Foreword

This chapter is based on lecture notes from coding theory courses taught by Venkatesan Guruswami at University at Washington and CMU; by Atri Rudra at University at Buffalo, SUNY and by Madhu Sudan at MIT.

This version is dated **February 21, 2017**. For the latest version, please go to

http://www.cse.buffalo.edu/ atri/courses/coding-theory/book/

The material in this chapter is supported in part by the National Science Foundation under CAREER grant CCF-0844796. Any opinions, findings and conclusions or recomendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

Chapter 2

A Look at Some Nicely Behaved Codes: Linear Codes

Let us now pause for a bit and think about how we can represent a code. In general, a code $C : [q]^k \longrightarrow [q]^n$ can be stored using nq^k symbols from $[q]$ (n symbols for each of the q^k codewords) or $nq^k \log q$ bits. For constant rate codes, this is exponential space, which is prohibitive even for modest values of k like $k = 100$. A natural question is whether we can do better. Intuitively, the code must have some extra structure that would facilitate a succinct representation of the code. We will now look at a class of codes called *linear codes* that have more structure than general codes which leads to some other nice properties. We have already seen binary linear codes in Section 1.5, that is, $C \subseteq \{0,1\}^n$ is linear code if for all $\mathbf{c}_1, \mathbf{c}_2 \in C$, $\mathbf{c}_1 + \mathbf{c}_2 \in C$, where the "+" denotes bit-wise EXOR.

Definition 2.0.1 (Linear Codes)**.** Let q be a prime power (i.e. $q = p^s$ for some prime p and integer $s \geq 1$). $C \subseteq \{0,1,\ldots,q-1\}^n$ is a *linear code* if it is a *linear subspace* of $\{0,1,\ldots,q-1\}^n$. If C has dimension k and distance d then it will be referred to as an $[n,k,d]_q$ or just an $[n,k]_q$ code.

Of course the above definition is not complete because we have not defined a linear subspace yet. We do that next.

2.1 Finite Fields

To define linear subspaces, we will need to work with (finite) fields. At a high level we need finite fields as when we talk about codes, we deal with finite symbols/numbers and we want to endow these symbols with the same math that makes arithmetic over reals work. Finite fields accomplish this precise task. We begin with a quick overview of fields.

Informally speaking, a field is a set of elements on which one can do addition, subtraction, multiplication and division and still stay in the set. More formally,

Definition 2.1.1. A field $\mathbb{F}$ is given by a triple $(S, +, \cdot)$, where S is the set of elements containing special elements 0 and 1 and $+, \cdot$ are functions $\mathbb{F} \times \mathbb{F} \rightarrow \mathbb{F}$ with the following properties:

- Closure: For every $a, b \in S$, we have both $a + b \in S$ and $a \cdot b \in S$.
- Associativity: $+$ and $\cdot$ are associative, that is, for every $a, b, c \in S$, $a+(b+c) = (a+b)+c$ and $a \cdot (b \cdot c) = (a \cdot b) \cdot c$.
- Commutativity: $+$ and $\cdot$ are commutative, that is, for every $a, b \in S$, $a + b = b + a$ and $a \cdot b = b \cdot a$.
- Distributivity: $\cdot$ distributes over $+$, that is for every $a, b, c \in S$, $a \cdot (b + c) = a \cdot b + a \cdot c$.
- Identities: For every $a \in S$, $a + 0 = a$ and $a \cdot 1 = a$.
- Inverses: For every $a \in S$, there exists its unique *additive inverse* $-a$ such that $a + (-a) = 0$. Also for every $a \in S \setminus \{0\}$, there exists its unique *multiplicative inverse* a^{-1} such that $a \cdot a^{-1} = 1$.

With the usual semantics for $+$ and $\cdot$, $\mathbb{R}$ (set of real number) is a field but $\mathbb{Z}$ (set of integers) is not a field as division of two integers can give rise to a rational number (the set of rational numbers itself is a field though– see Exercise 2.1). In this course, we will exclusively deal with *finite fields*. As the name suggests these are fields with a finite size set of elements. (We will overload notation and denote the size of a field $|\mathbb{F}| = |S|$.) The following is a well known result.

Theorem 2.1.1 (Size of Finite Fields)**.** *The size of any finite field is p^s for prime p and integer $s \geq 1$.*

One example of finite fields that we have seen is the field of two elements $\{0, 1\}$, which we will denote by $\mathbb{F}_2$ (we have seen this field in the context of binary linear codes). For $\mathbb{F}_2$, addition is the XOR operation, while multiplication is the AND operation. The additive inverse of an element in $\mathbb{F}_2$ is the number itself while the multiplicative inverse of 1 is 1 itself.

Let p be a prime number. Then the integers modulo p form a field, denoted by $\mathbb{F}_p$ (and also by $\mathbb{Z}_p$), where the addition and multiplication are carried out $\mod p$. For example, consider $\mathbb{F}_7$, where the elements are $\{0, 1, 2, 3, 4, 5, 6\}$. So we have $4 + 3 \mod 7 = 0$ and $4 \cdot 4 \mod 7 = 2$. Further, the additive inverse of 4 is 3 as $3 + 4 \mod 7 = 0$ and the multiplicative inverse of 4 is 2 as $4 \cdot 2 \mod 7 = 1$.

More formally, we prove the following result.

Lemma 2.1.2. *Let p be a prime. Then $\mathbb{F}_p = (\{0, 1, \ldots, p-1\}, +_p, \cdot_p)$ is a field, where $+_p$ and $\cdot_p$ are addition and multiplication $\mod p$.*

Proof. The properties of associativity, commutativity, distributivity and identities hold for integers and hence, they hold for $\mathbb{F}_p$. The closure property follows since both the “addition" and “multiplication" are done $\mod p$, which implies that for any $a, b \in \{0, \ldots, p-1\}$, $a +_p b, a \cdot_p b \in \{0, \ldots, p-1\}$. Thus, to complete the proof, we need to prove the existence of unique additive and multiplicative inverses.

Fix an arbitrary $a \in \{0, \ldots, p-1\}$. Then we claim that its additive inverse is $p - a \mod p$. It is easy to check that $a + p - a = 0 \mod p$. Next we argue that this is the unique additive inverse.

To see this note that the sequence $a, a+1, a+2, \ldots, a+p-1$ are p consecutive numbers and thus, exactly one of them is a multiple of p, which happens for $b = p - a \mod p$, as desired.

Now fix an $a \in \{1, \ldots, p-1\}$. Next we argue for the existence of a unique multiplicative universe a^{-1}. Consider the set of numbers $\{a \cdot_p b\}_{b \in \{1,\ldots,p-1\}}$. We claim that all these numbers are unique. To see this, note that if this is not the case, then there exist $b_1 \neq b_2 \in \{0, 1, \ldots, p-1\}$ such that $a \cdot b_1 = a \cdot b_2 \mod p$, which in turn implies that $a \cdot (b_1 - b_2) = 0 \mod p$. Since a and $b_1 - b_2$ are non-zero numbers, this implies that p divides $a \cdot (b_1 - b_2)$. Further, since a and $|b_1 - b_2|$ are both at most $p-1$, this implies that factors of a and $(b_1 - b_2) \mod p$ when multiplied together results in p, which is a contradiction since p is prime. Thus, this implies that there exists a unique element b such that $a \cdot b = 1 \mod p$ and thus, b is the required a^{-1}. □

One might think that there could be different fields with the same number of elements. However, this is not the case:

Theorem 2.1.3. *For every prime power q there is a* unique *finite field with q elements (up to isomorphism*[1]*).*

Thus, we are justified in just using $\mathbb{F}_q$ to denote a finite field on q elements.

2.2 Linear Subspaces

We are finally ready to define the notion of linear subspace.

Definition 2.2.1 (Linear Subspace)**.** $S \subseteq \mathbb{F}_q{}^n$ is a linear subspace if the following properties hold:

1. For every $\mathbf{x}, \mathbf{y} \in S$, $\mathbf{x} + \mathbf{y} \in S$, where the addition is vector addition over $\mathbb{F}_q$ (that is, do addition component wise over $\mathbb{F}_q$).

2. For every $a \in \mathbb{F}_q$ and $\mathbf{x} \in S$, $a \cdot \mathbf{x} \in S$, where the multiplication is over $\mathbb{F}_q$.

Here is a (trivial) example of a linear subspace of $\mathbb{F}_5^3$:

$$S_1 = \{(0,0,0), (1,1,1), (2,2,2), (3,3,3), (4,4,4)\}. \tag{2.1}$$

Note that for example $(1,1,1) + (3,3,3) = (4,4,4) \in S_1$ and $2 \cdot (4,4,4) = (3,3,3) \in S_1$ as required by the definition. Here is another somewhat less trivial example of a linear subspace over $\mathbb{F}_3^3$:

$$S_2 = \{(0,0,0), (1,0,1), (2,0,2), (0,1,1), (0,2,2), (1,1,2), (1,2,0), (2,1,0), (2,2,1). \tag{2.2}$$

Note that $(1,0,1) + (0,2,2) = (1,2,0) \in S_2$ and $2 \cdot (2,0,2) = (1,0,1) \in S_2$ as required.

Remark 2.2.1. Note that the second property implies that $\mathbf{0}$ is contained in every linear subspace. Further for any subspace over $\mathbb{F}_2$, the second property is redundant: see Exercise 2.4.

Before we state some properties of linear subspaces, we state some relevant definitions.

[1]An isomorphism $\phi : S \to S'$ is a map (such that $\mathbb{F} = (S, +, \cdot)$ and $\mathbb{F}' = (S', \oplus, \circ)$ are fields) where for every $a_1, a_2 \in S$, we have $\phi(a_1 + a_2) = \phi(a_1) \oplus \phi(a_2)$ and $\phi(a_1 \cdot a_2) = \phi(a_1) \circ \phi(a_2)$.

Definition 2.2.2 (Span)**.** Given a set $B = \{\mathbf{v}_1, \ldots, \mathbf{v}_\ell\}$. The *span* of B is the set of vectors

$$\left\{ \sum_{i=1}^{\ell} a_i \cdot \mathbf{v}_i | a_i \in \mathbb{F}_q \text{ for every } i \in [\ell] \right\}.$$

Definition 2.2.3 (Linear independence of vectors)**.** We say that $\mathbf{v}_1, \mathbf{v}_2, \ldots \mathbf{v}_k$ are *linearly independent* if for every $1 \le i \le k$ and for every $k-1$-tuple $(a_1, a_2, \ldots, a_{i-1}, a_{i+1}, \ldots, a_k) \in \mathbb{F}_q^{k-1}$,

$$\mathbf{v}_i \neq a_1 \mathbf{v}_1 + \ldots + a_{i-1}\mathbf{v}_{i-1} + a_{i+1}\mathbf{v}_{i+1} + \ldots + a_k \mathbf{v}_k.$$

In other words, $\mathbf{v}_i$ is not in the span of the set $\{\mathbf{v}_1, \ldots, \mathbf{v}_{i-1}, \mathbf{v}_{i+1}, \ldots, \mathbf{v}_n\}$.

For example the vectors $(1,0,1), (1,1,1) \in S_2$ are linearly independent.

Definition 2.2.4 (Rank of a matrix)**.** The *rank* of matrix in $\mathbb{F}_q^{k \times k}$ is the maximum number of linearly independent rows (or columns). A matrix in $\mathbb{F}_q^{k \times n}$ with rank $\min(k, n)$ is said to have *full* rank.

One can define the row (column) rank of a matrix as the maximum number of linearly independent rows (columns). However, it is a well-known theorem that the row rank of a matrix is the same as its column rank. For example, the matrix below over $\mathbb{F}_3$ has full rank (see Exercise 2.5):

$$G_2 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}. \tag{2.3}$$

Any linear subspace satisfies the following properties (the full proof can be found in any standard linear algebra textbook).

Theorem 2.2.1. *If $S \subseteq \mathbb{F}_q{}^n$ is a linear subspace then*

1. $|S| = q^k$ *for some* $k \ge 0$. *The parameter k is called the* dimension *of S.*

2. *There exists* $\mathbf{v}_1, \ldots, \mathbf{v}_k \in S$ *called* basis elements *(which need not be unique) such that every* $\mathbf{x} \in S$ *can be expressed as* $\mathbf{x} = a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots + a_n\mathbf{v}_n$ *where* $a_i \in \mathbb{F}_q$ *for* $1 \le i \le k$. *In other words, there exists a full rank $k \times n$ matrix G (also known as a* generator matrix*) with entries from* $\mathbb{F}_q$ *such that every* $\mathbf{x} \in S$, $\mathbf{x} = (a_1, a_2, \ldots a_k) \cdot G$ *where*

$$G = \begin{pmatrix} \longleftarrow \mathbf{v}_1 \longrightarrow \\ \longleftarrow \mathbf{v}_2 \longrightarrow \\ \vdots \\ \longleftarrow \mathbf{v}_k \longrightarrow \end{pmatrix}.$$

3. *There exists a full rank* $(n-k) \times n$ *matrix H (called a* parity check matrix*) such that for every* $\mathbf{x} \in S$, $H\mathbf{x}^T = \mathbf{0}$.

4. *G and H are* orthogonal, *that is,* $G \cdot H^T = \mathbf{0}$.

Proof Sketch.

Property 1. We begin with the proof of the first property. For the sake of contradiction, let us assume that $q^k < |S| < q^{k+1}$, for some $k \geq 0$. Iteratively, we will construct a set of linearly independent vectors $B \subseteq S$ such that $|B| \geq k+1$. Note that by the definition of a linear subspace the span of B should be contained in S. However, this is a contradiction as the size of the span of B is at least[2] $q^{k+1} > |S|$.

To complete the proof, we show how to construct the set B in a greedy fashion. In the first step pick $\mathbf{v}_1$ to be any non-zero vector in S and set $B \leftarrow \{\mathbf{v}_1\}$ (we can find such a vector as $|S| > q^k \geq 1$). Now say after the step t (for some $t \leq k$), $|B| = t$. Now the size of the span of the current B is $q^t \leq q^k < |S|$. Thus there exists a vector $\mathbf{v}_{t+1} \in S \setminus B$ that is linearly independent of vectors in B. Set $B \leftarrow B \cup \{\mathbf{v}_{t+1}\}$. Thus, we can continue building B till $|B| = k+1$, as desired.

Property 2. We first note that we can pick $B = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ to be any set of k linearly independent vectors– this just follows from the argument above for **Property** 1.1. This is because the span of B is contained in S. However, since $|S| = q^k$ and the span of B has q^k vectors, the two have to be the same.

Property 3. Property 3 above follows from another fact that every linear subspace S has a null space $N \subseteq \mathbb{F}_q^n$ such that for every $\mathbf{x} \in S$ and $\mathbf{y} \in N$, $\langle \mathbf{x}, \mathbf{y} \rangle = 0$. Further, it is known that N itself is a linear subspace of dimension $n-k$. (The claim that N is also a linear subspace follows from the following two facts: for every $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{F}_q^n$, (i) $\langle \mathbf{x}, \mathbf{y}+\mathbf{z} \rangle = \langle \mathbf{x}, \mathbf{y} \rangle + \langle \mathbf{x}, \mathbf{z} \rangle$ and (ii) for any $a \in \mathbb{F}_q$, $\langle \mathbf{x}, a\mathbf{y} \rangle = a \cdot \langle \mathbf{x}, \mathbf{y} \rangle$.) In other words, there exists a generator matrix H for it. This matrix H is called the parity check matrix of S.

Property 4. See Exercise 2.8. □

As examples, the linear subspace S_1 in (2.1) has as one of its generator matrices

$$G_1 = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

and as one of its parity check matrices

$$H_1 = \begin{pmatrix} 1 & 2 & 2 \\ 2 & 2 & 1 \end{pmatrix}.$$

Further, the linear subspace S_2 in (2.2) has G_2 as one of its generator matrices and has the following as one of its parity check matrices

$$H_2 = \begin{pmatrix} 1 & 1 & 2 \end{pmatrix}.$$

Finally, we state another property of linear subspaces that is useful.

[2]See Exercise 2.7.

Lemma 2.2.2. *Given matrix G of dimension $k \times n$ that is a generator matrix of subspace S_1 and matrix H of dimension $(n-k) \times n$ that is a parity check matrix of subspace S_2 such that $GH^T = \mathbf{0}$, then $S_1 = S_2$.*

Proof. We first prove that $S_1 \subseteq S_2$. Given any $\mathbf{c} \in S_1$, there exists $\mathbf{x} \in \mathbb{F}_q^k$ such that $\mathbf{c} = \mathbf{x}G$. Then,

$$\mathbf{c}H^T = \mathbf{x}GH^T = \mathbf{0},$$

which implies that $\mathbf{c} \in S_2$, as desired.

To complete the proof note that as H has full rank, its null space (or S_2) has dimension $n-(n-k) = k$ (this follows from a well known fact from linear algebra). Now as G has full rank, the dimension of S_1 is also k. Thus, as $S_1 \subseteq S_2$, it has to be the case that $S_1 = S_2$.[3] □

2.3 Properties of Linear Codes

The above theorem gives two alternate characterizations of an $[n,k]_q$ linear code C:

- C is generated by its $k \times n$ generator matrix G. As an example that we have already seen, the $[7,4,3]_2$ Hamming code has the following generator matrix:

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

- C is also characterized by an $(n-k) \times n$ parity check matrix H. We claim that the following matrix is a parity check matrix of the $[7,4,3]_2$ Hamming code:

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

  Indeed, it can be easily verified that $G \cdot H^T = \mathbf{0}$. Then Lemma 2.2.2 proves that H is indeed a parity check matrix of the $[7,4,3]_2$ Hamming code.

We now look at some consequences of the above characterizations of an $[n,k]_q$ linear code C. We started this chapter with a quest for succinct representation of a code. Note that both the generator matrix and the parity check matrix can be represented using $O(n^2)$ symbols from $\mathbb{F}_q$ (which is much smaller than the exponential representation of a general code). More precisely (see Exercise 2.10),

Proposition 2.3.1. *Any $[n,k]_q$ linear code can be represented with* $\min(nk, n(n-k))$ *symbols from $\mathbb{F}_q$.*

[3]If not, $S_1 \subset S_2$ which implies that that $|S_2| \geq |S_1| + 1$. The latter is not possible if both S_1 and S_2 have the same dimension.

There is an encoding algorithm for C that runs in $O(n^2)$ (in particular $O(kn)$) time– given a message $\mathbf{m} \in \mathbb{F}_q^k$, the corresponding codeword $C(\mathbf{m}) = \mathbf{m} \cdot G$, where G is the generator matrix of C. (See Exercise 2.11.)

Proposition 2.3.2. *For any $[n,k]_q$ linear code, given its generator matrix, encoding can be done with $O(nk)$ operations over $\mathbb{F}_q$.*

There is an error-detecting algorithm for C that runs in $O(n^2)$. This is a big improvement over the naive brute force exponential time algorithm (that goes through all possible codewords $\mathbf{c} \in C$ and checks if $\mathbf{y} = \mathbf{c}$). (See Exercise 2.12.)

Proposition 2.3.3. *For any $[n,k]_q$ linear code, given its parity check matrix, error detection can be performed in $O(n(n-k))$ operations over $\mathbb{F}_q$.*

Next, we look at some alternate characterizations of the distance of a linear code.

2.3.1 On the Distance of a Linear Code

We start with the following property, which we have seen for the special case of binary linear codes (Proposition 1.5.3).

Proposition 2.3.4. *For a $[n,k,d]_q$ code C,*

$$d = \min_{\substack{\mathbf{c} \in C, \\ \mathbf{c} \neq \mathbf{0}}} wt(\mathbf{c}).$$

Proof. To show that d is the same as the minimum weight we show that d is no more than the minimum weight and d is no less than the minimum weight.

First, we show that d is no more than the minimum weight. We can see this by considering $\Delta(\mathbf{0}, \mathbf{c}')$ where $\mathbf{c}'$ is the non-zero codeword in C with minimum weight; its distance from $\mathbf{0}$ is equal to its weight. Thus, we have $d \le wt(\mathbf{c}')$, as desired.

Now, to show that d is no less than the minimum weight, consider $\mathbf{c_1} \neq \mathbf{c_2} \in C$ such that $\Delta(\mathbf{c}_1, \mathbf{c}_2) = d$. Note that $\mathbf{c}_1 - \mathbf{c_2} \in C$ (this is because $-\mathbf{c_2} = -1 \cdot \mathbf{c}_2 \in C$, where -1 is the additive inverse of 1 in $\mathbb{F}_q$ and $\mathbf{c}_1 - \mathbf{c}_2 = \mathbf{c}_1 + (-\mathbf{c}_2)$, which by the definition of linear codes is in C). Now note that $wt(\mathbf{c_1} - \mathbf{c_2}) = \Delta(\mathbf{c}_1, \mathbf{c}_2) = d$, since the non-zero symbols in $\mathbf{c}_1 - \mathbf{c}_2$ occur exactly in the positions where the two codewords differ. Further, since $\mathbf{c}_1 \neq \mathbf{c}_2$, $\mathbf{c}_1 - \mathbf{c}_2 \neq \mathbf{0}$, which implies that the minimum Hamming weight of any non-zero codeword in C is at most d. □

Next, we look at another property implied by the parity check matrix of a linear code.

Proposition 2.3.5. *For any $[n,k,d]_q$ code C with parity check matrix H, d is the minimum number of linearly dependent columns in H.*

Proof. By Proposition 2.3.4, we need to show that the minimum weight of a non-zero codeword in C is the minimum number of linearly dependent columns. Let t be the minimum number of linearly dependent columns in H. To prove the claim we will show that $t \le d$ and $t \ge d$.

For the first direction, Let $\mathbf{c} \neq \mathbf{0} \in C$ be a codeword with $wt(\mathbf{c}) = d$. Now note that, by the definition of the parity check matrix, $H \cdot \mathbf{c}^T = \mathbf{0}$. Working through the matrix multiplication, this gives us that $\sum_{i=1}^{n} c_i H^i$, where

$$H = \begin{pmatrix} \uparrow & \uparrow & & \uparrow & & \uparrow \\ H^1 & H^2 & \cdots & H^i & \cdots & H^n \\ \downarrow & \downarrow & & \downarrow & & \downarrow \end{pmatrix}$$

and $\mathbf{c} = (c_1, \ldots, c_n)$. Note that we can skip multiplication for those columns for which the corresponding bit c_i is zero, so for this to be zero, those H^i with $c_i \neq 0$ are linearly dependent. This means that $d \geq t$, as the columns corresponding to non-zero entries in $\mathbf{c}$ are one instance of linearly dependent columns.

For the other direction, consider the minimum set of columns from H, $H^{i_1}, H^{i_2}, \ldots, H^{i_t}$ that are linearly dependent. This implies that there exists non-zero elements $c'_{i_1}, \ldots, c'_{i_t} \in \mathbb{F}_q$ such that $c'_{i_i} H^{i_1} + \ldots + c'_{i_t} H^{i_t} = \mathbf{0}$. (Note that all the c'_{i_j} are non-zero as no set of less than t columns are linearly dependent.) Now extend $c'_{i_1}, \ldots, c'_{i_t}$ to the vector $\mathbf{c}'$ such that $c'_j = 0$ for $j \notin \{i_1, \ldots, i_t\}$. Note that $\mathbf{c}' \in C$ and thus, $d \leq wt(\mathbf{c}') = t$ (where recall t is the minimum number of linearly independent columns in H). □

2.4 Hamming Codes

We now change gears and look at the general family of linear codes, which were discovered by Hamming. So far we have seen the $[7,4,3]_2$ Hamming code (in Section 1.5). In fact for any $r \geq 2$, there is a $[2^r - 1, 2^r - r - 1, 3]_2$ Hamming code. Thus in Section 1.5, we have seen this code for $r = 3$.

Consider the $r \times (2^r - 1)$ matrix $\mathbf{H}_r$ over $\mathbb{F}_2$, where the ith column $\mathbf{H}_r^i$, $1 \leq i \leq 2^r - 1$, is the binary representation of i (note that such a representation is a vector in $\{0,1\}^r$). For example, for the case we have seen ($r = 3$),

$$\mathbf{H}_3 = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}.$$

Note that by its definition, the code that has $\mathbf{H}_r$ as its parity check matrix has block length $2^r - 1$ and dimension $2^r - r - 1$. This leads to the formal definition of the general Hamming code.

Definition 2.4.1. The $[2^r - 1, 2^r - r - 1]_2$ Hamming code, denoted by $C_{H,r}$ has parity check matrix $\mathbf{H}_r$.

In other words, the general $[2^r - 1, 2^r - r - 1]_2$ Hamming code is the code $\{\mathbf{c} \in \{0,1\}^{2^r - 1} | H_r \cdot \mathbf{c}^T = \mathbf{0}\}$.

Next we argue that the above Hamming code has distance 3 (in Proposition 1.5.1 we argued this for $r = 3$).

Proposition 2.4.1. *The Hamming code* $[2^r - 1, 2^r - r - 1, 3]_2$ *has distance 3.*

Proof. No two columns in $\mathbf{H}_r$ are linearly dependent. If they were, we would have $\mathbf{H}_r^i + \mathbf{H}_r^j = \mathbf{0}$, but this is impossible since they differ in at least one bit (being binary representations of integers, $i \neq j$). Thus, by Proposition 2.3.5, the distance is at least 3. It is at most 3, since (e.g.) $\mathbf{H}_r^1 + \mathbf{H}_r^2 + \mathbf{H}_r^3 = \mathbf{0}$. □

Now note that under the Hamming bound for $d = 3$ (Theorem 1.6.1), $k \le n - \log_2(n+1)$, so for $n = 2^r - 1$, $k \le 2^r - r - 1$. Hence, the Hamming code is a perfect code. (See Definition 1.7.2.)

In Question 1.7.1, we asked which codes are perfect codes. Interestingly, the only perfect binary codes are the following:

- The Hamming codes which we just studied.
- The trivial $[n, 1, n]_2$ codes for odd n (which have 0^n and 1^n as the only codewords): see Exercise 2.22.
- Two codes due to Golay [24].

2.5 Family of codes

Till now, we have mostly studied specific codes, that is, codes with *fixed* block lengths and dimension. The only exception was the "family" of $[2^r - 1, 2^r - r - 1, 3]_2$ Hamming codes (for $r \ge 2$) that we studied in the last section. We will see shortly that when we do an asymptotic study of codes (which is what we will do), it makes more sense to talk about a family of codes. First, we define the notion of family of codes:

Definition 2.5.1 (Family of codes)**.** $C = \{C_i\}_{i \ge 1}$ is a family of codes where C_i is a $(n_i, k_i, d_i)_q$ code for each i (and we assume $n_{i+1} > n_i$). The rate of C is defined as

$$R(C) = \lim_{i \to \infty} \left\{ \frac{k_i}{n_i} \right\}.$$

The relative distance of C is defined as

$$\delta(C) = \lim_{i \to \infty} \left\{ \frac{d_i}{n_i} \right\}.$$

For example, C_H the family of Hamming code is a family of codes with $n_i = 2^i - 1, k_i = 2^i - i - 1, d_i = 3$ and thus,

$$R(C_H) = \lim_{i \to \infty} 1 - \frac{i}{2^i - 1} = 1,$$

and

$$\delta(C_H) = \lim_{i \to \infty} \frac{3}{2^i - 1} = 0.$$

We will mostly work with family of codes from now on. This is necessary as we will study the asymptotic behavior of algorithms for codes, which does not make sense for a fixed code. For example, when we say that a decoding algorithm for a code C takes $O(n^2)$ time, we would be implicitly assuming that C is a family of codes and that the algorithm has an $O(n^2)$ running time when the block length is large enough. From now on, unless mentioned otherwise, whenever we talk about a code, we will be implicitly assuming that we are talking about a family of codes.

Given that we can only formally talk about asymptotic run time of algorithms, we now also state our formal notion of efficient algorithms:

We'll call an algorithm related to a code of block length n to be efficient, if it runs in time polynomial in n.

For all the specific codes that we will study in this book, the corresponding family of codes will be a "family" in a more natural sense. In other words, all the specific codes in a family of codes will be the "same" code except with different parameters. A bit more formally, we will consider families $\{C_i\}_i$, where given i, one can compute a sufficient description of C_i efficiently.[4]

Finally, the definition of a family of code allows us to present the final version of the the big motivating question for the book. The last formal version of the main question we considered was Question 1.4.1, where we were interested in the tradeoff of rate R and distance d. The comparison was somewhat unfair because R was a ratio while d was an integer. A more appropriate comparison should be between rate R and the relative distance δ. Further, we would be interested in tackling in the main motivating question for families of codes, which results in the following final version:

Question 2.5.1. *What is the optimal tradeoff between $R(C)$ and $\delta(C)$ that can be achieved by some code family C?*

2.6 Efficient Decoding of Hamming codes

We have shown that Hamming code has distance of 3 and thus, by Proposition 1.4.1, can correct one error. However, this is a *combinatorial* result and does not give us an efficient algorithm. One obvious candidate for decoding is the MLD function. Unfortunately, the only implementation of MLD that we know is the one in Algorithm 1, which will take time $2^{\Theta(n)}$, where n is the block length of the Hamming code. However, we can do much better. Consider the following simple algorithm: given the received word $\mathbf{y}$, first check if it is indeed a valid codeword. If it is, we are done. Otherwise, flip each of the n bits and check if the resulting vector is a valid codeword. If so, we have successfully decoded from one error. (If none of the checks are successful,

[4]We stress that this is not *always* going to be the case. In particular, we will consider "random" codes where this efficient constructibility will not be true.

then we declare a decoding failure.) Algorithm 2 formally presents this algorithm (where $C_{H,r}$ is the $[2^r-1, 2^r-r-1, 3]_2$ Hamming code).[5]

Algorithm 2 Naive Decoder for Hamming Code

```
INPUT: Received word $\mathbf{y}$
OUTPUT: $\mathbf{c}$ if $\Delta(\mathbf{y},\mathbf{c}) \le 1$ else Fail

1: IF $\mathbf{y} \in C_{H,r}$ THEN
2:     RETURN $\mathbf{y}$
3: FOR $i = 1 \dots n$ DO
4:     $\mathbf{y}' \leftarrow \mathbf{y} + \mathbf{e}_i$                    ▷ $\mathbf{e}_i$ is the $i$th standard vector
5:     IF $\mathbf{y}' \in C_{H,r}$ THEN
6:         RETURN $\mathbf{y}'$
7: RETURN Fail
```

It is easy to check that Algorithm 2 can correct up to 1 error. If each of the checks $\mathbf{y}' \in C_{H,r}$ can be done in $T(n)$ time, then the time complexity of the proposed algorithm will be $O(nT(n))$. Note that since $C_{H,r}$ is a linear code (and dimension $k = n - O(\log n)$) by Proposition 2.3.3, we have $T(n) = O(n \log n)$. Thus, the proposed algorithm has running time $O(n^2 \log n)$.

Note that Algorithm 2 can be generalized to work for any linear code C with distance $2t+1$ (and hence, can correct up to t errors): go through all possible error vectors $\mathbf{z} \in [q]^n$ (with $wt(\mathbf{z}) \le t$) and check if $\mathbf{y} - \mathbf{z}$ is in the code or not. Algorithm 3 presents the formal algorithm (where C is an $[n, k, 2t+1]_q$ code). The number of error patterns $\mathbf{z}$ considered by Algorithm 3

Algorithm 3 Decoder for Any Linear Code

```
INPUT: Received word $\mathbf{y}$
OUTPUT: $\mathbf{c} \in C$ if $\Delta(\mathbf{y},\mathbf{c}) \le t$ else Fail

1: FOR $i = 0 \dots t$ DO
2:     FOR $S \subseteq [n]$ such that $|S| = i$ DO
3:         FOR $\mathbf{z} \in \mathbb{F}_q^n$ such that $wt(\mathbf{z}_S) = wt(\mathbf{z}) = i$ DO
4:             IF $\mathbf{y} - \mathbf{z} \in C$ THEN
5:                 RETURN $\mathbf{y} - \mathbf{z}$
6: RETURN Fail
```

is[6] $\sum_{i=0}^{t} \binom{n}{i}(q-1)^i \le O((nq)^t)$. Further by Proposition 2.3.3, Step 4 can be performed with $O(n^2)$ operations over $\mathbb{F}_q$. Thus, Algorithm 3 runs in with $O(n^{t+2}q^t)$ operations over $\mathbb{F}_q$, which for q being polynomial small in n, is $n^{O(t)}$ operations. In other words, the algorithm will have

[5]Formally speaking, a decoding algorithm should return the transmitted message $\mathbf{x}$ but Algorithm 2 actually returns $C_{H,r}(\mathbf{x})$. However, since $C_{H,r}$ is a linear code, it is not too hard to see that one can obtain $\mathbf{x}$ from $C_{H,r}(\mathbf{x})$ in $O(n^3)$ time– see Exercise 2.23. Further, for $C_{H,r}$ one can do this in $O(n)$ time– see Exercise 2.24.

[6]Recall (1.14).

polynomial running time for codes with constant distance (though the running time would not be practical even for moderate values of t).

However, it turns out that for Hamming codes there exists a decoding algorithm with an $O(n^2)$ running time. To see this first note that if the received word $\mathbf{y}$ has no errors then $H_r \cdot \mathbf{y}^T = \mathbf{0}$. If not, $\mathbf{y} = \mathbf{c} + \mathbf{e}_i$, where $\mathbf{c} \in C$ and $\mathbf{e}_i$ which is the unit vector with the only nonzero element at the i-th position. Thus, if H_r^i stands for the i-th column of H_r,

$$H_r \cdot \mathbf{y}^T = H_r \cdot \mathbf{c}^T + H_r \cdot (\mathbf{e}_i)^T = H_r \cdot (\mathbf{e}_i)^T = H_r^i,$$

where the second equality follows as $H_r \cdot \mathbf{c}^T = 0$, which in turn follows from the fact that $\mathbf{c} \in C$. In other words, $H_r \cdot \mathbf{y}^T$ gives the *location* of the error. This leads to Algorithm 4.

Algorithm 4 Efficient Decoder for Hamming Code

```
INPUT: Received word $\mathbf{y}$
OUTPUT: $\mathbf{c}$ if $\Delta(\mathbf{y},\mathbf{c}) \le 1$ else Fail

1: $\mathbf{b} \leftarrow H_r \cdot \mathbf{y}^T$.
2: Let $i \in [n]$ be the number whose binary representation is $\mathbf{b}$
3: IF $\mathbf{y} - \mathbf{e}_i \in C_H$ THEN
4:     RETURN $\mathbf{y} - \mathbf{e}_i$
5: RETURN Fail
```

Since Step 1 in Algorithm 4 is a matrix vector multiplication (which can be done in $O(n \log n)$ time as the matrix is $O(\log n) \times n$) and Step 3 by Proposition 2.3.3 can be performed in $O(n \log n)$ time, Algorithm 4 runs in $O(n \log n)$ time. Thus,

Theorem 2.6.1. *The $[n = 2^r - 1, 2^r - r - 1, 3]_2$ Hamming code is 1-error correctable. Further, decoding can be performed in time $O(n \log n)$.*

2.7 Dual of a Linear Code

Till now, we have thought of parity check matrix as defining a code via its null space. However, we are not beholden to think of the parity check matrix in this way. A natural alternative is to use the parity check matrix as a generator matrix. The following definition addresses this question.

Definition 2.7.1 (Dual of a code)**.** Let H be a parity check matrix of a code C, then the code generated by H is called the dual of C. For any code C, its dual is denoted by $C^{\perp}$.

It is obvious from the definition that if C is an $[n,k]_q$ code then $C^{\perp}$ is an $[n, n-k]_q$ code. The first example that might come to mind is $C^{\perp}_{H,r}$, which is also known as the *Simplex code* (we will denote it by $C_{Sim,r}$). Adding an all 0's column to H_r and using the resulting matrix as a generating matrix, we get the *Hadamard* code (we will denote it by C_{Had}, r). We claim that $C_{Sim,r}$ and $C_{Had,r}$ are $[2^r - 1, r, 2^{r-1}]_2$ and $[2^r, r, 2^{r-1}]_2$ codes respectively. The claimed block length and dimension follow from the definition of the codes, while the distance follows from the following result.

Proposition 2.7.1. *$C_{Sim,r}$ and $C_{Had,r}$ both have a distance of 2^{r-1}.*

Proof. We first show the result for $C_{Had,r}$. In fact, we will show something stronger: every non-zero codeword in $C_{Had,r}$ has weight exactly equal to 2^{r-1} (the claimed distance follows from Proposition 2.3.4). Consider a message $\mathbf{x} \neq 0$. Let its ith entry be $x_i = 1$. $\mathbf{x}$ is encoded as

$$\mathbf{c} = (x_1, x_2, \ldots, x_r)(H_r^0, H_r^1, \ldots, H_r^{2^r-1}),$$

where H_r^j is the binary representation of $0 \le j \le 2^r - 1$ (that is, it contains all the vectors in $\{0,1\}^r$). Further note that the jth bit of the codeword $\mathbf{c}$ is $\langle \mathbf{x}, H_r^j \rangle$. Group all the columns of the generator matrix into pairs $(\mathbf{u}, \mathbf{v})$ such that $\mathbf{v} = \mathbf{u} + \mathbf{e}_i$ (i.e. $\mathbf{v}$ and $\mathbf{u}$ are the same except in the ith position). Notice that this partitions all the columns into 2^{r-1} disjoint pairs. Then,

$$\langle \mathbf{x}, \mathbf{v} \rangle = \langle \mathbf{x}, \mathbf{u} + \mathbf{e}_i \rangle = \langle \mathbf{x}, \mathbf{u} \rangle + \langle \mathbf{x}, \mathbf{e}_i \rangle = \langle \mathbf{x}, \mathbf{u} \rangle + x_i = \langle \mathbf{x}, \mathbf{u} \rangle + 1.$$

Thus we have that exactly one of $\langle \mathbf{x}, \mathbf{v} \rangle$ and $\langle \mathbf{x}, \mathbf{u} \rangle$ is 1. As the choice of the pair $(\mathbf{u}, \mathbf{v})$ was arbitrary, we have proved that for any non-zero codeword $\mathbf{c}$ such that $\mathbf{c} \in C_{Had}$, $wt(\mathbf{c}) = 2^{r-1}$.

For the simplex code, we observe that all codewords of $C_{Had,3}$ are obtained by padding a 0 to the beginning of the codewords in $C_{Sim,r}$, which implies that all non-zero codewords in $C_{Sim,r}$ also have a weight of 2^{r-1}, which completes the proof. □

We remark that the family of Hamming code has a rate of 1 and a (relative) distance of 0 while the families of Simplex/Hadamard codes have a rate of 0 and a relative distance of 1/2. Notice that both code families either have rate or relative distance equal to 0. Given this, the following question is natural special case of Question 2.5.1:

Question 2.7.1. *Does there exist a family of codes C such that $R(C) > 0$ and $\delta(C) > 0$ hold* simultaneously*?*

Codes that have the above property are called *asymptotically good.*

2.8 Exercises

Exercise 2.1. Prove that the set of rationals (i.e. the set of reals of the form $\frac{a}{b}$, where both a and $b \neq 0$ are integers), denoted by $\mathbb{Q}$, is a field.

Exercise 2.2. Let q be a prime power. Let $x \in \mathbb{F}_q$ such that $x \notin \{0,1\}$. Then prove that for any $n \le q - 1$:

$$\sum_{i=0}^{n} x^i = \frac{x^{n+1} - 1}{x - 1}.$$

Exercise 2.3. The main aim of this exercise is to prove the following identity that is true for any $\alpha \in \mathbb{F}_q$:

$$\alpha^q = \alpha \tag{2.4}$$

To make progress towards the above we will prove a sequence of properties of *groups*. A group G is a pair $(S, \circ)$ where the operator $\circ : G \times G \to G$ such that $\circ$ is commutative[7] and the elements of S are closed under $\circ$. Further, there is a special element $\iota \in S$ that is the identity element and every element $a \in S$ has an inverse element $b \in S$ such that $a \circ b = \iota$. Note that a finite field $\mathbb{F}_q$ consists of an *additive group* with the + operator (and 0 as additive identity) and a *multiplicative* group on the non-zero elements of $\mathbb{F}_q$ (which is also denoted by $\mathbb{F}_q^*$) with the $\cdot$ operator (and 1 as the multiplicative identity).[8]

For the rest of the problem let $G = (S, \cdot)$ be a multiplicative group with $|G| = m$. Prove the following statements.

1. For any $\beta \in G$, let $o(\beta)$ be the smallest integer o such that $\beta^o = 1$. Prove that such an $o \le m$ always exists. Further, argue that $T = \{1, \beta, \ldots, \beta^{o-1}\}$ also forms a group. $(T, \cdot)$ is called a *sub-group* of G and $o(\beta)$ is called the *order* of β.

2. For any $g \in G$, define the *coset* (w.r.t. T) as

$$gT = \{g \cdot \beta | \beta \in T\}.$$

   Prove that if $g \cdot h^{-1} \in T$ then $gT = hT$ and $gT \cap hT = \emptyset$ otherwise. Further argue that these cosets partition the group G into disjoint sets.

3. Argue that for any $g \in G$, we have $|gT| = |T|$.

4. Using the above results or otherwise, argue that for any $\beta \in G$, we have

$$\beta^m = 1.$$

5. Prove (2.4).

Exercise 2.4. Prove that for $q = 2$, the second condition in Definition 2.2.1 is implied by the first condition.

Exercise 2.5. Prove that G_2 from (2.3) has full rank.

Exercise 2.6. In this problem we will look at the problem of solving a system of linear equations over $\mathbb{F}_q$. That is, one needs to solve for unknowns $x_1, \ldots, x_n$ given the following m linear equations (where $a_{i,j}, b_i \in \mathbb{F}_q$ for $1 \le i \le m$ and $1 \le j \le n$):

$$a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n = b_1.$$

$$a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,n}x_n = b_2.$$

$$\vdots$$

$$a_{m,1}x_1 + a_{m,2}x_2 + \cdots + a_{m,n}x_n = b_m.$$

[7]Technically, G is an *abelian* group.

[8]Recall Definition 2.1.1.

1. (*Warm-up*) Convince yourself that the above problem can be stated as $A \cdot \mathbf{x}^T = \mathbf{b}^T$, where A is an $m \times n$ matrix over $\mathbb{F}_q$, $\mathbf{x} \in \mathbb{F}_q^n$ and $\mathbf{b} \in \mathbb{F}_q^m$.

2. (*Upper Triangular Matrix*) Assume $n = m$ and that A is upper triangular, i.e. all diagonal elements ($a_{i,i}$) are non-zero and all lower triangular elements ($a_{i,j}$, $i > j$) are 0. Then present an $O(n^2)$ time[9] algorithm to compute the unknown vector $\mathbf{x}$.

3. (*Gaussian Elimination*) Assume that A has *full rank* (or equivalently a *rank* of n.)

   (a) Prove that the following algorithm due to Gauss converts A into an upper triangular matrix. By permuting the columns if necessary make sure that $a_{1,1} \neq 0$. (Why can one assume w.l.o.g. that this can be done?) Multiply all rows $1 < i \leq n$ with $\frac{a_{1,1}}{a_{i,1}}$ and then subtract $a_{1,j}$ from the (i, j)th entry $1 \leq j \leq n$. Recurse with the same algorithm on the $(n-1) \times (n-1)$ matrix A' obtained by removing the first row and column from A. (Stop when $n = 1$.)

   (b) What happens if A does not have full rank? Show how one can modify the algorithm above to either upper triangulate a matrix or report that it does not have full rank. (Convince yourself that your modification works.)

   (c) Call a system of equations $A \cdot \mathbf{x}^T = \mathbf{b}^T$ *consistent* if there exists a solution to $\mathbf{x} \in \mathbb{F}_q^n$. Show that there exists an $O(n^3)$ algorithm that finds the solution if the system of equations is consistent and A has full rank (and report "fail" otherwise).

4. ($m < n$ *case*) Assume that A has full rank, i.e. has a rank of m. In this scenario either the system of equations is inconsistent or there are q^{n-m} solutions to $\mathbf{x}$. Modify the algorithm from above to design an $O(m^2 n)$ time algorithm to output the solutions (or report that the system is inconsistent).

   - Note that in case the system is consistent there will be q^{n-m} solutions, which might be much bigger than $O(m^2 n)$. Show that this is not a problem as one can represent the solutions as system of linear equations. (I.e. one can have $n - m$ "free" variables and m "bound" variables.)

5. ($m > n$ *case*) Assume that A has full rank, i.e. a rank of n. In this scenario either the system of equations is inconsistent or there is a unique solution to $\mathbf{x}$. Modify the algorithm from above to design an $O(m^2 n)$ time algorithm to output the solution (or report that the system is inconsistent).

6. (*Non-full rank case*) Give an $O(m^2 n)$ algorithm for the general case, i.e. the $m \times n$ matrix A need not have full rank. (The algorithm should either report that the system of equations is inconsistent or output the solution(s) to $\mathbf{x}$.)

Exercise 2.7. Prove that the span of k linearly independent vectors over $\mathbb{F}_q$ has size exactly q^k.

[9]For this problem, any basic operation over $\mathbb{F}_q$ takes unit time.

Exercise 2.8. Let G and H be a generator and parity check matrix of the same linear code of dimension k and block length n. Then $G \cdot H^T = \mathbf{0}$.

Exercise 2.9. Let C be an $[n,k]_q$ linear code with a generator matrix with no all zeros columns. Then for every position $i \in [n]$ and $\alpha \in \mathbb{F}_q$, the number of codewords $\mathbf{c} \in C$ such that $c_i = \alpha$ is exactly q^{k-1}.

Exercise 2.10. Prove Proposition 2.3.1.

Exercise 2.11. Prove Proposition 2.3.2.

Exercise 2.12. Prove Proposition 2.3.3.

Exercise 2.13. A set of vector $S \subseteq \mathbb{F}_q^n$ is called t-wise independent if for every set of positions I with $|I| = t$, the set S projected to I has each of the vectors in $\mathbb{F}_q^t$ appear the same number of times. (In other words, if one picks a vector $(s_1, \dots, s_n)$ from S at random then any of the t random variables are uniformly and independently random over $\mathbb{F}_q$).

Prove that any linear code C whose dual $C^{\perp}$ has distance $d^{\perp}$ is $(d^{\perp} - 1)$-wise independent.

Exercise 2.14. A set of vectors $S \subseteq \mathbb{F}_2^k$ is called ε-biased sample space if the following property holds. Pick a vector $X = (x_1, \dots, x_k)$ uniformly at random from S. Then X has *bias* at most ε, that is, for every $I \subseteq [k]$,

$$\left| \Pr\left(\sum_{i \in I} x_i = 0 \right) - \Pr\left(\sum_{i \in I} x_i = 1 \right) \right| \le \varepsilon.$$

We will look at some connections of such sets to codes.

1. Let C be an $[n,k]_2$ code such that all non-zero codewords have Hamming weight in the range $\left[\left(\frac{1}{2} - \varepsilon\right) n, \left(\frac{1}{2} + \varepsilon\right) n\right]$. Then there exists an ε-biased space of size n.

2. Let C be an $[n,k]_2$ code such that all non-zero codewords have Hamming weight in the range $\left[\left(\frac{1}{2} - \gamma\right) n, \left(\frac{1}{2} + \gamma\right) n\right]$ for some constant $0 < \gamma < 1/2$. Then there exists an ε-biased space of size $n^{O(\gamma^{-1} \cdot \log(1/\varepsilon))}$.

Exercise 2.15. Let C be an $[n,k,d]_q$ code. Let $\mathbf{y} = (y_1, \dots, y_n) \in (\mathbb{F}_q \cup \{?\})^n$ be a received word[10] such that $y_i = ?$ for at most $d - 1$ values of i. Present an $O(n^3)$ time algorithm that outputs a codeword $\mathbf{c} = (c_1, \dots, c_n) \in C$ that agrees with y in all un-erased positions (i.e., $c_i = y_i$ if $y_i \neq ?$) or states that no such $\mathbf{c}$ exists. (Recall that if such a $\mathbf{c}$ exists then it is unique.)

Exercise 2.16. In the chapter, we did not talk about how to obtain the parity check matrix of a linear code from its generator matrix. In this problem, we will look at this “conversion" procedure.

(a) Prove that any generator matrix $\mathbf{G}$ of an $[n,k]_q$ code C (recall that $\mathbf{G}$ is a $k \times n$ matrix) can be converted into another equivalent generator matrix of the form $\mathbf{G}' = [\mathbf{I}_k | \mathbf{A}]$, where $\mathbf{I}_k$ is the $k \times k$ identity matrix and $\mathbf{A}$ is some $k \times (n-k)$ matrix. By “equivalent," we mean that the code generated by $\mathbf{G}'$ has a linear bijective map to C.

[10] A ? denotes an erasure.

Note that the code generated by $\mathbf{G}'$ has the message symbols as its first k symbols in the corresponding codeword. Such codes are called *systematic codes*. In other words, every linear code can be converted into a systematic code. Systematic codes are popular in practice as they allow for immediate access to the message symbols.

(b) Given an $k \times n$ generator matrix of the form $[\mathbf{I}_k|\mathbf{A}]$, give a corresponding $(n-k) \times n$ parity check matrix. Briefly justify why your construction of the parity check matrix is correct.

Hint: Try to think of a parity check matrix that can be decomposed into two submatrices: one will be closely related to $\mathbf{A}$ and the other will be an identity matrix, though the latter might not be a $k \times k$ matrix).

(c) Use part (b) to present a generator matrix for the $[2^r-1, 2^r-r-1, 3]_2$ Hamming code.

Exercise 2.17. So far in this book we have seen that one can modify one code to get another code with interesting properties (for example, the construction of the Hadamard code from the Simplex code from Section 2.7 and Exercise 1.7). In this problem you will need to come up with more ways of constructing new codes from existing ones.

Prove the following statements (recall that the notation $(n,k,d)_q$ code is used for general codes with q^k codewords where k need not be an integer, whereas the notation $[n,k,d]_q$ code stands for a *linear code* of dimension k):

1. If there exists an $(n,k,d)_{2^m}$ code, then there also exists an $(nm, km, d' \geq d)_2$ code.
2. If there exists an $[n,k,d]_{2^m}$ code, then there also exists an $[nm, km, d' \geq d]_2$ code.
3. If there exists an $[n,k,d]_q$ code, then there also exists an $[n-d, k-1, d' \geq \lceil d/q \rceil]_q$ code.
4. If there exists an $[n,k,\delta n]_q$ code, then for every $m \geq 1$, there also exists an $\left(n^m, k/m, \left(1-(1-\delta)^m\right)\cdot n^m\right.$ code.
5. If there exists an $[n,k,\delta n]_2$ code, then for every *odd* $m \geq 1$, there also exists an $\left[n^m, k, \frac{1}{2}\cdot\left(1-(1-2\delta)^m\right.\right.$ code.

Note: In all the parts, the only things that you can assume about the original code are only the parameters given by its definition– nothing else!

Exercise 2.18. Let C_1 be an $[n,k_1,d_1]_q$ code and C_2 be an $[n,k_2,d_2]_q$ code. Then define a new code as follows:

$$C_1 \ominus C_2 = \{(\mathbf{c}_1, \mathbf{c}_1 + \mathbf{c}_2) | \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2\}.$$

Next we will prove interesting properties of this operations on codes:

1. If G_i is the generator matrix for C_i for $i \in [2]$, what is a generator matrix for $C_1 \ominus C_2$?
2. Argue that $C_1 \ominus C_2$ is an $[2n, k_1+k_2, d \overset{\text{def}}{=} \min(2d_1, d_2)]_q$ code.

3. Assume there exists algorithms $\mathscr{A}_i$ for code C_i for $i \in [2]$ such that: (i) $\mathscr{A}_1$ can decode from e errors and s erasures such that $2e+s < d_1$ and (ii) $\mathscr{A}_2$ can decode from $\lfloor (d_2-1)/2 \rfloor$ errors. Then argue that one can correct $\lfloor (d-1)/2 \rfloor$ errors for $C_1 \ominus C_2$.
   Hint: Given a received word $(\mathbf{y}_1, \mathbf{y}_2) \in \mathbb{F}_q^n \times \mathbb{F}_q^n$, first apply $\mathscr{A}_2$ on $\mathbf{y}_2 - \mathbf{y}_1$. Then create an intermediate received word for $\mathscr{A}_1$.

4. We will now consider a recursive construction of a binary linear code that uses the $\ominus$ operator. For integers $0 \le r \le m$, we define the code $C(r,m)$ as follows:

   - $C(r,r) = \mathbb{F}_2^r$ and $C(0,r)$ is the code with only two codewords: the all ones and all zeroes vector in $\mathbb{F}_2^r$.
   - For $1 < r < m$, $C(r,m) = C(r,m-1) \ominus C(r-1,m-1)$.

   Determine the parameters of the code $C(r,m)$.

Exercise 2.19. Let C_1 be an $[n_1,k_1,d_1]_2$ binary linear code, and C_2 an $[n_2,k_2,d_2]$ binary linear code. Let $C \subseteq \mathbb{F}_2^{n_1 \times n_2}$ be the subset of $n_1 \times n_2$ matrices whose columns belong to C_1 and whose rows belong to C_2. C is called the tensor of C_1 and C_2 and is denoted by $C_1 \otimes C_2$.

Prove that C is an $[n_1 n_2, k_1 k_2, d_1 d_2]_2$ binary linear code.

Exercise 2.20. In Section 2.4 we considered the *binary* Hamming code. In this problem we will consider the more general q-ary Hamming code. In particular, let q be a prime power and $r \ge 1$ be an integer. Define the following $r \times n$ matrix $H_{q,r}$, where each column is an non-zero vector from $\mathbb{F}_q^r$ such that the first non-zero entry is 1. For example,

$$H_{3,2} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 2 \end{pmatrix}$$

In this problem we will derive the parameters of the code. Define the generalized Hamming code $C_{H,r,q}$ to be the linear code whose parity check matrix is $H_{q,r}$. Argue that

1. The block length of $C_{H,r,q}$ is $n = \frac{q^r-1}{q-1}$.

2. $C_{H,q,r}$ has dimension $n-r$.

3. $C_{H,q,r}$ has distance 3.

Exercise 2.21. Design the best 6-ary code (family) with distance 3 that you can.

Hint: Start with a 7-ary Hamming code.

Exercise 2.22. Prove that the $[n,1,n]_2$ code for odd n (i.e. the code with the all zeros and all ones vector as it only two codewords) attains the Hamming bound (Theorem 1.7.1).

Exercise 2.23. Let C be an $[n,k]_q$ code with generator matrix G. Then given a codeword $\mathbf{c} \in C$ one can compute the corresponding message in time $O(kn^2)$.

Exercise 2.24. Given a $\mathbf{c} \in C_{H,r}$, one can compute the corresponding message in time $O(n)$.

Exercise 2.25. Let C be an $(n,k)_q$ code. Prove that if C can be decoded from e errors in time $T(n)$, then it can be decoded from $n+c$ errors in time $O((nq)^c \cdot T(n))$.

Exercise 2.26. Show that the bound of kd of the number of ones in the generator matrix of any binary linear code (see Exercise 1.12) cannot be improved for every code.

Exercise 2.27. Let C be a linear code. Then prove that $\left(C^{\perp}\right)^{\perp} = C$.

Exercise 2.28. Note that for any linear code C, the codewords $\mathbf{0}$ is in both C and $C^{\perp}$. Show that there exists a linear code C such that it shares a non-zero codeword with $C^{\perp}$.

Exercise 2.29. We go into a bit of diversion and look at how finite fields are different from infinite fields (e.g. $\mathbb{R}$). Most of the properties of linear subspaces that we have used for linear codes (e.g. notion of dimension, the existence of generator and parity check matrices, notion of duals) also hold for linear subspaces over $\mathbb{R}$.[11] One trivial property that holds for linear subspaces over finite fields that does not hold over $\mathbb{R}$ is that linear subspaces over $\mathbb{F}_q$ with dimension k has size q^k (though this is a trivial consequence that F_q are finite field while $\mathbb{R}$ is an infinite field). Next, we consider a more subtle distinction.

Let $S \subseteq \mathbb{R}^n$ be a linear subspace over $\mathbb{R}$ and let $S^{\perp}$ is the dual of S. Then show that

$$S \cap S^{\perp} = \{\mathbf{0}\}.$$

By contrast, linear subspaces over finite fields can have non-trivial intersection with their duals (see e.g. Exercise 2.28).

Exercise 2.30. A linear code C is called *self-orthogonal* if $C \subseteq C^{\perp}$. Show that

1. The binary repetition code with even number of repetitions is self-orthogonal.
2. The Hadamard code $C_{Had,r}$ is self-orthogonal.

Exercise 2.31. A linear code C is called self dual if $C = C^{\perp}$. Show that for

1. Any self dual code has dimension $n/2$.
2. Prove that the following code is self-dual

$$\{(\mathbf{x},\mathbf{x}) | \mathbf{x} \in \mathbb{F}_2^k\}.$$

Exercise 2.32. Given a code C a *puncturing* of C is another code C' where the same set of positions are dropped in all codewords of C. More precisely, if $C \subseteq \Sigma^n$ and the set of punctured positions is $P \subseteq [n]$, then the punctured code is $\{(c_i)_{i \notin P} | (c_1, \ldots, c_n) \in C\}$.

Prove that a linear code with no repetitions (i.e. there are no two positions $i \neq j$ such that for every codeword $\mathbf{c} \in C$, $c_i = c_i$) is a puncturing of the Hadamard code. Hence, Hadamard code is the “longest" linear code that does not repeat.

[11]A linear subspace $S \subseteq \mathbb{R}^n$ is the same as in Definition 2.2.1 where all occurrences of the finite field $\mathbb{F}_q$ is replaced by $\mathbb{R}$.

Exercise 2.33. In this problem we will consider the *long code*. For the definition, we will use the functional way of looking at the ambient space as mentioned in Remark 1.2.1. A long code of dimension k is a binary code such that the codeword corresponding to $\mathbf{x} = \mathbb{F}_2^k$, is the function $f : \{0,1\}^{2^k} \rightarrow \{0,1\}$ defined as follows. For any $\mathbf{m} \in \{0,1\}^{\mathbb{F}_2^k}$, we have $f((m_\alpha)_{\alpha\in\mathbb{F}_2^k}) = m_{\mathbf{x}}$. Derive the parameters of the long code.

Finally, argue that the long code is the code with the longest block length such that the codewords do not have a repeated coordinate (i.e. there does not exists $i \neq j$ such that for every codeword $\mathbf{c}$, $c_i = c_j$). (Contrast this with the property of Hadamard code above.)

2.9 Bibliographic Notes

Finite fields are also called Galois fields (another common notation for $\mathbb{F}_q$ is $GF(q)$), named after Évariste Galois, whose worked laid the foundations of their theory. (Galois led an extremely short and interesting life, which ended in death from a duel.) For a more thorough treatment refer to any standard text on algebra or the book on finite fields by Lidl and Niederreiter [52].

The answer to Question 1.7.1 was proved by van Lint [74] and Tietavainen [73].