Coding Theory

G. Averkov

June 2, 2023

Contents

1 Introduction **4**
- 1.1 Coding theory in real life . . . 4
- 1.2 Sending data through discrete noisy channels . . . 5
- 1.3 Motivating examples . . . 5
  - 1.3.1 Encoding 2 to 3 bits for error detection . . . 5
  - 1.3.2 2 to 6 bits for error correction . . . 7
  - 1.3.3 2 to 5 bits for error correction . . . 7
  - 1.3.4 2 to 4 bits for error correction? . . . 7
- 1.4 Basic math notation and terminology . . . 7
- 1.5 Formal models of communication through noisy channels . . . 8

2 Bounds for block codes **9**
- 2.1 Parameters of block codes . . . 9
- 2.2 Hamming, Gilbert-Varshamov and singleton bounds . . . 11

3 Decoding for stochastic channels **14**
- 3.1 Probability over finite sample spaces in a nutshell . . . 14
- 3.2 Stochastic channels . . . 16
- 3.3 Maximal-likelihood decoding . . . 18
- 3.4 Shannon's theorem on the capacity of noisy channels . . . 19

4 Modular arithmetic: foundations and applications **27**
- 4.1 Abstract rings and fields in a nutshell . . . 27
- 4.2 Ideals and quotient rings . . . 28
- 4.3 Ring structures behind the modular arithmetic . . . 29
- 4.4 Examples of codes relying on modular arithmetic . . . 30
  - 4.4.1 Some codes over $\mathbb{Z}_2$ . . . 30
  - 4.4.2 Parity-check codes . . . 30
  - 4.4.3 ISIN and ISBN codes . . . 31
  - 4.4.4 The Hamming code of length 7 . . . 31

5 Linear codes **31**
- 5.1 Linear algebra . . . 32
  - 5.1.1 Vector spaces . . . 32
  - 5.1.2 Linear maps and matrices . . . 32
  - 5.1.3 Gaussian elimination . . . 33
- 5.2 Parameters of linear codes . . . 33
- 5.3 Syndrome decoding . . . 34
  - 5.3.1 Two versions of the minimum-distance decoding . . . 34
  - 5.3.2 Initialization of the dictionary . . . 34
  - 5.3.3 Decoding using the dictionary . . . 35
  - 5.3.4 Computational merits of the syndrome decoding . . . 35
- 5.4 Hamming codes . . . 36
- 5.5 Evaluation codes . . . 37
  - 5.5.1 Univariate polynomial interpolation . . . 37

5.5.2 Reed-Solomon codes 38
5.5.3 Multivariate polynomial interpolation on a hypercube 39
5.5.4 Reed-Muller code 40
5.6 General finite fields 40
5.6.1 Rings of univariate polynomials 40
5.6.2 Finite fields from irreducible polynomials 41
5.6.3 Fundamental theorem of finite fields 42
5.6.4 Finite fields in applications 42
5.7 LDPC codes 43
5.8 Cyclic codes 45
5.8.1 Cyclic codes and ideals 45

1 Introduction

1.1 Coding theory in real life

I suppose, in this audience we all are heavy users of coding theory. Indeed, almost no communication channel, whether wired or wireless is 100% reliable as well as no data storage, whether old fashioned or very fancy and modern. Thus, while transmitting the information to communicate or reading or writing the information from and to data storage, error may happen. Communication takes place at a micro and macro level, within a computer chip, when data are sent around through the world using cables and satellites that orbit around the Earth. Our mobile devices exchange data with cell towers when we chat with friends and the family over our favorite messenger. Technologies evolve. We used to have 2G and 3G mobile networks. 4G and 5G mobile networks are regularly available nowadays. All of them, the older and the newer ones, rely on error correction. When we go to a grocery to buy something for dinner, the bar codes of the items we bought get scanned, and this is also a communication process, where error may happen and have to be detected. When we scan a QR code on a poster, we use error correction as well. When we do all this, we use coding theory, which means we use exciting algorithmic and mathematical theory. Since 1950s multiple smart ways of error detection and correction have been established to cope with communication over a noisy channel. To address a particular example, do you know what kind of math is hidden behind QR codes?

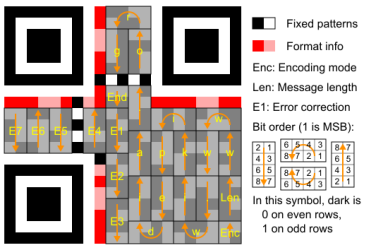


Image from Wikipedia, see [6].

The example shown above is a 19-dimensional linear subspace in the ambient space of dimension 26 over the finite field of size 256. More precisely, it is a Reed-Solomon code that is designed using evaluation of a polynomial at finitely many values of a given finite field and as such it is a maximum-distance separable code (which means 'very nice' but in a technical language of coding theory). You probably see that, in order to know the technology behind QR codes and other types of codes used in different contexts, you need to understand math. We need to understand vector spaces, to study properties of polynomials, to know about finite fields. Knowledge of linear and nonlinear algebra is really essential, but it is also desirable to know other types of math. The aim of this course is exactly this. Explain you the math relevant for designing coding schemes along with the explanation of coding schemes that are used in modern practical applications. My intention is to explain most of the math almost from scratch, so that even in the case that your math background is rather limited, you succeed in understanding the explanations. As a rule of thumb: the more math you know, the more comfortable you'll feel with topics in coding theory. If you are missing some math background and feel that the explanation in the lectures was too short, give you feedback and we'll see how we could help you.

1.2 Sending data through discrete noisy channels

Algebraic coding theory studies efficient ways of establishing coding schemes. These are procedures that fit into the following template. A message is encoded and then sent as a word (i.e. a sequence of symbols over an alphabet) through a noisy channel. Being exposed to noise, the word sent through the channel might get distorted and changed. That is, the word which arrives to the receiver might contain errors, meaning that some symbols of the arrived word are not the same as the symbol of the word that was sent. For example, it could go something like that:

funny $\rightsquigarrow$ bunny	1 error
funny $\rightsquigarrow$ fancy	2 errors

The objective is to find ways of encoding messages so that a desired level of error detection and/or correction is possible with a possible small overhead. Under the overhead one means the amount information that needs to be transmitted to the channel and the computational efforts required for encoding and decoding.

Algebraic coding theory relies on probability theory, discrete mathematics (including combinatorics), algebra (including linear algebra), theory of algorithms, game theory, complexity theory and optimization as fundamental tools. Codes are also used in public key cryptography, where the complexity theoretic aspects play a crucial role.

The main questions that are answered by coding theory are: what good codes? what are good coding schemes? Of course, the questions need a formalization, and we will establish the formalism shortly.

The milestones we achieve in this course are:

- *Bounds on codes.* There is a trade off between the number of errors we can detect and the amount of information we have to sent. Bound on codes describe the possibilities and the limitations between these quantities.

- *Shannon's theorem on the capacity of noisy channels.* The theorem answers the question, whether we can pick a coding scheme that would fit perfectly to the amount of noise we have in the channel. The theorem answers the question whether good codes exist. The answer is positive: good codes indeed exist them, but the proof of the theorem does not tell us how to construct good codes.

- *Families of codes that are used in practice.* This milestone is not quite one milestone but rather a cookbook of different approaches to constructing codes that work well in practice. This part could be viewed as the main part of the course, and it requires algebra as a background.

1.3 Motivating examples

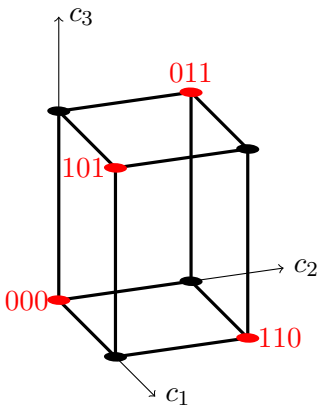
1.3.1 Encoding 2 to 3 bits for error detection

Consider, as an introductory example, the situation of sending two bits of information. That is, the set of possible messages we are going to transmit is $\{00, 01, 10, 11\}$. Frequently, one

would need to transmit much more than just two bits, but then we could group them into two-bit blocks and get back to the problem of transmitting two-bit chunks.

If we send just the two bits, and some of them gets modified by noise, we won't be able to detect that the error occurred. A rather straightforward idea to cope with this problem is to use redundant information. So, let's us supply the sequence of two bits $c_1c_2 \in \{0,1\}^2$, which we want to send, with another bit c_3 which is set to 1 whenever $c_1 = c_2$. This way we are sending one of the four strings from the set $C := \{000, 011, 101, 110\}$ being a subset of the set $\{0,1\}^3$ of three bit strings. A set of code words used for transmission is called a code. We are using a code C with strings of length 3 of size $|C| = 4$.

If the two bit string c_1c_2 is to be transmitted, it can first encoded as $c_1c_2c_3 \in C$ and then sent through the channel. If say, we start with 01 and then encode it as $011 \in C$, but then receive 111, we know that the error has occurred since the last bit is equal to 1, which is the indication for the fact that the first two bits should be equal, but they are not equal. So, we have been able to detect the error, which is merely the fact that $111 \notin C$. In fact, it will be able to detect any error, no matter what code word is sent and no matter what bit was flipped by the noise. Indeed, if exactly one bit of the code word is changed, we get a string which does not belong to the code anymore. If you like geometry, you can also visualize C as the set of four vertices of the three-dimensional cube $[0,1]^3$ with the property that none of these vertices is connected by an edge of the cube.



This code will also be able to detect three errors. If 011 is changed to 100, we also see that the received message is not in the code. But the code will not be able to detect two errors. If say 011 is transformed to 110 by noise, we are in trouble. Coding theory does not give a 100% guarantee that an error can be detected/corrected, but it provides ways to make sure that error detection/correction is likely to succeed.

Our example code C allows to detect one error. If each of the three bits of the code can be damaged independently with a small probability $0 < p < 1$, we probably do not have to worry about having more than one error, because that would be quite unlikely when p is small.

Exercise 1.1. Use your knowledge in probability theory to calculate the probability of having $k = 0, \ldots, n$ errors in an n-bit string, when each bit is changed independently of any other bit with the probability p. What probabilities for k errors do you have for $p = 1/10 = 10\%$ and $n = 3$?

We should also pay attention to the price we have to pay for the ability to detect an error. If say our channel has the speed of 100 Megabits per second (Mbps), then with this encoding scheme, we transmit 2 bits via 3, which means that the speed we will achieve is $100 \cdot \frac{2}{3}$ Mbps. That's a slowdown that we have to pay for detection of errors.

Coding theory is devoted to studying ways of getting the maximum possible performance within the given limitations.

1.3.2 2 to 6 bits for error correction

What if we want not only to be able to detect an error when transmitting two bits, but also to recover the original message? A quite trivial idea that can be employed is repeating the two-bit string several times. Say, we can encode c_1c_2 and $c_1c_2c_3c_4c_5c_6$ with $c_1c_2 = c_3c_4 = c_5c_6$. It's quite an expensive encoding, since 2 bits get transformed to 6, but we can now not only detect error in one bit, but also correct it. When one bit is changed, then one of the two sub-strings among c_1c_2, c_2c_3 and c_4c_5 is changed, but the other two remain the same. So, we know that the error has occurred and, under the assumption that we know it was exactly one error, we also know that the two substrings that are equal to each other comprise the original message.

1.3.3 2 to 5 bits for error correction

Now we may start to wonder if paying the price of 6 bits for sending two was too high. Maybe we could send over a smaller number of bits and still be able to correct one error. What we can do is a kind of mixture of the two coding schemes employed above. We do one repetition and add one "check bit". That is, $c_1c_2 \in \{0,1\}^2$ is transformed to $c_1c_2c_3c_4c_5$, where $c_1c_2 = c_3c_4$ and c_5 is equal to 1 whenever $c_1 = c_2$. Again, assume that one of the bits was changed by the noise. It is either a bit in c_1c_2 or a bit in c_3c_4 or the check bit c_5. If the sub-strings c_1c_2 and c_3c_4 have not been changed they remain the same, and we know that c_5 was changed, by seeing that the check bit is set improperly as compared to $c_1c_2 = c_3c_4$. Thus, the error is detected and we know that c_1c_2 is the two-bit string we encoded. If one of the two sub-strings c_1c_2 or c_3c_4 was changed, then we know this by the fact that after a change the two sub-strings are not equal anymore. Which one has remained undisturbed, can be figured out by looking at the check bit c_5. Only one of the two sub-strings received will be compatible with the way c_5 is set. We thus see that we can detect one error and recover the original message. The price we pay is naturally higher than the price we paid for detection of an error without any correction.

1.3.4 2 to 4 bits for error correction?

An important questions about the inherent limitations arises by looking at the toy problem of sending two bits, we consider. Can we somehow encode a two bit string via four bits and still be able to correct one error? It turns out that we cannot. We will see why later.

1.4 Basic math notation and terminology

We need some basic math notation and terminology, before we dive into coding theory. The cardinality or the size of a set K is the number of elements in K, denoted by $|K|$. The

cardinality can be zero, positive integer or infinity. We use the floor and the ceiling notation for rounding down and rounding up, respectively. We denote inclusion as $\subseteq$ and the cartesian product operation on sets by $\times$. We also use standard notions like map, injective map, surjective map, bijective map, binary operation, relations etc. If you happen not to know any of these notions, please ask and I will clarify the meaning.

We use the notation $\mathbb{N} := \{1, 2, 3, \ldots\}$ and call the elements of $\mathbb{N}$ natural numbers. Furthermore, we use $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$ to denote the set of non-negative integers. For a set K, we use K^n with $n \in \mathbb{N}_0$ to denote the set of all n-tuples of elements in K. When K is a finite non-empty set viewed as an alphabet, then it is common to call elements of K symbols of the alphabet and the elements of K^n strings or words of length n over the alphabet K. When elements of K^n are interpreted this way, we often write $s = s_1 \cdots s_n$ rather than $s = (s_1, \ldots, s_n)$ to represent an element $s \in K^n$. For example, we could write 01 rather than $(0, 1)$, to save space and time while writing.

We use $x' = \operatorname{argmin}_{x \in X} f(x)$ to denote that x' is a minimizer of the optimization problem $\min_{x \in X} f(x)$ that, is $x' \in X$ and $f(x') = \min_{x \in X} f(x)$. As there is not always a unique minimizer, our way of writing abuses notation in the same way as the common abuse of notation when describing the asymptotic relations between functions in terms of the O and Ω asymptotics.

1.5 Formal models of communication through noisy channels

Let's describe a general template for error correction that could be specialized in many different ways.

Let M be a finite set of messages to be sent, and W some finite space of words that can be transmitted over a channel. Models of for noisy communication can be described by this diagram.

$$M \xrightarrow[\text{encoding}]{f} W \xrightarrow{\text{noise}} W \xrightarrow[\text{decoding}]{g} M$$

Def 1.2. We define the encoder-decoder coding scheme over a finite set W is defined as the pair (f, g) of maps $f : M \to W$ and $g : M \to W$, where is called a set of messages, W is called a set of words and $c = f(a)$ is called the code of the message $a \in M$. The map f is called an encoder or an encoding scheme and g is called a decoder or a decoding scheme.

There are two basic models for noise. One involves randomness: then $y(c)$ is a random variable, so it depends on $c = f(a)$ and chance. Another is an adversarial model in which the noise is the adversary that can change c to $y(c)$ within prescribed rules. In the probabilistic model, the objective is to provide the encoding scheme, in which the probability of the correct decoding, which is the probability of $g(y(f(a)) = a$, is maximized. The adversarial model is a game of the noise against sender and receiver with the condition $f(y(g(a)) = a$ indicating that the sender and the receiver have won the game against the noise. One also considers the model, in which the message is not deterministic but a random variable distributed in M.

Rem 1.3. If $f : M \to W$ is injective, which usually a reasonable choice, one can identify M with its image $f(M) \subseteq W$. Such an image $C = f(M)$ is called a code and the elements of C are called codewords. This motivates the following definition.

Def 1.4. We define a code-decoder coding scheme over a finite set W as a pair (C, g), where $C \subseteq W$ and $g : W \to C$. The set C is called a code and g is called a decoder or a decoding scheme for the code C.

Furthermore, if we focus on error detection, we can make use of the following definition.

Def 1.5. We define a code-error-detector coding scheme over W as a pair (C, h) where $C \subseteq W$ is called a code, and $h : W \to \{\text{false}, \text{true}\}$ is called the error detector.

If $y(c)$ represents the word received after sending $c \in C$ through a communication channel, the objective of the error detector is to determine if $y(c) = c$ holds.

2 Bounds for block codes

In this section, we use the binomial coefficents $\binom{n}{j}$, n choose j, and their combinatorial meaning, i.e, the number of j-element subsets of an n-element sets. More informally, $\binom{n}{j}$ is the number of ways to choose j sorts of cakes out of n available sorts.

2.1 Parameters of block codes

Def 2.1. For a finite set K with $q = |K| > 0$, a non-empty subset C of K^n is called a block code, or merely a code, over the alphabet K. Elements of C are called code words, n is called the length of the code C. The code C is called binary resp. ternary if $q = 2$ resp. $q = 3$.

We have seen three examples of block codes above.

Def 2.2. A function $d : X \times X \to \mathbb{R}$ is called a metric if it satisfies the conditions

(a) $d(u, v) \geq 0$ with $d(u, v) = 0$ if, and only if, $u = v$.

(b) $d(u, v) = d(v, u)$, (symmetry)

(c) $d(u, v) \leq d(u, t) + d(t, v)$, (triangle inequality)

for all $u, v, t \in X$. For $\rho \in \mathbb{R}_{\geq 0}$ and $c \in X$, the set

$$B(c, \rho) = \{x \in X \,:\, d(c, x) \leq \rho\}$$

is called the (closed) ball of radius ρ with the center at c.

Prop 2.3. *If $d : X \times X \to \mathbb{R}$ is a metric and $\rho > 0$, then $B(a, \rho) \cap B(b, \rho) = \emptyset$ when $a, b \in X$ satisfy $d(a, b) > 2\rho$.*

Proof. Exercise. □

In what follows, we assume $n \in \mathbb{N}$ and K to be non-empty and finite.

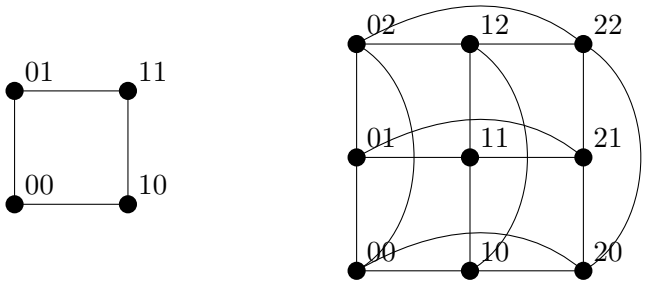
Def 2.4. The hamming distance is defined as the function $d : K^n \times K^n \to \mathbb{R}$, given by

$$d(u, v) = |\,\{i = 1, \ldots, n \,:\, u_i \neq v_i\}\,|,$$

which is the number of positions, in which strings $u = u_1 \cdots u_n$ and $v = v_1 \cdots v_n$ differ. The value $d(u, v)/n$ representing the quotient of components, in which u and v differ, is called the relative hamming distance of u and v. The value $\text{wt}(u) := d(u, 0)$ is called the weight of u.

Rem 2.5. With some knowledge of graph theory, you will be able to interpret the hamming distance as the shortest-path distance on a graph. For $q = |K|$ the hamming distance is the shortest-path distance of the n-fold Cartesian power of the complete graph on q nodes. This graph has K^n as the vertex set and two words $u, v \in K^n$ are connected by an edge if and only if their differ in exactly one position. When $q = 2$, this graph is called the n-dimensional hypercube graph.

Here are examples of the graphs corresponding to the hamming distances on K^2 with $K = \{0, 1\}$ and $K = \{0, 1, 2\}$, respectively.



Prop 2.6. *The hamming distance is a metric.*

Proof. The proof is left as an exercise. The only condition that requires somewhat more than a trivial justification is the triangle inequality. □

Rem 2.7. Our space of words K^n is endowed with a metric, the metric metric, and it also has the natural measure, which is just the counting measure, the cardinality $|S|$ of a set $S \subseteq K^n$ can be viewed as the discrete counterpart of the volume. We will be interested in the volumes of balls in K^n.

Prop 2.8. *Let d be the hamming distance on K^n, and let $\rho = 0, \ldots, n$. The respective ball $B(c, \rho)$ has the cardinality*

$$\big|B(c, \rho)\big| = V_q^n(\rho) := \sum_{j=0}^{\rho} \binom{n}{j} (q-1)^j$$

for every $c \in K^n$ and for $q := |K|$.

Proof. Exercise in combinatorics. □

When a code $C \subseteq K^n$ is considered, then for each code word $c \in C$, the ball $B(c, \rho)$ represents the set of all messages that can be received, when c is sent, in the case of at most ρ errors.

Def 2.9. If a code $c \in C$ sent through a communication channel is transformed to $y \in K^n$, then $d(c, y)$ is called the number of errors in y.

Def 2.10. For $t, c \in \mathbb{N}$, a code $C \subseteq K^n$ is called:

(a) t-error detecting if $d(c, c') > t$ holds for all $c, c' \in C$ with $c \neq c'$.

(b) e-error correcting if $B_e(c) \cap B_e(c') = \emptyset$ holds for all $c, c' \in C$ with $c \neq c'$.

Rem 2.11. A t-error detecting code can indeed detect up to t errors. If for a code $c \in C$ sent the word y received has at least one and up to t errors, then y is not in C, since $d(c, c') > t$ for each $c' \in C$ with $c' \neq c$, while for y one has $1 \leq d(c, y) \leq t$. Thus, if the word y received is in C, it either coincides with the message c sent or there have been more than t errors.

Rem 2.12. An e-error correcting code can indeed be supplied with a decoder that corrects up to e errors. Let $g : K^n \to C$ be a decoder that associates to each $y \in K^n$ a code $c \in C$ that minimizes the hamming distance $d(y, c)$. If, for a code $c \in C$ sent through the channel, the received word y has at most e errors, then $y = g(c)$, since $d(c, y) \leq e$ and for every $c' \in C \setminus \{c\}$, one has $d(y, c') > e$, as otherwise y would be in $B_e(c) \cap B_e(c')$ contradicting the definition of the e-error correcting code.

Def 2.13. For a code $C \subseteq K^n$, the value

$$d(C) = \min \left\{ d(c, c') \, : \, c, c' \in C, \; c \neq c' \right\}$$

is called the minimum distance of the code C. Furthermore, C is said to be a (n, M, d) code if $M = |C|$ and $d = d(C)$. That is, the triple (n, M, d) represents the length, the size and the minimum distance of the code C. The quotient d/n is called the relative minimum distance of C.

The following proposition shows that the minimum distance allows one to assess how many errors can be detected or corrected.

Prop 2.14. *A code $C \subseteq K^n$ is e-error correcting when $e \in \mathbb{N}_0$ satisfies $2e < d(C)$ and t-error detecting when $t \in \mathbb{N}_0$ satisfies $t < d(C)$. In particular, C is e-error correcting with $e = \left\lfloor \frac{d(C)-1}{2} \right\rfloor$ and t-error detecting with $t = d(C) - 1$.*

We know that C is error correcting for $2e+1 \leq d(C)$, $e \in \mathbb{N}_0$. It is interesting to study the interplay between $d(C)$, $|C|$ and n: the ability to correct errors, the amount of information that C can transmit and the amount of information that will have to be sent through the channel. We have calculated $|B_\rho(c)|$ for the hamming distance, which is the number, independent of $c \in K^n$.

Rem 2.15. The code $C = \{(k, \ldots, k) \in K^n \, : \, k \in K\}$ over a finite alphabet K is called a repetition code of length n. Its minimum distance is $d(C) = n$ and so it can correct $\left\lfloor \frac{n-1}{2} \right\rfloor$ errors, however, at a high price of sending n symbols instead of one to ensure error-correction.

2.2 Hamming, Gilbert-Varshamov and singleton bounds

How many code words can we transmit if we want to correct e errors and use a q-ary words of length n. It's quite difficult to figure out exactly. But at least we can provide bounds. Here is an upper bound.

Thm 2.16 (Hamming bound)**.** *For a code $\neq C \subseteq K^n$ over a q-ary alphabet K and every $e \in \mathbb{N}_0$ satisfying $2e + 1 \leq d(C)$, one has*

$$|C| \leq \frac{q^n}{V_q^n(e)},$$

Furthermore, if the latter inequality is attained with equality, then every word in K^n belongs to exactly one hamming ball of radius e with the center in some $c \in C$ or, in other words, K^n is the disjoint union of all $B(c, e)$ with $c \in C$.

Def 2.17. Codes C satisfying the equality $q^n = |C| \cdot V_q^n(e)$ in the previous theorem are called *perfect*.

Rem 2.18. For a perfect code, the space K^n of all words is decomposed into the disjoint union of balls of radius e with the centers in the points of the code. Thus, the decoding can be organized nicely as the map sending a word to the center of the ball it is contained in. However, it turns out that perfect codes are quite rare. Nevertheless, one can analyze how close to being perfect one can get. This is done in the following theorem.

Thm 2.19 (Gilbert-Varshamov bound = GV bound)**.** *For given $n, q, d \in \mathbb{N}$ with $1 \leq d \leq n$ and $q \geq 2$ and an alphabet K with $q = |K|$, there is a q-ary code $\emptyset \neq C \subseteq K^n$ with the minimum distance d that satisfies*

$$|C| \geq \frac{q^n}{V_q^n(d-1)}$$

Proof. Consider the q-ary code $C \subseteq K^n$ with the distance d and the maximal $|C|$. Then every $u \in K^n$ is a ball of radious $d-1$ with the center in a point of C. Indeed, if u did not satisfy this property it could be added to C without changing the distance and increasing the cardinality of C. Consequently, K^n is equal to the union of all balls with centers in C of radus $d-1$. It follows that

$$q^n \leq |C| V(n, q, d-1),$$

giving the desired assertion. □

Rem 2.20. A code satisfying the above bound can be constructed using a greedy strategy: add a pair of words at distance d to each other to C, then iterating by adding a word in $K^n \setminus \bigcup_{c \in C} B(c, d-1)$ to C, until the balls $B(c, d-1)$ with $c \in C$ cover the whole K^n.

Rem 2.21. The Hamming and the GV bound show that a q-code maximizing $|C|$ for given $d = d(C)$ and block length n has the size $|C|$ in the range determined by the following bounds:

$$\frac{q^n}{V_q^n(d-1)} \leq |C| \leq \frac{q^n}{V_q^n(\lfloor \frac{d-1}{2} \rfloor)}.$$

The size of a Hamming ball grows with the radius, and the growth might be rather quick. The discrepancy between the bounds is quantified by the discrepancy of the two radii $d-1$ and $\lfloor \frac{d-1}{2} \rfloor$.

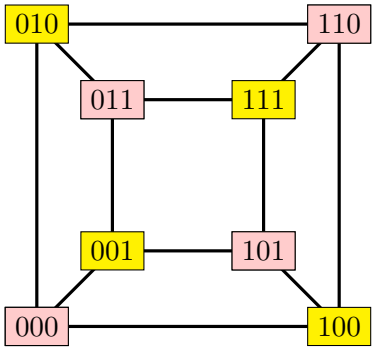
Thm 2.22 (Singleton bound)**.** *Let $C \subseteq K^n$ be a q-ary code with $d = d(C)$ and $q \geq 2$. Then $|C| \leq q^{n-d+1}$, or equivalently*

$$d \leq n - \log_q |C| + 1.$$

Proof. Consider the map $T : C \to K^{n-d+1}$ given by $T(c_1 \cdots c_n) = c_d \cdots c_n$. This map is injective, since any two distinct words differ in at least d components, but only $d-1$ components are dropped by T. Hence $q^{n-d+1} = |K^{n-d+1}| \geq |C|$. □

Def 2.23. Codes satisfying $d = n - \log_d |C| + 1$ are called maximum-distance separable or MDS codes.

Example 2.24. The code $C = \{(c_1, c_2, c_1 \oplus c_2) : c_1, c_2 \in \{0,1\}\} \subseteq \{0,1\}^3$ is an example of an MDS code; in this picture the code words are highlighted in pink:



It's worth noticing that the MDS codes are good examples of codes that can deal with unreadable symbols. Like with this code: assume that you receive 1?1, where the equation mark indicates that that symbol is not readable. Assuming the readable symbols are correct, you are able to determine the unreadable part: it is the 0. An MDR code of distance d is able to recover up to $d-1$ unreadable symbols. In QR codes (which are special MDS codes) such a facility is used among other situations, when a logo is placed into a QR code. Ability to recover unreadable parts is also essential for the distributed data storage. Imagine having two drives with a very valuable information that you don't want to lose. For the case that the drives will be damaged you might get an extra backup drive for each of them to copy the data. A less wasteful approach is to use the above MDR code, which would allow you to reduce the number of backup drives from two to one. If x_i is the i-th bit on the first drive and y_i the i-th bit on the second drive, just store $z_i := x_i \oplus y_i$ as the i-th bit on the backup drive. Then $x_i \oplus y_i \oplus z_i = 0$. If any other three drives fails, you would be able to recover its contents. Say, if the first drive fails, you know that $x_i = y_i \oplus z_i$ and you get the contents back. If the second one fails, you know that $y_i = x_i \oplus z_i$. If, however, two of the three fail, you are in trouble. In distributed storage research, one tries to come up with codes that are able to restore sufficiently much unreadable parts but also would allow an efficient recovery procedure.

Exercise 2.25. Let's return to the question whether one error can be corrected by transmitting two bits of information via code of length four. Show that no code $C \subseteq \{0,1\}^4$ with $|C| = 4$ can correct 1 error.

Exercise 2.26. Show that the minimum distance of every perfect code C with at least two elements is an odd number.

Exercise 2.27. Construct a binary MDS codes with the parameters $(4, 2^3, 2)$.

Exercise 2.28. Show that every perfect binary code of minimum distance 7 has block length $n \in \{7, 23\}$.

Exercise 2.29. Develop an enumeration program, in the computer language you prefer, to find a binary code, whose size is as large as you are able to find, of length 6 and minimum distance 3. That's a computer enumeration challenge.

3 Decoding for stochastic channels

Error correcting is usually done under an assumption that the noise of the channel is random and is distributed in a certain specific way and the frequency of the use of the code words of a given code is also described via a (probabilistic) distribution. Thus, we need a certain amount of the probability theory as a background, but we stay in the finite setting which allows us to avoid the measure-theoretic subtleties of the probability theory. That is, our probability space is finite. We start by recalling the basics of the probability theory. If you are math students, you've had all that in a probability course. If you are CS student at the BTU, you've got no probability theory before. If you are an AI student, you might have taken a probability class before, but it's instructive seeing how this theory works in this new context.

3.1 Probability over finite sample spaces in a nutshell

Def 3.1. A finite probability space is a pair (Ω, P), where Ω is a finite non-empty set, called the sample space and $P : \Omega \to \mathbb{R}$ is a function that satisfies $P(\omega) \geq 0$ and $\sum_{\omega\in\Omega} P(\omega) = 1$. Elements of Ω are called elementary events, and $A \subseteq \Omega$ is called a random event. The value $P(A) = \sum_{\omega\in A} P(\omega)$ is called a probability of the event A. Note that, usually, the notation $P(A)$ is used to denote the image of A under the map P, but in the case of P, we deviate from this convention and use $P(A)$ in the above sense. Essentially, Ω represents all possible manifestations of chance (involved in our considerations). Events are frequently described by a sentence/condition whose validity depends on the chance. Set-theoretically the condition corresponds to the set of all $\omega \in \Omega$, for which the condition is true.

Rem 3.2. One can view (Ω, P) as a kind of device that one can invoke and it will produce an outcome ω. That would be described, in words, as carrying out the random experiment and obtaining a random outcome ω, whereas the value $P(\omega)$ attached to ω is the probability of obtaining ω. The elementary event ω is the most detailed record of the outcome of the randomness we are dealing with, when we carry out our random experiment. In probability one frequently likes not to specify' Ω explicitly and to hide the respective $\omega \in \Omega$. The reasons for this is that one just assumes that there is a kind of primary source of randomness, from which all other random events and variables are derived, but one might not have a direct access to that primary source but only to some of its derivations that do not keep the track of the whole ω. As a small visual example, imagine a person going through a maze, deciding whether to go left or right each minute by tossing a coin. Imagine, you observe the person entering and then leaving the maze after a certain random number of minutes. Then you've observed the random number of minutes taken to go through the maze, but you did not see the outcomes of the coin tosses that determine the time take to go through the maze.

Example 3.3. A toss of a fair coin is captured by $\Omega = \{0, 1\}$, $P(0) = P(1) = \frac{1}{2}$. That is, when a fair coin is tossed, we observe either heads, represented as zero, or a tail, represented as 1, both having equal probability of 1/2. A sequenc eof n independent tosses of a fair coin can be modeled with $\Omega = \{0, 1\}^n$, and $P(\omega) = \frac{1}{2^n}$ for every $\omega \in \Omega$. The event that all n tosses had the same result is the two-element subset $A = \{0 \cdots 0, 1 \cdots 1\}$ of this Ω, its probability is $P(A) = \frac{1}{2^{n-1}}$.

Example 3.4. A biased coin with the tail probability equal to $p \in (0, 1)$ can be modeled by $\Omega = \{0, 1\}$ with $P(1) = p$ and $P(0) = 1 - p$. Again, also here, we can described n tosses of a

biased coin with the tail probability equal to p via $\Omega = \{0,1\}^n$ and

$$P(\underbrace{\omega_1 \cdots \omega_n}_{\omega}) = p^{\mathrm{wt}(\omega)}(1-p)^{n-\mathrm{wt}(\omega)}.$$

In what follows, when we deal with probabilities we assume that an underlying (discrete) probability space is given.

Def 3.5. Events A, B are called independent if $P(AB) = P(A)P(B)$, where AB stands for $A \cap B$. More generally, a system of finitely many events $A_1, \ldots, A_m$ is said to be (mutually) independent if $P(A_{i_1} \cdots A_{i_l}) = P(A_{i_1}) \cdots P(A_{i_l})$ holds for any choice of $1 \leq i_1 < \cdots < i_l \leq m$.

Def 3.6. We call a function $X : \Omega \to S$ a random variable in S or a random value in S, and we write $X \in_R S$. The function $s \in S \mapsto P(X = s)$ is called the distribution of X. The random variable is said to be uniformly distributed in S if $P(X = s) = \frac{1}{|S|}$ for every $s \in S$. Depending on how S is chosen, one can consider random real or complex-valued variables, random words, random graphs, random functions etc.

Def 3.7. When $P(B) > 0$, the conditional probability of A, given B, is defined by the formula

$$P(A|B) = \frac{P(AB)}{P(B)}.$$

That the probability of A occurring under the assumption B occur.

Rem 3.8. If A and B are independent and $P(B) > 0$, then $P(A|B) = P(A)$. Informally: A "does not care" if B occurred or not.

Rem 3.9. It is easy to see that for $P(A), P(B) > 0$, one can link $P(A|B)$ and $P(B|A)$ via the formula

$$\mathsf{P}\,(A \,|\, B) = \frac{P(B|A)}{P(B)} P(A),$$

known as the Bayes Formula.

Prop 3.10. *If $X \in_R S$ and S is finite and A is an event, then*

$$\mathsf{P}(A) = \sum_{s \in S}^{m} \mathsf{P}\,(A \,|\, X = s)\, P(X = s),$$

where the terms with $P(A_i) = 0$ should be interpreted as zero.

Def 3.11. A system of random variables $X_1, \ldots, X_m$ is said to be independent if for any choice of sets S_i with $i = 1, \ldots, m$, one has

$$P(X_1 \in S_1, \ldots, X_m \in S_m) = \prod_{i=1}^{m} P(X_i \in S_i).$$

Def 3.12. For $X : \Omega \to \mathbb{R}$, the values

$$E(X) = \sum_{\omega \in \Omega} X(\omega) P(\omega)$$

and

$$V(X) = E((X - E(X))^2)$$

are called the expectation and the variance of X, respectively.

Prop 3.13. *For a finite $S \subseteq \mathbb{R}$ and $X : \Omega \to S$, one has $E(X) = \sum_{s \in S} s\, P(X = s)$.*

Thm 3.14 (Markov's inequality)**.** *For $X : \Omega \to \mathbb{R}_{\geq 0}$ and $a > E(X)$, one has*

$$P(X \geq a) \leq \frac{E(X)}{a}.$$

Proof.

$$aP(X \geq a) = a \sum_{\omega \in \Omega : X(\omega) \geq a} P(\omega) \leq \sum_{\omega \in \Omega : X(\omega) \geq a} X(\omega) P(\omega) \leq \sum_{\omega \in \Omega} X(\omega) P(\omega) = E(X).$$

□

Rem 3.15. Markov's inequality estimates the probability of a general random variable to be over its expected value. That's the probability of being over the average.

Thm 3.16 (Chebyshev's inequality)**.** *For $X : \Omega \to \mathbb{R}$ and $b > 0$, one has*

$$P(|X - E(X)| \geq b) \leq \frac{b^2}{V(X)}.$$

Proof. Apply Markov's inequality to estimate the probability of $(X - E(X))^2 \geq b^2$. □

Thm 3.17 (Union bound)**.** *For $A_1, \ldots, A_m \subseteq \Omega$,*

$$\mathsf{P}\left(\bigcup_{i=1}^{m} A_i \right) \leq \sum_{i=1}^{m} \mathsf{P}(A_i)$$

Proof.

$$\mathsf{P}\left(\bigcup_{i=1}^{m} A_i \right) = \sum_{\omega \in A_1 \cup \cdots \cup A_m} P(\omega) \leq \sum_{i=1}^{m} \sum_{\omega \in A_i} P(\omega) = \sum_{i=1}^{m} \mathsf{P}(A_i),$$

where $\leq$ holds, since every term $P(\omega)$ on the left-hand side of $\leq$ occurs at least once on the right-hand side. □

Thm 3.18. *If $X_1, \ldots, X_m \in_R \mathbb{R}$ and independent, then $E(X_1 \cdots X_m) = \prod_{i=1}^{m} E(X_i)$ and $V(X_1 + \cdots + X_m) = \sum_{i=1}^{m} V(X_i)$.*

3.2 Stochastic channels

Def 3.19. We define a random function $F : V \xrightarrow{R} W$ as a function with values in a set W that depends on the input $v \in V$ and the sample point ω, that is, one has $F : V \times \Omega \to W$. It is customary to not indicate the dependence on ω explicitly and write $F(v)$ rather than $F(v, \omega)$. Another way to interpret $F : V \xrightarrow{R} W$ is as a system of random variables $F(v) \in_R W$ indexed by $v \in V$.

Def 3.20. A stochastic channel on a finite word space W is a random function $\mathrm{CH} : W \xrightarrow{R} W$. For $x, y \in W$, we call the probability of $\mathrm{CH}(x) = y$ the probability of the transition from x to y, when passing x from the channel CH.

Rem 3.21. A communication model with a random input for a stochastic channel is a model, in which a random $X \in_R W$ (stochastically) independent of the channel CH is sent through the channel CH and the output of the channel is $Y := \mathrm{CH}(X)$. Informally, this means that the noise of CH is independent on the random choice of the input. In this case, we have

Def 3.22. Binary symmetric channel with the cross over probability $0 < p < 1$ is defined as $\mathrm{CH} : \{0,1\} \xrightarrow{R} \{0,1\}$ by

$$\mathrm{CH}(x_1, \ldots, x_n) = (x_1 \oplus N_1, \ldots, x_n \oplus N_n),$$

where $N_1, \ldots, N_n \in_R \{0,1\}$ are independent equally distributed random bits with $P(N_i = 1) = p$. Somewhat informally, take $x_1, \ldots, x_n$ make n tosses of a coin with the tails probability p and flip the i-th bit if the i-th toss was tails.

Binary stochastic channels can be generalized to q-ary stochastic channels:

Def 3.23. For $q \in \mathbb{N}$, $q \geq 2$, a q-ary symmetric channel with the cross over probability $0 < p < 1$ is defined as $\mathrm{CH} : K \xrightarrow{R} K$ with $q = |K|$ where $\mathrm{CH}(x_1, \ldots, x_n) = (Y_1, \ldots, Y_n)$, where $Y_1, \ldots, Y_n \in_R K$ are independent and such that $\mathsf{P}(Y_i = x_i) = 1-p$ and $\mathsf{P}(Y_i = x_i') = \frac{p}{1-q}$ for every $x_i' \in K \setminus \{x_i\}$.

Prop 3.24. *The probability of transition from $x \in K^n$ to $y \in K^n$ in a q-ary stochastic channel with $q \in \mathbb{N}, q \geq 2$ and the crossover probability $p \in (0,1)$ is equal to*

$$p_{x \to y} = (1-p)^{n - d(x,y)} \left(\frac{p}{q-1} \right)^{d(x,y)}.$$

Proof. Let $x = x_1 \cdots x_n$ and $y = y_1 \cdots y_n$. The event A_i that x_i is transmitted as y_i through the channel is $1-p$ if $x_i = y_i$ and $\frac{p}{q-1}$ if $x_i \neq y_i$. The events $A_1, \ldots, A_n$ are independent so that the event $A_1 \cdots A_n$ that x_i is transmitted to y_i for each $i = 1, \ldots, n$ has the probability $p_{x \to y}$ as described in the assertion. □

Prop 3.25. *For a nonempty finite word space W a channel $\mathrm{CH} : W \xrightarrow{R} W$ with the transition probabilities $p_{x \to y} = P(\mathrm{CH}(x) = y)$, and a random variable $X \in_R W$ independent of* CH *and the random variable $Y = \mathrm{CH}(X)$, we have*

$$\mathsf{P}(Y = y \,|\, X = x) = p_{x \to y}$$

for all $x, y \in W$.

Proof. We have

$$\begin{aligned}
\mathsf{P}(Y = y \,|\, X = x) &= \mathsf{P}(\mathrm{CH}(X) = y \,|\, X = x) && \text{(since } Y = \mathrm{CH}(X)\text{)} \\
&= \mathsf{P}(\mathrm{CH}(x) = y \,|\, X = x) && \text{(in view of } X = x\text{)} \\
&= P(\mathrm{CH}(x) = y) && \text{(since CH is independent of } X\text{)} \\
&= p_{x \to y}
\end{aligned}$$

□

Rem 3.26. If the input of a stochastic channel is modeled as a random variable, we can just forget about the details of how define $Y = \mathrm{CH}(X)$ and just think about the random input X and the random output Y with the respective conditional probabilities $p_{x \to y} = \mathsf{P}(Y = y \,|\, X = x)$ being given.

3.3 Maximal-likelihood decoding

Consider a code $\emptyset \neq C \subseteq W$ in a finite word space W and let $X \in_R C$ be a random input of the channel that has a known distribution $p_x := \mathsf{P}(X = x)$ for $x \in C$ and $Y \in_R W$ the respective random output of the channel with $P(Y = y) > 0$ for each $y \in W$. Let $p_{x\to y} = \mathsf{P}(Y = y \,|\, X = x)$ – the probability of receiving y under the condition that x is sent – be given. When a concrete $y \in W$ is received, it is observed by the receiver, but the receiver does not know which $x \in C$ was sent. We want to decode y to an $x \in C$ that has the highest probability of being sent under the condition that x was received. That is we want to decode $y \in W$ to $D(y) \in \mathrm{Argmax}_{x\in C}\, \mathsf{P}(X = x \,|\, Y = y)$. This is called the maximal likelyhood decoding, or ML decoding. In terms of the data p_x and $p_{x\to y}$ with $x \in C$ and $y \in W$, which is assumed to be given, we can express the objective function $\mathsf{P}(X = x \,|\, Y = y)$ to be maximized over $x \in C$ as follows:

$$\begin{aligned}\mathsf{P}(X = x \,|\, Y = y) &= \frac{\mathsf{P}(Y = y \,|\, X = x)}{\mathsf{P}(Y = y)}\mathsf{P}(X = x)\\ &= \frac{\mathsf{P}(Y = y \,|\, X = x)}{\sum_{x\in C} \mathsf{P}(Y = y \,|\, X = x)\,\mathsf{P}(X = x)}\mathsf{P}(X = x)\\ &= \frac{p_{x\to y}}{\sum_{c\in C} p_c \cdot p_{c\to y}}p_x\end{aligned}$$

When X is uniformly distributed in C, which is a common assumption, the latter expression can be simplified to

$$\mathsf{P}(X = x \,|\, Y = y) = \frac{p_{x\to y}}{\sum_{c\in C} p_{c\to y}}.$$

The value in the denominator is independent on x so that under the assumption that a code word which is sent is uniformly distributed in C (that is, all code words are equaliy likely) we can choose $D(y) \in \mathrm{Argmax}_{x\in C}\, p_{x\to y}$.

Exercise 3.27. Determine an ML decoder for $C = \{0, 1\}$ and $W = \{0, 1, 2\}$ in the case $p_0 = p_1 = \frac{1}{2}$ and $p_{x\to y}$ given by

$p_{x\to y}$	$x = 0$	$y = 1$	$y = 2$
$x = 0$	0.8	0.2	0
$x = 1$	0.1	0.8	0.1
$x = 2$	0	0.2	0.8

Determine the probability of the correct recovery with the ML decoder under the above assumptions. Solve the same problems for the code $C = \{0, 2\}$ with $p_0 = p_2 = \frac{1}{2}$.

Def 3.28. For a code $\emptyset \neq C \subseteq K^n$ in, a decoder $D : W \to C$ satisfying $D(y) \in \mathrm{Argmax}_{x\in W}\, d(x, y)$ is called da minimum-distance decoder.

The following assertion shows that if , for a q-symmetric channel a symbol $s \in K$ is received as s has probability higher than $> 1/q$, while for each other symbol $s' \in K \setminus \{s\}$, the probability that s received as s' is $< 1/q$, then ML decoders are exactly the minimum distance decoders.

Maximal likelihood estimate should have been defined less ambigously. There are two cases: where one does and one does not fix a distribution on code words. What should be changed? Terminology?

Prop 3.29. *When a word $x \in K^n$ is sent through a symmetric q-ary channel with a cross-over probability $p \in (0, 1 - \frac{1}{q})$, then a decoder for this channel is an ML decoder if and only if it is is a minimum-distance decoder.*

Proof. This first part of the assertion is clear. To show the second part of the assertion note that

$$p_{x \to y} = \left(\frac{p}{(1-p)(q-1)} \right)^{d(x,y)} (1-p)^n,$$

where the condition $0 < p < 1 - \frac{1}{q}$ ensures

$$0 < \frac{p}{(1-p)(q-1)} < 1$$

which shows that the maximization of $p_{x \to y}$ for $x \in C$ is equivalent to minimizaion of $d(x, y)$ for $x \in C$. □

3.4 Shannon's theorem on the capacity of noisy channels

Change entropy to q-ary entropy. Sort stuff. Adjust the formulation of Shannon's theorem (rate bound defined not with ϵ but with a rate).

Imagine sending a code of the from $\underbrace{0 \cdots 0}_{k} \underbrace{1 \cdots 1}_{k}$. What happens when after its distorted version arrives as an output of the binary symmetric channel with the crossover probability $0 \leq p \leq \frac{1}{2}$? If $p = 0$, just the original code. If $p = \frac{1}{2}$, it just a random string uniformly distributed in $\{0, 1\}^{2k}$ and the original information gets completely destroyed by the noise. If p is somewhere in between, the information survives but the output gets gradually closer to a total mess as p grows and gets closer to $1/2$:

If we encode k bits via n to cope with the noisy channel, then the ratio k/n quantifies the "informativeness" of our code, meaning the amount of the code which is the "pure" information. Even more generally, we introduce the following definition with respect to any finite alphabet K:

Def 3.30. For a q-ary code C of length n, the value

$$R(C) = \frac{\log_q |C|}{n}$$

is called the information rate, or simply the rate, of C.

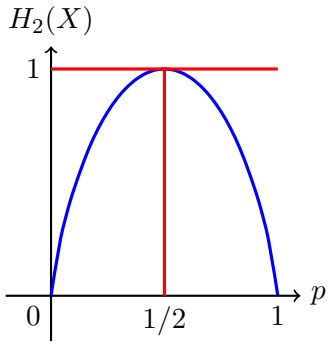


Figure 1: Binary entropy of a biased coin $X \in_R \{0,1\}$ as a function of the probability of the tails

In what follows, we interpret $p \log_q p$ as zero for $p = 0$. This represents the extension of the latter expression by contunity from positive values of p to the value $p = 0$.

Def 3.31. For a random variable $X \in_S S$ distributed in a non-empty finite set S, we define the q-ary entropy as

$$H_q(X) = -\sum_{x \in S} P(X = x) \log_q P(X = x).$$

where the terms corresponding to the probability $P(X = x) = 0$ are interpreted as zero.

Rem 3.32. Informally, $H_q(X)$ measures the expected surprise associated with observing values of X, where the reference value of the q-ary entropy equal one is associated with the expected surprise of observing a random uniformly distributed symbol of the q-ary alphabet. Or to put it differently: $H_q(X)$ is the expected number of random symbols of the q-ary alphabet provided by X.

If we have X recording the outcome of a toss of a biased coin with the probability of the head equal to to $p \in (0,1)$ such that $X = 0$ for heads and $X = 1$ for tails, then $H_2(X) = -p \log_2 p - (1-p) \log_2 (1-p)$. When p is close to 0 or close to 1, the outcome of X is close to being definite so that observation of X does not provide that many random bits on the average. Indeed, one can see that $H(X) \to 0$ when $p \to 0$ or $p \to 1$ in this situation.

For a random variable X that takes only finitely many values $v_1, \ldots, v_k$ with the positive probabilities $p_i = P(X = v_i)$, the value $H(X) = -\sum_{i=1}^{k} p_i \log_2 p_i$ is called the entropy of X. Roughly speaking, $H(X)$ describes how surprising X is on the average (the expectation of surprise). Consider, for example, the variables that encodes a toss of a coin X with $P(X = \text{ head}) = p$ and $P(X = \text{tail}) = 1 - p$. Its entropy is $-p \log_2 p - (1-p) \log_2 p$. When $p \to 0$, we have $H(X) \to 0$, that is when tails is very likely and head not, we would usually have a tail, when tossing such a coin, and it would be no surprise. When $p = \frac{1}{2}$ we have a fair coin, for which $H(X) = 1$. That quantifies that the coin gives us "one bit of random information".

As a next example, consider the random variable X that describes the random transition of symbols sent through a q-ary symmetric channel.

A symbol $k \in K$ of a q-ary channel is replaced by $k' \in K \setminus \{k\}$ with probability $p/(q-1)$ and remains the same is $1-p$. We can associate to this action a random variable with q values having probabilities $\frac{p}{q-1}, \ldots, \frac{p}{q-1}, 1-p$. The entropy of this variable is

$$-(1-p)\log_2(1-p) - \sum_{i=1}^{q-1} \frac{p}{q-1} \log_2 \frac{p}{q-1} = -(1-p)\log_2(1-p) - p\log_2 \frac{p}{q-1}$$

A symbol of K carries roughly $\log_2 q$ bits of information. In the q-ary symmetric channel it is distorted by the bits of the noise $k \in K$, whose expected number we've calculated above.

We are now motivated enough to introduce the capacity of the q-ary symmetric channel to be the difference of $\log_2 q$ and entropy $H(X)$ of the variable X that describes the action of noise on the symbols of K. This difference $\log_2 q - H(X)$ is the expected number of bits carried by a symbol of the alphabet that survive the noise.

Def 3.33. For $q \in \mathbb{N}, q \geq 2$ and $0 < p < 1$. The value

$$\kappa_q(p) := \log_2 q + p\log_2 \frac{p}{q-1} + (1-p)\log_2(1-p).$$

is called the information rate or the capacity of the q-ary symmetric channel with the crossover probability p.

S

Thm 3.34 (Shannon's noisy-channel coding theorem)**.** *Let K be a q-ary alphabet with $q \in \mathbb{N}, q \geq 2$. $1-p \in (1/q, p)$. Then for every $\epsilon > 0$ there exist a coding scheme (C, g) for a code $C \subseteq K^n$ of some (sufficiently large) block length $n \in \mathbb{N}$ and the rate $R(C) \geq \kappa_q(p) - \epsilon$ such that the probability of the error of the coding scheme (C, D), with $D : K^n \to C$, on the q-ary symmetric channel with the crossover probability p is at most ϵ.*

Rem 3.35. There is also a converse theorem that says, roughly speaking, the capacity $\kappa_q(p)$ of the channels cannot be exceeded. That is, one cannot have $R(C) > \kappa_q(p)$ and still be able to transmit and decode a word with a good accuracy.

Rem 3.36. The theorem says that there is a coding scheme that works almost flawlessly on a given symmetric channel, in the sense that decoding gives almost no errors and the capacity of the channel is almost completely used up. However, the proof of this theorem is of existential nature. That is, finding such flawless codes is a hard problem.

We are going to derive a version of this theorem for $q = 2$. We start with a prepreration. We will have to deal with binomial coefficients, and will need to know roughly how to bound them roughly by simple analytic expressions. It turns out that the entropy of a toss of a biased coin is the right analytic expression for that purpose. Furthermore, it is well-known and not hard to show that the binomial coefficients $\binom{n}{0}, \ldots, \binom{n}{n}$ form a so-called unimodular sequence, which is symmetric up to the order reversion.

Lem 3.37 (binomial coefficients - entropic bound and unimodularity)**.** For every $i \in \{0, \ldots, n\}$, the binomial coefficient $\binom{n}{i}$ can be bounded from above via

$$\binom{n}{i} \leq 2^{h(\frac{i}{n})n},$$

where $h : [0,1] \to \mathbb{R}$ is given by

$$h(p) := -p \log_2 p - (1-p) \log_2 (1-p).$$

Furthermore, $\binom{n}{i}$ is monotonically increasing in i for $i \in \{0, \ldots, \lfloor \frac{n}{2} \rfloor\}$ and monotonically decreasing in i for $i \in \{\lceil \frac{n}{2} \rceil, \ldots, n\}$.

Proof. Write n^n as $((n-i)+i)^n$, then expand this n-th power of the sum of $n-i$ and i using the binomial theorem:

$$n^n = (n-i+i)^n = \sum_{k=0}^{n} \binom{n}{k} (n-i)^{n-k} i^k.$$

Leaving the term for $k = i$ only, we arrive at the estimate

$$n^n \geq \binom{n}{i} (n-i)^{n-i} i^i.$$

Clearly, the latter inequality an upper bound on the i-th binomial coefficient, since we can rewrite it as the estimate:

$$\binom{n}{i} \leq \frac{n^n}{(n-i)^{n-i}\, i^i}.$$

The asserted upper bound on the binomial coefficient is then derived by a slight reformulation of the latter upper bound. One has

$$\frac{n^n}{(n-i)^{n-i} i^i} = \frac{n^{n-i}\, n^i}{(n-i)^{n-i}\, i^i} = \left(\frac{n}{n-i}\right)^{n-i} \left(\frac{n}{i}\right)^i$$

and one can see that the binary logarithm of the latter value is exactly $h\left(\frac{i}{n}\right) n$, yielding the desired bound.

The second part of the assertion is left as an exercise. □

There are even more precise analytic estimates, both from above and below, for the value of the binomial coefficients, which are obtained by writing down the binomial coefficient in terms of the factorials and then applying the so-called Stirling formula to the factorials. For our purposes, however, the above estimate, we have derived using elementary methods, is good enough.

Integrate the proof idea into the actual proof

Proof of Shannon's theorem for $q = 2$. Let $R := \kappa_2(p) - \epsilon$ the lower bound on the rate $R(C)$ we want to achieve. Rather than constructing a code $\emptyset \neq C \subseteq K^n$ and a decoder $D : K^n \to C$, we prefer to construct a pair (f, g) with an encoder $f : K^m \to K^n$ from the message space K^m to the word space K^n and a decoder $g : K^n \to K^m$. We only give a proof in the binary case $q = 2$. So, we fix our alphabet K to be $K = \{0, 1\}$. We consider the length of the message $m \in \mathbb{N}$ and the length of the code word $n \in \mathbb{N}$ such that $m \leq n$. Values for m and n will be fixed later in the proof. We will chose the encoder f to be an injective map, so that the corresponding code $C = f(K^m)$ has exactly 2^m codewords and its binary rate is m/n. Once f and g are fixed, the decoder $D : K^n \to C$ corresponding to the pair (f, g) is not hard to fix:

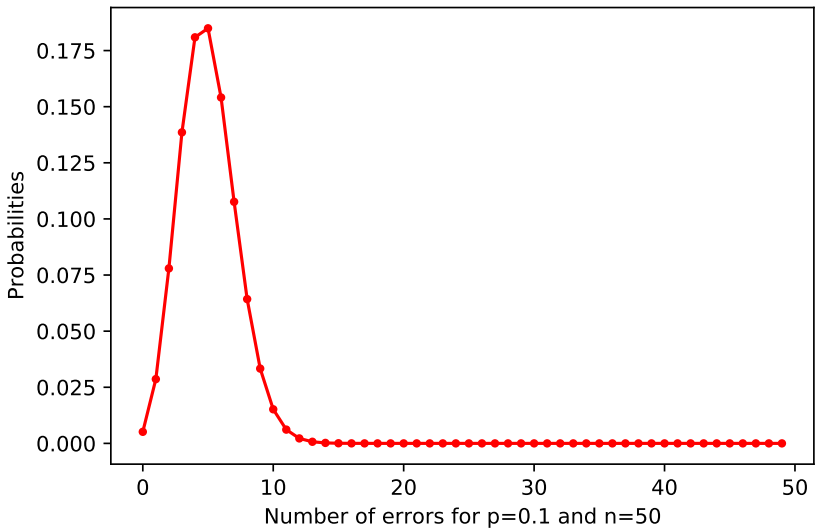
it is just $D(y) = f(g(y))$ (meaning that one decodes to a message and then determines the code word of that message).

The noise of our binary channel is a vector $N = (N_1, \ldots, N_n)$ that stores the n tosses $N_1, \ldots, N_n \in_R \{0, 1\}$ of a biased coin with the probability of the tails $P(N_i = 1)$ equal to $p \in (0, \frac{1}{2})$. We use the addition modulo two, denoted as $\oplus$, for vectors in the componentwise sense. Thus, for $y = (y_1, \ldots, y_n) \in \{0, 1\}^n$ by $y \oplus N$ we mean the noisy version of y, given by $y \oplus N = (y_1 \oplus N_1, \ldots, y_n \oplus N_n)$. If y is sent through the channel, then $y \oplus N$ is received, and its number of errors is $\mathrm{wt}(N) = \sum_{i=1}^{n} N_i$. It is not hard to calculate the expecation and the variance of the number of errors. One has

$$E(\mathrm{wt}(N)) = \sum_{i=1}^{n} E(N_i) = np, \qquad \text{(by additivity of } E\text{)}.$$

$$V(\mathrm{wt}(N)) = \sum_{i=1}^{n} V(N_i) = np(1-p). \qquad \text{(by additivity of } V \text{ for independent variables)}$$

An important facotu about $\mathrm{wt}(N)$ is that its distribution is concentrated around its mean value np, when n. This means that $\mathrm{wt}(N)$ is likely to be close to np so that the probability of $\mathrm{wt}(N)$ being larger thatn $n\rho$ for some constant $\rho > p$ is very small. Here is an illustration for the distrubution of $\mathrm{wt}(N)$ for concrete choice of p and n being $p = 0.1$ and $n = 50$



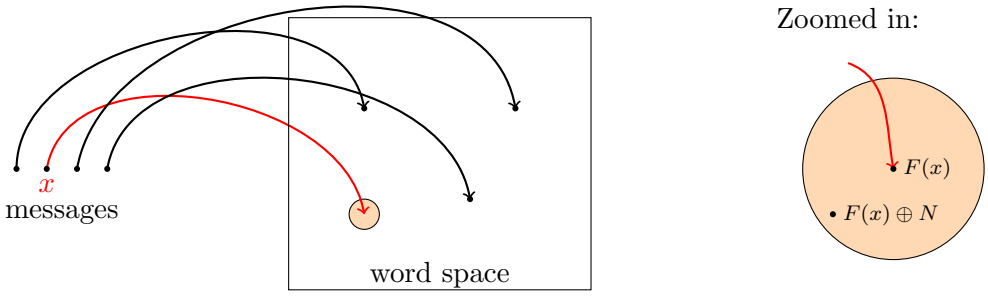
Strong concentration of the binomially distributed error count $\mathrm{wt}(N)$ around its expected value $E(\mathrm{wt}(N)) = np = 5$.

In terms of the word spaces, the latter means that it is typical for the distorted word $y \oplus N$ to be not too far away from y. Therefore fixing $\rho > p$, we are likley to have $y \oplus N$ within the hamming ball of radius ρn centered in y. This way of reasoing needs to be supported by inequalties from the probabiliy theorem. One could make use of inequalities that directly address the way $\mathrm{wt}(N)$ is distributed (it has the so-called binomial distribution), but since we do not intened to provide optimal estimates of probabilities, we will stick to Chebyshev

inequality, which works for any distrubtion and requires the knowledge of the expectation and the variance only.

Our proof approach employs so-called *probabilistic method*, which is a method in combinatorics that demonstrates the existence of a combinatorial object with a given property by considering that an approriately defined random object has a desired property with a positive probability. The method is very powerful, but not constructive. In our approach, in order to pick an encoding map $f : K^m \to K^n$, we consider the random uniformly distributed injective map $F : K^m \xrightarrow{R} K^n$. This means, that each message $x \in K^m$ gets a random code word $F(x)$ assigned to it, where there are no preferences as to what code word it will get, but if $x \neq x'$ then $F(x) \neq F(x')$. informally, the latter means that the code words of the 2^m messages are scattered randomly over the word space K^n.

The proof idea is as follows. If x is encoded as $F(x)$ and $F(x) \oplus N$ is received, then since the number of errors is not large in general, $F(x) \oplus N$ is very likely to be in a ball of some quite small radius ρn cencetered at $F(x)$. But on the other hand, since this ball is so small compared to the size of the word space K^n, all the other code words $F(x')$ with $x \neq x'$ scattered in K^n are very likely to be outside of the mentioned ball. That means, that the minimum-distance decoding is likely to tranasform $F(x) \oplus N$ to $F(x)$. Here is a pictogram for this explanation:



The ball $B(F(x), \rho n)$ is so tiny compared to the whole word space K^n that the code word $F(x')$ with for $x' \neq x$ are unlikely to be in this ball.

As already mentioned above, we use the minimum-distance decoding which means that

$$g(y) \in \text{Argmin}_{x' \in K^n} d(y, F(x')).$$

When x is sent, the docoding is correct when $g(F(x) \oplus N) = x$. In particular, this is the case, when $d(F(x) \oplus N, F(x')) > d(F(x), F(x) \oplus N)$ for each x' not equal to x.

The analysis of the probability of the error is based on fixing a threshold $\rho \in (p, 1/2)$ on the relative hamming distance of the code words to $F(x) \oplus N$. The value ρ will be fixed in the end of the proof along with n and m, but we assume that ρ is chosen in such a way that ρn is a natural number, because we want consider the hamming distance ρn. Later on, when we fix n, m and ρ, we will take care about fulfilling this condition on ρn.

We fix an arbitrary $x \in K^m$ and upper bound the probability of the random event E_x that x is decoded incorrectly using the minimum-distance decoding with respect to the encoder F. When x is fixed, the randomness in our considerations arises from the random encoder F and the noise N of the channel and we assume them to be independent of each other. If $F(x) \oplus N$

is at distance at most ρn to $F(x)$ and the code words $F(x')$ of all the other messages $x' \neq x$ are further away from $F(x) \oplus N$ than ρn, the decoding is correct. Taking contraposition, we thus see that if the error E_x for decoding x occurs, then the noise was too large, meaning that the distance of the received word $F(x) \oplus N$ to $F(x)$ is larger than ρn, or otherwise, for some message x' different from x, the distance of $F(x')$ to $F(x) \oplus N$ is too small, menaing that $d(F(x') \oplus N, F(x)) \leq \rho n$ holds. The former event of distance of $F(x)$ and $F(x) \oplus N$ being larger than ρn can be expressed as $\mathrm{wt}(N) > \rho n$, since the distance of $F(x)$ to $F(x) \oplus N$ is exactly $\mathrm{wt}(N)$. We denote the event that the distance of $F(x) \oplus N$ to $F(x')$ is at most ρn by $B_{x,x'}$.

The union bound gives

$$\mathsf{P}\left(E_x\right) \leq \mathsf{P}\left(\mathrm{wt}(N) > \rho n\right) + \sum_{x' \in K^m \setminus \{x\}} \mathsf{P}\left(B_{x,x'}\right).$$

Chebyshev's inequality yields the estimate

$$\mathsf{P}\left(\mathrm{wt}(N) \geq \rho n\right) \leq \mathsf{P}\left(|\,\mathrm{wt}(N) - pn| \geq (\rho - p)n\right) \leq \frac{p(1-p)n}{(\rho-p)^2 n^2} \leq \frac{p(1-p)}{(\rho-p)^2 n}.$$

To estimate $\mathsf{P}\left(B_{x,x'}\right)$, we condition on the value of N first and obtain, taking into account the independence of N from F, that

$$\mathsf{P}\left(B_{x,x'}\right) = \sum_{z \in K^n} \mathsf{P}\left(d(F(x)+z, F(x')) \leq \rho n\right) \mathsf{P}\left(N = z\right)$$

As a next step, we condition on the value of $F(x)$ and obtain

$$\mathsf{P}\left(B_{x,x'}\right) = \sum_{y,z \in K^n} \mathsf{P}\left(d(y+z, F(x')) \leq \rho n \,\middle|\, F(x) = y\right) \mathsf{P}\left(F(x) = y\right) \mathsf{P}\left(N = z\right)$$

It is not hard to see that under the condition $F(x) = y$, the vector $F(x')$ is uniformly distributed in $K^n \setminus \{y\}$. This is clear for the symmetry reasons: if an injective map is built up, and a value y of the map F on x has been fixed, the value on another point x' can be any other vector in $K^n \setminus \{y\}$ with the probability of all these vectors being equal . This gives the estimate:

$$\begin{aligned}
\mathsf{P}\left(d(y+z, F(x')) \leq \rho n \,\middle|\, F(x) = y\right) &= \frac{|B(y+z, \rho n) \setminus \{y\}|}{2^n - 1} \\
&\leq \frac{|B(y+z, \rho n)|}{2^n - 1} \\
&\leq \frac{1}{2^n - 1} \sum_{i=0}^{\rho n} \binom{n}{i} \\
&\leq \frac{1}{2^n - 1} (\rho n + 1) \binom{n}{\rho n} \\
&\leq \frac{1}{2^n - 1} (2\rho n) 2^{h(\rho) n}.
\end{aligned}$$

Consequently,

$$\mathsf{P}\left(B_{x,x'}\right) \leq \frac{(2\rho n)2^{h(\rho)n}}{2^n-1} \sum_{y,z\in K^n} \mathsf{P}\left(F(x)=y\right)\mathsf{P}\left(N=z\right) = \frac{(2\rho n)2^{h(\rho)n}}{2^n-1}.$$

The latter bound gives

$$\sum_{x'\in K^m\setminus\{x\}} \mathsf{P}\left(B_{x,x'}\right) \leq \frac{2^m-1}{2^n-1}\cdot(2\rho n)2^{h(\rho)n} \leq \frac{2^m}{2^n}\cdot(2\rho n)2^{h(\rho)n} = 2^{m-(1-h(\rho))n}(2\rho n).$$

The above estimates amount to the bound

$$\mathsf{P}\left(E_x\right) \leq \frac{p(1-p)}{(\rho-p)^2 n} + 2^{m-(1-h(\rho))n}(2\rho n).$$

We want to have $m \geq Rn$ to achieve the rate at least R. Since $R < \kappa_2(p) = 1-h(p)$, there exists $\rho \in (p, 1/2)$ close enough to p that satisfies $R < 1-h(\rho) < 1-h(p)$, as h is continuous and is strictly increasing on $(0, 1/2)$. We can choose such a ρ to be a rational number, in order to ensure that $\rho n \in \mathbb{N}$ holds for infinitely man choices of $n \in \mathbb{N}$. Fix $m := \lceil Rn \rceil$, which gives the estimate $m \leq Rn+1$ and results into the estimate

$$\mathsf{P}\left(E_x\right) \leq \frac{p(1-p)}{(\rho-p)^2 n} + 2^{Rn+1-(1-h(\rho))n}(2\rho n) = \frac{p(1-p)}{(\rho-p)^2 n} + \frac{4\rho n}{2^{(1-h(\rho)-R)n}} \xrightarrow[n\to\infty]{} 0.$$

Thus, we have $\mathsf{P}\left(E_x\right) \leq \epsilon$ by choosing $n \in \mathbb{N}$ sufficiently large and such that $\rho n \in \mathbb{N}$ holds.

We have shown that the probability of the error when decoding a fixed arbitrary message x can be made arbitrarily small. To conclude the proof, we need to show that when a random message X uniformly distributed in K^m is sent through the channel, the decoding error is still small. We assume that this X is independent of our random encoder F and the noise N.

Consider the event E that when X is sent, the minimum-distance decoding with respect to the encoder F is incorrect. We make use of our estimate on $P(E_x)$ for an arbitrary fixed x. We have

$$\mathsf{P}\left(E\right) = \sum_{x\in K^m} \mathsf{P}\left(E \,|\, X=x\right)\mathsf{P}\left(X=x\right) = \sum_{x\in K^m} P(E_x)P(X=x) \leq \epsilon \sum_{x\in K^m} \mathsf{P}\left(X=x\right) = \epsilon.$$

Our implementation of the probabilistic method is concluded by showing that exists an injective encoder $f : K^m \to K^n$, with respect to which the error of decoding the random message X incorrectly is at most ϵ. Condititioning on the possible realizations of the random encoder F, we have

$$\sum_{f:K^m\to K \text{ injective}} \mathsf{P}\left(E \,|\, F=f\right)\mathsf{P}\left(F=f\right) = \mathsf{P}\left(E\right) \leq \epsilon.$$

When the average of finitely many values is at most ϵ, there is at least one value among them which is also at most ϵ. Hence, there exists an injective map $f : K^m \to K^n$, for which $\mathsf{P}\left(E \,|\, F=f\right) \leq \epsilon$. We have thus shown the existence of an injective map f with the property that the error of the minimum-distance decoding of the uniformly distributed random vector X in K^m is at most ϵ. □

4 Modular arithmetic: foundations and applications

A large number of codes is based on finite algebraic structures. An algebraic structure is comprised of a ground set and operations on that set satisfying a certain system of rules. In coding theory one uses such structure over a finite ground set.

4.1 Abstract rings and fields in a nutshell

This time K may or may not be finite. On a ground set K, with two binary operations $+, \cdot : K^2 \to K$ consider the following rules:

		Addition	Multiplication
neutral element	(N)	$a + 0 = a$	$a \cdot 1 = a.$
associative law	(A)	$(a + b) + c = a + (b + c)$	$(a \cdot b) \cdot c = a \cdot (b \cdot c)$
commutative law	(C)	$a + b = b + a$	$a \cdot b = b \cdot a$
inversion	(I)	$a + (-a) = 0$	$a \neq 0 \Rightarrow a \cdot a^{-1} = 1.$

To emphasize that 0 and 1 are elements of the particular structure K we work in, it sometimes make sense to write $0 = 0_K$ and $1 = 1_K$. Apart from that rules phrased independently for $+$ and $\cdot$, there are two laws that link addition with multiplication:

distributive	(D)	$(a + b) \cdot c = a \cdot c + b \cdot c,$
non-triviality		$0 \neq 1,$

with the non-triviality making use of 0 and 1 from (N).

We should clarify how to read these laws. (N) for the addition declares the existence of a unique element in K, denoted by 0, for which $a + 0 = a$ holds for every $a \in K$. (N) for multiplication is interpreted similar. For (A),(C) we just declare the validity of the respective equations for all $a, b, c \in K$. (I) for addition declares that for every $a \in K$ there is a unique element in K, denoted as $-a$, that satisfies $-a \in K$. (I) for multiplication declares for every $a \in K$ with $a \neq 0$ the existence of a unique element in K, denoted by a^{-1}, that satisfies $a \cdot a^{-1} = 1$.

It turns out that when combining such laws to declare an algebraic structure, the uniqueness assumption is redundant and can be derived as a consequence of the laws.

Def 4.1. A structure $(K, +)$ with one operation $+ : K^2 \to K$ satisfying (N), (A), (C), (I) for $+$ is called an Abelian group (or commutative group).

Def 4.2. A structure $(K, +, \cdot)$ with operations $+, \cdot : K^2 \to K$ satisfying (N), (A), (C), (I) for $+$ and $\cdot$, (D) and $0 \neq 1$ is called a *field*.

Def 4.3. A structure $(K, +, \cdot)$ with operations $+, \cdot : K^2 \to K$ satisfying (N), (A), (C) for $+$ and $\cdot$, (I) for $+$, (D) and $0 \neq 1$ is called a *unitary commutative ring*.

From the above definitions we see that a field, is a unitary commutative ring, in which every non-zero element is invertible with respect to multiplication.

The fields of real and rational numbers are known examples of fields. The ring of integer numbers is a well-known example of a unitary commutative ring that is not a field. Those are infinite structures, whereas in coding theory one interested in finite ones. We will develops means to generating such finite structures.

4.2 Ideals and quotient rings

Here, rather than unitary commutative ring, we merely say ring. That is, by default, the ring will mean a unitary commutative ring. In a ring R, the following notations will be useful to write down different properties concisely:

$$A + B = \{a + b : a \in A, b \in B\} \qquad f + B = \{f + b : b \in B\},$$
$$A \cdot B = \{a \cdot b : a \in A, b \in B\} \qquad fB = \{a \cdot b : b \in B\},$$

where $A, B \subseteq R$ and $f \in R$.

Def 4.4. Let R be a ring. Then $I \subseteq R$ is called an ideal if the following hold:

(a) $0 \in I$ (the zero is in I)

(b) $I + I \subseteq I$ (sum of elements of I is in I)

(c) $R \cdot I \subseteq I$ (sum of any ring element and an element in I is in I).

It is not hard to see that for $f \in R$, the set $I = fR = \{fr : r \in R\}$ is an ideal, this is called the ideal in R generated by f. One can also consider the ideals $f_1R + \cdots + f_mR$ generated by finitely many elements $f_1, \ldots, f_m \in R$.

One writes $a \equiv b \mod I$ if $a - b \in I$ and one writes $a \equiv b \mod f$ if $a - b \in fR$ saying that a and b are congruent modulo I or congruent modulo f, respectively. For an ideal I, one introduces the set

$$R/I = \{[r] : r \in R\},$$

where

$$[r] = [r]_I = r + I = \{r + a : a \in I\}$$

is the so called coset of r modulo I. On R/I natural operations $+$ and $\cdot$ can be defined by

$$[a] + [b] := [a + b],$$
$$[a] \cdot [b] := [a \cdot b].$$

These operations are well-defined, in the sense, that the choice of a concrete representative of the coset has no impact on the result. With these operations, if the ideal I is not equal to R, then R/I, endowed with $+$ and $\cdot$ as above, is a unitary commutative ring. One has $0_{R/I} = [0]_I$ and $1_{R/I} = [1]_I$.

Rem 4.5. Intuitively, R/I is a structure obtained from R, by mapping I to zero or, roughly speaking, by declaring elements of I to be zero in R/I. Analogously, R/fR i s a structure obtained from R by mapping $f \in R$ to zero or, roughly speaking, declaring f to be zero in R/fR.

Def 4.6. An ideal I in the ring R is called maximal, if there exists no ideal strictly containing I other than the whole ring R.

Thm 4.7. *For a ring R and an ideal I, the quotient ring R/I is a field if and only if I is a maximal ideal.*

Proof. Assume, I is maximal. Consider $a \in R$ with $a \neq I$ so that the coset $[a]$ is an arbitrary non-zero element of R/I. Since the ideal I is maximal, the ideal $I + aR$ that contains I as a subset, but not equal to I since it has the element a, is equal to R. From the equality $I + aR = R$, we obtain $1 \in I + aR$, meaning for some $b \in R$ one has $1 \in I + ab$. But then $[a] \cdot [b] = [1]$. Thus, we have shown that $[b] = [a]^{-1}$.

If I is not maximal, we need to show that R/I is not a field. Consider an ideal J with $I \subsetneq J \subsetneq R$. Pick an element a belonging to J but not to I. Then $I + aR$ is yet another ideal that strictly contains I as a subset. If $[a]_I$ were invertible, we had $b \in R$ such that $ab \in 1 + I$. But then $1 \in ab + I \subseteq aR + I$ would be an element of $I + aR$, which would imply that $I + aR = R$, a contradiction to $J \neq R$. □

4.3 Ring structures behind the modular arithmetic

Thm 4.8. *Every ideal of $\mathbb{Z}$ has the form $m\mathbb{Z}$ with $m \in \mathbb{N}_0$.*

Proof. Let I be an ideal in $\mathbb{Z}$. If $I = \{0\}$ the assertion is clear, so assume that I contains elements other than zero. Take to consecutive elements $a, b \in I$ with $a < b$ whose distance $m := b - a$ is minimal. It is clear that $m \in I$ and by this $m\mathbb{Z} \subseteq I$. If the latter inclusion was strict, we had an element c in I but not in $m\mathbb{Z}$. But then its distance to the closest element in $m\mathbb{Z}$ is smaller than m, which contradicts the choice of m. Hence, $I = m\mathbb{Z}$. □

Def 4.9. For $m \in \mathbb{N}$, $m \geq 2$, the ring $\mathbb{Z}_m := \mathbb{Z}/m\mathbb{Z}$ is called the ring of integers modulo m.

Prop 4.10. *For $l, m \in \mathbb{N}$, the inclusion $l\mathbb{Z} \subseteq m\mathbb{Z}$ holds if and only if l is divisible by m.*

Proof. If $l\mathbb{Z} \subseteq m\mathbb{Z}$, then $l \in m\mathbb{Z}$ and so l is divisible by m. Conversely, if l is divisible by m, then $l/m \in \mathbb{Z}$ and so $l\mathbb{Z} = m(l/m)\mathbb{Z} \subseteq m\mathbb{Z}$. □

Rem 4.11. The ring $\mathbb{Z}_m := \mathbb{Z}/m\mathbb{Z}$ has exactly m elements $[0], \ldots, [m-1]$. Imagine the $\mathbb{Z}$ rolled onto the rotary switch with m positions $0, \ldots, m-1$. Rotating by m positions does not change anything. So, rotating by m positions is as good as rotation by 0 positions. That should provide intuition for arithmetic operations in $\mathbb{Z}/m\mathbb{Z}$. Arithmetic in $\mathbb{Z}/m\mathbb{Z}$ is called modular. Time keeping on the clock uses arithmetic modulo 12 and modulo 60. Fixed size registers in computer use modular arithmetic modulo a power of two: for example, the 8-bit register arithmetic is the arithmetic is the arithmetic in the ring $\mathbb{Z}_{2^8} = \mathbb{Z}_{256}$.

Rem 4.12. The ring $\mathbb{Z}_2$ has only two elements, $0 = 0_{\mathbb{Z}_2}$ and $1 = 1_{\mathbb{Z}_2}$, with the addition given by $0 + 0 = 0, 0 + 1 = 1 + 0 = 1$ and $1 + 1 = 0$ while the multiplication given by $0 \cdot 0 = 0 \cdot 1 = 1 \cdot 0 = 0$ and $1 \cdot 1 = 1$. It is easy to see that $\mathbb{Z}_2$ is a field.

Thm 4.13. *For $m \in \mathbb{N}$, the ring $\mathbb{Z}/m\mathbb{Z}$ is a field if and only if m is a prime number.*

Proof. Clearly, for $\mathbb{Z}/m\mathbb{Z}$ being a field, it is necessary to have $m \geq 2$, as for $m = 1$ one has $[0] = [1]$. So, we assume $m \geq 2$.

Prop 4.10 allows to characterize the maximal ideals of $\mathbb{Z}$ as ideals $m\mathbb{Z}$, where $m \in \mathbb{N}$ is prime. The assertion follows from Thm 4.7. □

Rem 4.14. If p is prime, then other common notations for $\mathbb{Z}_p$ are $\mathbb{F}_p$ and $\mathrm{GF}(p)$. Here, GF stand for the Galois Field. Finite fields are called Galois Field. There are more general Galois Fields than $\mathrm{GF}(p)$.

Rem 4.15. There are more efficient ways of showing that $\mathbb{Z}/m\mathbb{Z}$ is field when m is prime. Those ways, based on the so-called Euclidean Algorithm, suggest an efficient algorithm for inversion of the non-zero elements of $\mathbb{Z}/m\mathbb{Z}$.

4.4 Examples of codes relying on modular arithmetic

4.4.1 Some codes over $\mathbb{Z}_2$

We've presented such examples in the introduction already. The coding scheme $c_1c_2 \mapsto c_1c_2c_3$ with $K = \mathbb{Z}_2$ and $c_3 = c_1 + c_2$ was the first example we considered. We did not phrase it in terms of $\mathbb{Z}_2$ though. Note that the condition $c_3 = c_1 + c_2$ can also be formulated as $c_1 + c_2 + c_3 = 0$ in $\mathbb{Z}_2$, by adding c_3 to both sides of the equation. Thus, the respective code C is the solution set of the homogeneous linear equation $c_1 + c_2 + c_3 = 0$ over $\mathbb{Z}_2$. This gives a link to the linear algebra, employed over a finite field. We will explore this important connection more systematically in what follows.

As another interpretation we can think about C as the image of the $\mathbb{Z}_2$-linear map $(c_1, c_2) \mapsto (c_1, c_2, c_1 + c_2)$, which also allows us to represent the code words in the form

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

as linear combinations of two vectors with the coefficients $c_1, c_2 \in \mathbb{Z}_2$. Surely, we can also use the matrix notation:

$$\begin{pmatrix} c_1 & c_2 \end{pmatrix} \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

to describe the code word arising from c_1 and c_2.

In the introduction, we also consider the code with the coding scheme

$$(c_1, c_2) \mapsto (c_1, c_2, c_1, c_2, c_1 + c_2),$$

realizable as a linear map from K^2 to K^5 with $K = \mathbb{Z}_2$, that can be analyzed similarly.

For a general $n \in \mathbb{N}$, the well-known code from the praxis is the parity check code of length n over the alphabet $K = \mathbb{Z}_2$. It is defined by

$$C = \{(c_1, \cdots, c_n) \in K^n \,:\, c_1 + \cdots + c_n = 0\}$$

4.4.2 Parity-check codes

Let K be a finite Abelian group, used as an alphabet let $\pi_1, \ldots, \pi_n : K \to K$ be bijective maps and $a \in K$. The codes of the form

$$C = \left\{ (c_1, \ldots, c_n) \in K^n \,:\, \sum_{i=1}^{n} \pi_i(c_i) = a \right\}$$

are called parity-check codes. The parity-check code with $K = \mathbb{Z}_2$ and $\pi_1, \ldots, \pi_n$ is a standard example The equality defining the parity-check code is called the parity-check equality.

Parity-check codes are 1-error detecting. Their rate is close to one, but they cannot do much more than just detecting one error.

4.4.3 ISIN and ISBN codes

Consider $K = \{0, \dots, 9\}$, which identified naturally with $\mathbb{Z}_{10}$ and fix the map $\pi : K \to K$ that maps $k \in K$ to the sum of the digits of $2k$. For example, 7 is mapped to $1 + 4 = 5$ by π since $2 \cdot 7 = 14$. It turns out that π is a bijection. The ISIN-Code (International Securities Identification Number) is a code of length 14 over K given by the equality

$$\pi(c_1) + c_2 + \pi(c_3) + \cdots + \pi(c_{13}) + c_{14} \equiv 0 \mod 10.$$

Text representation of this code is given by two letters followed 10 digits in K, with c_1c_2 and c_3c_4 encoding the first and the second letter respectively (via $10 \mapsto A$, $11 \mapsto B$, $\dots 35 \mapsto Z$) This is a control code and so it detects one error. But it also detect a swap in two neighboring symbols unless these symbols are 0 and 9.

Exercise 4.16. Show that the ISIN-Code C detects the exchange of c_i and c_{i+1} in the code word $c = c_1 \cdots c_{14}$ unless $\{c_i, c_{i+1}\} = \{0, 9\}$.

The ISBN10 code, uses the alphabet $K = \{0, 1, \dots, 9, X\}$, interpreted as $\mathbb{Z}_{11}$. It is a control code given by the equality

$$\sum_{i=1}^{10} (11 - i)c_i = 0$$

in $\mathbb{Z}_{11}$.

Exercise 4.17. Show that the ISBN10 code can detect the exchange of any two symbols c_i and c_j with $1 \le i < j \le 10$.

4.4.4 The Hamming code of length 7

The hamming code of length 7 over $K = \mathbb{Z}_2$ is a subset C of K^7 described by a system of equations:

$$\begin{aligned} c_1 + c_4 + c_6 + c_7 &= 0, \\ c_2 + c_4 + c_5 + c_7 &= 0, \\ c_3 + c_5 + c_6 + c_7 &= 0. \end{aligned}$$

over K.

Exercise 4.18. Compute $|C|$ and $d(C)$ for the hamming code of length 7 and show that C is a perfect code.

5 Linear codes

It's good if you are already comfortable with basic linear algebra. If not, we'd need to cope with this somehow. We give an overview of basic notions and facts in linear algebra, to be prepared to the discussion of linear codes.

5.1 Linear algebra

5.1.1 Vector spaces

If K is a field, then a set V endowed with operations $+ : V^2 \to V$ and $\cdot : K \times V \to V$, called vector addition and scalar multiplication respectively, is called a vector space if $(V, +)$ is an Abelian group (that is, the four laws for the Abelian group are fulfilled), and, furthermore, the laws

$$\alpha(u + v) = \alpha u + \beta v$$
$$(\alpha + \beta)u = \alpha u + \beta u$$

and

$$\alpha(\beta v) = (\alpha\beta)v,$$
$$1v = v$$

hold for all $\alpha, \beta \in K$ and $u, v \in V$.

If U is a subset of a vector space V such that the vector addition and scalar multiplication on V can be restricted to U making it a vector space, then U is called a vector subspace of V, which we denote as $U \leq V$. For U to be a vector subspace of V it is necessary and sufficient that $0 \in U$ holds and $\alpha u + \beta v \in U$ is fulfilled for all $u, v \in U$ and all $\alpha, \beta \in U$.

For $n \in \mathbb{N}$, the set K^n is a vector space with respect to the componentwise addition of vectors and the componentwise multiplication of a vector with a scalar.

The expression $\lambda_1 v_1 + \cdots + \lambda_k v_k$ is called a linear combination of vectors $v_1, \ldots, v_k \in V$ with the coefficients $\lambda_1, \ldots, \lambda_k \in K$. The linear hull of a subset M of V, is the set of all linear combinations of vectors from M. The linear hull of M is a subspace of V. A system of vectors $v_1, \ldots, v_k$ is called linearly dependent some linear combination of these vectors, in which not all of the coefficients are equal to zero, gives a zero vector. A system of vectors which is not linearly dependent is called linearly independent. If a vector space V is a linear hull of a system $\{v_1, \ldots, v_n\}$ of linearly independent vectors, then $v_1, \ldots, v_n$ is called a basis of V. All bases of a given vector space have the same cardinality, and this carinality is called the dimension of the vector space. The dimension of K^n is n and the system of vectors $e_1, \ldots, e_n$, with $e_i \in K^n$ having the i-th component equal to 1 and all the remaining components equal to 0, is called the standard basis of K^n.

5.1.2 Linear maps and matrices

A map $F : V \to W$ defined on K-vector spaces V and W is called linear if $F(\alpha a + \beta b) = \alpha F(a) + \beta F(b)$ holds for all $a, b \in V$ and all $\alpha, \beta \in K$.

A matrix is a table of numbers, in our cases we consider matrices $A = (a_{ij}) \in K^{m \times n}$ with m rows and n columns, which have an entry $a_{ij} \in K$ in the i-th row and j-th column. Such a matrix is a way to store m vectors in K^n as rows of A and a way to store n vectors from K^m as columns of A. A matrix A can be multiplied with a vector $x = (x_1, \ldots, x_n) \in K^n$ giving the vector $y = Ax \in K^n$ that has $y_i = \sum_{j=1}^{n} a_{ij} x_j$ as the j-th component. On the other hand, one can calculate $w := uA$ for a vector $u = (u_1, \ldots, u_m)$ with $w_j = \sum_{i=1}^{m} u_i a_{ij}$ being the j-th component of w. By A we can defined the linear map $u \mapsto uA$ from K^m to K^n and the linear map $x \mapsto Ax$ from K^n to K^m.

For a linear map $F : V \to W$, the set $\{v \in V : F(v) = 0\}$ is called the kernel of F and the set $F(V)$ is called the image of F. The kernel is a vector subspace of V, while the image is a vector subspae of W. If V and W are finite-dimensional one can link dimension of V, the kernel and the image via

Thm 5.1 (rank-nullity theorem)**.** *Consider a linear map $F : V \to W$ on finite dimensional vector spaces. If n is the dimension of V, r is the dimension of the image of F and k is the dimension of the dimension of the kernel of F, then $n = r + k$.*

The intuition behind this theorem is as follow: F maps the n dimension of V, but "swallows" k of the dimensions so that only $r = n - k$ dimensions "survive".

The dimension of the image of F is called the rank of F.

The rank of a matrix A is defined as the dimension of the linear hull of its columns, which turns out to be the same as the dimension of the linear hull of its rows.

5.1.3 Gaussian elimination

For given $A \in K^{m\times n}, b \in K^n$ and an vector unknown $x \in K^n$, the condition $Ax = b$ is called a system of m linear equations in n variables. To solve such a system one frequently uses the so-called Gaussian elimination. To this end, the coefficients of the system are stored in the so-called extended matrix $(A \,|\, b) \in K^{m\times(n+1)}$ of the system. Multiplication of a mutliple of one of the rows of the matrix, rescaling a matrix by a non-zero factor and changing the order of rows of teh matrix are the so called elemetnary row operations, which correspond to the transformation of the m linear equations according to the well-known principles. The Gaussian elimination method in its several variations uses the above transformations to gradually transform the system into a form, from which the solution set can be read off effortlessly. Here is a brief outline of the Gauss-Jordan method. A variable occurring in some of the equation is picked and eleminated from all the remaining equations. Such a variable is called a basic variable. If there is equality that contains some variable that is not (yet) basic, such variable is picked, eliminated from all other equalities and declared to be basic. This procedure is repeated iteratively until all equalities that contain no basic variables have the left hand side identically equal to zero. When this method terminates, it gives an explicit dependence of the basic variables on the non-basic ones. In particular, if there are no non-basic variables, it just gives the solution of the system $Ax = b$.

Gaussian elimination allows one to solve various problems in linear algebraic. For example, it can be used to determine a basis of vector sub-space determined by a system $Ax = 0$ of homogeneous linear equations (homogeneous means that the right-hand sides of the equations are equal to zero).

5.2 Parameters of linear codes

Def 5.2. If K is a finite field and C is a vector subspace of K^n, then C is called a linear code. If $C \subseteq K^n$ is a linear code with $\dim(C) = k$ and $d = d(C)$, then C is called an $[n, k]$-code or $[n, k, d]$-code or $[n, k, d]_q$-code, when $q = |K|$. These values are called the parameters of linear code C.

Def 5.3. We define the weight of $\emptyset \neq C \subseteq K^n$ as $\mathrm{wt}(C) = \min\{\mathrm{wt}(c) : c \in C \setminus \{0\}\}$.

Prop 5.4. *For a linear code $C \subseteq K^n$, one has $d(C) = \mathrm{wt}(C)$ and $|C| = |K|^{\dim(C)}$.*

Proof. Exercise. □

There are two standards ways to represent a vector subspace of K^n, as a linear hull vectors or as a solution set of a system of homogeneous linear equations. In both cases, it is natural to require non-redundancy by taking the convex hull of linearly independent vectors and by taking a system of equations that does not contain any redundant equations. This leads to the following

Def 5.5. If a $[n, k]$ code C over K is the linear hull of k linearly independent vectors $g_1, \ldots, g_k$, then the matrix G with the rows $g_1, \ldots, g_k$ is called the generator matrix of C. That is, $C = K^k G := \{uG : u \in K^k\}$.

A matrix $H \in K^{(n-k)\times n}$ whose kernel $\ker H := \{u \in K^n : Hu = 0\}$ is C, is called the parity-check matrix of the code C.

Rem 5.6. The generator matrix can be used for encoding messages. Control matrices can be used to check the received messages for correctness.

Exercise 5.7. Determine generate and parity-check matrices of the parity-check code $C \subseteq K^n$. Are these matrices unique?

The parity-check matrix can also be used to determine the minimum distance of the code.

Thm 5.8. *For an $[n, k]$ linear code C over K with a check matrix H, the minimum distance $d(C)$ of C is equal to*

(a) The minimum r such that r columns in H are linearly dependent.

(b) The minimum r such that any $r - 1$ columns of H are linearly independent.

5.3 Syndrome decoding

5.3.1 Two versions of the minimum-distance decoding

We consider a linear code $C \subseteq K^n$, for which a check matrix $H \in K^{m\times n}$ of full row rank. is available and we consider the following interrelated tasks:

(a) Implement a minimum distance decoding.

(b) Assuming that the code C is e-error correcting, use the minimum distance decoding if the output y of the channel is at distance at most e to some code word or report a decoding error, otherwise.

5.3.2 Initialization of the dictionary

For both problems, the so-called syndrome decoding can be used. It consists of an initialization stage, in which a certain dictionary is created, and the stage, in which for a given output y of the channel, the decoding of y is carried out.

The initialization in the case (a) is as follows. Since H has full row rank, the linear system $Hv = s$ with the right-hand side $s \in K^m$ has a solution $v \in K^n$ and we determine a solution

$v = v(s)$ with the minimum weight. Note that computing $v(s)$ may take a certain large amount of computational time, but since it is an initialization, we do it only once. We store all of the computed $h(s)$ in a dictionary as key-value pairs with the key $s \in K^m$ and the respective value $v(s) \in K^n$. There are different efficient implementations of the dictionary data structure available, including hash tables and different kinds of search trees, sorted lists with a binary search etc. (for more details on this topic, see [2]).

The initialization in the case (b) is analogous. For each $v \in B_q(0, e)$, we compute Hv and store the key-value pair with the key $s = Hv$ and the value $v(s) = v$ in a dictionary. Note that every key is used only once in this process, since under the assumption that C is e-error correcting, we have $Hv \neq Hw$ for all $v, w \in B_q(0, e)$ with $v \neq w$. Indeed, if we had $Hv = Hw$, then $H(v - w) = 0$, which means that $c := v - w \in C \setminus \{0\}$. But then $v \in B_q(0, e) \cap B_q(c)$ since $v \in B_q(0, e)$ and $d(v, v - w) = d(0, w) \leq e$. The latter is a contradiction to assumption that C is e-error correcting.

5.3.3 Decoding using the dictionary

We now turn our attention to the decoding part of the syndrome-decoding procedure.

In the case (a), the decoding $D(y) := y - v(Hy)$ is a minimum distance decoding. First, observe that $D(y) \in C$ since $H(y - v(Hy)) = Hy - Hv(Hy) = s - Hv(s)$, with $s = Hv$, where $s - Hv(s) = 0$ by the choice of $v(s)$. To see that D is a minimum distance decoding consider $d(y, c)$ for an arbitrary $c \in C$. We have $d(y, c) = \mathrm{wt}(y - c)$, where $H(y - c) = Hy = s$. As $y - c$ is a solution of the linear system $Hv = s$ in the unknown v, by our choice of $v(s)$, we have $d(y, y - v(s)) = \mathrm{wt}(v(s)) \leq \mathrm{wt}(y - c)$. Hence $y - v(s)$ is indeed a closest code to y.

In the case (b), we again use the decoding $D(y) := y - v(Hy)$, but here it may turn out that Hy is not a key of the dictionary. This the case when y has distance large than s to every code word. In this, case we terminate with the error message "the output y of the channel has more than e errors". Indeed, if the output had at most e errors, then we had $d(y, c) \leq e$ for some $c \in C$, which would imply that $Hy = H(y - c)$ with $y - c \in B_q(0, e)$ would be a key of the dictionary. If $s = Hy$ is a key of the dictionary, then $H(y - v(s)) = Hy - Hv(s) = s - Hv(s) = 0$, by the choice of $v(s)$, which implies that $y - v(s) \in C$ and one has $d(y, y - v(s)) = d(0, v(s)) \leq s$, which shows that $y - v(s)$ is the unique code word at distance at most s to y.

5.3.4 Computational merits of the syndrome decoding

In the setting (a), we compare the computational advantages of the syndrome decoding to the direct brute-force approach.

We have fixed the check matrix H to be the $m \times n$ matrix of rank n, which means that $k := n - m$ is the dimension of C. When the generator matrix of size $k \times n$ is computed (computing G from H can be done via Gaussian elimination, which takes a polynomial number of operations over K), the closest code word to a given $y \in K^n$ can be calculated by iterating through all of the q^k code words of C and determining the hamming distances to the received word y for each of the code words. This requires $O(nq^k)$ arithmetic operations over K. In the syndrome decoding, we would store the dictionary of $q^m = q^{n-k}$ key-value pairs and then just look up the value associated to a given key. The computational costs of such a retrieval for efficiently organized dictionaries are very low: if they are logarithmic in the number of keys,

then the retrieval needs about $O(m) = O(\log n)$ operations. Even in the case of a dictionary organized as an unsorted list, we save computational time when $q^{n-k} < q^n$ holds, that is, when the dimension k of the code satisfies $k \geq n/2$.

The brute-force implementation of the minimum distance decoding in the setting (b) can be organized as organized as follows. Iterate over all $v \in B_q(s)$ and stop as soon as v with $y - v \in C$ is found, returning the code word $c = y - v$. If such v is not found, terminate with the error message. This approach requires $V_q^n(e) = |B_q(0, e)|$ iterations in the worst case. The number $V_q^n(e)$ can be quite big, depending on how e and n are chosen. Again, the syndrome decoding can be more efficient even when the dictionary is implemented as an unsorted list of key-value pairs. If $q^{n-k} < V_q^n(e)$, the number of key-value pairs is lower than the number of iterations in the brute-force implementation.

5.4 Hamming codes

For $u \in K \setminus \{0\}$, the one-dimensional vector space $Ku = \{ku : k \in K\}$ is a line (passing through the origin).

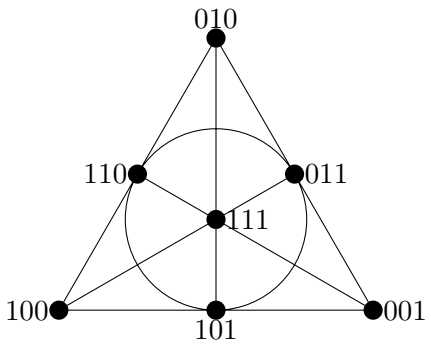
Prop 5.9. *The vector space K^m over a finite field K q elements has exactly*

$$\frac{q^m - 1}{q - 1}$$

lines through the origin.

Proof. Each of the $q^m - 1$ vectors $u \in K^m \setminus \{0\}$ generates a line (through the origin). A line is generated by any of the $q - 1$ it contains. Thus, the map $u \mapsto Ku$ from $K^m \setminus \{0\}$ is $q - 1$ to 1. This determines the number of lines. □

Over the binary field, there is a one-to-one correspondence between lines and the non-zero vectors, since each line consists of a zero vector and a non-zero vector. This diagram (called the Fano plane) shows linearly dependencies between non-zero vectors of K^3 for $K = \mathbb{Z}_2$. Every triple of vectors that is connected in a common segment or the circle curve is linearly dependent over $\mathbb{Z}_2$:



Def 5.10. If $h_1, \ldots, h_n$ are non-zero vectors that span all n lines of K^m, where $m \geq 2$, then the linear code

$$C = \{(c_1, \ldots, c_n) \in K^n \,:\, c_1 h_1 + \cdots + c_n h_n = 0\}$$

is called a hamming code.

Prop 5.11. *The parity-check matrix of the hamming code in the above definition is* $H = (h_1 \cdots h_n) \in K^{m\times n}$. *It is a perfect*$[n, n-m, 3]$ *code with* $n = \frac{q^m-1}{q-1}$ *and* $q = |K|$.

Proof. It is clear that C is the kernel of H. So, for H to be a parity-check matrix it remains to check that it has a full row rank, that is it should have the rank m. This is true since multiples of all unit vectors occur as columns of H. The dimension $n-m$ of this code is determined by the rank-nullity theorem. Now two columns of H are linearly dependent, since they span different lines (by construction). But there are three linearly independent columns, say, the multiple of e_1, the multiple of e_2 and the multiple of $e_1 + e_2$.

It remains to show that the code is perfect. We need to calculate the total number of elements of the balls of radius 1 with centers in C. This number is

$$|C|(1 + n(q-1)) = q^{n-m}(1 + n(q-1)) = q^{n-m}(1 + q^m - 1) = q^n$$

and thus the total number of elements in K^n, which confirms that the code is indeed perfect. □

Rem 5.12. As a corollary we see that the hamming code can correct one error and each word received has a unique closest word in C.

5.5 Evaluation codes

5.5.1 Univariate polynomial interpolation

Def 5.13. Let K be a field, and x a symbol, then $K[x]$ is the set of all formal expressions $f = \sum_i c_i x^i$, where the sum is over $i \in \mathbb{N}_0$, $c_i \in K$ and $c_i \neq 0$ only for finitely many choices of i. Such f is called a polynomial, $c_i x^i$ is called the term of degree i with the coefficient c_i and the monomial x^i. R is viewed as a subset of $K[x]$. The largest i such that $c_i \neq 0$ is called the degree of f, denoted by $\deg(f)$. One defines $\deg(0) = -\infty$. For $f \neq 0$, the coefficient $c_{\deg(f)}$ is called the leading coefficient and if it is equal to 1, then f is called monic. On $K[x]$ one defines addition and multiplication in the obvious way:

$$\sum_i c_i x^i + \sum_i d_i x^i := \sum_i (c_i + d_i) x^i$$

$$\left(\sum_i c_i x^i\right)\left(\sum_j d_j x^j\right) := \sum_{i,j} c_i d_j x^{i+j}$$

By $K[x]_d$ we denote the set of all elements of $K[x]$ that have degree at most d. Polynomials can be evaluated by substituting a value for x, by $f(r) \in K$ we denote the result of the substitution $x = r$ into $f \in K[x]$. If $f(r) = 0$, then we say that $r \in K$ is a zero or a root of $f \in K[x]$.

Prop 5.14. *If* K *is a field and* x *a symbol, then* $K[x]$ *is a commutative unitary ring and an infinite-dimensional vector space over* K. *Furthermore,* $K[x]_d$ *with* $d \in \mathbb{N}_0$ *is a vector space over* K *of dimension* $d+1$.

Prop 5.15. *Let* K *be a field and let* $A = \{a_1, \ldots, a_k\}$ *be a* k*-element subset of* K *with* $k \in \mathbb{N}$ *and* $k \leq |K|$. *Then the evaluation ma* $\mathrm{ev}_A : K[x]_{k-1} \to K^k$ *with* $\mathrm{ev}_A(f) := (f(a_1), \ldots, f(a_k))$ *is a linear bijection.*

Proof. The map is linear, since evaluation of a polynomial f in a given point $a \in K$ is linear in f. Let's show that the map is surjective by showing that each standard unit vector $e_i \in K^k$ is in the image. Indeed, the polynomial

$$f_i := \prod_{\substack{j=1,\dots,k \\ j \neq i}} \frac{x - a_j}{a_i - a_j}$$

of degree $k-1$ is equal to zero on each a_j with $j \neq 1$ and is equal to 1 on a_i. Thus, $\mathrm{ev}_A(f_i) = e_i$ and an arbitrary $y = (y_1, \dots, y_{k-1}) \in K_{k-1}$ satisfies $y = \mathrm{ev}_A(y_1 f_1 + \cdots + y_k f_k)$. Thus, the map ev_A is surjective. But then, by the rank-nulity theorem it is also injective, since

$$\dim \ker(\mathrm{ev}_A) := \dim(K^k) - \dim \mathrm{ev}_A(K[x]_{k-1}) = \dim(K^k) - \dim(K^k) = k - k = 0.$$

□

Rem 5.16. The polynomials $f_1, \dots, f_k$ from the above proof are called the Lagrange interpolating polynomials. With this polynomials the task of finding a polynomial $f \in K[x]_{k-1}$ that satisfies $f(a_i) = y_i$ for the so-called interpolation nodes $(a_i, y_i) \in K^2$ with $i = 1, \dots, k$ can be solved. In numerical mathematics, the usual choice of the underlying field for the interpolation task is $K = \mathbb{R}$, but the interpolation can also be carried out in arbitrary fields, including finite fields.

5.5.2 Reed-Solomon codes

Thm 5.17. *Let K be a field and let k, n be natural numbers with Let $1 \leq k \leq n \leq |K|$. Then, for every subset A of K with n elements, the vector space* $\dim \mathrm{ev}_A(K[x]_{k-1}) = k$.

Proof. We show that $\ker(\mathrm{ev}_A) = \{0\}$. Let B be a subset of A with exactly n elements. If $\mathrm{ev}_A(f) = 0$, then $\mathrm{ev}_B(f) = 0$. But, we have shown that ev_B is bijective and so the latter implies that $f = 0$. This shows that $\dim \mathrm{ev}_A(K[x]_{k-1}) = \dim(K[x]_{k-1}) = k$. □

Thm 5.18. *Let K be a finite field, let $q := |K|$. Consider integers $1 \leq k \leq n \leq q$ and fix an n-element set $A \subseteq K$. Then code $C = \{\mathrm{ev}_A(f) : f \in K[x]_{k-1}\}$ is a linear code with the parameters $[n, k, n-k+1]$.*

Proof. $C \subseteq K^n$ and $\dim(C) = k$ has been shown. Consider an arbitrary non-zero code word $\mathrm{ev}_A(f)$ with $f \in K[x]_{k-1}$. If f is equal to zero on at least k elements of A, we can fix a k-element set $B \subseteq A$ with $\mathrm{ev}_B(f) = 0$. But sine $\mathrm{ev}_B : K[x]_{k-1} \to K^k$ is a bijection, we conclude that $f = 0$, which is a contradiction to $\mathrm{ev}_A(f) \neq 0$. Hence, $\mathrm{ev}_A(f)$ has at most $k-1$ components equal to zero and so the weight of $\mathrm{ev}_A(f)$ is at least $n-k+1$. This shows $d(C) \geq n-k+1$. The bound is attained by taking $k-1$ elements $a_1, \dots, a_{k-1}$ and considering the polynomial $f = (x-a_1)\cdots(x-a_{k-1})$, which yields the code word $\mathrm{ev}_A(f)$ of weight $n-k+1$. □

Def 5.19. The code of the previous theorem is called a $[n, k, n-k+1]$-Reed-Solomon code.

We would like to reprove the singleton bound for linear codes. Even though singleton bound applies to arbitrary codes, it is instructive to see its proof for linear codes and compare the proof with the construction of the Reed-Solomon codes.

Thm 5.20 (Singleton bound reproved for linear codes)**.** *Let K be a finite field and C a k-dimensional vector subspace of K^n, where $1 \leq k \leq n$. Then $d(C) \leq n - k + 1$.*

Proof. One can describe C as an image of a linear map $f : K^k \to K^n$. For example, one can pick a basis $b_1, \ldots, b_k$ of C and define $f(a_1, \ldots, a_k) = a_1 b_1 + \cdots + a_k b_k$. Let $d = d(C)$. We introduce the projection $\pi : K^n \to K^{n-d+1}$ by $\pi(y_1, \ldots, y_n) := (y_1, \ldots, y_{n-d+1})$. The map $\pi \circ f$ is injective. Indeed if $a \in K^k \setminus \{0\}$, then $f(a) \neq 0$ and so as an element of C it has at least d non-zero components. Thus, $f(a)$ has at most $n - d$ non-zero components. Thus, one of the components of $\pi(f(a))$ is non-zero. Since $\pi \circ f : K^k \to K^{n-d+1}$ is injective, we have $k = \dim(K^k) \leq \dim(K^{n-d+1}) = n - d + 1$, yielding the desired bound. □

Rem 5.21. Note that Reed-Solomon codes are MDS codes, because their satisfy the singleton bound with equality.

5.5.3 Multivariate polynomial interpolation on a hypercube

Def 5.22. Let K be a field and let $x = (x_1, \ldots, x_n)$ be a list of n symbols For $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}_0^n$ we call $x^\alpha := x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ a multivariate monomial of degree $|\alpha| := \alpha_1 + \cdots + \alpha_n$, and α is called the exponent vector of this monomial. By $K[x_1, \ldots, x_n]$ we denote the set of formal expressions of the form

$$f = \sum_\alpha c_\alpha x^\alpha,$$

the sum is taken over $\alpha \in \mathbb{N}_0^n$ and $c_\alpha \neq 0$ for all but finitely many α's. Elements of $K[x_1, \ldots, x_n]$ are called multivariate polynomials in variables $x_1, \ldots, x_n$. Equality of multivariate polynomials is defined by comparison of coefficients. The sum and the product of multivariate polynomials are introduced in a standard way. The notions like coefficent, term and the monomial are defined analogously to the univariate case. The degree of a non-zero polynomial is the largest degree of a monomial occuring in the polynomial with a non-zero coefficient. The degree of the zero polynomial is declared to be $-\infty$. A polynomial f is said to be multilinear if f the exponent vectors of all monomials that occur in f with a non-zero coefficient belong to $\{0, 1\}^n$ (a more appropriate name would be multi-affine, but 'multilinear' is a common terminology). Polynomials f can be evaluated on points K^n by substituting $x_1 = a_1, \ldots, x_n = a_n$ for $a = (a_1, \ldots, a_n) \in K^n$. The result of the evaluation is denoted by $f(a)$.

Prop 5.23. *$K[x_1, \ldots, x_n]$ is a ring.*

Thm 5.24. *Let K be a field, let $A := \{0_K, 1_K\}^m \subseteq K^A$ with $n = 2^m$ and let V be the K-vector space of all multilinear polynomials in $K[x_1, \ldots, x_m]$. then $\mathrm{ev}_A : V \to K^n$ given by $\mathrm{ev}_A(f) = (f(a))_{a \in A}$ is a linear bijection.*

Proof. Consider the $f_0(t) = 1 - t \in K[t]$ and $f_1(t) = t \in K[t]$. Note that for $u, v \in \{0, 1\}$ one has $f_u(v) \in \{0, 1\}$ with the equality $f_u(v) = 1$ if and only if $u = v$.

If $p = (p_1, \ldots, p_n) \in A$, we define $f_p := f_{p_1}(x_1) \cdots f_{p_n}(x_n) \in V$. By construciton $f_p(q) \in \{0, 1\}$ for all $q \in A$ with the quality $f_p(q) = 1$ if and only if $p = q$. It follows for every $y = (y_a)_{a \in A} \in K^A$, we can write every we have $\mathrm{ev}_A(f) = y$, when

$$f = \sum_{a \in A} y_a f_a.$$

Thus, ev_A is surjective. Since $\dim(K^A) = \dim(V) = 2^m$, it is also bijective. □

5.5.4 Reed-Muller code

Thm 5.25. *Consider the $A = \mathbb{Z}_2^m$, let V be the vector space of all multlinear polynomials polynomials in $\mathbb{Z}_2[x_1, \ldots, x_m]$ and let V_r the subspace of V consisting of polynomials of degree at most r, where $1 \leq r \leq m$. Then $C = \{\mathrm{ev}_A(f) : f \in V_r\}$ is a linear code with the parameters $[2^m, \sum_{j=0}^{r} \binom{m}{j}, 2^{m-r}]$.*

Proof. We have seen that the evaluation map is bijective on V, so it is injective on $V_r \subseteq V$. It is clear that $\dim(V_r) = \sum_{j=0}^{r} \binom{m}{j}$. Thus, C being the image of V_r under the evaluation map has the same dimension as V_r. It remains to determine $d = d(C)$. The evaluation of the polynomial $x_1, \ldots, x_r \in V_r$ on the 2^{m-r} points of the form $(1, \ldots, 1, p_{r+1}, \ldots, p_m)$ gives 1. So, $d \geq 2^{m-r}$. We show that every polynomial $f \in V_r \setminus \{0\}$ is non-zero on at least 2^{m-r} points of A. We use induction on m. The assertion is trivial for $m = 0$. In general, we write $f = g + hx_m$ with multilinear $g, h \in K[x_1, \ldots, x_{m-1}]$. Note that $\deg g \leq r$ and $\deg h \leq r - 1$.

- If $g \neq 0 = h$, then the induction hypothesis yields $\mathrm{wt}(g) \geq 2^{(m-1)-r} = 2^{m-r-l}$. But then, since x_m can be evaluated at 0 and 1 with no influence on the value of f, we have $\mathrm{wt}(f) = 2\,\mathrm{wt}(g)$.
- If $g = 0 \neq h$, then $\mathrm{wt}(f) = \mathrm{wt}(h)$. The induction hypothesis yields $\mathrm{wt}(h) \geq 2^{(m-1)-(r-1)} = 2^{m-1}$.
- If $g \neq 0 \neq h$ and $g = h$, then evaluation of x_m at 0, gives $\mathrm{wt}(f) \geq \mathrm{wt}(g)$ and the induction hypothesis applied to g (which has $\deg(g) = \deg(h) \leq r - 1$) yields $\mathrm{wt}(g) \geq 2^{(m-1)-(r-1)} = 2^{m-r}$.
- If $g \neq 0 \neq h$ and $g \neq h$, the the substitutions $x_m = 0$ and $x_m = 1$, give g and $g + h$ respectively. The induction hypothesis gives $\mathrm{wt}(g) \geq 2^{(m-1)-r}$ and $\mathrm{wt}(g+h) \geq 2^{(m-1)-r}$. Hence $\mathrm{wt}(f) = \mathrm{wt}(g) + \mathrm{wt}(h) \geq 2^{m-r}$.

□

5.6 General finite fields

5.6.1 Rings of univariate polynomials

Prop 5.26. *Let K be a field and $g \in K[x] \setminus \{0\}$. Then for every $f \in K[x]$, there exists a unique decomposition $f = qg + r$ with $q, r \in K[x]$ and $\deg(r) < \deg(g)$. These q and r are called the quotient and the remainder, respectively, of the division of f by g.*

Def 5.27. In a ring R, an element $f \in R$ is called divisible by $g \in R \setminus \{0\}$ if $f = qg$ for some $q \in R$.

Def 5.28. An element of a ring $a \in R$ is said to be invertible or a unit if it has an element $b \in R$ satisfying $ab = 1$. One uses the notation a^{-1} to denote this element b (it is a unique element).

Exercise 5.29. The only units of $\mathbb{Z}$ are 1 and -1.

Prop 5.30. *If K is a field and x a symbol, then the set of all invertible elements of $K[t]$ coincides with $K \setminus \{0\}$ (the set of all invertible elements of K).*

Def 5.31. An element a of a ring R is called irreducible if it is not a unit and it is not a product of two elements that are both not units.

Exercise 5.32. The irreducible elements of $\mathbb{Z}$ are of the form $\pm p$, where p is a prime number.

Prop 5.33. *Let K be a field. Then $a \in K$ is a zero of $f \in K[x]$ if and only if f is divisible by $x - a$.*

Prop 5.34. *Let K be a field and x a symbol. Then every $f \in K[x] \setminus \{0\}$ is a product of finitely many irreducible elements. In particular, f can be factorized as $f = cf_1 \cdots f_t$, where $c \in K \setminus \{0\}$, $f_1, \ldots, f_k$ are irreducible and monic and this factorization is unique up to the order of the factors $f_1, \ldots, f_k$.*

Prop 5.35. *Let K be a field and x a symbol. A polynomial $f \in K[x] \setminus \{0\}$ has at most $\deg(f)$ zeros in K.*

5.6.2 Finite fields from irreducible polynomials

Prop 5.36. *Let K be a field and $g \in R := K[x] \setminus \{0\}$. Then R/gR is a vector space over K of dimension $\deg(g)$. In particuar, if K is finite, then R is a finite ring with $|K|^{\deg(g)}$ elements.*

Proof. Let α be the coset of x modulo gR and let $d := \deg(g)$. It is straightforward to see that $\alpha^0, \ldots, \alpha^{d-1}$ form a basis of R/gR. □

Def 5.37. An non-zero element a of a commutative unitary ring R is called a zero divisor if $ab = 0$ holds for some non-zero element b of R.

Prop 5.38. *If K is a finite commutative ring without zero divisors, then K is a field.*

Proof. It suffices to show that every $a \in K \setminus \{0\}$ is invertible. The map $f(x) = ax$ is well-defined as the map $R \setminus \{0\} \to \mathbb{R} \setminus \{0\}$. Indeed if $x \in \mathbb{R} \setminus \{0\}$, then $ax \neq 0$ since $a \neq 0 \neq x$ and K has no zero divisors. If $x, x' \in K$ and $x \neq x'$, then $ax \neq ax'$, since $a(x - x') \neq 0$ as K has no zero divisors. We conclude that f is injective. But since $R \setminus \{0\}$ is finite, it is bijective. Hence, one has $ax = 1_R$ for some $x \in R$. □

Prop 5.39. *If K is a finite field and g is an irreducible polynomial in $R := K[x]$, then R/gR is a field.*

Proof. R/gR is a commutative unitary ring and it is finite, since it has exactly $|K|^{\deg(g)}$ elements. It suffices to show that R/gR has no zero divisors. Consider two polynomials $f, h \in K[x] \setminus \{0\}$ such that their cosets $[f], [g]$ modulo g satisfy $[f][g] = [0]$. That means, fg is divisible by g. That means, $fg = gq$ for some $q \in K[x]$. From the unique factorization theorem and the fact that g is irreduicible, we conclude that g is factor of f or g. That is, $[f] = [0]$ or $[g] = [0]$ holds. This shows that R/gR has no zero divisors. Consequently, R/gR is a field. □

Rem 5.40. The previous proposition holds also without the finiteness assumption on the field K.

5.6.3 Fundamental theorem of finite fields

Def 5.41. Fields K and L are said to be isomorphic if there exists a bijective map $f : K \to L$ satisfying $f(1_K) = 1_L$ and $f(x+y) = f(x)+f(y)$, $f(x \cdot y) = f(x) \cdot f(y)$ for all $x, y \in K$. Such an f is called an isomorphism of the fields K and L.

Prop 5.42. *For every finite field K there exists $t \in \mathbb{N}$ such that $\sum_{i=1}^{t} 1_K = 0_K$. The minimal such t is a prime number and is called the characteristic of K.*

Proof. The map $t \mapsto \sum_{i=1}^{t} 1_K = 0_K$ acts from the infinite set $\mathbb{N}$ to the finite set K. Hence, $\sum_{i=1}^{a} 1_K = \sum_{i=1}^{b} 1_K$ holds for some $a, b \in \mathbb{N}$ with $a < b$. Consequently, $\sum_{i=1}^{t} 1_K = 0_K$ for $t = b - a$. Now, consider the minimal $t \in \mathbb{N}$ with $\sum_{i=1}^{t} 1_K = 0_K$. It is clear that $t \neq 1$, since $1_K \neq 0_K$. If t were not prime, we had $t = mn$ with $m, n \in \mathbb{N}$ and $m, n \geq 2$. But then $0_K = \sum_{i=1}^{t} 1_K = \sum_{i=1}^{mn} 1_K = \sum_{i=1}^{mn} 1_K = \sum_{i=1}^{m} \sum_{j=1}^{n} 1_K = (\sum_{i=1}^{m} 1_K)(\sum_{j=1}^{n} 1_K)$. So, one of the two factors $\sum_{i=1}^{m} 1_K$ or $\sum_{j=1}^{n} 1_K$ is 0_K. Since $m, n < t$, we obtain a contradiction to the minimality of t. Hence, t is prime. □

Thm 5.43.

(a) For every prime p and every $n \in \mathbb{N}$ there exists a finite field of the form R/gR, where $R = \mathbb{Z}_p[x]$ and $g \in \mathbb{Z}_p[x]$ is irreducible of degree n; such a field has p^n elements.

(b) Every finite field K has p^n elements, where $n \in \mathbb{N}$ and p is the characteristic of K.

(c) If K and L are finite fields with $|K| = |L|$, then K and L are isomorphic.

5.6.4 Finite fields in applications

Fields of characteristic two are quite popular in applications since they fit into the model of storing data as a sequence of bits. For example, one can use a byte (i.e., the sequence of 8 bits) to encode element of $\mathbb{F}_{2^8}$. Note however that the modular arithmetic on bites is the arithmetic of $\mathbb{Z}_{2^8}$, which is a ring but not a field.

UNDER CONSTRUCTION,

see [5] Let $\mathrm{RS}_q(n, k)$ denote a Reed-Solomon code with the parameters $[n, k, n-k+1]_q$. QR codes are a family of bar codes, that involve $\mathrm{RS}_{2^8}(26, k)$. The choice of the dimension k can be $k \in \{19, 16, 13, 9\}$, these choices give different levels of error correction capability and are represented by the labels L (low), M (medium), Q (quartile), H (high).

An example of a QR-Code structure: $n = 26$, $k = 19$ and $q = 2^8$. The code consists of 17 data bytes + 7 error correction bytes, Enc and End field that in total occupy one byte and the message length byte.

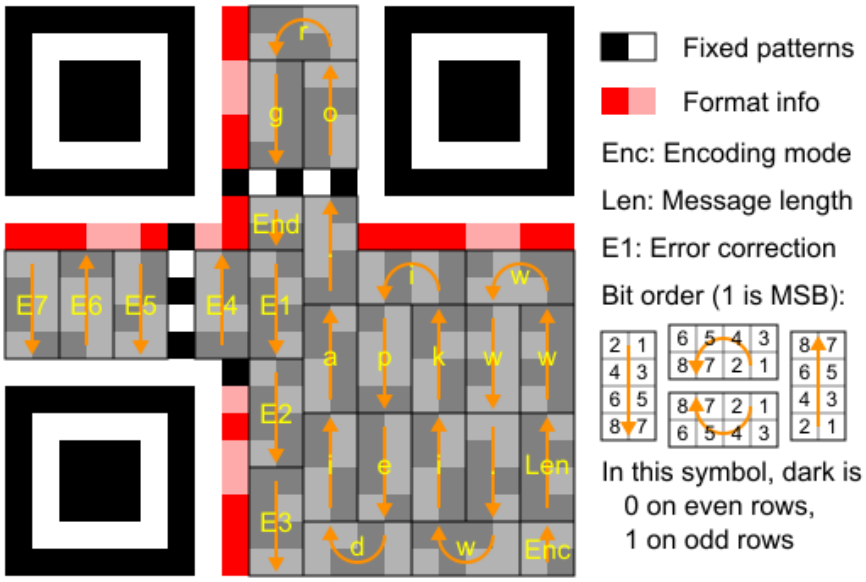


Image from Wikipedia, see [6].

The distance of this code is $n-k+1=8$. That means, the code can detect up to 7 erroneous bytes and correct up to 3 erroneous bytes.

5.7 LDPC codes

LDPC is an abbreviation for low density parity check. We are going to discuss codes over $\mathbb{Z}_2$ that are based on parity checks involving a small number of variables only.

Consider a linear code $C=\{x\in\mathbb{Z}_2^n : Hx=0\}$ given by a matrix $H\in\mathbb{Z}_2^{m\times n}$. There is a natural correspondence between matrices H over $\mathbb{Z}_2$ and bipartite graphs, whose two parts corresponds to the variables and the equations of the system $Hx=0$, respectively, with a variable and an equation connected by an edge if and only if the variable is contained in the equation.

A bipartite graph with two parts, which we call the left and right part, is said to be (l,r) regular if the degree of each node of the left part is l and the degree of each node of the right part is r. In terms of the system $Hx=0$, that means that each variable is contained in exactly l equations and each equation contains exactly r variables. by double counting, we have $nl=mr$, which allows to compute the rate of such a code.

Prop 5.44. *Let $C=\{x\in\mathbb{Z}_2^n : Hx=0\}\subseteq\mathbb{Z}_2^n$ be such the system $Hx=0$ given by the matrix $H\in\mathbb{Z}_2^{m\times n}$, with $m<n$, has the property that each variable is contained in exactly l equations and each equation contains exactly m variables. Then $R(C)\geq 1-\frac{r}{l}$.*

Proof. As noted above $ml=nr$. One has $\dim(C)\geq n-m$, since each equation potentially removes one dimension. Then $R(C)=\dim(C)/n=1-\frac{m}{n}=1-\frac{r}{l}$. □

Def 5.45. Let $0<\alpha<1$, $\delta>1$ and $n\in\mathbb{N}$ and let $\alpha n\geq 1$. An (l,r)-regular bipartite graph, that has n nodes in the left part, is said to be (n,l,r,α,δ) expander if for every subset U of nodes from the left part satisfying $0<|U|\leq\alpha n$, the set ∂U of nodes of the right part

that are connected with some node in U by an edge satisfies the estimate $|\partial U| > \delta|U|$. An $(n, l, r, \alpha, \delta)$-expander code is a code of the form $C = \{x \in \mathbb{Z}_2^n : Hx = 0\}$, where the matrix H corresponds to an $(n, l, r, \alpha, \delta)$-expander graph. The value δ is called the expansion factor.

Rem 5.46. Since the constraint αn is an upper bound, the constant α can be lowered so that the respective change αn rounding down. Such a change does not affect the expansion property. Note that the expansion factor δ is at most l, the degree of nodes of th left part, which can be seen by taking $|U| = 1$.

Prop 5.47. *Every $(n, l, r, \alpha, l/2)$-expander code C satisfies $d(C) > \alpha n$. In particular, C can correct up to $\frac{\alpha}{2}n$ errors.*

Proof. Consider the respective expander graph that describes which variables are contained in which equations. If the assertion were wrong, we had a word $c \in C \setminus \{0\}$ with $\mathrm{wt}(c) \leq \alpha n$. Let $c = (c_1, \ldots, c_n)$ and U be the set of variables x_i with with $c_i = 1$. Since $|U| \leq \alpha n$, the expansion property gives $|\partial U| > \frac{l}{2}|U|$. That is, the variables set to 1 are contained in at least $\frac{l}{2}|U|$ equations. Each equation that contains a variable set to 1, must contain at least two such variables, otherwise the equation would not be valid. Each variable in U is present in l copies in equations, so that the variables in U occur $l|U|$ times in total in the equations from ∂U. As each equation contains at least two variables in U, there are at most $\frac{l}{2}|U|$ equations. We have obtained $\frac{l}{2}|U| \geq |\partial U| > \frac{l}{2}|U|$, which is a contradiction.

The statement regarding the correction of errors is clear. □

Prop 5.48. *For every $(n, l, r, \alpha, 3l/4)$-expander code C and every $v \in \mathbb{Z}_2^n$ that satisfies $d(v, c)$ for some $c \in C$ with $d(c, v) \leq \frac{1}{2}(\alpha n - 1)$, the code c can be determined from v and the expander graph that defines C by means of $O((ln)^2)$ operations over $\mathbb{Z}_2$.*

Proof. Let $v = (v_1, \ldots, v_n)$ and $c = (c_1, \ldots, c_n)$. Consdier the matrix $H \in \mathbb{Z}_2^{m \times n}$ with the sparsity structure of the expander graph, whose kernel is C. Let $H_{i,j}$ be the entries of H and let $(Hx)_i = 0$ be the i-th equation of the system $Hx = 0$. Consider

$$\begin{aligned} T &:= \{j = 1, \ldots, n : v_j \neq c_j\}, & t &:= |T|, \\ S &:= \{i = 1, \ldots, m : (Hv)_i = 1\}, & s &:= |S|, \\ K &:= \{i = 1, \ldots, m : (Hv)_i = 0 \text{ and } H_{i,j} = 1 \text{ for some } j \in T\}, & k &:= |K|. \end{aligned}$$

We assume that $t > 0$ as otherwise $v = c \in C$, and there is nothing to be done.

The set T indexes the variables at which error occurred. the set S indexes equations that are not satisfied by v, and the set K indexes the equations that are satisfied but that contain variables that are set incorrectly. Note that the above sets and their cardinalities depend on v and will change during the iterative process that is described below. The are lt copies of variables index by T that occur in equations. There is at least one such copy occurring in each equation indexed by S, and at least two copies occurring in the equations indexed by K. Hence $lt \geq 2k + s$. Whenever $t = |T| < \alpha n$, the number of equations involving variables in T is at least $\frac{3l}{4}|T| = \frac{3}{4}lt$ by the expansion property, which yields $s + k > \frac{3}{4}lt$ (since $S \cup K$ indexes all the equations that involve variables indexed by T).

Under the assumption $t < \alpha n$, we thus have $s \leq lt - 2k \leq lt < \alpha ln$ and we also have $lt - s \geq 2k > \frac{3}{2}lt - 2s$, which yields $s > \frac{1}{2}lt$. For $j \in T$, let $S_j \subseteq S$ be the indices of the equations that are not valid on v and that contain the j-th variable. One has $t \max_{j=1,\ldots,t} |S_j| \geq |S| = s > \frac{1}{2}lt$,

which means that some variable indexed by j occurs in more invalid than valid inequalities. Now, when we want to correct v to obtain c, we don't have access to T, because we don't know c, so what we can do is just flipping any v_j for the variable x_j that occurs in more incorrect than the correct inequalities. After this change s gets smaller, because more equations are made valid than those that are made invalid. The value t however may get smaller or larger by one. For the termination of the process we are about to develop, we want to see that t never gets beyond a certain threshold.

We start the iteration with $t \leq \frac{1}{2}(\alpha n - 1)$ which implies that $s \leq lt \leq \frac{l}{2}(\alpha n - 1)$. These are our initial bounds on t and s. When we do a change described above, the bound on s remains valid as s gets smaller.

Whenever $1 \leq t < \alpha n$, we iterate as described above making s smaller by each iteration. A priori, there are two possibilities that the iterative process stops: either $t = 0$, which implies that $v = c \in C$ or $t \geq \alpha n$. We rule out the second possibility arguing by contradiction. It would mean that we reach the iteration in which $t < \alpha n$ and after which t would be increased by one giving the value $t + 1 \geq \alpha n$.

We obtain

$$\frac{l}{2}(\alpha n - 1) \geq s > \frac{1}{2}lt \geq \frac{l}{2}(\alpha n - 1),$$

which gives a contradiction. Hence, during the iteration terminates with $t = 0$, yielding $v = c$.

If the matrix H is given through the bipartite graph, the size of the graph is of order nl, which is the number of edges of the graph. During the calculation we traverse the graph and so an iteration takes $O(nl)$ operations in $\mathbb{Z}_2$. The initial bound on s is of order $O(ln)$ as well and with each iteration, it gets smaller. So we end up with $O((ln)^2)$ operations over $\mathbb{Z}_2$. □

5.8 Cyclic codes

Use [3], Ch. 6?

Def 5.49. A linear code $C \subseteq K^n$ is called cyclic if for every $c = (c_0, \ldots, c_{n-1}) \in C$, the cyclic shift $(c_1, \ldots, c_{n-1}, c_0)$ also belongs to C.

Thm 5.50. *Every ideal I in $R := K[x]$ has the form $I = gR$.*

Proof. If $I = \{0\}$ this is clear. If $I \neq \{0\}$, pick a non-zero polynomial in I of the minimum degree. We have $gR \subseteq I$ by construction. To show the converse, consider $f \in I$ and divide f by g with rest obtaining $g = qf + r$ with $q, f \in R$ and $\deg(r) < \deg(g)$. Since $g, f \in I$, one also has $r \in I$. If r were not zero, then we had $0 \leq \deg(r) < \deg(g)$ contradicting the choice of g. Hence $r = 0$, which shows $f \in gR$ and concludes the proof of the inclusion $I \subseteq gR$. □

5.8.1 Cyclic codes and ideals

References

[1] Alon and Spencer: The probabilistic method

[2] Cormen, Leiserson, Rivest, Stein.

[3] Lint

[4] Willems

[5] `http://simoneparisotto.com/math/misc/qrcode/qrcode.pdf`

[6] `https://en.wikipedia.org/wiki/QR_code`