0
α	0	1	0
α^2	0	0	1
α^3	1	1	0
α^4	0	1	1
α^5	1	1	1
α^6	1	0	1

Table B.1: $GF(2^3)$ generated by α, $\alpha^3 + \alpha + 1 = 0$.

	1	α	α^2	α^3
1	1	0	0	0
α	0	1	0	0
α^2	0	0	1	0
α^3	0	0	0	1
α^4	1	1	0	0
α^5	0	1	1	0
α^6	0	0	1	1
α^7	1	1	0	1
α^8	1	0	1	0
α^9	0	1	0	1
α^{10}	1	1	1	0
α^{11}	0	1	1	1
α^{12}	1	1	1	1
α^{13}	1	0	1	1
α^{14}	1	0	0	1

Table B.2: $GF(2^4)$ generated by α, $\alpha^4 + \alpha + 1 = 0$.

	1	α	α^2	α^3	α^4
1	1	0	0	0	0
α	0	1	0	0	0
α^2	0	0	1	0	0
α^3	0	0	0	1	0
α^4	0	0	0	0	1
α^5	1	0	1	0	0
α^6	0	1	0	1	0
α^7	0	0	1	0	1
α^8	1	0	1	1	0
α^9	0	1	0	1	1
α^{10}	1	0	0	0	1
α^{11}	1	1	1	0	0
α^{12}	0	1	1	1	0
α^{13}	0	0	1	1	1
α^{14}	1	0	1	1	1

α^{15}	1	1	1	1	1
α^{16}	1	1	0	1	1
α^{17}	1	1	0	0	1
α^{18}	1	1	0	0	0
α^{19}	0	1	1	0	0
α^{20}	0	0	1	1	0
α^{21}	0	0	0	1	1
α^{22}	1	0	1	0	1
α^{23}	1	1	1	1	0
α^{24}	0	1	1	1	1
α^{25}	1	0	0	1	1
α^{26}	1	1	1	0	1
α^{27}	1	1	0	1	0
α^{28}	0	1	1	0	1
α^{29}	1	0	0	1	0
α^{30}	0	1	0	0	1

Table B.3: $GF(2^5)$ generated by α, $\alpha^5 + \alpha^2 + 1 = 0$.

	1	α	α^2	α^3	α^4	α^5
0	0	0	0	0	0	0
1	1	0	0	0	0	0
α^1	0	1	0	0	0	0
α^2	0	0	1	0	0	0
α^3	0	0	0	1	0	0
α^4	0	0	0	0	1	0
α^5	0	0	0	0	0	1
α^6	1	1	0	0	0	0
α^7	0	1	1	0	0	0
α^8	0	0	1	1	0	0
α^9	0	0	0	1	1	0
α^{10}	0	0	0	0	1	1
α^{11}	1	1	0	0	0	1
α^{12}	1	0	1	0	0	0
α^{13}	0	1	0	1	0	0
α^{14}	0	0	1	0	1	0
α^{15}	0	0	0	1	0	1
α^{16}	1	1	0	0	1	0
α^{17}	0	1	1	0	0	1
α^{18}	1	1	1	1	0	0
α^{19}	0	1	1	1	1	0
α^{20}	0	0	1	1	1	1
α^{21}	1	1	0	1	1	1
α^{22}	1	0	1	0	1	1
α^{23}	1	0	0	1	0	1
α^{24}	1	0	0	0	1	0
α^{25}	0	1	0	0	0	1
α^{26}	1	1	1	0	0	0
α^{27}	0	1	1	1	0	0
α^{28}	0	0	1	1	1	0
α^{29}	0	0	0	1	1	1
α^{30}	1	1	0	0	1	1

	1	α	α^2	α^3	α^4	α^5
α^{31}	1	0	1	0	0	1
α^{32}	1	0	0	1	0	0
α^{33}	0	1	0	0	1	0
α^{34}	0	0	1	0	0	1
α^{35}	1	1	0	1	0	0
α^{36}	0	1	1	0	1	0
α^{37}	0	0	1	1	0	1
α^{38}	1	1	0	1	1	0
α^{39}	0	1	1	0	1	1
α^{40}	1	1	1	1	0	1
α^{41}	1	0	1	1	1	0
α^{42}	0	1	0	1	1	1
α^{43}	1	1	1	0	1	1
α^{44}	1	0	1	1	0	1
α^{45}	1	0	0	1	1	0
α^{46}	0	1	0	0	1	1
α^{47}	1	1	1	0	0	1
α^{48}	1	0	1	1	0	0
α^{49}	0	1	0	1	1	0
α^{50}	0	0	1	0	1	1
α^{51}	1	1	0	1	0	1
α^{52}	1	0	1	0	1	0
α^{53}	0	1	0	1	0	1
α^{54}	1	1	1	0	1	0
α^{55}	0	1	1	1	0	1
α^{56}	1	1	1	1	1	0
α^{57}	0	1	1	1	1	1
α^{58}	1	1	1	1	1	1
α^{59}	1	0	1	1	1	1
α^{60}	1	0	0	1	1	1
α^{61}	1	0	0	0	1	1
α^{62}	1	0	0	0	0	1

Table B.4: $GF(2^6)$ generated by α, $\alpha^6 + \alpha + 1 = 0$.

Appendix C

Solutions to the problems

C.1 Solutions to Chapter 1

1.4.1 When 000000 has been transmitted, the receiver will not detect transmission errors if the error pattern is 001111, 110011 or 111100. This happens with probablity $p^4(1-p)^2 + p^4(1-p)^2 + p^4(1-p)^2$.

One finds the same probability of undetected error when one of the other three words has been transmitted. So the overall probability of undetected error is equal to $3p^4(1-p)^2$.

1.4.2 It follows from

$$p(-1|-1)p(0.01|-1)p(0.01|-1) =$$
$$= \left(\frac{1}{\sqrt{2\pi}}\right)^3 e^{-(0^2+1.01^2+1.01^2)/2}$$
$$\approx 0.0229.$$

and

$$p(-1|+1)p(0.01|+1)p(0.01|+1) =$$
$$= \left(\frac{1}{\sqrt{2\pi}}\right)^3 e^{-(2^2+0.99^2+0.99^2)/2}$$
$$\approx 0.0032.$$

that any reasonable soft decision decoding rule will decode $(-1, 0.01, 0.01)$ into $(-1, -1, -1)$.

With hard decision decoding the received sequence is transformed into $(-1, +1, +1)$. Now $(+1, +1, +1)$ is the most likely transmitted word, because the probability that $(+1, +1, +1)$ is transmitted given that $(-1, +1, +1)$ is received is $p(1-p)^2$ with $p = \frac{1}{\sqrt{2\pi}} \int_{x \geq 0} e^{-(x+1)^2/2} dx$. This exceeds the probability that $(-1, -1, -1)$ is transmitted given that $(-1, +1, +1)$ is received, which is $p^2(1-p)$.

C.2   Solutions to Chapter 2

2.5.1 According to (2.4) $|B_1(\underline{x})| = 1 + 6(2-1) = 7$.

If there are 9 words of length 6 at mutual distance at least 3, then at least $\lceil 9/2 \rceil = 5$ of them will have the same first coordinate and at least $\lceil 9/4 \rceil = 3$ of them will have the same first two coordinates. Deleting the first two coordinates in these 3 words (a technique called shortening in Definition 3.1.2) yields three words of length 4 and at mutual distance at least 3. However to two words of length 4 and at distance three no third word can be added.

2.5.2 If such a code contains two words at distance 5 or 6 (w.l.o.g. these words are 000000 and 11111?) each other codeword will have distance 3 or less to at least one of these. So the minimum distance of such a code is at most 4 and in this case each pair of codewords must have distance exactly 4. Starting with 000000 and 111100 one finds (up to isomorphism) a unique (6,4,4)-code. This code, say C, is linear, consist of the words 000000, 111100, 110011 and 001111 and has parity check matrix

$$H = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

C has one coset with unique coset leader of weight 0, six cosets with unique coset leader of weight 1 (namely with syndromes $abc*^T$ with abc of weight 1), six cosets with two coset leaders of weight 2 (namely with syndromes $abc*^T$ with abc of weight 2), one coset with three coset leaders of weight 2 (namely with syndrome 0001^T), and two cosets with four coset leaders of weight 3 (namely with syndromes $111*^T$).

So the probability of correctly decoding a received word with a maximum likelihood decoding algorithm is given by

$$(1-p)^6 + 6p(1-p)^5 + 6p^2(1-p)^4 + p^2(1-p)^4 + 2p^3(1-p)^3.$$

2.5.3 It follows from Theorem 2.1.8 that a ternary $(4, 9, 3)$ code is systematic on each two coordinates. This makes it easy to find the following code:

0000	1012	2021
0111	1120	2102
0222	1201	2210

A code with these parameters is perfect, which also makes it easy to find it. The above code is also linear with parity check matrix

$$H = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 2 \end{pmatrix}.$$

In Chapter 3 this code will be called the ternary Hamming code of length 4 (see Definition 3.2.1).

2.5.4 If two codewords have Hamming distance 1, there is only one coordinate where they differ. An erasure on that coordinate can not be corrected. We conclude that the minimum distance has to be at least 2. Conversely, if the distance between any two codewords is at least 2, one can correct any single erasure.

The maximum size code of length 5 and minimum distance 2 is the even weight code. This can be seen directly but also follows from Theorem 2.1.8.

If a received word contains at most one erasure, there is a unique codeword coinciding with it on the other positions. The possible coordinate with the erasure can be determined by the rule that the codeword has even weight. Hence, the probability of correctly decoding is $p^5 + 5qp^4 = 0.91854$.

Received words with more erasures can also be decoded with some probability of success. Indeed, if a received word contains t erasures, one can guess $t - 1$ coordinates and fill in the last one in such a way that the weight becomes even. In this way the probability of decoding correctly is

$$p^5 + \sum_{t=1}^{5} \frac{1}{2^{t-1}} \binom{5}{t} q^t p^{5-t} = 0.95707.$$

2.5.5 Consider two binary vectors, called $\underline{x}$ and $\underline{y}$, of even weight, say $2v$ and $2w$. Let their inner product be i, so without loss of generality:

$$
\begin{array}{ccccc}
 & & \overbrace{i} & \overbrace{2v-i} & \overbrace{2w-i} \\
\underline{x} & = & 1\cdots 1 & 11\cdots 1 & 00\cdots 0 & 0\cdots 0 \\
\underline{y} & = & 1\cdots 1 & 00\cdots 0 & 11\cdots 1 & 0\cdots 0 \\
\hline
\underline{x}+\underline{y} & = & 0\cdots 0 & 11\cdots 1 & 11\cdots 1 & 0\cdots 0 \\
\end{array}
$$

So $\underline{x} + \underline{y}$ has weight $2v + 2w - 2i$, which is also even.

Since each vector in C is a sum of rows of G, each of even weight, it follows (with an induction argument) that each word in C has even weight.

2.5.6 a) Vector (111101000) has syndrome $(1111)^T$, which is the 5-th column of H. So its coset leader is (000010000).

b) Vector (110101011) has syndrome $(0101)^T$, which is not equal to one of the columns of H. It can be written as sum of two columns of H in the following ways: 1+9, 2+6, 4+8. So the coset leaders are (100000001), (010001000) and (000100010).

c) Vector (010010010) has syndrome $(0000)^T$, so it is a codeword. Its coset leader is (000000000).

2.5.7 Take two rows from G, say $\underline{x}$ and $\underline{y}$. Use the same approach as in the solution of Problem 2.5.5. The weights of $\underline{x}$ and $\underline{y}$ can now be written as $4v$ and $4w$ and their inner product $(\underline{x}, \underline{y})$ as $2i$, because C is self dual and thus each inner product is even. It follows that $\underline{x} + \underline{y}$ has weight $4v + 4w - 2 \cdot 2i$, which is again a multiple of four.

For an arbitrary vector in the code, which is the sum of rows of G, one again uses an induction argument.

2.5.8 If all the codewords have a weight divisible by 4, it follows that $\underline{1}$ is in the dual code $C^{\perp}$. Since $C = C^{\perp}$ we can conclude that $\underline{1}$ is an element in C.

Let $A_i, 0 \leq i \leq 24$, be the weight enumerator of C. Since $\underline{1} \in C$ it follows that $A_i = A_{24-i}$ From $d = 8$ and the divisibility of all weights by 4, the non-zero coefficients in the weight enumerator are $A_0 = A_{24} = 1$, $A_8 = A_{16}$ and A_{12}. Since $|C| = 2^{12}$ also the relation $2 + 2A_8 + A_{12} = 4096$ holds. We need one more relation.

The coefficient of z^2 in the MacWilliams relation (2.22) gives:

$$
\begin{aligned}
0 &= \binom{24}{2} + A_8 \left(\binom{8}{2} - 8 \right) + A_{12}(-12) + \\
&\quad + A_8 \left(\binom{8}{2} - 8 \right) + \binom{24}{2},
\end{aligned}
$$

i.e.

$$-10A_8 + 3A_{12} = 138.$$

Combining the two relations yields the weight enumerator $A_0 = A_{24} = 1$, $A_8 = A_{16} = 759$ and $A_{12} = 2576$.

2.5.9 A vector $\underline{c}$ is in the code C if and only if $H\underline{c}^T = \underline{0}^T$, i.e. if and only if the columns in H corresponding to the non-zero coordinates of $\underline{c}$ are linearly dependent.

A vector $\underline{x}$ is at distance δ from the code C if it is in a coset with coset leader of weight δ, i.e. if it (or the coset leader) can be written as linear combination of δ columns of H but not of less.

2.5.10 Consider a k-tuple of coordinates. Puncturing the other $d - 1$ coordinates in the code yields, according to the proof of Theorem 2.1.8, a $(k, q^k, 1)$ code. So C is systematic on these k coordinates.

Consider any d-tuple of coordinates, put $\alpha \neq 0$ on the leftmost of these coordinates and put zeros on the complementary $n - d = k - 1$ positions. Since C is systematic on these k coordinates, it follows that exactly one codeword exists that coincides with these k entries. From the minimum distance d one can conclude that this

unique codeword has weight d and has all its nonzero entries on the chosen d-tuple of coordinates. It follows that

$$A_d = \binom{n}{d}(q - 1).$$

2.5.11 Apply Theorems 2.1.9, 2.1.8 and 2.1.3.

$n = 16$	$d = 3$	$d = 5$	$d = 7$	$d = 9$
Gilbert	479	27	5	2
Gilbert for linear codes	2^9	2^5	2^3	2^1
Singleton	2^{14}	2^{12}	2^{10}	2^8
Hamming	3855	478	94	26

2.5.12 Without loss of generality one gets

$$H = \begin{pmatrix} 0000100101101111 \\ 0001001010110111 \\ 0010010011011011 \\ 0100011100011101 \\ 1000011111100001 \end{pmatrix}.$$

The length n is equal to the number of odd weight vectors of length m, so $n = 2^{m-1}$.

The all-zero vector is a codeword, so has distance 0 to the code. The vector 1000000000000000 has distance 1 to the code (because its syndrome is 00001^T, which is the first column) and the vector 1100000000000000 has distance 2 to the code (because its syndrome is 00011^T, which does not occur as column, but is the sum of the first two columns).

The matrix H contains the m (independent) vectors of weight 1, so C has dimension $n - m = 2^{m-1} - m$. All columns in H are different, so $d \geq 3$. Moreover, the sum of three binary, odd-weight vectors again has odd weight, so no three columns add up to $\underline{0}$. In other words $d \geq 4$. That $d = 4$ follows from the existence of codewords of weight 4. For instance, put ones where H has columns $0 \cdots 0001^T$, $0 \cdots 0010^T$, $0 \cdots 0100^T$ and $0 \cdots 0111^T$.

Consider any non-zero vector $\underline{s}$ of length m. If the weight of $\underline{s}$ is odd, $\underline{s}$ is equal to a column of H. If its weight is even, $\underline{s}$ is equal to the sum of two columns $\underline{a}$ and $\underline{b}$ of H. Indeed, take $\underline{a}$ arbitrary of odd weight, then also $\underline{b} = \underline{s} - \underline{a}$ has odd weight. It follows that each length n vector lies at distance at most two from C, so the covering radius ρ is 2.

APPENDIX C. SOLUTIONS TO THE PROBLEMS

Vector $00\cdots0^T$ is the syndrome of a coset (the code) with unique coset leader of weight 0. Each of the $n = 2^{m-1}$ vectors $\underline{s}$ of odd weight is the syndrome of a coset with unique coset leader of weight 1: its only 1-coordinate is where column $\underline{s}$ occurs in H.

As already observed, each even weight syndrome $\underline{s}$ and every choice of odd weight vector $\underline{a}$ gives a unique different odd weight vector $\underline{b}$ with $\underline{s} = \underline{a}+\underline{b}$ and vice versa. Hence, there are $2^m - 1 - n = 2^{m-1} - 1$ cosets with $n/2 = 2^{m-2}$ coset leaders of weight 2.

2.5.13 The vector 00001010001010000100010 has syndrome 00000^T, so it is a codeword. The vector 100010001000100010000001 has syndrome 00111^T, which is not a column in H but is the sum of (among others) the 1-st and 8-th columns (01000^T and 01111^T). So one can change the 1-st and 8-th coordinates to obtain a codeword. The vector 010100100110000010100000 has syndrome 01101^T, which is the 6-th column. So correct the 6-th coordinate.

The coset with syndrome $\underline{0}$ is equal to C and has the origin as unique coset leader. The 24 cosets with syndrome $\underline{s}$ with $(s_1, s_2) \neq (0,0)$ have a unique coset leader of weight 1. The 1-coordinate of this coset leader is exactly where $\underline{s}$ appears as column in H.

The 7 cosets with non-zero syndrome $\underline{s}$ with $(s_1, s_2) = (0,0)$ have twelve coset leaders of weight 2. Indeed, take an arbitrary column $\underline{a}$ from H (so $(a_1, a_2) \neq (0,0)$) and compute $\underline{b} = \underline{s} - \underline{a}$. Then $(b_1, b_2) = (a_1, a_2) \neq (0,0)$, while $\underline{b} \neq \underline{a}$, and thus $\underline{b}$ is a column of H different from $\underline{a}$. There are 24 choices for $\underline{a}$, so twelve pairs $\underline{a}, \underline{b}$.

The probability of correctly decoding a transmitted codeword is given by

$$(1-p)^{24} + 24p(1-p)^{23} + 7p^2(1-p)^{22}.$$

Since each word is at distance at most 2 from the code one has $\rho = 2$.

Let $\underline{x}$ have distance 1 to C, and let its syndrome be $\underline{s}$. Since $(s_1, s_2) \neq (0,0)$ one can write $\underline{s} = \underline{a} + \underline{b}$ with $\underline{a}$ and $\underline{b}$ columns of H, as long as $(a_1, a_2) \neq (0,0)$ and $(a_1, a_2) \neq (s_1, s_2)$. So, for $\underline{a}$ there are 16 possibilities, leading to 8 pairs $\underline{a}, \underline{b}$. In other words, $\underline{x}$ has distance 2 to exactly eight codewords.

If $\underline{x}$ has distance 2 to C, we know already from the above that its has distance 2 to exactly 12 codewords.

2.5.14 The shortened code has dimension one less than C, so C^{sh} has dimension $n-m-1$. The minimum distance does not decrease by shortening, so $d \geq 3$. On the other hand, d can not be 4, because then puncturing C^{sh} would contradict the Hamming bound (Theorem 2.1.3). We conclude that $d = 3$. Finally, consider a word $\underline{x}$ of length $n - 1 = 2^m - 2$. If $(0, \underline{x})$ or $(1, \underline{x})$ is at distance at most 1 from a codeword $(0, \underline{y})$ in C, then $\underline{x}$ is at distance at most 1 from C^{sh}. If $(1, \underline{x})$ is at distance ≥ 2

from the codewords $(0, \underline{y})$ in C, then $(1, \underline{x})$ is at distance ≤ 1 from a codeword $(1, \underline{y})$ in C (because C is perfect). So $(0, \underline{x})$ is at distance 2 from from another codeword $(0, \underline{z})$ in C (again because C is perfect) and thus $\underline{x}$ is at distance 2 from $\underline{z}$ in C^{sh}. It follows that $\rho = 2$.

Let $d(\underline{x}, C^{sh}) = 1$. Without loss of generality $\underline{x} = 10 \cdots 0$ and $\underline{c} = \underline{0}$ is in C^{sh}. The codewords at distance 2 from $\underline{x}$ must have weight 3, their first coordinate has to be equal to 1 and the other two 1-coordinates of these words must all different. Their number $B(\underline{x}, 2)$ is thus at most $\lfloor (n-2)/2 \rfloor = (n-3)/2$.

The same reasoning when $d(\underline{x}, C^{sh}) = 2$ gives $B(\underline{x}, 2) \leq (n-1)/2$.

It follows that

$$\sum_{\underline{x}, d(\underline{x}, C^{sh}) \geq 1} (B(\underline{x}, 1) + B(\underline{x}, 2)) \leq$$
$$\leq |C^{sh}|(n-1)\{1 + \tfrac{n-3}{2}\} +$$
$$+ \left(2^{n-1} - |C^{sh}|(1 + (n-1))\right) \{0 + \tfrac{n-1}{2}\} =$$
$$= (2^{n-1} - |C^{sh}|)\tfrac{n-1}{2}.$$

On the other hand

$$\sum_{\underline{x}, d(\underline{x}, C^{sh}) \geq 1} (B(\underline{x}, 1) + B(\underline{x}, 2)) =$$
$$= \sum_{\underline{x}, d(\underline{x}, C^{sh}) \geq 1} \sum_{\underline{c} \in C^{sh}, \ 1 \leq d(\underline{c}, \underline{x}) \leq 2} 1 =$$
$$= \sum_{\underline{c} \in C^{sh}} \sum_{\underline{x}, \ 1 \leq d(\underline{x}, \underline{c}) \leq 2} 1 =$$
$$= |C^{sh}|\{\binom{n-1}{1} + \binom{n-1}{2}\}.$$

It follows from the above two expressions that $|C^{sh}|\{\binom{n}{1} + \binom{n}{2}\} \leq (2^{n-1} - |C^{sh}|)\tfrac{n-1}{2}$. Substitution of $|C^{sh}| = 2^{n-m-1}$ and $n = 2^m - 1$ yields a left and right hand side that are equal to each other. It follows that all inequalities in the derivation can be replaced by equalities. In particular:

$$d(\underline{x}, C^{sh}) = 1 \quad \Rightarrow \quad B(\underline{x}, 2) = (n-3)/2,$$

$$d(\underline{x}, C^{sh}) \geq 1 \quad \Rightarrow \quad B(\underline{x}, 2) = (n-1)/2.$$

2.5.15 All non-trivial linear combinations of the rows of G have weight 8 on the first 16 coordinates. Their weight on the last 4 coordinates is at least 2 if and only if at least one of the top three rows is used. So, the bottom vector spans the linear subcode of weight 8. The weight ≤ 10 vectors span the whole code. So C is a LUEP code with separation vector $(10, 10, 10, 8)$.

An optimal generator matrix of C is G.

C.3   Solutions to Chapter 3

3.4.1 Take the $[4, 3, 2]$ even weight code C_1 and the $[4, 1, 4]$ repetition code C_2 and apply Theorem 3.1.6.

3.4.2 Let $\underline{c}$ be a codeword of weight w, $w \geq d$. Put it in the top row of a generator matrix of the code C and permute all the ones of $\underline{c}$ to the front. One obtains

$$G = \left(\begin{array}{cccc|cccc} 1 & 1 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ \hline & & G^1 & & & & G^2 & \end{array} \right) \qquad \underline{c} \text{ of weight } w$$

Then exactly the same proof as of Theorem 3.1.4 yields that the code generated by the last $n - d$ columns of G, i.e. by G^2, has parameters $[n - w, k - 1, \geq d - \lfloor w/2 \rfloor]$ for $d \leq w \leq 2d - 1$. For $w \geq 2d$ nothing can be said about the dimension or minimum distance of this code.

3.4.3 Again the generator matrix can be assumed to have the form as in the proof of Theorem 3.1.4, where one may have had to multiply some of the columns by a non-zero constant to get only ones as non-zero elements in the top row. So, again

$$G = \left(\begin{array}{cccc|cccc} 1 & 1 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ \hline & & G^1 & & & & G^2 & \end{array} \right) \qquad \underline{c} \text{ of weight } d$$

Let $(\underline{a}^1, \underline{a}^2)$ be a non-trivial linear combination of the last $k - 1$ rows of G and let u_α denote the number of coordinates in $\underline{a}^1$ equal to α. So $\sum_\alpha u_\alpha = d$. Since $(\underline{a}^1, \underline{a}^2)$ must have distance at least d to $\alpha \underline{c}$, one has

$$(d - u_\alpha) + w(\underline{a}^2) \geq d.$$

The sum of these inequalities over all α in $GF(q)$ yields $q w(\underline{a}^2) \geq \sum_\alpha u_\alpha = d$, and thus $w(\underline{a}^2) \geq \lceil d/q \rceil$. This proves that the residual code of C with respect to a weight d code word has parameters $[n - d, k - 1, \geq \lceil d/q \rceil]$.

3.4.4 Assume that a $[101, 7, 50]$ code exists. Taking successive residual codes with respect to minimum weight codewords implies the existence of the following sequence of codes: $[51, 6, 25]$, $[26, 5, 13]$, $[13, 4, 7]$, $[6, 3, 4]$, $[2, 2, 2]$ and $[0, 1, 1]$. However the last code obviously does not exist, so neither does a $[101, 7, 50]$ code.

One can also apply Corollary 3.1.5.

3.4.5 Clearly, $N = n + d - \rho$. Also, since $d - \rho \geq 1$ it follows that the dimension increases by 1, i.e. $K = k + 1$. Finally consider any non-trivial linear combination of the rows of the given generator matrix of G^*. If it only involves the top k rows, its weight is at least d by the assumption on C. If it does involve the bottom row, then the weight of this linear combination is equal to $d - \rho$ plus the distance of $\underline{x}$ to a linear combination of the rows of G. This distance is at least ρ, because $\underline{x}$ has distance ρ to C. It follows that any non-trivial linear combination of the given generator matrix has weight at least d and thus $D = d$.

We conclude that G^* is a $[n + d - \rho, k + 1, d]$ code.

3.4.6 By permuting the columns of the generator matrix G^* of C^* one may assume that the first columns are equal to $\underline{s}^T$. Since $\underline{s} \neq \underline{0}$, there must be a row in G^* that starts with s identical non-zero elements. By permuting the rows of G^* one can assume that this is the last row and by a suitable scalar multiplication this non-zero element can be assumed to be 1. So we have the following matrix:

$$
G^* = \left(
\begin{array}{ccc|cccc}
s_1 & \cdots & s_1 & & & & \\
\vdots & & \vdots & & \multicolumn{2}{c}{G} & \\
s_k & \cdots & s_k & & & & \\
\hline
1 & \cdots & 1 & x_1 & x_2 & \cdots & x_n
\end{array}
\right).
$$

By subtracting s_i times the last row from row i, $1 \leq i \leq k$, one can even assume that $s_1 = \ldots = s_k = 0$. It follows that G generates an $[n, k, d]$ code C and that $\underline{x}$ has distance at least $d - s$ to all codewords in C.

3.4.7 Shortening a linear code with respect to the symbol 0 on coordinate i amounts to:

1) add as row to the parity check matrix the vector that consists of zeros everywhere, except for a one at the i-th coordinate,

2) puncture the i-th coordinate in the subcode defined by this new parity check matrix (i.e. this subcode consists of all codewords with a zero on the i-th coordinate).

Clearly, the new code has parameters $[n - 1, \geq k - 1, \geq d]$.

Since each codeword in C is orthogonal to $\underline{x}$, it follows that the subcode of C consisting of all codewords that are zero on $w - 1$ of w one-coordinates of $\underline{x}$ are also zero on the w-the coordinate. So, one only has to add $w - 1$ equations (see figure below) to the parity check matrix (the dimension of the subcode becomes $k - (w - 1)$) and one obtains w coordinates where all the codewords in this subcode are zero (so the length becomes $n - w$).

parity check equation $\underline{x}$	$0\cdots 0$	$111\cdots 111$	$0\cdots 0$
new parity check equation	$0\cdots 0$	$100\cdots 000$	$0\cdots 0$
new p.c. equation 2	$0\cdots 0$	$010\cdots 000$	$0\cdots 0$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
new p.c. equation $w-1$	$0\cdots 0$	$000\cdots 010$	$0\cdots 0$

3.4.8 By (3.8) $A(z)$ is given by

$$A(z) = \frac{1}{32}\left\{(1+z)^{31} + 31(1+z)^{15}(1-z)^{16}\right\}.$$

Since $A_i = A_{31-i}, 0 \le i \le 31$, $A_0 = 1$ and $A_1 = A_2 = 0$ it suffices to give

$$
\begin{aligned}
A_3 &= 155 \\
A_4 &= 1,085 \\
A_5 &= 5,208 \\
A_6 &= 22,568 \\
A_7 &= 82,615 \\
A_8 &= 247,845 \\
A_9 &= 628,680 \\
A_{10} &= 1,383,096 \\
A_{11} &= 2,648,919 \\
A_{12} &= 4,414,865 \\
A_{13} &= 6,440,560 \\
A_{14} &= 8,280,720 \\
A_{15} &= 9,398,115.
\end{aligned}
$$

It follows that

$$
\begin{aligned}
\frac{A_5}{\binom{31}{5}/2^5} &= 0.981, \\
\frac{A_{10}}{\binom{31}{10}/2^5} &= 0.998, \\
\frac{A_{15}}{\binom{31}{15}/2^5} &= 1.001.
\end{aligned}
$$

Remark: it can be shown that for $m \to \infty$ the ratios $\dfrac{A_i}{\binom{2^m-1}{i}/2^m}$ tend to 1.

3.4.9 That both codes are perfect follows from Theorem 2.1.7 and the observations

$$2^{12}\left\{1+\binom{23}{1}+\binom{23}{2}+\binom{23}{3}\right\}=2^{23},$$

$$3^6\left\{1+\binom{11}{1}2+\binom{11}{2}2^2\right\}=3^{11}.$$

For the recurrence relations of the weight enumerators of these perfect codes we generalize the result in equation (3.6).

For the binary perfect $[23, 12, 7]$ code one gets, by observing that each word of weight q in the vector space must be at distance at most 3 to a unique codeword.

$\binom{23}{w} = A_w + (w+1)A_{w+1} + (23-w+1)A_{w-1} + \binom{w+2}{2}A_{w+2} + w(23-w)A_w + \binom{23-w+2}{2}A_{w-2} + \binom{w+3}{3}A_{w+3} + \binom{w+1}{2}(23-w+1)A_{w+1} + (w+1)\binom{23-w+1}{2}A_{w-1} + \binom{23-w+3}{3}A_{w-3},$

where the A_i's are assumed to be zero for $i < 0$ and $i > 23$.

Similarly, for the ternary perfect $[11, 6, 5]$ code one gets,

$\binom{11}{w}2^w = A_w + (w+1)A_{w+1} + wA_w + 2(11-w+1)A_{w-1} + \binom{w+2}{2}A_{w+2} + (w+1)wA_{w+1} + 2w(11-w)A_w + \binom{w}{2}A_w + (11-w+1)(w-1)2A_{w-1} + \binom{11-w+2}{2}2^2A_{w-2},$

where one has to realize that there are two ways of changing a zero into a nonzero coordinate and one way to change a nonzero coordinate into another nonzero coordinate.

3.4.10 Puncturing a binary $[24, 12, 8]$ code yields a $[23, 12, \geq 7]$ code containing $\underline{0}$. This code is perfect by Theorem 2.1.7 and thus its weight enumerator satisfies the recurrence relation of order seven given in the preceding problem. Since $A_0 = 1$ and $A_1 = \ldots = A_6 = 0$, this recurrence relation has a unique solution.

Consider the 759 codewords of weight 8 in the $[24, 12, 8]$ code and put them in a 759×24 array. Then each column will contain A_7 ones and $759 - A_7$ zeros, since puncturing that column must yield the codewords of weight 7 and 8.

Counting the number of ones in the array columnwise gives $24A_7$ but rowwise one gets 759×8. It follows that $A_7 = 253$ and thus that $A_8 = 506$.

In exactly the same way one gets $24A_{11} = 2576 \times 12$, and thus $A_{11} = A_{12} = 1288$. Similarly, $24A_{15} = 759 \times 16$, and thus $A_{15} = 506$ and $A_{16} = 253$. From $24A_{23} = 1 \times 24$ it follows that $A_{23} = 1$ (of course).

3.4.11 That a $(90, 2^{78}, 5)$ binary code has to be perfect follows from Theorem 2.1.7 and the observation

$$2^{78} \left\{ 1 + \binom{90}{1} + \binom{90}{2} \right\} = 2^{90}.$$

The recurrence relation for the weight enumerator $A(z)$ of C is given by

$$
\begin{aligned}
\binom{90}{w} = {} & A_w + (w+1)A_{w+1} + (90 - w + 1)A_{w-1} + \\
& + \binom{w+2}{2} A_{w+2} + w(90 - w)A_w + \\
& + \binom{90 - w + 2}{2} A_{w-2}.
\end{aligned}
$$

Let $\underline{c}$ be one of the codewords in C. Translate the C over $-\underline{c}$. One obtains a new code with the same parameters (thus also perfect) containing $\underline{0}$. So, the weight enumerator of this code obviously satisfies $A_0 = 1$, $A_1 = \ldots = A_4 = 0$. For the other coefficients we use the recurrence relation with $w = 3, 4$ and 5.

Substitution of $w = 3$ in the recurrence relation yields: $\binom{90}{3} = \binom{5}{2}A_5$, and thus $A_5 = 11748$.

Substitution of $w = 4$ in the recurrence relation yields: $\binom{90}{4} = \binom{6}{2}A_6 + 5A_5$, and thus $A_6 = 166430$.

Substitution of $w = 5$ in the recurrence relation yields: $\binom{90}{5} = \binom{7}{2}A_7 + 6A_6 + 5 \times 85A_5$, and thus $A_7 = 1807513\frac{5}{7}$.

Since the coefficient A_w in the weight enumerator of a code counts the number of codewords of weight w this number has to be an integer. From the fact that A_7 is not an integer we can conclude that a binary (perfect) $(90, 2^{78}, 5)$ code does not exist.

3.4.12 Consider the Boolean function $\sum_{1 \leq i < j \leq m} a_{ij}x_i x_j + \sum_{1 \leq i \leq m} b_i x_i + c$ with at least one of the a'_{ij}s not equal to 0. We first want to demonstrate that a suitable invertible affine transformation maps this function to one of exactly the same form but with $a_{12} \neq 0$.

If $a_{12} = 1$ there is nothing to do. If $a_{1j} = 1$ with $j > 2$ apply $x'_2 = x_j$, $x'_j = x_2$ and $x'_k = x_k$ for $k \neq 2, j$. If $a_{i2} = 1$ with $i > 2$, interchange x_1 and x_i in a similar way. Finally, if $a_{1k} = 0$ for all k and $a_{k2} = 0$ for all k while $a_{ij} = 1$, apply the mapping that interchanges both 1 and i and 1 and j.

Now $\sum_{1 \leq i < j \leq m} a_{ij}x_i x_j + \sum_{1 \leq i \leq m} b_i x_i + c$ with $a_{12} = 1$ can be rewritten as

$(x_1 + \sum_{i \geq 3} a_{2i} x_i + b_2)(x_2 + \sum_{j \geq 3} a_{1j} x_j + b_1) + (\sum_{i \geq 3} a_{2i} x_i + b_2)(\sum_{j \geq 3} a_{1j} x_j + b_1) + \sum_{3 \leq i < j \leq m} a_{ij} x_i x_j + \sum_{3 \leq i \leq m} b_i x_i + c.$

This means that the invertible affine transformation

$$
\begin{aligned}
x_1' &= x_1 + \sum_{i \geq 3} a_{2i} x_i + b_2, \\
x_2' &= x_2 + \sum_{j \geq 3} a_{1j} x_i + b_1, \\
x_k' &= x_k, \quad k \geq 3,
\end{aligned}
$$

transforms the above function into

$$
x_1 x_2 + \sum_{3 \leq i < j \leq m} a_{ij} x_i x_j + \sum_{3 \leq i \leq m} b_i x_i + c,
$$

where we have left out all primes in the notation.

Note that the product of several invertible affine transformations, one applied one after another, still is an invertible affine transformation.

If one of the terms a_{ij} with $3 \leq i < j \leq m$ is not equal to 1, one can find in exactly the same way an invertible affine transformation that yields the form $x_1 x_2 + x_3 x_4 + \sum_{4 \leq i < j \leq m} a_{ij} x_i x_j + \sum_{4 \leq i \leq m} b_i x_i + c$ Continuing in this way one arrives at

$$
x_1 x_2 + \cdots + x_{2l-1} x_{2l} + \sum_{2l+1 \leq i \leq m} b_i x_i + c.
$$

If one of the $b_i \neq 0$, $i \geq 2l + 1$, one can apply an invertible affine transformation that maps the above form into $x_1 x_2 + \cdots + x_{2l-1} x_{2l} + x_{2l+1}$. So the following standard forms can be found:

1) $\underline{0}$
2) $\underline{1}$
3) x_1
4) $x_1 x_2 + \ldots + x_{2l-1} x_{2l}, \ 2l \leq m,$
5) $x_1 x_2 + \ldots + x_{2l-1} x_{2l} + 1, , \ 2l \leq m,$
6) $x_1 x_2 + \ldots + x_{2l-1} x_{2l} + x_{2l+1}, , \ 2l + 1 \leq m.$

It follows from the above that each word in $\mathcal{RM}(2,5)$ can be mapped by an invertible affine transformation into one of the following words: $\underline{0}$, $\underline{1}$, x_1, $x_1 x_2$, $x_1 x_2 + x_3 x_4$, $x_1 x_2 + 1$, $x_1 x_2 + x_3 x_4 + 1$, $x_1 x_2 + x_3$ or $x_1 x_2 + x_3 x_4 + x_5$.

Since an invertible affine transformation permutes the coordinates, such a transformation will not affect the weight of words. So, as one can easliy check, these standard forms give rise to the following weights:

weight	standardforms
0	$\underline{0}$
8	$x_1 x_2$
12	$x_1 x_2 + x_3 x_4$
16	x_1
	$x_1 x_2 + x_3$
	$x_1 x_2 + x_3 x_4 + x_5$
20	$x_1 x_2 + x_3 x_4 + 1$
24	$x_1 x_2 + 1$
32	$\underline{1}$

For instance, $x_1 x_2$ is equal to 1 if and only if $x_1 = x_2 = 1$, independent of the values of x_3, x_4 and x_5, i.e. independent of the eight possible values of (x_3, x_4, x_5).

That $\mathcal{RM}(2,5)$ is a selfdual code follows directly from Theorem 3.3.4. If follows that the weight enumerator of $\mathcal{RM}(2,5)$ is equal to that of its dual code. There are only five unknown coefficients: $A_8, A_{12}, A_{16}, A_{20}$ and A_{24}. Comparing the coefficients of z^i, $0 \le 7$, in the MacWilliams relations (Theorem 2.3.4) will yield all these coefficients.

3.4.13 The coefficient a_{13} is given by the innerproduct of $\underline{c}$ with any of the eight products $(x_2 + u_2)(x_4 + u_4)(x_5 + u_5)$. The choice $u_2 = u_4 = u_5 = 1$ results in the coordinates with $x_2 = x_4 = x_5 = 0$, i.e. coordinates 0, 4, 16 and 20. In other words $a_{13} = c_0 + c_4 + c_{16} + c_{20}$. Together with the other choices of u_2, u_4 and u_5 one gets the following equalities.

$$
\begin{aligned}
a_{13} &= c_0 + c_4 + c_{16} + c_{20} \\
a_{13} &= c_1 + c_5 + c_{17} + c_{21} \\
a_{13} &= c_2 + c_6 + c_{18} + c_{22} \\
a_{13} &= c_3 + c_7 + c_{19} + c_{23} \\
a_{13} &= c_8 + c_{12} + c_{24} + c_{28} \\
a_{13} &= c_9 + c_{13} + c_{25} + c_{29} \\
a_{13} &= c_{10} + c_{14} + c_{26} + c_{30} \\
a_{13} &= c_{11} + c_{15} + c_{27} + c_{31}
\end{aligned}
$$

Similarly for a_3 one finds that it should be equal to each of: $c_0 + c_4$, $c_1 + c_5$, $c_2 + c_6$, $c_3 + c_7$, $c_8 + c_{12}$, $c_9 + c_{13}$, $c_{10} + c_{14}$, $c_{11} + c_{15}$, $c_{16} + c_{20}$, $c_{17} + c_{21}$, $c_{18} + c_{22}$, $c_{19} + c_{23}$ $c_{24} + c_{28}$, $c_{25} + c_{29}$, $c_{26} + c_{30}$, $c_{27} + c_{31}$ (the primes in c_i' have been left out).

Finally for $a_\emptyset$ one has the 32 equations $a_\emptyset = c_i$, $0 \le i \le 31$ (the primes in c_i'' have been left out).

C.4   Solutions to Chapter 4

4.7.1 If f and g are q-ary polynomials of degree less than k and u and v are elements in $GF(q)$, then also $uf + vg$ is a q-ary polynomial of degree degree less than k. So C is a linear code. That the dimension of C is k follows from the fact that the only polynomial of degree less than n that is identical to zero on all the non-zero elements of $GF(q)$ is the polynomial 0. Since there are q^k polynomials of degree less than k, one can conclude that C is a k-dimensional code.

Any non-zero polynomial of degree less than k has at most $k - 1$ zeros, so each non-zero vector in C has weight at least $n - (k-1)$. So, C is an $[n, k, d \geq n - k + 1]$ code. From (2.17) it follows that $d = n - k + 1$ and that C is an MDS code.

The polynomials x^i, $0 \leq i < k$, give rise to the following base of the code C over $GF(q)$:

$$(1, \alpha^i, \ldots, \alpha^{i(n-1)}), \qquad 0 \leq i < k.$$

Cyclically shifting the i-th vector yields $(\alpha^i, \alpha^{2i}, \ldots, \alpha^{i(n-1)}, \alpha^{in})$, which can be written as $\alpha^i(1, \alpha^i, \ldots, \alpha^{i(n-1)})$. This vector also lies in C and is the image of the polynomial $\alpha^i x^i$. It follows that the cyclic shifts of the basis vectors are again in the code and also generate it, so C is cyclic. To make this more explicit: the cyclic shift of the codeword defined by the polynomial $f(x)$ is the codeword defined by the polynomial $f(\alpha x)$, which is of the same degree as $f(x)$ is.

In fact C is the Reed Solomon code.

4.7.2 Let α be a primitive 11-th root of unity in an extension field of $GF(3)$. Then $x^{11} - 1 = (x - 1)(x - \alpha) \ldots (x - \alpha^{10})$. Let $m_1(x)$ be the minimal polynomial of α. Then the conjugates of α are α^3, α^9, $\alpha^{27} = \alpha^5$ and $\alpha^{15} = \alpha^4$. Similarly the conjugates of $\alpha^{-1} = \alpha^{10}$ are α^8, α^2, α^6 and α^7.

So the factorization of $x^{11} - 1$ over $GF(3)$ is given by

$$x^{11} - 1 = (x - 1)m_1(x)m_{-1}(x),$$

with

$$
\begin{aligned}
m_1(x) &= (x - \alpha)(x - \alpha^3)(x - \alpha^9)(x - \alpha^5)(x - \alpha^4) \\
m_{-1}(x) &= (x - \alpha^{10})(x - \alpha^8)(x - \alpha^2)(x - \alpha^6)(x - \alpha^7),
\end{aligned}
$$

both of degree 5.

The smallest extension field of $GF(3)$ that contains a primitive 11-th root of unity is $GF(3^m)$, with m minimal such that 11 divides $3^m - 1$, so $m = 5$.

Let ω be a primitive element in $GF(3^5)$. Then $\alpha = \omega^{(3^5-1)/11}$ is a primitive 11-th root of unity in $GF(3^5)$. So the zeros of $m_1(x)$ are $\omega^{22}, \omega^{66}, \omega^{198}, \omega^{110}, \omega^{88}$, those of $m_{-1}(x)$ are $\omega^{220}, \omega^{176}, \omega^{44}, \omega^{132}, \omega^{154}$.

4.7.3 Let α be a primitive, 93-th root of unity in an extension field of $GF(2)$. The generator polynomial $g(x)$ must have α^i, $1 \leq i \leq 12$, as zeros. Grouping the conjugates of these zeros in their cyclotomic cosets, one gets:

cyclotomic cosets	exponents i of the zeros α^i
$\mathcal{C}_1$	$\underline{1}$, $\underline{2}$, $\underline{4}$, $\underline{8}$, 16, 32, 64, 35, 70, 47
$\mathcal{C}_3$	$\underline{3}$, $\underline{6}$, $\underline{12}$, 24, 48
$\mathcal{C}_5$	$\underline{5}$, $\underline{10}$, 20, 40, 80, 67, 41, 82, 71, 49
$\mathcal{C}_7$	$\underline{7}$, $\underline{14}$, 28, 56, 19, 38, 76, 59, 25, 50
$\mathcal{C}_9$	$\underline{9}$, 18, 36, 72, 51
$\mathcal{C}_{11}$	$\underline{11}$, 22, 44, 88, 83, 73, 53, $\underline{13}$, 26, 52

The dimension of C is 93 minus the number of zeros, so it is $93 - 4 \times 10 - 2 \times 5 = 43$.

Since $g(x)$ also has α^{13} and α^{14} as zeros, the actual minimum distance will be at least 15 by the BCH bound.

4.7.4 The generator polynomial $g(x)$ of C must have α^i, $1 \leq i \leq 4$, as zeros, where α is a primitive 15-th root of unity. So, $g(x) = m_1(x)m_3(x)$, with $m_1(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^4)(x - \alpha^8)$ and $m_3(x) = (x - \alpha^3)(x - \alpha^6)(x - \alpha^{12})(x - \alpha^9)$.

The smallest extension field of $GF(2)$ that contains a primitive 15-th root of unity is $GF(2^4)$. If $GF(2^4)$ is represented as $GF(2)[x]/(x^4 + x + 1)$, one has that $m_1(x) = x^4 + x + 1$ and $m_3(x) = x^4 + x^3 + x^2 + x + 1$, so $g(x) = 1 + x^4 + x^6 + x^7 + x^8$.

4.7.5 Since the smallest m for which 33 divides $2^m - 1$ is 10, the smallest extension field of $GF(2)$ containing α is $GF(2^{10})$.

The cyclotomic coset $\mathcal{C}_1$ containing 1 has as elements 1, 2, 4, 8, 16, -1, -2, -4, -8, -16. Together with $\mathcal{C}_0 = \{0\}$ one has 11 zeros of the generator polynomial $g(x)$. So the dimension of C is $33 - 11 = 22$.

The generator polynomial has the five consecutive zeros α^i, $-2 \leq i \leq 2$, so the BCH bound gives that $d \geq 6$.

If $d = 7$ or more, one gets a contradiction with the Hamming bound (Theorem 2.1.3), since

$$\sum_{i=0}^{3} \binom{33}{i} = 6018 > 2^{11}.$$

4.7.6 The conjugates of α are $\alpha, \alpha^2, \alpha^4, \alpha^8$, and those of α^3 are $\alpha^3, \alpha^6, \alpha^{12}, \alpha^9$. So the generator polynomial $g(x)$ has the four consecutive zeros: $\alpha, \alpha^2, \alpha^3, \alpha^4$ and thus the the minimum distance d is at least 5 by the BCH-bound.

Now consider the syndrome $s_1 = \alpha^7$ and $s_3 = \alpha^{14}$. We follow Algorithm 4.2.3. Note that $s_3 \neq s_1^3$. With a logarithm table of $GF(2)[x]/(x^4 + x + 1)$ one can find that $(s_3 + s_1^3)/s_1 = (\alpha^{14} + \alpha^6)/\alpha^7 = \alpha$.

The polynomial $z^2 + \alpha^7 z + \alpha$ has the the zeros: α^6 and α^{10}. So the most likely error pattern is

$$0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0.$$

4.7.7 Consider the coordinate permutation that maps a word $(c_0, c_1, \ldots, c_{62})$ into $(c'_0, c'_1, \ldots, c'_{62}) = (c_0, c_5, c_{10}, \ldots, c_{5\times62})$, where the indices have to be taken modulo 63. Note that this is indeed a coordinate permutation since 5 and 63 are coprime.

Clearly, $c(\alpha^l) = 0$, i.e. $\sum_{i=0}^{62} c_i \alpha^{il} = 0$, can be rewritten as $\sum_{i=0}^{62} c_{5i} \alpha^{5il} = 0$ or $\sum_{i=0}^{62} c'_i \alpha^{5il} = 0$ i.e. $c'(\alpha^{5l}) = 0$. So, $c(\alpha) = 0$ if and only if $c'(\alpha^5) = 0$ and, for the same reason, $c(\alpha^3) = 0$ if and only if $c'(\alpha^{15}) = 0$. We conclude that $C_{1,3}$ and $C_{5,15}$ are equivalent codes.

For $C_{3,9}$ a similar reasoning is not applicable, since $i \rightarrow 3i \pmod{63}$ is not a 1-1 mapping and thus does not yield a coordinate permutation.

That $C_{1,3}$ is indeed not equivalent to $C_{3,9}$ follows, for instance, from a dimension argument. The code $C_{1,3}$ has redundancy 6+6, while $C_{3,9}$ has redundancy 6+3, since the minimal polynomial of α^9 has degree 3.

4.7.8 In the notation of the problem, let $i(x) = a(x)g(x) = 1 - b(x)h(x)$. Then

a) From $i(x) = a(x)g(x)$ it follows that $i(x)$ is a codeword,
b) Since $g(x)h(x) \equiv 0 \pmod{x^n - 1}$, it follows that $i(x)c(x) \equiv (1 - b(x)h(x))c(x) \equiv c(x) \pmod{x^n - 1}$ for each codeword $c(x)$ in C.
c) This follows directly from b).
d) Since $i(x)$ is also a codeword, this follows directly from b) by taking $c(x) = i(x)$.
e) Let $i'(x)$ also satisfy a) and b). Then $i'(x) \equiv i'(x)i(x) \equiv i(x) \pmod{x^n - 1}$.

4.7.9 Let $g(x)$ denote the greatest common divisor of $a(x)$ and $x^n - 1$ and let C' be the cyclic code of length n with generator polymomial $g(x)$ (note that C' is well defined since $g(x)$ divides $x^n - 1$. It suffices to show that $C' = C$ (C is the smallest cyclic code of length n containing $a(x)$).

By the extended version of Euclid's Algorithm polynomials $u(x)$ and $v(x)$ exist such that

$$u(x)a(x) + v(x)(x^n - 1) = g(x).$$

It follows that $g(x)$ is an element of C and thus that C', the code generated by $g(x)$, is a subset of C. The minimality of C now implies that $C' = C$.

4.7.10 We follow Algorithm 4.5.8 with $G(x) = x^6$. From $S_1 = \alpha^{10}$, $S_3 = \alpha^{22}$ and $S_5 = \alpha^{25}$ one gets $S_2 = S_1^2 = \alpha^{20}$, $S_4 = S_2^2 = \alpha^9$ and $S_6 = S_3^2 = \alpha^{13}$, so

$$S(x) = \alpha^{10} + \alpha^{20}x + \alpha^{22}x^2 + \alpha^9 x^3 + \alpha^{25}x^4 + \alpha^{13}x^5.$$

The next step is to apply Euclid's Algorithm 4.5.3 until the degree$(s_i(x)) \le 3$. One gets

$$
\begin{aligned}
s_0(x) &= G(x) = x^6, \\
v_0(x) &= 0,
\end{aligned}
$$

$$
\begin{aligned}
s_1(x) &= S(x) = \\
&= \alpha^{10} + \alpha^{20}x + \alpha^{22}x^2 + \alpha^9 x^3 + \alpha^{25}x^4 + \alpha^{13}x^5, \\
v_1(x) &= 1,
\end{aligned}
$$

$$
\begin{aligned}
q_2(x) &= \alpha^{30} + \alpha^{18}x, \\
s_2(x) &= \alpha^9 + \alpha^4 x + \alpha^{20}x^2 + \alpha^{26}x^3 + \alpha^{22}x^4, \\
v_2(x) &= \alpha^{30} + \alpha^{18}x,
\end{aligned}
$$

$$
\begin{aligned}
q_3(x) &= \alpha^{15} + \alpha^{22}x, \\
s_3(x) &= \alpha^{23} + \alpha^{27}x + \alpha^{30}x^2 + \alpha^{20}x^3, \\
v_3(x) &= \alpha^{13} + \alpha^{13}x + \alpha^9 x^2,
\end{aligned}
$$

$$
\begin{aligned}
q_4(x) &= \alpha^2 + \alpha^2 x, \\
s_4(x) &= \alpha^{18} + \alpha^{11}x^2 \\
v_4(x) &= \alpha^8 + \alpha^{18}x + \alpha^{21}x^2 + \alpha^{11}x^3.
\end{aligned}
$$

We next have to find the zeros of $v_4(x)$. Trying out the successive elements in $GF(2^5)$ one finds α^8, α^{22} and α^{29}.

The reciprocals of these zeros give the error locations: 2, 9 and 23.

4.7.11 First compute the syndrome of $\underline{r}$. One obtains $S_1 = \alpha^3$, $S_2 = \alpha^4$, $S_3 = \alpha^4$, $S_4 = 0$.

We follow Algorithm 4.5.8 with $G(x) = x^4$ and $S(x) = \alpha^3 + \alpha^4 x + \alpha^4 x^2$. We get:

$$
\begin{aligned}
s_0(x) &= G(x) = x^4, \\
v_0(x) &= 0,
\end{aligned}
$$

$$
\begin{aligned}
s_1(x) &= S(x) = \alpha^3 + \alpha^4 x + \alpha^4 x^2 \\
v_1(x) &= 1,
\end{aligned}
$$

$$
\begin{aligned}
q_2(x) &= \alpha^5 + \alpha^3 x + \alpha^3 x^2, \\
s_2(x) &= \alpha + x \\
v_2(x) &= \alpha^5 + \alpha^3 x + \alpha^3 x^2.
\end{aligned}
$$

We conclude that $\sigma(x) = v_2(x)/\alpha^3 = x^2 + x + \alpha^2$ and $\omega(x) = s_2(x)/\alpha^3 = \alpha^5 + \alpha^4 x$. The zeros of $\sigma(x)$ are α^4 and α^5, so the error values follow from substituting these values in $\omega(x)$. We get $\omega(\alpha^4) = \alpha^6$ and $\omega(\alpha^5) = \alpha^3$. The error locations are given by the reciprocals of the zeros of $\sigma(x)$, so they are α^2 and α^3. We find the following error pattern:

$$
\underline{e} = (0, 0, \alpha^3, \alpha^6, 0, 0, 0).
$$

Subtracting $\underline{e}$ from $\underline{r}$ gives the codeword

$$
\underline{c} = (\alpha^3, \alpha, \alpha, 1, 0, \alpha^3, 1).
$$

4.7.12 In Example 4.6.2 one has $s = t = 1$ and thus $g = 0$. It follows from Corollary 4.6.5 that for $m \geq 0$ both the space $L(m)$ and the AG-code $C(\mathcal{P}, m)$ have dimension $m + 1$.

In Example 4.6.3 one has $s = 3$ and $t = 2$. Hence $g = (3 - 1)(2 - 1)/2 = 1$. It follows from Corollary 4.6.5 that for $m > 0$ both the space $L(m)$ and the AG-code $C(\mathcal{P}, m)$ have dimension m.

4.7.13 By Corollary 4.6.5, $g = 6$ and for $m > 10$ the code $C(\mathcal{P}, m)$ has dimension $m - 5$.

For $m = 10$ we find solutions $x^i y^j$ to Equation (4.42) for $(i, j) = (0,0), (0,1), (0,2),$ $(1,0), (1,1)$ $(2,0)$. It follows from (4.42) that $1 \in L(0)$, $x^5 \in L(5)$, $x^{10} \in L(10)$, $y \in L(4)$, $x^5 y \in L(9)$ and $y^2 \in L(8)$. So, for $0 \le m \le 10$, we get the following list of dimensions for $L(m)$:

m	0	1	2	3	4	5	6	7	8	9	10
dim $L(m)$	1	1	1	1	2	3	3	3	4	5	6

4.7.14 We have the following functions $\phi_u(x, y) = x^i y^j$

u	1	2	3	4	5	6
(i, j)	$(0,0)$	$(0,1)$	$(0,2)$	$(1,0)$	$(0,3)$	$(1,1)$

u	7	8	9	10	11
(i, j)	$(0,4)$	$(1,2)$	$(2,0)$	$(1,3)$	$(2,1)$

1. The pairs (u, v) in $\mathbf{T}$ for which (4.53) holds but not (4.54) are given by $(1,2), (2,1)$ and $(3,3)$.

2. The pairs (u, v) with $T_{u,v}$ of degree 12 are given by $(1,11), (2,9), (3,7), (4,6),$ $(5,5), (6,4), (7,3), (9,2), (11,1)$.

   The pair $(1,11)$ tells us that $S_{i,j}$ can be computed with $(i, j) = (0,0) + (2,1)$. The other pairs result in the same $S_{2,1}$ as we shall see in (d).

3. Of the nine $(= 11 + 2 - 2 \times 2)$ pairs above, six lie in the same row or column as one of the three known discrepancies. What remains are the pairs: $(4,6),$ $(5,5), (6,4)$. Under the assumption that these three candidates are correct, one finds $T_{4,6} = T_{6,4} = 1$ and $T_{5,5} = 0$.

4. By (4.50) we get:
   i) $\phi_4 \cdot \phi_6 = x^1 y^0 \cdot x^1 y^1 = x^2 y^1$, so $T_{4,6} = 1$ implies $S_{2,1} = 1$.
   ii) $\phi_6 \cdot \phi_6 = x^1 y^1 \cdot x^1 y^0 = x^2 y^1$, so $T_{6,4} = 1$ implies $S_{2,1} = 1$.
   iii) $\phi_5 \cdot \phi_5 = x^0 y^3 \cdot x^0 y^3 = x^0 y^6 = (1 + x + x^2)y$, so $T_{6,4} = 1$ implies $S_{0,1} + S_{1,1} + S_{2,1} = 0$. Since $S_{0,1} = T_{1,2} = 1$ and $S_{1,1} = T_{1,6} = 1$, we get as estimate from $T_{5,5} = 0$ that $S_{2,1} = 0$.

   The majority decision yields $S_{2,1} = 1$.

5. $x^0 y^0 \cdot x^2 y^1 = x^2 y^1$, so $T_{1,11} = T_{11,1} = S_{2,1} = 1$.
   $x^0 y^1 \cdot x^2 y^0 = x^2 y^1$, so $T_{2,9} = T_{9,2} = S_{2,1} = 1$.
   $x^0 y^2 \cdot x^1 y^2 = x^2 y^1$, so $T_{3,8} = T_{8,3} = S_{2,1} = 1$.
   $x^0 y^2 \cdot x^0 y^4 = x^0 y^6 = (1 + x + x^2)y$, so $T_{3,7} = T_{7,3} = S_{0,1} + S_{1,1} + S_{2,1} = 1 + 1 + 1 = 1$.
   $x^1 y^0 \cdot x^1 y^1 = x^2 y^1$, so $T_{4,6} = T_{6,4} = S_{2,1} = 1$.
   $x^0 y^3 \cdot x^0 y^3 = x^0 y^6 = (1 + x + x^2)y$, so $T_{5,5} = S_{0,1} + S_{1,1} + S_{2,1} = 1 + 1 + 1 = 1$.

4.7.15 Taking the points P_j, $1 \leq j \leq 8$, in $\mathcal{P}$ in the same order as in Example 4.6.3, the code $C(\mathcal{P}, m)$, $1 \leq m \leq 7$, in Example 4.6.3 is generated by the first m rows of

$$
G = \begin{pmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & \alpha & \alpha & \alpha^2 & \alpha^2 \\
\alpha & \alpha^2 & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 1 & 1 & \alpha^2 & \alpha^2 & \alpha & \alpha \\
0 & 0 & 0 & 1 & 0 & \alpha & 0 & \alpha^2 \\
\alpha^2 & \alpha & 0 & 1 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & \alpha^2 & 0 & \alpha \\
0 & 0 & 0 & 1 & 0 & \alpha & 0 & \alpha^2
\end{pmatrix}
\qquad
\begin{array}{l}
\phi_1 = 1 \\
\phi_2 = y \\
\phi_3 = x \\
\phi_4 = y^2 \\
\phi_5 = xy \\
\phi_6 = x^2 \\
\phi_7 = xy^2 \\
\phi_8 = x^2 y
\end{array}.
$$

It follows from $1 + \alpha + \alpha^2 = 0$ that each of the eight rows of G is orthogonal to the top row of G and thus that $\sum_{j=1}^{8} \phi_i(P_j) = 0$ for $1 \leq i \leq 8$.

To check that the second row of G is orthogonal to the first 6 rows of G, it suffices to observe that $\phi_2\phi_3 = \phi_5, \phi_2\phi_4 = y^3 = 1 + x + x^2 = \phi_1 + \phi_3 + \phi_6, \phi_2\phi_5 = \phi^7$ and $\phi_2\phi_6 = \phi^8$ and for all these functions we already know that $\sum_{j=1}^{8} \phi_i(P_j) = 0$.

To check that the third row of G is orthogonal to the first 5 rows of G, we observe that $\phi_3\phi_4 = \phi_7$ and $\phi_3\phi_5 = \phi_8$ and again for these functions we already know that $\sum_{j=1}^{8} \phi_i(P_j) = 0$.

The above shows that the code generated by the first $8 - m$ rows of G is in fact the dual code of the code generated by the first m rows of G, i.e. $C(\mathcal{P}, 8 - m)$ is equal to $C^*(\mathcal{P}, m)$.

C.5 Solutions to Chapter 5

5.6.1 A burst of length 9 affects at most three consecutive 4-tuples of coordinates. Since $16 = 2^4$, each symbol in the Reed-Solomon code represents a binary 4-tuple. It follows that e needs to be 3.

A burst of length 10 can affect four consecutive 4-tuples of coordinates (starting at the last coordinate of the first 4-tuple and ending at the first coordinate of the fourth 4-tuple), but never more. So e needs to be 4.

5.6.2 By Theorem 5.3.3, the binary Fire code generated by $(x^5 - 1)(1 + x + x^3)$ can correct bursts of length up to 3. By Theorem 5.2.1, interleaving this code at depth 5 creates a code that can correct bursts of length up to $3 \times 5 = 15$.

The length of this code is $5 \times 35 = 175$, its dimension is $5 \times (35 - (3 + 5)) = 135$.

5.6.3 The length of the binary Fire code generated by $g(x) = (x^7 - 1)(x^4 + x + 1)$ is $\mathrm{lcm}(7, 2^4 - 1) = 105$ by Lemma 5.3.2. This code can correct bursts of length up to 4 by Theorem 5.3.3.

Write the received word $r(x)$ as $c(x) + x^i B(x)$, where $c(x)$ is a codeword and $x^i B(x)$ the burst. The syndrome of $r(x)$ can be found by reducing $r(x) \pmod{g(x)}$ further modulo $x^7 - 1$ and modulo $x^4 + x + 1$. One gets

$$
\begin{aligned}
s_1(x) &\equiv r(x) \equiv x + x^2 + x^4 + x^5 + x^8 \equiv \\
&\equiv 2 + x^4 + x^5 \pmod{x^7 - 1},
\end{aligned}
$$

$$
\begin{aligned}
s_2(x) &\equiv r(x) \equiv x + x^2 + x^4 + x^5 + x^8 \equiv \\
&\equiv x + x^2 \pmod{x^4 + x + 1}.
\end{aligned}
$$

The gap of length at least 3 in $s_1(x)$ is at positions $x^6, 1, x$ modulo $x^7 - 1$, so the burst starts at coordinate $i' = 2$ modulo 7 and has pattern $B(x) = 1 + x^2 + x^3$.

To find i one has to solve

$$
x^{2+j7}(1 + x^2 + x^3) \equiv s_2(x) \equiv x + x^2 \pmod{x^4 + x + 1}.
$$

From the logarithm table of $GF(2)[x]/(1 + x + x^4)$, one gets $1 + x^2 + x^3 \equiv x^{13} \pmod{1 + x + x^4}$ and $x + x^2 \equiv x^5 \pmod{1 + x + x^4}$. So, one needs to solve

$$
2 + j7 + 13 \equiv 5 \pmod{15}.
$$

The solution $j = 5$ gives the burst $x^{2+5 \times 7}(1 + x^2 + x^3) = x^{37} + x^{39} + x^{40}$.

The redundancy of this Fire code is 7+4=11. The Reiger bound (Theorem 5.1.4) yields that the redundancy is at least $2 \times b = 8$.

5.6.4 The coordinates in $\mathcal{A}(7, 10)$ are ordered in the following way.

0	7	14	21	28	35	42	49	56	63
64	1	8	15	22	29	36	43	50	57
58	65	2	9	16	23	30	37	44	51
52	59	66	3	10	17	24	31	38	45
46	53	60	67	4	11	18	25	32	39
40	47	54	61	68	5	12	19	26	33
34	41	48	55	62	69	6	13	20	27

Two bursts of length at most 6 that have the same syndrome (see the proof of Lemma 5.4.2) are for example one that has its ones at coordinates 0 and 5, while the other has its ones at coordinates 35 and 40.

5.6.5 We follow Algorithm 5.4.4. So first we compute the horizontal and vertical syndromes $\underline{h}$ and $\underline{v}$.

$$R = \begin{array}{|c|c|c|c|c|c|c|c|c|c|}
\hline
1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
\hline
1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\
\hline
0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
\hline
1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\
\hline
0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\
\hline
1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline
\end{array}$$
$$\begin{array}{cccccccccc}
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 &
\end{array}$$

The one-coordinates in $\underline{h}$ are at $i_1 = 1$, $i_2 = 3$, $i_3 = 5$ and $i_4 = 6$.

In $\underline{v}$ a gap of size 4 occurs at coordinates 1, 2, 3 and 4. So, one gets $j_1 = 5$, $j_2 = 7$, $j_3 = 8$ and $j_4 = 9$.

We find the burst with ones at positions (i_l, j_l), $1 \leq l \leq 4$, so at: (1,5), (3,7), (4,8) and (5,9). This is the burst of length 5 starting at position (4,8) with burst pattern 11101.

5.6.6 First compute the syndrome $s_e = r(x) \pmod{1 + x + x^2}$ and $s_p = r(x) \pmod{1 + x + x^4}$. One gets $s_e = x$ and $s_p = 1 + x^3$.

It follows from $s_e = x$ that there are three bursts of degree at most 2 with $s_e = x$, namely $b_1(x) = 1x^{1+3l}$, $b_2(x) = (1 + x)x^{3l+2}$ and $b_3(x) = (1 + x^2)x^{3l}$.

Now, $1 + x^3 \equiv x^{14} \pmod{1 + x + x^4}$, $1 + x \equiv x^4 \pmod{1 + x + x^4}$. and $1 + x^2 \equiv x^8 \pmod{1 + x + x^4}$.

The equation

$$b_1(x) \equiv 1x^{1+3l} \equiv x^{14} \pmod{1 + x + x^4}$$

can be rewritten as

$$1 + 3l \equiv 14 \pmod{15},$$

which has no solution.

The equation

$$b_2(x) \equiv (1 + x)x^{3l+2} \equiv x^4 x^{3l+2} \equiv x^{14} \pmod{1 + x + x^4}$$

can be rewritten as

$$6 + 3l \equiv 14 \pmod{15},$$

which has no solution.

The equation

$$b_3(x) \equiv (1+x^2)x^{3l} \equiv x^8 x^{3l} \equiv x^{14} \pmod{1+x+x^4}$$

can be rewritten as

$$8 + 3l \equiv 14 \pmod{15},$$

which has $l \equiv 2 \pmod 5$ as solution.

We conclude that the burst of length $(1+x^2)x^6$ has occurred and that the transmitted codeword is $r(x) - (1+x^2)x^6 = 1 + x^2 + x^4 + x^8 + x^{10} + x^{13} + x^{14}$.

5.6.7 Since $e(x) = 1 + x^3$ has period $3 = 2^2 - 1$ it follows that the generator polynomial $g(x)$ must have the form $g(x) = (1 + x^3)p(x)$, with $p(x)$ is a primitive polynomial of even degree m.

Possible burst patterns of length at most 4, that are not the product of smaller degree patterns are: $1, 1 + x, 1 + x + x^2, 1 + x + x^3$ and $1 + x^2 + x^3$.

Define a, b, c and d by

$$
\begin{aligned}
1 + x &\equiv x^a &&(\operatorname{mod} p(x)), \\
1 + x + x^2 &\equiv x^b &&(\operatorname{mod} p(x)), \\
1 + x + x^3 &\equiv x^c &&(\operatorname{mod} p(x)), \\
1 + x^2 + x^3 &\equiv x^d &&(\operatorname{mod} p(x)).
\end{aligned}
$$

Consider the burst patterns $B_1(x) = 1$ and $B_2(x) = 1 + x + x^3$. Clearly, $B_1(x) \equiv x^l B_2(x) \pmod{1 + x^3}$, i.e. $1 \equiv x^l(1 + x + x^3) \pmod{1 + x^3}$ if and only if $l \equiv 2 \pmod 3$. From the condition $B_1(x) \not\equiv x^l B_2(x) \pmod{p(x)}$, i.e. $1 \not\equiv x^l x^c \pmod{p(x)}$ for all $l \equiv 2 \pmod 3$ it now follows that $0 \not\equiv 2 + c \pmod 3$, i.e. $c \not\equiv 2 \pmod 3$.

Similarly, the burst patterns $B_1(x) = 1$ and $B_2(x) = 1 + x^2 + x^3$ yield the condition $d \not\equiv 2 \pmod 3$.

The burst patterns $B_1(x) = 1 + x$ and $B_2(x) = 1 + x^2 = (1+x)^2$ yield the condition $a \not\equiv 1 + 2a \pmod 3$ and thus $a \not\equiv 2 \pmod 3$.

The burst patterns $B_1(x) = 1 + x + x^3$ and $B_2(x) = 1 + x^2 + x^3$ yield the condition $c \not\equiv 2 + d \pmod 3$ and thus $c + 2d \not\equiv 2 \pmod 3$.

It just so turns out that none of the other burst patterns give rise to further relations, so that we have the following AES conditions in the a, c and d defined above (and not in b):

$$a \not\equiv 2 \quad (\text{mod } 3),$$
$$c \not\equiv 1 \quad (\text{mod } 3),$$
$$d \not\equiv 2 \quad (\text{mod } 3),$$
$$c + 2d \not\equiv 2 \quad (\text{mod } 3),$$

Modulo 3 there are four solutions: $(a, c, d) \equiv$ (0,0,0), (1,0,0), (0,2,1) and (1,2,1).

C.6   Solutions to Chapter 6

6.6.1 The received sequence 11, 11, 00, 11, 10, 01, 00, 11, 10 gives rise to distances in Figure 6.3 that can be found Figure C.1.
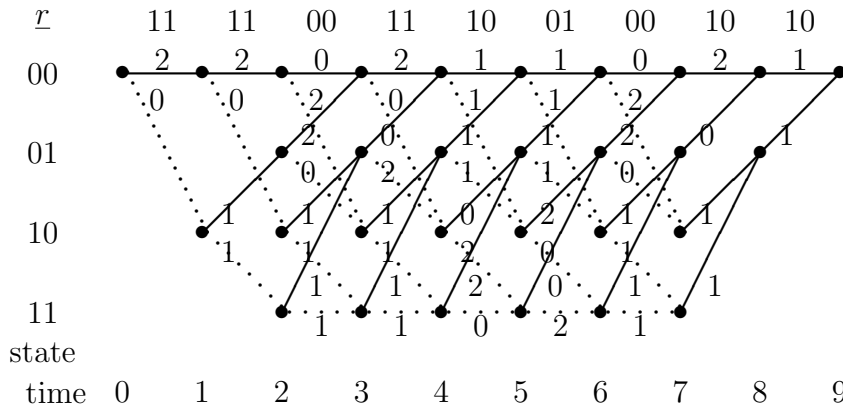
Figure C.1: Distances to sequence 11, 11, 00, 11, 10, 01, 00, 11, 10

In Figure C.2 the lowest weight path to every state can be found.

So, the path closest to the received sequence is

11 10 00 01 10 01 00 10 11

which is at distance 4. The corresponding information sequence is 1 0 1 1 1 0 1 0 0.
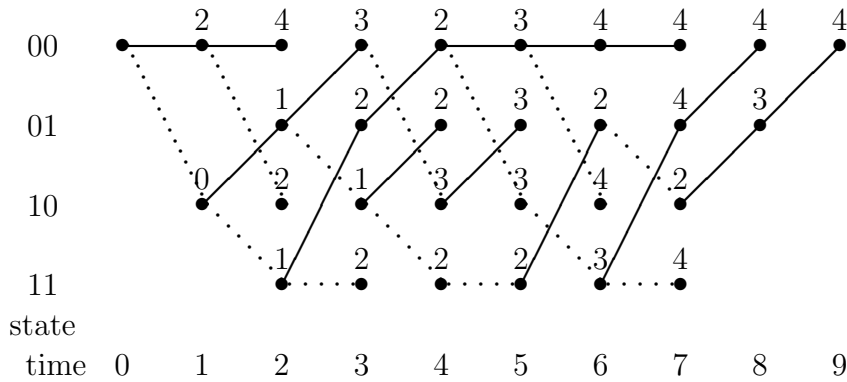
Figure C.2: Lowest weight paths

6.6.2 Elementary row operations yield the matrix

$$
G = \begin{pmatrix}
1 & 1+x & 1+x^2 & 1 & 1+x+x^2 \\
0 & x+x^2 & 1+x^2 & 1+x & 1+x \\
0 & 0 & 1+x^2 & 0 & 1+x^2
\end{pmatrix}.
$$

Elementary column operations yield the matrix

$$
G = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & x+x^2 & 1+x^2 & 1+x & 1+x \\
0 & 0 & 1+x^2 & 0 & 1+x^2
\end{pmatrix}.
$$

Interchanging columns 2 and 4 and further column operations yield

$$
G = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1+x & 0 & 0 & 0 \\
0 & 0 & 1+x^2 & 0 & 1+x^2
\end{pmatrix}.
$$

Subtracting column 3 from column 5 yields

$$
G = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 1+x & 0 & 0 & 0 \\
0 & 0 & 1+x^2 & 0 & 0
\end{pmatrix},
$$

from which the invariant factors $1, 1+x$ and $1+x^2$ can be read.

6.6.3 The input sequence $1 + x + x^2 + x^3 + \cdots = \frac{1}{1+x}$ of infinite weight results in the output $(1, 1+x)$ of weight 3.
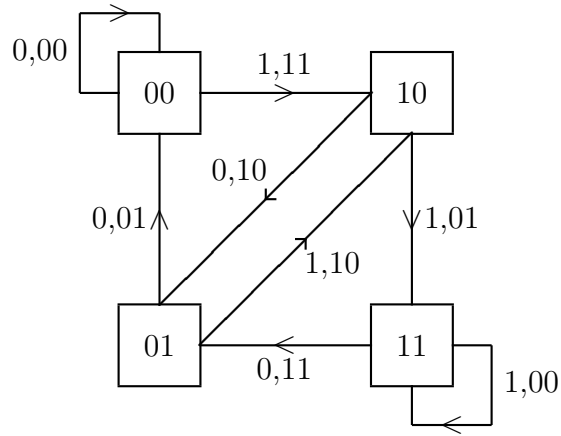
Figure C.3: State diagram of the code in Problem 6.6.3
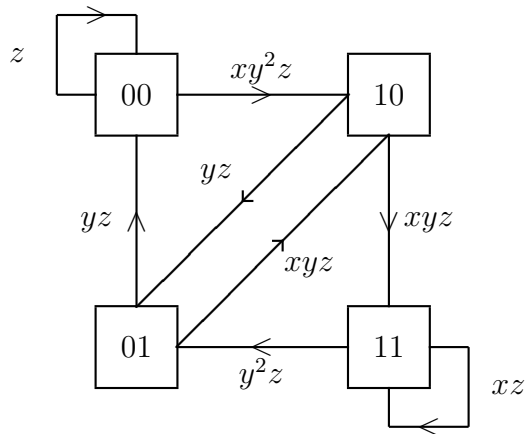
Figure C.4: Labeled diagram of the code in Problem 6.6.3

The state diagram of this code can be found in Figure C.3.

The corresponding labeled diagram of this code can be found in Figure C.4.

The four fundamental path enumerators satisfy the followoing relations:

$$
\begin{aligned}
A_{10}(x,y,z) &= yzA_{01}(x,y,z) + xyzA_{11}(x,y,z), \\
A_{00}(x,y,z) &= xy^2zA_{10}(x,y,z), \\
A_{01}(x,y,z) &= yz + xyzA_{10}(x,y,z), \\
A_{11}(x,y,z) &= y^2zA_{01}(x,y,z) + xzA_{11}(x,y,z).
\end{aligned}
$$

This gives rise to the following matrix equation:

$$
\begin{pmatrix}
1 & 0 & -xy^2z & 0 \\
0 & 1 & -xyz & 0 \\
0 & -yz & 1 & -xyz \\
0 & -y^2z & 0 & 1-xz
\end{pmatrix}
\begin{pmatrix}
A_{00}(x,y,z) \\
A_{01}(x,y,z) \\
A_{10}(x,y,z) \\
A_{11}(x,y,z)
\end{pmatrix}
=
\begin{pmatrix}
0 \\
yz \\
0 \\
0
\end{pmatrix}.
$$

With Kramer's Rule it is now simple to determine $A_{00}(x,y,z)$.

$$
A_{00}(x,y,z) = \frac{
\begin{vmatrix}
0 & 0 & -xy^2z & 0 \\
yz & 1 & -xyz & 0 \\
0 & -yz & 1 & -xyz \\
0 & -y^2z & 0 & 1-xz
\end{vmatrix}
}{
\begin{vmatrix}
1 & 0 & -xy^2z & 0 \\
0 & 1 & -xyz & 0 \\
0 & -yz & 1 & -xyz \\
0 & -y^2z & 0 & 1-xz
\end{vmatrix}
},
$$

and thus

$$
A_{00}(x,y,z) = \frac{xy^4z^3(1-xz-xy^2z)}{1-xz(1+y^2z+xy^4z^2-xy^2z^2)}.
$$

From $1-xz$ in the denominator one can see that in the expansion of $A(x,y,z)$ terms $x^{i+a}y^bz^{i+c}$ occur for all i, corresponding to paths with $i+a$ input ones out of $i+c$ inputs, with output weight just b. This reflects the catastrophic character of this convolutional code.
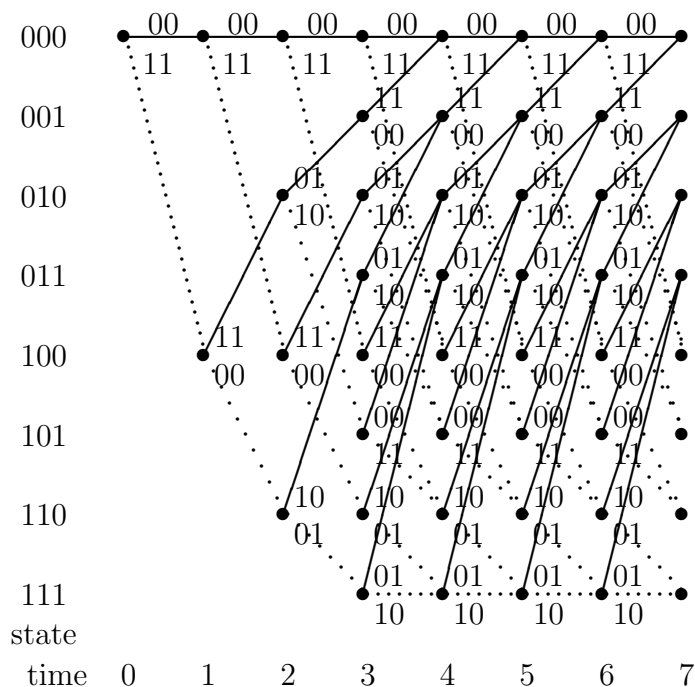
Figure C.5: Trellis of code in Problem 6.6.4

6.6.4 The trellis for the (2,1)-convolutional code C generated by

$$G = \left(\begin{array}{cc} 1 + x + x^3 & 1 + x + x^2 + x^3 \end{array} \right)$$

is given in Figure C.5.

The free distance of C is the minimum weight of the non-trivial paths from state 000 to state 000. The input sequence 11000 starting in state 000 gives rise to the path that also ends in state 000 and has output 11 00 10 10 11 of weight 6. In this small case one can easily check that six is indeed the minimum.

6.6.5 Omitting the subscripts, the received sequence

$$\cdots, a, a, d, c, d, a, a, \cdots$$

corresponds, according to Figure 6.9, to to the binary sequence

$$\cdots \quad 00 \quad 00 \quad 11 \quad 10 \quad 11 \quad 00 \quad 00 \quad \cdots.$$

This is the output sequence of the $(2, 1)$-convolutional code of Figure 6.1 when the input sequence is:

$$\cdots \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad \cdots.$$

175

Putting the binary image of the subscripts in front of these bits results in the input sequence:

$$\cdots \quad 000 \quad 000 \quad 011 \quad 110 \quad 100 \quad 000 \quad 000 \quad \cdots .$$

C.7 Solutions to Appendix A

A.6.1 For $n = 3$ one has $\gcd(F_3, F_2) = \gcd(2, 1)$. Since $2 = 2 \times 1 + 0$, the calculation of this gcd by Euclid's Algorithm involves $1 = n - 2$ steps.

For $n > 3$ the assertion follows with an induction argument. Indeed, the recurrence relation $F_n = 1 \times F_{n-1} + F_{n-2}$ implies that the first step in Euclid's Algorithm applied to the pair (F_n, F_{n-1}) yields the pair (F_{n-1}, F_{n-2}).

A.6.2 Note that f is the largest root of $x^2 = x + 1$, the characteristic equation for the recurrence relation $F_n = F_{n-1} + F_{n-2}$.

The statement can easily be verified for small values of b. We proceed with induction on b. Write $b = qa + r$, $0 \le r < a$. We distinguish two cases.

Case 1: $a \le b/f$.

By the induction hypothesis Euclid's Algorithm computes $\gcd(a, r)$ using at most $\log_f(a)$ iterations. So for the computation of $\gcd(b, a)$ at most

$$1 + \log_f(a) = \log_f(f \cdot a) \le \log_f(b)$$

iterations are needed.

Case 2: $b/f < a \le b$.

It follows that $q = 1$, $b = a + r$ and $0 \le r = b - a < b(1 - f^{-1}) = b/f^2$. Writing $a = q'r + r'$, $0 \le r' < r$, it follows from the induction hypothesis that Euclid's Algorithm involves at most $\log_f(r)$ iterations to compute $\gcd(r, r')$. So for $\gcd(a, b)$ at most

$$2 + log_f(r) = log_f(f^2.r) < log_f(b)$$

iterations are needed.

An explicit formula for F_n is given by

$$F_n = \tfrac{1}{\sqrt{5}} \cdot \left(\tfrac{1+\sqrt{5}}{2}\right)^n - \tfrac{1}{\sqrt{5}} \cdot \left(\tfrac{1-\sqrt{5}}{2}\right)^n .$$

A.6.3 For $k = 2$ the first assertion is equivalent to Corollary A.1.4. To give a proof by induction, one has to apply Corollary A.1.4 again to $a = a_1$ and $b = a_2 \ldots a_k$. It follows that either p divides a_1 or one of the factors a_i, $2 \le i \le k$.

That any integer can be written as product of powers of primes follows more or less directly from the definition of a number being prime (either n is a prime, or it is divisible by a smaller prime p; now continue with n/p). The uniqueness of this factorization again follows with an induction argument from the first part of the problem.

Equations (A.9) and (A.10) follow directly from equation (A.8) and in turn imply (A.11).

A.6.4 Take as set S the set of integers $\mathcal{Z}$ and as subset U of $S \times S$ the set of points $\{(i, i + lm) \mid i \in \mathcal{Z} \text{ and } l \in \mathcal{Z}\}$.

A.6.5 One first has to check that equation (A.23) yields a proper definition of an addition on $\mathcal{Z}_m$, in other words that the result of the addition is independent of the choice of a in $< a >$ and b in $< b >$. Indeed, if a' is in $< a >$ and b' in $< b >$, then $a' = a + im$ and $b' = b + jm$ for some integers i and j, so $a' + b' = a + b + (i + j)m$, which is element of the same congruence class $< a + b >$ that contains $a + b$.

Associativity directly follows from $< a > +(< b > + < c >) = < a > + < b + c > = < a + b + c > = < a + b > + < c > = (< a > + < b >) + < c >$, where the associativity in $\mathcal{Z}$ has been used. Commutivity follows similarly from the commutivity in $\mathcal{Z}$.

The unit-element of course is $< 0 >$, since $< a > + < 0 > = < a >$. Similarly the additive inverse of $< a >$ is given by $< -a >$.

A.6.6 Suppose that (H1), (H2), and (H3) hold and let g and h be both elements in H. By (H3) the element h^{-1} is in H and by (H1) also the product gh^{-1} is in H. This proves (H).

Conversely, assume that only (H1) and (H) hold. Apply (H) with $h = g$ to establish that the inverse e of G is in fact an element of H (so (H2) holds). Apply (H) with $g = e$ to find that the inverse of an element of H also lies in H (so (H3) holds).

A.6.7 The nontrivial subgroups of $(\mathcal{Z}_{15}, +)$ have orders 3 or 5 and consists of the classes represented by $\{0, 5, 10\}$ respectively $\{0, 3, 6, 9, 12\}$.

The nontrivial subgroups of $(\mathcal{Z}_{17}^*, \times)$ have orders 2, 4 or 8 and consists of the classes represented by $\{1, 16\}$, respectively $\{1, 4, 13, 16\}$ and $\{1, 2, 4, 8, 9, 13, 15, 16\}$.

Since 3 generates $\mathcal{Z}_{17}^*$ (it is a primitive element of $GF(17)$), the multiplicative order of the various elements in $\mathcal{Z}_{17}^*$ follows from Lemma A.3.12 and is given by

element	1	2	3	4	5	6	7	8
order	1	8	16	4	16	16	16	8

element	9	10	11	12	13	14	15	16
order	8	16	16	16	4	16	8	2

A.6.8 One has to check properties (E1), (E2) and (E3) in Definition A.2.1.

(E1) $s \sim s$ (reflexivity) translates by (A.25) into $ss^{-1} \in H$, which is true since $ss^{-1} = e$ (and (H2) holds).

(E2) $(s \sim t) \Rightarrow (t \sim s)$ (symmetry) translates into $st^{-1} \in H \Rightarrow ts^{-1} \in H$. This implication holds since the inverse of any element in H must also lie in H (see (H3)) and ts^{-1} is the inverse of st^{-1}, as can be checked directly.

(E3) $(s \sim t \wedge t \sim u) \Rightarrow (s \sim u)$ (transitivity) is equivalent to $(st^{-1} \in H \wedge tu^{-1} \in H) \Rightarrow (su^{-1} \in H)$ and this follows from $su^{-1} = (st^{-1})(tu^{-1})$ (and (H1)).

To show the second assertion, note that b is in the same class as a if and only if $b \sim a$, i.e. iff $ba^{-1} \in H$, i.e. iff $b \in Ha = \{ha \mid h \in H\}$.

A.6.9 To prove that the set of multiples of $1 + x^3$ form a subring of $\mathcal{F}_2[x]$ and is even an ideal in $\mathcal{F}_2[x]$ is completely analogous to showing that $m\mathcal{Z}$ is an ideal in $\mathcal{Z}$. We shall only prove property (I) in Definition A.3.8.

Let $i(x) \in (1+x^3)$ and $r(x) \in \mathcal{F}_2[x]$. Then $i(x) = a(x)(x^3+1)$ for some polynomial a(x). It follows that $i(x)r(x) = a(x)(x^3 + 1)r(x) = (a(x)r(x))(x^3 + 1)$ is also in $(1 + x^3)$.

From the factorization $1+x^3 = (1+x)(1+x+x^2)$ one can deduce the zero-divisors $< 1+x >$ and $< 1+x+x^2 >$. Indeed, $< 1+x >< 1+x+x^2 >=< 1+x^3 >=< 0 >$ in the residue class ring $(\mathcal{F}_2[x]/(1 + x^3), +, \times)$.

A.6.10 From $(1 + x^2 + x^4)(1 + x + x^3) = 1 + x + x^2 + x^4 + x^7 = x^2(1 + x^2 + x^5) + (1+x)$ it follows that

$$(1 + x^2 + x^4)(1 + x + x^3) \equiv 1 + x \ (\text{mod } 1 + x^2 + x^5).$$

A.6.11 Of the four binary polynomials of degree 2, three are divisible by the first degree irreducible polynomials x or $x + 1$, namely $x^2, x(x + 1)$, and $(x + 1)^2$. So $I_2(2) = 4 - 3 = 1$. The only binary, irreducible polynomial of degree 2 is $x^2 + x + 1$.

Of the eight binary polynomials of degree 3, four are the product of first degree polynomials namely $x^i(x + 1)^{3-i}$, $0 \le i \le 3$. Further two of them are the product of the only irreducible polynomial of degree 2 and one of the two first degree polynomials (namely $x(x^2 + x + 1)$ and $(x + 1)(x^2 + x + 1)$). So, $I_2(3) = 8 - 4 = 2$ (these two irreducible polynomials of degree 3 are $x^3 + x^2 + 1$ and $x^3 + x + 1$).

Of the sixteen binary polynomials of degree 4, five are the product of first degree polynomials, three of them are the product of the only irreducible polynomial of degree 2, and two first degree polynomials, one is the product of two irreducible polynomials of degree 2 and two times two of them are the product of a third degree, irreducible polynomial with a first degree polynomial. So $I_2(4) = 16 - 5 - 3 - 1 - 4 = 3$ (these are $x^4 + x^3 + 1, x^4 + x + 1$ and $x^4 + x^3 + x^2 + x + 1$).

To check (A.38), observe that indeed $I_2(2) = \frac{1}{2}(2^2 - 2^1) = 1$, $I_2(3) = \frac{1}{3}(2^3 - 2^1) = 2$ and $I_2(4) = \frac{1}{4}(2^4 - 2^2 + 0 \cdot 2^1) = 3$.

A.6.12 When applying Euclid's Algorithm (Algorithm A.1.3) to $1+x+x^3$ and $1+x^2+x^5$ to find the multiplicative inverse of $1 + x + x^3$ modulo $1 + x^2 + x^5$ one does not need to evaluate the v_n-sequence. One gets the following iterations:

$$s_0(x) = x^5 + x^2 + 1, \qquad u_0(x) = 0,$$

$$s_1(x) = x^3 + x + 1, \qquad u_1(x) = 1,$$

$$q_2(x) = x^2 + 1, \qquad s_2(x) = x, \qquad u_2(x) = x^2 + 1,$$

$$q_3(x) = x^2 + 1, \qquad s_3(x) = 1, \qquad u_3(x) = x^4,$$

$$q_4(x) = x, \qquad s_4(x) = 0,$$

Now that $s_4(x) = 0$ and $s_3(x) = 1$ one has by (A.7) that the gcd of $1 + x + x^3$ and $1 + x^2 + x^5$ is indeed 1 and that

$$x^4(x^3 + x + 1) \equiv u_3(x)(x^3 + x + 1) \equiv 1 \ (\mathrm{mod}\ x^5 + x^2 + 1).$$

So, the multiplicative inverse of $1 + x + x^3 \ (\mathrm{mod}\ 1 + x^2 + x^5)$ is given by x^4.

A.6.13 By computing the successive powers of x modulo $1 + x + x^2 + x^3 + x^4$ one gets $1, x, x^2, x^3, x^4$ and then $x^5 \equiv 1 \ (\mathrm{mod}\ 1 + x + x^2 + x^3 + x^4)$, so x has order 5.

On the other hand, $(x^2 + x^3)^2 \equiv 1 + x^2 + x^3 \ (\mathrm{mod}\ 1 + x + x^2 + x^3 + x^4)$, and $(x^2 + x^3)^3 \equiv 1 \ (\mathrm{mod}\ 1 + x + x^2 + x^3 + x^4)$, so $x^2 + x^3$ has order 3.

The product is $x(x^2 + x^3) \equiv 1 + x + x^2 \ (\mathrm{mod}\ 1 + x + x^2 + x^3 + x^4)$. It clearly has an order dividing $3 \times 5 = 15$. On the other hand the order must be a multiple of both 3 and 5, so it is equal to 15. (Indeed, if the product of two elements has order k and also one of the elements has order k then so will the other element have an order dividing k.)

A.6.14 The log table of $GF(3)[x]/(2 + 2x + x^2)$ is given by

	1	x
0	0	0
1	1	0
x^1	0	1
x^2	1	1
x^3	1	2
x^4	2	0
x^5	0	2
x^6	2	2
x^7	2	1

A.6.15 The minimal polynomial of α^i, i.e. $m_i(x)$, has as zeroes all the conjugates of α^i. Since α has order 63, one gets

$m_1(x)$ has zeroes $\alpha^1, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \alpha^{32}$;

$m_3(x)$ has zeroes $\alpha^3, \alpha^6, \alpha^{12}, \alpha^{24}, \alpha^{48}, \alpha^{96} = \alpha^{33}$;

$m_9(x)$ has zeroes $\alpha^9, \alpha^{18}, \alpha^{36}$.

A.6.16 The multiplicative order of 3 modulo 11 is 5, since $3^5 \equiv 1 \pmod{11}$ and no smaller positive power of 3 has this property. It follows that α, a primitive 11-th root of unity in an extension field of $GF(3)$, has the 5 conjugates: $\alpha, \alpha^3, \alpha^9, \alpha^{27} = \alpha^5, \alpha^{15} = \alpha^4$.

The minimal polynomial of α is given by

$$(x - \alpha)(x - \alpha^3)(x - \alpha^9)(x - \alpha^5)(x - \alpha^4).$$

and has degree 5.

A.6.17 By Corollary A.5.26, the field $GF(625) = GF(5^4)$ contains as subfields $GF(5^m)$ with $m|4$, so $GF(5)$ and $GF(5^2)$.

The (ground)field $GF(5)$ contains as elements the solutions of $x^5 = x$, so 0 and the fourth roots of unity ω^i, $0 \le i \le 3$, with $\omega = \alpha^{(5^4-1)/(5-1)} = \alpha^{156}$.

The subfield $GF(5^2)$ contains as elements the solutions of $x^{25} = x$, so 0 and the 24-th roots of unity β^i, $0 \le i \le 23$, with $\beta = \alpha^{(5^4-1)/(5^2-1)} = \alpha^{26}$.

A.6.18 Let α be a primitive 17-th root of unity. Its conjugates are

$$\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \alpha^{15}, \alpha^{13}, \alpha^9.$$

It follows that $m_1(x)$, the minimal polynomial of α, has degree 8.

The minimal polynomial $m_3(x)$ of α^3 has zeroes

$$\alpha^3, \alpha^6, \alpha^{12}, \alpha^7, \alpha^{14}, \alpha^{11}, \alpha^5, \alpha^{10}.$$

We conclude that the two different minimal polynomials of primitive 17-th roots of unity are $m_1(x)$ and $m_3(x)$. Of course

$$x^{17} - 1 = (x - 1)m_1(x)m_3(x).$$

From

$$x^8 m_1(x^{-1}) = x^8 \prod_{i=0}^{7}(x^{-1} - \alpha^{2^i}) = \prod_{i=0}^{7}(1 - \alpha^{2^i}x)$$

$$= \alpha^{1+2+4+8+16+15+13+9} \prod_{i=0}^{7}(\alpha^{-2^i} - x)$$

$$= \prod_{i=0}^{7}(x - \alpha^{-2^i}) = \prod_{i=0}^{7}(x - \alpha^{2^i}) = m_1(x).$$

It follows that $m_1(x)$ is its own reciprocal. The same holds for $m_3(x)$.

So, we can write $m_1(x)$ and $m_3(x)$ as follows:

$$m_1(x) = 1 + ax + bx^2 + cx^3 + dx^4 + cx^5 + bx^6 + ax^7 + x^8,$$

$$m_3(x) = 1 + sx + tx^2 + ux^3 + vx^4 + ux^5 + tx^6 + sx^7 + x^8.$$

Since $m_1(x)m_3(x) = (x^{17} - 1)/(x - 1) = 1 + x + x^2 + \ldots + x^{16}$, we get the following equations:

(x):	$a + s = 1,$
(x^2):	$t + as + b = 1,$
(x^3):	$u + at + bs + c = 1,$
(x^4):	$v + au + bt + cs + d = 1,$
(x^5):	$u + av + bu + ct + ds + c = 1,$
(x^6):	$t + au + bv + cu + dt + cs + b = 1,$
(x^7):	$s + at + bu + cv + du + ct + bs + a = 1,$
(x^8):	$dv = 1,$

From (x^8) we conclude that $d = v = 1$. Now, without loss of generality, (x) yields $a = 1$ and $s = 0$ (otherwise, interchange the role of $m_1(x)$ and $m_3(x)$, i.e. take α^3 as primitive 17-th root of unity).

Substituting these values in (x^2) yields $t + b = 1$ and thus $tb = 0$. Equation (x^4) now reduces to $u = 1$ and (x^3) to $t = c$. Next, (x^5) yields (since $tc = cc = c$) $b = 1$. Finally, (x^2) gives $t = 0$ and (x^3) gives $t = c$.

Summarizing, one has obtained

$$m_1(x) = 1 + x + x^2 + x^4 + x^6 + x^7 + x^8,$$

$$m_3(x) = 1 + x^3 + x^4 + x^5 + x^8.$$

A.6.19 We use the same numbering as in the problem.

1. It follows from Corollary A.5.13 and Corollary A.5.4 that

$$(Tr(x))^p = (x + x^p + x^{p^2} + \cdots + x^{p^{m-2}} + x^{p^{m-1}})^p =$$
$$= (x^p + x^{p^2} + x^{p^3} + \cdots + x^{p^{m-1}} + x) = Tr(x).$$

Theorem A.5.14 now implies that $Tr(x) \in GF(p)$ for all $x \in GF(p^m)$.

2. Let x and y be elements in $GF(q)$ and a and b in $GF(p)$. Then, by Corollary A.5.13 and Theorem A.5.14,

$$Tr(ax + by) =$$
$$= (ax + by) + (ax + by)^p + \cdots + (ax + by)^{p^{m-1}}$$

$$= (ax + by) + (a^p x^p + b^p y^p) + \cdots + (a^{p^{m-1}} x^{p^{m-1}} +$$

$$+ b^{p^{m-1}} y^{p^{m-1}})$$

$$= (ax + by) + (ax^p + by^p) + \cdots + (ax^{p^{m-1}} + by^{p^{m-1}})$$

$$= a(x + x^p + \cdots + x^{p^{m-1}}) + b(y + y^p + \cdots + y^{p^{m-1}})$$

$$= aTr(x) + bTr(y).$$

This proves the linearity of the trace mapping.

3. Since $Tr(x)$ is a linear mapping from $GF(p^m)$ to $GF(p)$, it either is the all-zero mapping or it takes on every value in $GF(p)$ equally often. However, $Tr(x)$ is a polynomial of degree p^{m-1}, so it has at most p^{m-1} zeroes by Theorem A.4.14. Consequently, $Tr(x)$ is not identical to zero and thus it takes on every value in $GF(p)$ equally often.

Bibliography

[1] Arazi, B., *A commonsense approach to the theory of error correcting codes,* MIT Press Series in Computer Systems, Cambridge MA, 1988.

[2] Beker, H.J. and F.C. Piper, *Cryptography and coding,* Clarendon Press, Oxford, Great. Britt., 1989.

[3] Berlekamp E.R., *Algebraic coding theory,* Revised 1984 edition, Aegean Park Press, 1984.

[4] Berlekamp E.R., *Key Papers in the Development of Coding Theory,* IEEE Press Selected Reprint Series, New York, 1974.

[5] Blahut, R.E., *Theory and practice of error control codes,* (reprint), Addison-Wesley Publishing Company, Reading MA, 1984.

[6] Blake, I.F., *Algebraic coding theory,* Benchmark Papers in Electrical Engineering and Computer Science, Hutchinson & Ross, Inc., Dowden, 1973.

[7] Blake, I.F. and R. Mullin, *The mathematical theory of coding,* Academic Press, New York, 1975.

[8] Cameron, P.J. and J.H. van Lint, *Designs, graphs, codes and their links,* London Mathematical Society Student Texts 22, Cambridge University Press, Cambridge, 1991.

[9] Chambers, W.G., *Basics of communications and coding,* Clarendon Press, Oxford, Great. Britt., 1985.

[10] Clark, Jr., G.C. and J.B. Cain, *Error-correction coding for digital communications,* Plenum Press, New York etc., 1981.

[11] Gallager, R.G., *Information theory and reliable communication,* Wiley, New York, 1968.

[12] Hamming, R.W., *Coding and information theory,* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1980.

[13] Hill, R., *A first course in coding theory,* Clarendon Press, Oxford, Great. Britt., 1986.

[14] Lin, S. and D.J. Costello, Jr., *Error control coding; fundamentals and applications,* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1983.

[15] Lint, J.H. van, *Coding theory,* Springer Lecture Notes, Vol. 201, Springer Verlag, Berlin etc., 1971.

[16] Lint, J.H. van, *Introduction to coding theory,* Graduate Texts in Mathematics, Vol. 86, Springer-Verlag, New York etc., 1982; second edition 1992.

[17] Lint, J.H. van, and G. van der Geer, *Introduction to coding theory and algebraic geometry,* Birkhäuser, Basel, 1988.

[18] MacWilliams, F.J. and N.J.A. Sloane, *The theory of error-correcting codes,* North Holland, Amsterdam etc., 1977.

[19] McEliece, R.J., *The theory of information and coding,* Encyclopedia of Math. and its Applications, Vol. 3, Addison-Wesley Publishing Company, Reading, Mass., 1977.

[20] Moreno, C.J., *Algebraic curves over finite fields,* Cambridge University Press, Cambridge etc., 1991.

[21] Peterson, W.W. and E.J. Weldon, *Error-correcting codes* (2nd ed.), MIT Press, Cambridge, Mass., 1972.

[22] Pless, V., *Introduction to the theory of error-correcting codes* (2nd ed.), John Wiley and Sons, New York etc., 1989.

[23] Pretzel, O., *Error-correcting codes and finite fields,* Clarendon Press, Oxford, Great. Britt., 1992.

[24] Rao, T.R.N. and E. Fujiwara, *Error-control coding for computer systems,* Prentice-Hall Series in Computer Engineering, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1989.

[25] Rhee, M.Y., *Error-correcting coding theory,* McGraw–Hill Publishing Company, New York, 1989.

[26] Sweeney, P., *Error control coding: an introduction,* Prentice Hall, New York, 1991.

[27] Thompson, T.M., *From error-correcting codes through sphere packings to simple groups,* Mathematical Association of America, Providence R.I., 1983.

[28] Tsfasman, M.A. and S.G. Vlăduţ, *Algebraic-geometric codes*, Kluwer Academic Publishers, Dordrecht, 1991.

[29] Vanstone, S.A. and P.C. van Oorschot, *An introduction to error correcting codes with applications,* Kluwer Academic Publishers, Boston MA, 1989.

[30] Welsh, D., *Codes and cryptography,* Clarendon Press, Oxford, Great. Britt., 1988.

Index