ALGEBRAIC CODING THEORY

ARISTIDE LAUNOIS

ABSTRACT. This paper includes excerpts of a group project, in collaboration with Akil Al-Khafaji, Joshua Brown, Heran Du and Lucas Papadopoulos under the supervision of Agata Smoktunowicz, designed to introduce the basics of coding theory to undergraduates with a background in algebra. These excerpts introduce the basic definitions of a code in the first section and lead to a proof of the famous MacWilliams identity in the second. Note that some definitions and proofs are missing from these excerpts, however, these can be found in any relevant textbook or lecture notes.

1. INTRODUCTION TO CODES

Algebraic Coding Theory is the study of error-correcting codes using tools from algebra. Originally developed to address issues in data communication and storage, coding theory has since found applications in diverse fields such as cryptography, data compression, and network security. An error-correcting code is used to ensure safe transmission of information via noisy channels i.e. there is some chance for error. For example, consider the following situation:

Say we want to send a 'yes' or 'no' answer to a question and we want to send this via some noisy channel. We will send this using bits by setting the bit 1 and 0 to mean 'yes' and 'no', respectively. In doing so, we have 'encoded' the messages as bits. However, this is not a very good code. Had we sent a 1 and one error occurs through transmission then the receiver would receive a 0 and 'decode' this as a 'no'. Thus, even though we wanted to say 'yes' the receiver thinks we said 'no'.

Coding theory aims to find ways to solve the above problem. It does this by finding ways of encoding the messages that allows the receiver to correctly decode the received data. Often times it is useful to use algebraic methods to do this.

The most simple code that we can easily see works is known as the **3-bit repetition code**. Here instead of encoding the message 'yes' by 1 we encode it with the binary string 111 and similarly 'no' is encoded with 000. Now, if one error is made in transmission of the word 'yes', say 101 was received, the receiver would be able to decode this correctly to 'yes' by noting that 101 is 'closer' to 111 than it is to 000.

In practice, the messages that we want to send ('yes' and 'no' in our example) will also be binary strings. The key for making error-correction possible is that these strings will be shorter than those we encode them as. This is called adding **redundancy**. However, the sending of extra bits is not a desirable property as it lengthens transmission time. This is often referred to as "the main coding theory problem".

1.1. Introduction.

Here we introduce some fundamental algebraic definitions and concepts, assuming the reader has prior familiarity with them. Additionally, we establish the notation for these, as they will be referenced throughout the paper.

1.1.1. *Rings & Fields.* The following are standard definitions that can be found in [10].

Definition 1.1.1. A **ring** R is a set with binary operations

$$\text{Addition:} \quad + : R \times R \to R : (a, b) \mapsto a + b$$
$$\text{Multiplication:} \quad \cdot : R \times R \to R : (a, b) \mapsto a \cdot b$$

such that $(R, +)$ forms an abelian group, we denote its identity by 0_R, or just 0; and $(R\backslash\{0\}, \cdot)$ forms a monoid. That is, for all $a, b, c, \in R$

(1) Multiplication is associative. That is, $a \cdot (b \cdot c) = (a \cdot b) \cdot c$
(2) There exists a multiplicative identity. That is, there exists $1_R \in R$ (or just 1) such that $1_R \cdot a = a \cdot 1_R = a$
(3) Multiplication is distributive over addition. That is, $a \cdot (b + c) = a \cdot b + a \cdot c$.

If moreover $(R\backslash\{0\}, \cdot)$ forms an abelian group (i.e. the ring is commutative and there exists multiplicative inverses) then we call R a **field**. We denote that **finite field** with q elements by $\mathbb{F}_q$ throughout this paper.

Remark 1.1.2. See Stewart [10] for a classification of finite fields.

excerpts based on a group project written by Akil Al-Khafaji, Joshua Brown, Heran Du, Aristide Launois, and Lucas Papadopoulos under the supervision of Agata Smoktunowicz.

We will borrow the following well-known result about finite fields.

Theorem 1.1.3. *The multiplicative group of a finite field $\mathbb{F}_q$, $\mathbb{F}_q^\times = (\mathbb{F}_q \backslash \{0\}, \cdot)$, is cyclic.*

1.1.2. *Modules & Vector spaces.*

Definition 1.1.4. A left (respectively right) **R-module**, V, is a pair consisting of an abelian group $V = (V, +)$ and an operation $R \times V \to V : (\lambda, v) \mapsto \lambda v$, called **scalar multiplication**, such that for all $\lambda, \mu \in R$ and $v, w \in V$ the following identities hold:

$$\begin{aligned} \lambda(v + w) &= \lambda v + \lambda w \ ; \\ (\lambda + \mu)v &= \lambda v + \mu v \ ; \\ \lambda(\mu v) &= (\lambda\mu)v \ ; \\ 1_R v &= v. \end{aligned}$$

For a right module we reverse the order of scalar multiplication.

If R is a field then V is a called a vector space over R or simply an **R-vector space**.

Definition 1.1.5. A subset U of a left (respectively right) R-module, V, is called a left (respectively right) **R-submodule** if U is closed under the left (respectively right) scalar multiplication of V. That is, a submodule is a subset of a module which is a module in its own right under the same operations with the same identity elements.

If R is a field then V is a vector space and we call U a **vector (or linear) subspace**

1.2. **Basics of Codes.** Here we begin formally defining the basic objects of interest in coding theory.

Definition 1.2.1. A (block) **code** C of **length n over a finite ring** R **(or field)** is a non-empty subset of the set of n-tuples R^n, where the elements in C are called **codewords** and R is called the **alphabet**.

We sometimes call the ambient space R^n the **codespace** and refer to its elements as **words**.

For a code over the finite field $\mathbb{F}_q$ with $q = 2$ or $q = 3$, the code is described as a **binary code** or a **ternary code**, respectively. More generally, we call codes over the field $\mathbb{F}_q$, **q-ary** codes.

We will mostly discuss codes over fields throughout this paper.

It is conventional to write the codewords as row vectors. Throughout this paper we will often not write words/vectors in bold and simply write $x \in R^n$ unless it will help with clarity.

Example 1.2.2.
(1) The code $C_1 = \{(0,0,0,0), (0,0,0,1)\}$ is a binary code of length 4.
(2) The code $C_2 = \{(1,2), (2,2), (0,2)\}$ is a ternary code of length 2.
(3) The code $C_3 = \mathbb{F}_4^5$ is a 4-ary code of length 5.
(4) The code $C_4 = \{(0,0)(1,0), (0,0,0,1)\}$ is not a block code as the codewords do not have fixed length.

Remark 1.2.3. There is a difference between a 4-ary code and a **quaternary** code. As in the above definition, a 4-ary code is a code over the <u>field</u>, $\mathbb{F}_4$. A quaternary code refers to a code over the <u>ring</u> of integers modulo 4, $\mathbb{Z}_4$. During this paper, we will mainly focus on the prior.

The terms binary and ternary code, however, are not ambiguous since $\mathbb{Z}_2 = \mathbb{F}_2$ and $\mathbb{Z}_3 = \mathbb{F}_3$.

When discussing binary codes it is customary to denote codewords without brackets. For example C_1 above would be written

$$C_1 = \{0000, 0001\}.$$

We saw this in the introduction to this chapter.

Definition 1.2.4. A left (respectively right) **linear code** of length n over a finite ring (or field) R is an *R-submodule*, C, of the codespace R^n.

Example 1.2.5.
(1) The binary code $C_1 = \{001, 011\}$ is not a linear code. Observe that it does not contain the zero codeword.
(2) The binary code $C_2 = \{0000, 1010, 0101, 1111\}$ is a linear code of dimension 2. It has a basis $\mathcal{B} = \{1010, 0101\}$.

1.3. **Parameters of Codes.** We will restrict our attention to codes over finite fields for this section. The definitions for rings are analogous.

Here we define the parameters of a code. These are the basic properties of a code that we are interested in. One such parameter is the **length** of a code, see Definition 1.2.1.

Definition 1.3.1. The **size** of a code C, $M = |C|$, is the number of codewords in the code. Note that if C is linear of dimension k, then $M = q^k$.

Now we construct the third and final parameter of a code, the minimum distance. This is the parameter that is responsible for determining the error-correcting capabilities of the code. To do this we need the following crucial definition.

Definition 1.3.2. The (Hamming) **weight** of a codeword $x = (x_1, x_2, \dots, x_n) \in \mathbb{F}_q^n$, denoted by $w_H(x)$ or just $w(x)$, is the number of nonzero elements in x

$$w_H(x) = \big|\{i \in \{1, \dots, n\} : x_i \neq 0\}\big|,$$

where $|\cdot|$ denotes the cardinality of a set.

Example 1.3.3. For $x = (1, 0, 1, 1, 0)$, the number of nonzero elements is $w_H(x) = 3$.

Definition 1.3.4. The (Hamming) **distance** between two words $x, y \in \mathbb{F}_q^n$ is the number of positions in which they differ, formally defined as:

$$d_H(x, y) = d(x, y) := \big|\{i \in \{1, \dots, n\} : x_i \neq y_i\}\big|.$$

The distance and weight are related in the following way.

Proposition 1.3.5. *For all $x, y \in \mathbb{F}_q^n$, $d_H(x, y) = w_H(x - y)$.*

Proof. The coordinate positions in which two codewords differ, $x_i \neq y_i$, are exactly the coordinate positions where $x_i - y_i \neq 0$, which is exactly the i-th coordinate position of the difference vector $x - y$. □

Example 1.3.6. Consider the following two binary words of length 5:

$$c_1 = (1, 0, 1, 1, 0), \quad c_2 = (1, 1, 0, 1, 0).$$

The two codewords differ in two positions so the Hamming Distance is

$$d_H(c_1, c_2) = w_H(c_1 - c_2) = w_H((0, 1, 1, 0, 0)) = 2.$$

Proposition 1.3.7. *The Hamming distance defines a metric [9]. That is, it satisfies the following properties: For all $x, y, z \in \mathbb{F}_q^n$*

(1) $d_H(x, y) \geq 0$ and $d_H(x, y) = 0$ if and only if $x = y$.
*(2) **Symmetric:** $d_H(x, y) = d_H(y, x)$,*
*(3) **Triangle Inequality:** $d_H(x, z) \leq d_H(x, y) + d_H(y, z)$.*

Proof. (1) By definition of the weight 1.3.2 we see that

$$d_H(x, y) = w(x - y) \geq 0$$

for all x, y.

Note that, if $x = y$ then $d_H(x, y) = w(x - y) = w(0) = 0$. Conversely, we know by the definition of the weight, that the only vector with weight zero is the zero vector itself. So,

$$d_H(x, y) = w(x - y) = 0 \Rightarrow x - y = 0 \Rightarrow x = y.$$

(2) We note that the weight function is even. That is, writing $x = (x_1, \dots, x_n) \in \mathbb{F}_q^n$ in coordinates

$$w(-x) = \big|\{-x_i : x_i \neq 0\}\big| = \big|\{x_i : x_i \neq 0\}\big| = w(x).$$

It follows directly that

$$d_H(x, y) = w(x - y) = w(-(x - y)) = w(y - x) = d_H(y, x)$$

(3) Observe that,

$$w(u + v) \leq w(u) + w(v).$$

This follows since if the coordinate $(u + v)_i = u_i + v_i \neq 0$, then at least one of u_i or v_i is nonzero. Hence,

$$d_H(x, z) = w(x - z) = w((x - y) + (y - z)) \leq w(x - y) + w(y - z) = d_H(x, y) + d_H(y, z).$$

□

Remark 1.3.8. We can use different weight and distance functions such as the Lee weight and metric. This is most natural when considering quaternary codes, as in Remark 1.2.3. For the interested reader this is covered in Chapter 3 and 8 of [5].

Definition 1.3.9. The **minimum distance** d of a (not necessarily linear) code is the minimum distance between any two of its codewords. Formally, for a code C,

$$d := \min\{d_H(x, y) : x, y \in C, x \neq y\}.$$

Example 1.3.10. The binary code of length 4, $C = \{0111, 1011, 1101, 1110, 1111\}$, has minimum distance $d = 1$.

The following is one way in which the structure of linear codes is useful.

Proposition 1.3.11. *The minimum distance of a linear code is equal to the minimum weight of its nonzero codeword. Formally, for a linear code C with minimum distance d,*

$$d = \min\{w_H(x) : x \in C, x \neq 0\}.$$

Remark 1.3.12. Note that the assumption that the code is linear is important here. In Example 1.3.10, we looked at a non-linear binary code. The minimum distance was 1, but its minimum weight was 3.

Proof. Put $w := \min\{w_H(x) : x \in C, x \neq 0\}$. By Proposition 1.3.5, we have that $d = d(x, y) = w(x-y) \geq w$ for some $x, y \in C$. Here we have used the fact that C is linear to see that $x-y \in C$. Similarly, $w = w_H(x) = d_H(x, 0) \geq d_H(x, y) \geq d$. Hence, $w = d$. □

1.3.1. *Basics of Decoding.* Once the receiver has received a (possibly erroneous) word from the sender, they must then decide which codeword the sender originally sent. In the introduction to this Chapter, we did this when we said that 101 was closer to 111 than 000. We would like to write this formally. What we did implicitly, was assign each word $w \in \mathbb{F}_q^n$ a codeword $c \in C$.

Definition 1.3.13. A **decoding scheme** is a function $\sigma : \mathbb{F}_q^n \to C$.

The simplest kind of decoding scheme is a **minimal distance decoding scheme**. This assigns to each word w the closest codeword with respect to Hamming distance.

Definition 1.3.14. A **minimal distance (or maximum likelihood) decoding scheme** is a function $\sigma : \mathbb{F}_q^n \to C$ such that $d(w, \sigma(w)) \leq d(w, c)$ for all $w \in \mathbb{F}_q^n$ and $c \in C$.

Unless otherwise stated we assume the use of a maximum likelihood decoding scheme.

We can now illustrate how the minimum distance of a code and its error-correcting capacities are related. To do this we need a useful concept known as Sphere-packing.

Definition 1.3.15. We define the **ball of radius r centered at a point** $x \in \mathbb{F}_q^n$ to be the set

$$B_r(x) := \{y \in \mathbb{F}_q^n : d_H(x, y) \leq r\},$$

where d_H refers to the Hamming distance, Definition 1.3.4.

Lemma 1.3.16. *Let C be a code with minimum distance d and $2r < d$. Then the balls of radius r, $B_r(c)$ and $B_r(c')$ are disjoint for all $c, c' \in C$.*

Proof. Suppose for contradiction that they are not disjoint. Let z be in their intersection. Then by the triangle inequality, 1.3.7,

$$d(c, c') \leq d(c, z) + d(z, c') \leq 2r < d,$$

a contradiction. □

Theorem 1.3.17 (Error-Correcting Capability)**.** *A code with minimum distance d can correct up to $\lfloor (d-1)/2 \rfloor$ errors (using a minimal distance decoding scheme), where $\lfloor \cdot \rfloor$ is the usual floor function.*

Moreover, it can detect up to $d-1$ errors. That is, the receiver will know an error has been made in transmission provided no more than $d-1$ errors have been made. The receiver can detect an error whenever they have been sent a word which is not a codeword.

Proof. For error correction, suppose w is the received word and c the sent codeword. If there are no more than r errors in the received codeword, then the correct codeword must be within r Hamming distance. That is, $w \in B_r(c)$. So, $2r + 1 \leq d$, in particular, $2r < d$. By Lemma 1.3.16,

$$B_r(c) \cap B_r(c') = \emptyset,$$

for all $c' \in C$. Hence, using a minimal distance decoding scheme, 1.3.14, the receiver would correctly assign c to w, since

$$d(w, c) \leq r \leq d(w, c')$$

for all $c' \in C$.

For error detection, suppose t errors occur and $0 < t < d-1$ (we can assume $t \neq 0$ since otherwise there are no errors to detect). Then the received word is still not equal to another codeword, by definition of the minimum distance, so the receiver will detect the error. □

We conclude this section by introducing some very important notation that we will use constantly throughout the paper.

Definition 1.3.18. The **parameters of a code over a finite field** $\mathbb{F}_q$ is the tuple (n, M, d) of its length, size, and minimum distance. We call a code of length n, size M, and minimum distance d, an (n, M, d) code.

If the code is linear we will use square brackets. So, a linear code of length n, dimension k, and distance d is called an $[n, q^k, d]$ code or, sometimes an $[n, k, d]_q$ code or $[n, k, d]$ code [2].

If the minimum distance is not known, then we omit it and refer to such a codes as (n, M), $[n, q^k]$,$[n, k]_q$ or $[n, k]$ codes.

Example 1.3.19.
(1) The binary code $C_1 = \{001, 110, 101, 111\}$ is a non-linear $(3, 4, 1)$ code.
(2) The ternary code $C = \text{span}\{(1, 0, 0, 0), (0, 0, 2, 0)\}$ is a linear $[4, 3^2, 1]$ or $[4, 2, 1]_3$ code.

2. The MacWilliams Identity

In this chapter we will prove a foundational result in coding theory due to F.J. MacWilliams [6], the MacWilliams Identity. We will begin with a brief overview and some background in order to justify to the reader why the result is considered to be foundational. We will need to introduce some important group theory that is the backbone of the proof. We will then see how the result can be applied with a worked example computing a special polynomial associated with the $\text{Ham}(3,2)$ code that encodes the distribution of the weights of its codewords.

We start by defining, analgously to vector spaces, the dual of a code:

Definition 2.0.1. Let $C \subset \mathbb{F}_q^n$ be a code. The **dual code** is defined as

$$C^{\perp} := \{v \in \mathbb{F}_q^n : \langle v, c\rangle = 0 \text{ for every } c \in C\}.$$

where $\langle \cdot , \cdot \rangle$ is the usual inner product vectors.

Note that the dual code is always a linear code. It is easy to see that the dual code is unique and hence there is no ambiguity in saying 'the' dual code.

2.1. Background & Motivation.

The result that we are trying to prove is known as the MacWilliams Identity. In particular, we will prove it for linear codes over finite fields with respect to Hamming weight.

To understand what the result states we need to introduce a couple of new concepts that will be formally defined later. The **Hamming weight distribution** of a code is the family of numbers $\{Q_i\}_{i=1}^n$ whose members Q_i are equal to the amount of codewords of weight i in the code.

Remark 2.1.1. For a linear code all the parameters can be recovered from its weight distribution.

We will then define the **Hamming weight enumerator** of a code to be the polynomial in z whose z^i coefficient is Q_i for all $0 \leq i \leq n$. This encodes the weight distribution as a polynomial.

From this we can derive the following identity between that of a linear code over $\mathbb{F}_q$ and its dual, denoted $Q(z)$ and $P(z)$ respectively,

$$P(z) = q^{-k}\,(1 + (q-1)z)^n\, Q\left(\frac{1-z}{1+(q-1)z}\right). \tag{1}$$

This will be explored more deeply in sections 2.3, 2.4 and 2.5. This result tells us the surprising fact that, the Hamming weight distribution of a linear q-ary code is completely determined by that of its dual.

This is not the only reason that this result is considered foundational, as we are about to see.

2.1.1. *Linear vs Non-Linear.*

Here we will illustrate the importance of non-linear codes and how the MacWilliams identity inspired results that are useful in decoding such codes.

Proposition 2.1.2. *No length 11 binary linear code of dimension 5 with minimum distance 5 exists.*

Proof. Suppose that such a code exists. The size of the code is $2^5 = 32$. We compute the RHS of the Sphere-Packing bound, see [5],

$$\frac{2^{11}}{\sum_{i=0}^{\lfloor\frac{5-1}{2}\rfloor}\binom{11}{i}(2-1)^i} = \frac{2^{11}}{\sum_{i=0}^{2}\binom{11}{i}} = \frac{2^{11}}{1+11+55} = 30.567\cdots \leq 32.$$

So, no such code can exist.

□

As a direct corollary, the largest binary linear code with minimum distance 5 has dimension at most 4. Hence, has at most $2^4 = 16$ codewords. If we allow ourselves to consider non-linear codes, we can find a binary code of the same length with higher size and minimum distance, called the length 11 Hadamard code [7]. Clearly, higher minimum distance and size for the same fixed length is a desirable property of a code.

Moreover, whilst we will not prove it here, using the MacWilliams Identity we can derive a new bound called the Plotkin bound (this was originally proved without the MacWilliams Identity by Plotkin) which is stronger than the Singleton bound [5]. It turns out the length 11 Hadamard code actually achieves this bound.

So, we can find non-linear codes with desirable properties. The natural question to ask next is how we decode such codes efficiently.

Encoding and decoding schemes we saw rely on the parity-check matrix, which is defined as the generator of the dual code. For us to generalize the identity (1) to non-linear codes over fields, we need to understand what the 'dual code' is. If we were to do this naively and define, for a general (block) code C over $\mathbb{F}_q$ of length n, the dual code

$$C^{\perp} := \{v \in \mathbb{F}_q^n : \langle u, v\rangle = 0 \text{ for all } u \in C\}.$$

We find that $(C^{\perp})^{\perp} = \operatorname{span}\{C\}$ which is, of course, a linear code. Thus, $(C^{\perp})^{\perp} = C$ if and only if C is a linear code. Simply put, the naive extension of our definition doesn't give us any information about the non-linear code C.

However, the MacWilliams identity inspired an extension to the notion of the dual, from linear binary codes to all binary codes that contain the zero codeword for which $(C^{\perp})^{\perp} = C$ does hold [8]. This has proved useful in finding efficient decoding schemes for such codes.

Remark 2.1.3. The notion of dual shown in [8] gives a definition of the "weight enumerator of the dual" but does not guarantee the existence of a "dual code" that satisfies this weight enumerator. This still provides a structure that can aid in decoding. However, as we are about to see for codes related to linear codes over $\mathbb{Z}_4$ we can actually find such codes.

2.1.2. *The Kerdock & Preparata Codes.* Here we discuss how the MacWilliams identity inspired the study of linear codes over more general rings. In the previous section, we discussed how considering codes with less structure (not assuming the structure of a vector space) can allow us to find better codes. The idea now is to weaken the structure of the alphabet from that of a field to a more general ring in order to help us see how it was used to solve big questions in coding theory.

The Kerdock and Preparata codes are families of non-linear binary codes. It turns out that such codes have Hamming weight enumerators that satisfy the MacWilliams Identity as in (1). Historically, this was of great interest to coding theorists in the 1970s and was eventually explained by the following results.

It turns out that these non-linear codes over $\mathbb{F}_2$ can be related to linear codes over $\mathbb{Z}_4$ via what is known as the **Gray map**:

$$\tilde{g} : \mathbb{Z}_4^n \to \mathbb{F}_2^{2n}$$

which is the natural extension of the map $g : \mathbb{Z}_4 \to \mathbb{Z}_2^2$ defined by

$$0 \mapsto (0,0), \quad 1 \mapsto (0,1), \quad 2 \mapsto (1,1), \quad 3 \mapsto (1,0)$$

That is, given a Kerdock code, K and a Preparata code, P, there exists linear codes over $\mathbb{Z}_4$, C, C' such that $\tilde{g}(C) = K$ and $\tilde{g}(C') = P$. Moreover, $C' = C^{\perp}$. We say that K and P are $\mathbb{Z}_4$**-duals**. It is known that codes who are $\mathbb{Z}_4$-duals must have weight enumerators that satisfy the MacWilliams identity in (1), see Chapter 8 of [5].

This then sparked great interest for linear codes over $\mathbb{Z}_4$ known as quaternary codes and other codes over more general rings. Eventually, it was shown that a MacWilliams-type identity holds for all linear codes over finite Frobenius rings with respect to Hamming weight [3].

Remark 2.1.4. We have talked about weakening the structure of both the alphabet (fields to rings) and the code itself (linear vs non-linear). It is important to note that everything we talked about was with respect to the Hamming weight. However, sometimes it is useful to consider different weight functions. For example, the most natural weight function for decoding linear codes over $\mathbb{Z}_4$ is the Lee weight not the Hamming weight [5].

2.2. **Characters of Finite Abelian Groups.** In this section we develop the necessary group theory we need to prove our main result. Throughout this section we denote by $\mathbb{T}$ the group of unit length complex numbers $\mathbb{T} := \{z \in \mathbb{C} : |z| = 1\}$ with the usual multiplication in $\mathbb{C}$. We use the letter $\mathbb{T}$ as it represents the one dimensional torus i.e. the circle.

Definition 2.2.1. Let G be a group. A **character on** G is a group homomorphism $\chi : G \to \mathbb{T}$. We denote the set of all characters on G by $\hat{G}$ and call it the **character** or **dual 'group' of G**.

Example 2.2.2.
(1) We denote by χ_0 the **trivial** or **principal character on G** defined by $\chi_0(g) = 1$ for all $g \in G$.
(2) Given a character χ we can define a new character $\overline{\chi}$ by

$$\overline{\chi}(g) = \overline{\chi(g)} = (\chi(g))^{-1}$$

for all $g \in G$, where $\overline{\chi(g)}$ denotes the usual complex conjugate.

With these examples in mind we can justify the name **character 'group'**:

Proposition 2.2.3. *The set of characters on G, $\hat{G}$, is an abelian group under the operation, $\hat{G} \times \hat{G} \to \hat{G} : (\chi_1, \chi_2) \to \chi_1\chi_2$ where*

$$\chi_1\chi_2(g) = \chi_1(g)\chi_2(g)$$

for all $\chi_1, \chi_2 \in \hat{G}$ and $g \in G$.

Proof. The identity element is χ_0 the principal character since,

$$\chi_0\chi(g) = \chi_0(g)\chi(g) = 1 \cdot \chi(g) = \chi(g) = \chi(g) \cdot 1 = \chi(g)\chi_0(g) = \chi\chi_0(g)$$

for all $g \in G$ and $\chi \in \hat{G}$. The inverses are given by $\chi^{-1} := \overline{\chi}$ as in Example 2.2.2. Associativity follows from the associativity of complex multiplication and the definition of the group operation. □

Remark 2.2.4. Notice that when we restrict our attention to finite groups G with identity e, Lagrange's Theorem says that $g^{|G|} = e$ for all $g \in G$. It follows that

$$1 = \chi(e) = \chi(g^{|G|}) = (\chi(g))^{|G|}.$$

So, characters of finite groups must take values that are $|G|$-th roots of unity. We saw this was the case in the above example.

The following property will be used many times in the proof of the MacWilliams identity:

Proposition 2.2.5. *Let G be a finite group with identity e, and let χ_0 be the trivial character on G. Then the following holds for all $\chi \in \hat{G}$,*

$$\sum_{g \in G} \chi(g) = \begin{cases} |G| & \text{if } \chi = \chi_0, \\ 0 & \text{otherwise.} \end{cases}$$

Moreover, if $\chi \neq \chi_0$, then $\sum_{g \in G \setminus \{e\}} \chi(g) = -1$.

Proof. Suppose that $\chi = \chi_0$. Then,

$$\sum_{g \in G} \chi(g) = \sum_{g \in G} \chi_0(g) = \sum_{g \in G} 1 = |G|$$

This concludes the first case. Assume now that $\chi \neq \chi_0$. Let $h \in G$, then

$$\chi(h) \sum_{g \in G} \chi(g) = \sum_{g \in G} \chi(h)\chi(g) = \sum_{g \in G} \chi(hg) = \sum_{g' \in G} \chi(g')$$

Thus,

$$(\chi(h) - 1) \sum_{g \in G} \chi(g) = 0 \tag{2}$$

and since χ is not trivial there exists some $h' \in G$ such that $\chi(h') \neq 1$. Substituting into (2) yields

$$\left[(\chi(h') - 1) \sum_{g \in G} \chi(g) = 0 \right] \Rightarrow \sum_{g \in G} \chi(g) = 0.$$

For any character we have that $\chi(e) = 1$. So, by what we have just shown for χ non-trivial,

$$\sum_{g \in G \setminus \{0\}} \chi(g) = -\chi(e) + \sum_{g \in G} \chi(g) = -1.$$

□

Characters have applications in many areas of mathematics due to their nice properties. It is not obvious that an arbitrary group has any non trivial characters. In fact, we can find non-abelian groups which have none in the sense of our definition. For example, the quaternion group, Q_8, has no non-trivial characters in the sense of Definition 2.2.1.

However, when restricting our attention to finite abelian groups, G, this is not the case. One very important property is that,

$$G \cong \hat{G}$$

In particular, this tells us that $|G| = |\hat{G}|$. As an easy corollary, we note that any non trivial finite abelian group has non trivial characters. We will not prove either result here however we will state the weaker result and use it in the proof of the MacWilliams Identity.

Theorem 2.2.6. *Let G be a finite abelian group. Then,*

$$|G| = |\hat{G}|.$$

Proof. See Appendix A. □

Example 2.2.7. We will describe the character group of the multiplicative group $G = (\mathbb{F}_q \setminus \{0\}, \cdot)$.

The group G is cyclic, by Theorem 1.1.3, with generator 2. We define the following distinct characters on G

$$\chi_0(2) = 1$$
$$\chi(2) = -1.$$

These are the only characters on G by Theorem 2.2.6. So, $\hat{G} = \{\chi_0, \chi\}$ as a set and with the multiplication of characters we have $\hat{G} \cong C_2$ as a group.

In Example 2.2.7, we found two distinct characters on the multiplicative group $\mathbb{F}_3\backslash\{0\}$, which verifies the result, Theorem 2.2.6. We emphasize once more that these results only hold for finite abelian groups.

We finish this subsection by looking to Fourier analysis and defining a discrete version of the Fourier transform:

Definition 2.2.8. Let G be a finite abelian group. Let V be a $\mathbb{C}$-vector space. Let $f : G \to V$ be some map. The **Hadamard** or **Discrete Fourier Transform of f** is a map $\hat{f} : \hat{G} \to V$ given by

$$\hat{f}(\chi) := \sum_{g \in G} f(g)\chi(g)$$

for all $\chi \in \hat{G}$.

2.3. Weight Enumerators.

Here we go back to coding theory and introduce a way of encoding lots of information about a code using polynomials.

Definition 2.3.1. Given a $[n, k]$ code, C, over $\mathbb{F}_q$ the (Hamming) **weight enumerator of** C is the polynomial,

$$Q(z) = \sum_{c \in C} z^{w(c)} = \sum_{i=0}^{n} Q_i z^i$$

where $Q_i = |\{c \in C : w(c) = i\}|$. The ordered family, $\{Q_i\}_{i=0}^{n}$, of $n+1$ integers is called the **weight distribution of** C. We think of $Q(z)$ as an element of the polynomial ring $\mathbb{C}[z]$. We can also define the **homogeneous weight enumerator**,

$$H_C(x, y) = \sum_{c \in C} x^{n-w(c)} y^{w(c)}.$$

The following Proposition describes the relationship between the homogeneous and non-homogeneous weight enumerators:

Proposition 2.3.2. *Let C an $[n, k]$ code over $\mathbb{F}_q$. Then*

$$H_C(1, y) = Q(y)$$
$$x^n Q(x^{-1}y) = H_C(x, y)$$

*We say that we have **dehomogenized** $H(x, y)$ to find $Q(y)$ and that we have **homogenized** $Q(y)$ to find $H_C(x, y)$.*

Proof. We substitute $x = 1$ into the homogeneous weight enumerator to find

$$H_C(1, y) = \sum_{c \in C} 1^{n-w(c)} y^{w(c)} = \sum_{c \in C} y^{w(c)} = Q(y).$$

Substituting now $z = x^{-1}y$ into the weight enumerator and multiplying through by x^n we see

$$x^n Q(x^{-1}y) = x^n \sum_{c \in C} (x^{-1}y)^{w(c)} = x^n \sum_{c \in C} x^{-w(c)} y^{w(c)} = H_C(x, y).$$

□

This lets us use both forms of the weight enumerator for whatever best suits our purposes. We will say the 'weight enumerator of a C' to mean either form and hope it will be clear from the context.

Example 2.3.3. We end this section by computing the weight enumerators of the following simple binary codes:

(1) The weight enumerator of the 3-bit repetition code $C_1 = \{000, 111\}$ is the polynomial $Q_1(z) = 1 + z^3$.
  Homogenizing this we find $H_{C_1}(x, y) = x^3 Q_1(x^{-1}y) = x^3 + y^3$ to be the homogeneous weight enumerator of C_1.
  The weight distribution is $\{1, 0, 0, 1\}$.
(2) The weight enumerator of the length 4 binary linear code $C_2 = \{0000, 0001, 1000, 1001\}$ is the polynomial $Q_2(z) = 1 + 2z + z^2$.
  Homogenizing this we find $H_{C_2}(x, y) = x^4 Q_2(x^{-1}y) = x^4 + 2x^3y + x^2y^2$ to be the homogeneous weight enumerator of C_2.
  The weight distribution is $\{1, 2, 1, 0, 0\}$.

2.4. **The Proof.** Now we are ready to begin proving the main result of this chapter, the MacWilliams Identity. We fix a linear $[n, k]$ code over $\mathbb{F}_q$, C, for the remainder of this section. We also ensure that C is not the trivial zero code that contains only the zero codeword by setting $k \geq 1$.

Theorem 2.4.1 (MacWilliams Identity, 1963)**.** *Let C be an $[n, k]$ $(k \geq 1)$ code over the field $\mathbb{F}_q$. Denote the weight enumerator of C by $Q(z)$ and the weight enumerator of $C^\perp$ by $P(z)$. Then*

$$|C| \cdot P(z) = [1 + (q-1)z]^n \, Q\left(\frac{1-z}{1+(q-1)z}\right).$$

The proof of this theorem is quite complicated. We use a similar approach to both [5] and [1]. However, the presentation of our proof will differ in that we will assume much less familiarity with algebraic methods. We begin with some linear algebra over finite fields.

Lemma 2.4.2. *Let $f : C \to \mathbb{F}_q$ be a surjective linear map. Then f takes all the values in $\mathbb{F}_q$ the same number of times. Concretely, the preimages of the singleton sets (the fibres) have cardinality,*

$$|f^{-1}\{\alpha\}| = q^{k-1}$$

for all $\alpha \in \mathbb{F}_q$, where $f^{-1}\{\alpha\} := \{u \in C : f(c) = \alpha\}$.

Proof. Note that $\ker(f) \subset C$ is a subspace and, in particular, a subgroup of the additive group of C. Since f is surjective, we have that $\operatorname{rank}(f) = 1$ and so by rank-nullity theorem, $\dim(\ker(f)) = k - 1$. Thus, $|\ker(f)| = q^{k-1}$.

Since f is surjective, we can choose $u_\alpha \in C$ such that $f(u_\alpha) = \alpha$ for each $\alpha \in \mathbb{F}_q$. So, we have that the additive cosets $u_\alpha + \ker(f)$ all have q^{k-1} elements also. This is exactly what we wanted since,

$$\begin{aligned} u_\alpha + \ker(f) &= \{u_\alpha + v \in C : f(v) = 0\} = \{u_\alpha + v \in C : f(u_\alpha + v) = \alpha\} \\ &= \{u \in C : f(u) = \alpha\} = f^{-1}\{\alpha\}. \end{aligned}$$

□

We now begin to construct the main object of interest in the proof:

Lemma 2.4.3. *Let $v \in C^\perp$. Then the linear map $C \to \mathbb{F}_q : u \mapsto \langle u, v\rangle$ is surjective.*

Proof. Suppose the $v \notin C^\perp$. Write $f_v(u) = \langle u, v\rangle$. This is the same mapping $f_v : C \to \mathbb{F}_q : u \mapsto \langle u, v\rangle$. Note that the dimension of the codomain is 1.

Since $v \notin C^\perp$ it follows that there exists some $u \in C$ such that $f_v(u) \neq 0$ and so $\ker(f_v) \neq C \Rightarrow \dim(\ker(f_v)) < k$. By rank-nullity theorem we then see that $\dim(\operatorname{im}(f_v)) \neq 0 \Rightarrow \dim(\operatorname{im}(f_v)) = 1$ and thus f_v is surjective. □

From now on we will fix χ to be a character on the additive group $(\mathbb{F}_q, +)$.

Lemma 2.4.4. *Let $v \in \mathbb{F}_q^n$. Then $\chi_u(v) := \chi(\langle u, v\rangle)$ is a character on $(\mathbb{F}_q^n, +)$.*

Proof. This is an (additive) character by linearity of the inner product. □

Definition 2.4.5. Let f be a function defined on $(\mathbb{F}_q^n, +)$ given by $f(v) = z^{w(v)}$. We apply the Hadamard Transform, Definition 2.2.8, and define for $u \in C$

$$l(u) = l(u; \chi) := \hat{f}(\chi_u(v)) = \sum_{v \in \mathbb{F}_q^n} z^{w(v)} \chi(\langle u, v\rangle).$$

We will call this an **l-function on C** or the **l-function on C (at χ)**.

The idea now is to sum the l-function over all codewords in two different ways.

Lemma 2.4.6. *Let C be an $[n, k]$ code over $\mathbb{F}_q$, $(k \geq 1)$. Then*

$$\sum_{u \in C} l(u) = |C| \cdot P(z),$$

where $l(u)$ is the l-function on C.

Proof. Since the term $z^{w(v)}$ depends only on v and not u we can switch the order of summation as follows,

$$\sum_{u \in C} l(u) = \sum_{u \in C} \sum_{v \in \mathbb{F}_q^n} z^{w(v)} \chi(\langle u, v\rangle) = \sum_{v \in \mathbb{F}_q^n} z^{w(v)} \sum_{u \in C} \chi(\langle u, v\rangle). \tag{3}$$

Now we consider case wise on v.

Firstly, if $v \in C^\perp$ then $\langle u, v\rangle = 0$, so the sum

$$\sum_{u \in C} \chi(\langle u, v\rangle) = \sum_{u \in C} \chi(0) = \sum_{u \in C} 1 = |C|.$$

Now suppose $v \notin C^{\perp}$. By Lemma 2.4.2 and 2.4.3 we have that,

$$\sum_{u \in C} \chi(\langle u, v \rangle) = \sum_{u \in C} \chi(f_v(u)) = q^{k-1} \sum_{\alpha \in \mathbb{F}_q} \chi(\alpha) = 0.$$

which vanishes by Proposition 2.2.5.

Hence,

$$\sum_{u \in C} l(u) = |C| \sum_{u \in C^{\perp}} z^{w(u)} = |C| \cdot P(z).$$

□

Now, we evaluate the same sum without exchanging the order of summation first. It is convenient to extend the usual (Hamming) weight function to a function on the ground field, $\mathbb{F}_q$, which will allow us to analyse our sum using coordinates. We do this, in the natural way, by setting $w(a) = 0$ if $a = 0$ and $w(a) = 1$ otherwise. Thus, for all $v = (v_1, v_2, \ldots, v_n) \in \mathbb{F}_q^n$,

$$w(v) = \sum_{i=1}^{n} w(v_i).$$

In words, the weight of a codeword is the sum of the weight of its coordinates.

Lemma 2.4.7. *Let C be an $[n, k]$ code over $\mathbb{F}_q$ $(k \geq 1)$. Then*

$$\sum_{u \in C} l(u) = [1 + (q-1)z]^n \, Q\left(\frac{1-z}{1+(q-1)z}\right),$$

where $l(u)$ is the l-function on C.

Proof. Recall that χ is a character on the additive group $(\mathbb{F}_q, +)$. Write $u = (u_1, \ldots, u_2) \in C$ using coordinates then the l-function on C can be expanded

$$\begin{aligned} l(u) &= \sum_{(v_1, v_2, \ldots, v_n) \in \mathbb{F}_q^n} z^{w(v_1)+w(v_2)+\cdots+w(v_n)} \chi(u_1 v_1 + \cdots + u_n v_n) \\ &= \sum_{(v_1, v_2, \ldots, v_n) \in \mathbb{F}_q^n} z^{w(v_1)} \chi(u_1 v_1) z^{w(v_2)} \chi(u_2 v_2) \cdots z^{w(v_n)} \chi(u_n v_n) \\ &= \sum_{v_1 \in \mathbb{F}_q} \sum_{v_2 \in \mathbb{F}_q} \cdots \sum_{v_n \in \mathbb{F}_q} \prod_{i=1}^{n} z^{w(v_i)} \chi(u_i v_i). \end{aligned}$$

Furthermore, by repeatedly applying the formula $\sum_i \sum_j (a_i b_j) = (\sum_i a_i)(\sum_j b_j)$ for finite sequences $\{a_i\}, \{b_j\}$, we find

$$\sum_{v_1 \in \mathbb{F}_q} \sum_{v_2 \in \mathbb{F}_q} \cdots \sum_{v_n \in \mathbb{F}_q} \prod_{i=1}^{n} z^{w(v_i)} \chi(u_i v_i) = \prod_{i=1}^{n} \sum_{v_i \in \mathbb{F}_q} z^{w(v_i)} \chi(u_i v_i) = \prod_{i=1}^{n} \sum_{\alpha \in \mathbb{F}_q} z^{w(\alpha)} \chi(u_i \alpha).$$

Now, we consider the value of the inner sum $\sum_{\alpha \in \mathbb{F}_q} z^{w(\alpha)} \chi(u_i \alpha)$ case-wise depending on the value of u_i.

If $u_i = 0$ then $\chi(u_i \alpha) = \chi(0) = 1$. So,

$$\sum_{\alpha \in \mathbb{F}_q} z^{w(\alpha)} \chi(u_i \alpha) = \sum_{\alpha \in \mathbb{F}_q} z^{w(\alpha)} = z^0 + \sum_{\alpha \in \mathbb{F}_q \setminus \{0\}} z^1 = 1 + (q-1)z.$$

Suppose now that $u_i \neq 0$. Then,

$$\sum_{\alpha \in \mathbb{F}_q} z^{w(\alpha)} \chi(u_i \alpha) = 1 + \sum_{\alpha \in \mathbb{F}_q \setminus \{0\}} z^{w(\alpha)} \chi(u_i \alpha) = 1 + z \sum_{\beta \in \mathbb{F}_q \setminus \{0\}} \chi(\beta)$$

The second equality holds similar to the previous case for $u_i = 0$.

Thus, by Proposition 2.2.5 it follows that,

$$\sum_{\alpha \in \mathbb{F}_q} z^{w(\alpha)} \chi(u_i \alpha) = 1 + z(-1) = 1 - z$$

for non-zero u_i.

Since $u_i \neq 0$ exactly $w(u)$ times we have the following formula for the l-function,

$$l(u) = \prod_{i=1}^{n} \sum_{\alpha \in \mathbb{F}_q} z^{w(\alpha)} \chi(u_i \alpha) = (1-z)^{w(u)} (1 + (q-1)z)^{n - w(u)}.$$

Our last step is to sum over all codewords in an attempt to retrieve a weight enumerator for our code,

$$\sum_{u\in C} l(u) = \sum_{u\in C}(1-z)^{w(u)}(1+(q-1)z)^{n-w(u)}$$
$$= [1+(q-1)z]^n \sum_{u\in C}(1-z)^{w(u)}(1+(q-1)z)^{-w(u)}$$
$$= [1+(q-1)z]^n \sum_{u\in C}\left(\frac{(1-z)}{(1+(q-1)z)}\right)^{w(u)} = [1+(q-1)z]^n\, Q\left(\frac{(1-z)}{(1+(q-1)z)}\right).$$

□

Proof of Theorem 2.4.1. This follows directly by combining the above two lemmas. □

There is also a homogeneous version of the identity:

Corollary 2.4.8 (Homogeneous MacWilliams Identity)**.** *Let C be an $[n,k]$ code over the field $\mathbb{F}_q$ $(k \geq 1)$. Then*

$$H_{C^\perp}(x,y) = \frac{1}{|C|} H_C\left(x-y, x+(q-1)y\right).$$

Proof. This follows from homogenizing the MacWilliams Identity 2.4.1, as in Proposition 2.3.2.

$$H_{C^\perp}(x,y) = x^n P(x^{-1}y) = \frac{x^n}{|C|}\left[1+(q-1)x^{-1}y\right]^n Q\left(\frac{1-x^{-1}y}{1+(q-1)x^{-1}y}\right)$$
$$= \frac{x^n\cdot x^{-n}}{|C|}\left[x+(q-1)y\right]^n Q\left(\frac{x-y}{x+(q-1)y}\right) = \frac{1}{|C|}\left[x+(q-1)y\right]^n \sum_{i=1}^{n} Q_i\left(\frac{x-y}{x+(q-1)y}\right)^i$$
$$= \frac{1}{|C|}\sum_{i=1}^{n} Q_i\left(x-y\right)^i\left(x+(q-1)y\right)^{n-i} = \frac{1}{|C|}H_C\left(x-y, x+(q-1)y\right).$$

□

This concludes the beautiful proof of these identities. We end this section by verifying the MacWilliams Identities with a couple simple examples.

Example 2.4.9. We look back to the codes C_1, C_2 of Example 2.3.3.

(1) We see that the dual code $C_1^\perp$ is given by $C_1^\perp = \{000, 110, 101, 011\}$. We can calculate its (homogeneous) weight enumerator to be the polynomial $H_{c_1^\perp}(x,y) = x^3 + 3xy^2$. Finally, we can compare and verify the (homogeneous) MacWilliams Identity. Recall that $H_{C_1}(x,y) = x^3 + y^3$. So,

$$\frac{1}{2}H_{c_1}(x+y, x-y) = \frac{1}{2}\left((x+y)^3 + (x-y)^3\right)$$
$$= \frac{1}{2}\left((x^3+3x^2y+3xy^2+y^3) + (x^3-3x^2y+3xy^2-y^3)\right)$$
$$= x^3 + 3xy^2 = H_{c_1^\perp}(x,y).$$

(2) The dual code to C_2 is $C_2^\perp = \{0000, 0100, 0010, 0110\}$. We can calculate its weight enumerator to be the polynomial $P(z) = 1+2z+z^2$. Finally, we can compare and verify the MacWilliams Identity. Recall that the weight enumerator of C_2 is $Q(z) = 1+2z+z^2$. So, we can verify Theorem 2.4.1 with $q=2$,

$$(1+z)^4 Q\left(\frac{1-z}{1+z}\right) = (1+z)^4\left(1+2\left(\frac{1-z}{1+z}\right)+\left(\frac{1-z}{1+z}\right)^2\right)$$
$$= (1+z)^2\left((1+z)^2 + 2(1-z)(1+z) + (1-z)^2\right)$$
$$= (1+z)^2\cdot 4 = 4(1+2z+z^2) = |C|\cdot P(z).$$

2.5. **The Weight Enumerator of Ham(3,2).**

We discussed, briefly, at the start of this chapter, Chapter 2, that the MacWilliams Identity 2.4.1 can be used to derive the Plotkin bound. We also discussed that we can used the MacWilliams Identity to extend the notion of 'dual code' to the class of non-linear binary codes that contain the zero codeword. Another application of the MacWilliams Identity is its essential role in the proof of the non-existence of the projective plane of order 10 (see [1], Chapter 13).

In order to investigate a more direct application, we will end this chapter with a worked example. We will use the MacWilliams Identity to analyse a member of a familiar family of linear codes, the Hamming codes, see [5] for a definition. We ask a simple question about the binary hamming code with redundancy 3:

"What is the weight enumerator of Ham$(3,2)$?"

We will answer this by examining the weight enumerator of the dual code $\Sigma(3,2) := \text{Ham}^{\perp}(3,2)$. In general the dual codes $\Sigma(r,q) := \text{Ham}^{\perp}(r,q)$ are known as the **Simplex codes**.

Proposition 2.5.1. *All non-zero codewords of $\Sigma(3,2)$ have weight 4. Moreover, the (homogeneous) weight enumerator is given by*

$$H_{\Sigma(3,2)}(x,y) = x^7 + 7x^3y^4.$$

Proof. From the definition of Ham$(3,2)$ code we have a specific form for the parity-check matrix i.e. a generator matrix of $\text{Ham}^{\perp}(3,2) = \Sigma(3,2)$. Such a matrix is given by

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

Since this is a generator matrix the rows form a basis for the code. We notice that each of these vectors has weight 4. Moreover, all linear combination of rows over $\mathbb{F}_2$ are given by the rows themselves and,

$$1001101 + 0101011 = 1100110$$
$$1001101 + 0010111 = 1011010$$
$$0101011 + 0010111 = 0111100$$
$$1001101 + 0101011 + 0010111 = 1110001$$

these all still have weight 4. These are all the codewords so this concludes the first part.

From the definition of the weight enumerator and what we have just proved we have

$$H_{\Sigma(3,2)}(x,y) = \sum_{c\in\Sigma(3,2)} x^{7-w(c)}y^{w(c)} = x^{7-0}y^0 + \sum_{v\in\Sigma(3,2)\setminus\{0\}} x^7y^{7-w(v)} = x^7 + 7x^3y^4.$$

□

Corollary 2.5.2. *The weight enumerator of the binary Hamming code with redundancy 3, Ham$(3,2)$ is*

$$H_{Ham(3,2)}(x,y) = x^7 + 7x^4y^3 + 7x^3y^4 + y^4.$$

Proof. We put $C = \Sigma(3,2)$ and apply the homogeneous MacWilliams Identity (Corollary 2.4.8) to find the weight enumerator of $C^{\perp} = \text{Ham}(3,2)$.

$$H_{\text{Ham}(3,2)}(x,y) = H_{C^{\perp}}(x,y) = \frac{1}{|\Sigma(3,2)|}H_{\Sigma(3,2)}(x+y, x-y)$$
$$= \frac{1}{8}\left((x+y)^7 + 7(x+y)^3(x-y)^4\right) = x^7 + 7x^4y^3 + 7x^3y^4 + y^7.$$

□

This weight enumerator encodes all the information about the weights of the codewords of the binary Hamming code with redundancy 3. We can read off the weight distribution of Ham$(3,2)$ to find $\{1,0,0,7,7,0,0,1\}$. In particular, this tells us that the codeword $1111111 \in \text{Ham}(3,2)$. In fact, we could show, more generally, that any binary Hamming code with redundancy r has the codeword $\underbrace{111\cdots1}_{\times 2^r-1}$.

References

[1] P.J. Cameron and J.H. van Lint. *Graphs, codes and designs.* Vol. 43. Cambridge University Press, 1980.

[2] R. Hill. *A first course in coding theory.* Oxford University Press, 1986.

[3] T. Honold and I. Landjev. 'MacWilliams identities for linear codes over finite Frobenius rings'. In: *Finite Fields and Applications: Proceedings of The Fifth International Conference on Finite Fields and Applications F q 5, held at the University of Augsburg, Germany, August 2–6, 1999.* Springer. 2001, pp. 276–292.

[4] G.J.O. Jameson. *The prime number theorem.* 53. Cambridge University Press, 2003.

[5] J.H. van Lint. *Introduction to Coding Theory.* Graduate Texts in Mathematics. Springer Berlin Heidelberg, 1998. ISBN: 9783540641339.

[6] F.J. MacWilliams. 'A theorem on the distribution of weights in a systematic code'. In: *Bell System Technical Journal* 42.1 (1963), pp. 79–94.

[7] F.J. MacWilliams and N.J.A. Sloane. *The theory of error-correcting codes.* Vol. 16. Elsevier, 1977.

[8] F.J. MacWilliams, N.J.A. Sloane and J-M. Goethals. 'The MacWilliams identities for non-linear codes'. In: *Bell System Technical Journal* 51.4 (1972), pp. 803–819.

[9] R. Magnus. 'Metric spaces'. In: *Metric Spaces: A Companion to Analysis.* Springer, 2021, pp. 1–27.

[10] I. Stewart. *Galois theory.* Chapman and Hall/CRC, 2022.

Appendix Appendix A Characters of Finite Abelian Groups

Here we prove that for any finite abelian group, G, we have $|G| = |\hat{G}|$.

Proposition A.0.1. *Let G be a finite, abelian group. Given $\chi_1, \chi_2 \in \hat{G}$ then the sum,*

$$\frac{1}{|G|}\sum_{g\in G}\chi_1(g)\overline{\chi_2(g)} = \begin{cases}1 & \chi_1 = \chi_2\\ 0 & \chi_1 \neq \chi_2\end{cases} \tag{OR}$$

Proof. Suppose $\chi_1 = \chi_2$. Then for all $g \in G$

$$\chi_1(g)\overline{\chi_2(g)} = \chi_1(g)\overline{\chi_1(g)} = |\chi_1(g)|^2 = 1.$$

Now, if $\chi_1 \neq \chi_2$ then we notice

$$\chi_1(g)\overline{\chi_2(g)} = \chi_1(g)\overline{\chi_2}(g) = (\chi_1\overline{\chi_2})(g) = (\chi_1\chi_2^{-1})(g).$$

Since the characters are distinct the character $\chi_1\chi_2^{-1}$ is non trivial. Hence, by our first property, Proposition 2.2.5, the sum

$$\sum_{g\in G}\chi_1(g)\overline{\chi_2(g)} = \sum_{g\in G}(\chi_1\chi_2^{-1})(g) = 0.$$

□

Remark A.0.2. The orthogonality relations in Proposition A.0.1 are a key tool that are used in the proof of Dirichlet's Theorem on primes in arithmetic progressions [4].

We also have a nice extension property. This proof requires some more sophisticated group theory compared to the rest of the paper.

Proposition A.0.3. *Let G be a finite abelian group and $H \leq G$ a subgroup. Given a character χ on H we can extend this to a character $\tilde{\chi}$ on G.*

Proof. Pick an element $a \in G\backslash H$. We extend χ to a homomorphism on the subgroup generated by H and a, $\langle H, a\rangle$. If $\langle H, a\rangle = G$ we are done. Otherwise, we repeat the process picking an element $a' \in G\backslash\langle H, a\rangle$. Eventually, we will get a character on G as the process will terminate due to G being finite.

Note that since G is abelian the subgroup H is normal [10]. Thus, we can consider the element $aH \in G/H$ the factor group. Note that since G finite G/H is finite. Thus, by Lagrange's theorem there exists an $m \in \mathbb{N}$ such that $(aH)^m = a^mH = H$. That is, $a^m \in H$.

We extend χ to a character $\chi_1 : \langle H, a\rangle \to \mathbb{T}$. Since $\chi(a^m)$ is already defined we need $\chi_1(a) = \omega$ such that $\omega^m = \chi(a^m)$. An element $g \in \langle H, a\rangle$ is of the form $g = ha^r$ for some $h \in H$ and $r \in \mathbb{Z}$. Put,

$$\chi_1(g) = \omega^r\chi(h).$$

We need to check this is well-defined. That is, write $g = h'a^s$ for some $h' \in H$ and $s \in \mathbb{Z}$ we want $\omega^r\chi(h) = \omega^s\chi(h')$. Now, $a^{s-r} = h(h')^{-1} \in H$ so $s - r = km$ for some k. Hence, $h = h'a^{km}$. Thus,

$$\chi(h) = \chi(h')\omega^{km} = \chi(h')\omega^{s-r}.$$

Multiplying through by ω^r yields $\omega^r\chi(h) = \omega^s\chi(h')$.

It suffices now to check that this is indeed a group homomorphism. Let $g_1 = h_1a^r$ and $g_2 = h_2a^s$. Then

$$\chi_1(g_1)\chi_1(g_2) = \omega^r\chi(h_1)\cdot\omega^s\chi(h_2) = \omega^r\omega^s\chi(h_1)\chi(h_2) = \omega^{r+s}\chi(h_1h_2) = \chi_1(h_1h_2a^{r+s}) = \chi_1(g_1g_2).$$

This concludes the proof. □

Corollary A.0.4. *Given distinct elements g_1, g_2 of a finite abelian group G, there exists a character $\chi \in \hat{G}$ such that $\chi(g_1) \neq \chi(g_2)$.*

Proof. Put $h = g_1g_2^{-1}$ and let h have order m i.e. $h^m = e$ where e is the identity of G. Consider the cyclic subgroup H generated by h. This subgroup has m elements $H = \{e, h, h^2, \ldots, h^{m-1}\}$.

We define a character on H by

$$\chi(h) := e^{\frac{2\pi i}{m}}.$$

Clearly, this maps into $\mathbb{T}$. It is a homomorphism since for all $h^k, h^t \in H$

$$\chi(h^kh^t) = \chi(h^{k+t}) = \chi(h)^{k+t} = \chi(h)^k\chi(h)^t = \chi(h^k)\chi(h^t).$$

Since $g_1 = hg_2$ we have

$$\chi(g_1) = \chi(h)\chi(g_2) = e^{\frac{2\pi i}{m}}\chi(g_2) \neq \chi(g_2).$$

If $H \neq G$ then we extend this character to a character on G using Proposition A.0.3.

□

We recall that we are aiming to prove that for any finite abelian group, $|G| = |\hat{G}|$. We begin with showing $|G| \leq |\hat{\hat{G}}|$, equivalently, finding an injection $G \to \hat{\hat{G}}$.

Proposition A.0.5. *Let G be a finite, abelian group. Given an element $g \in G$ we define a character, $\hat{g} : \hat{G} \to \mathbb{T}$ by setting*

$$\hat{g}(\chi) = \chi(g)$$

for all characters χ. Moreover, the map $G \to \hat{\hat{G}} : g \mapsto \hat{g}$ is injective.

Proof. The map $\hat{g}$ is a homomorphism since,

$$\hat{g}(\chi_1\chi_2) = (\chi_1\chi_2)(g) = \chi_1(g)\chi_2(g) = \hat{g}(\chi_1)\hat{g}(\chi_2).$$

This tells us we have a well defined map $G \to \hat{\hat{G}} : g \mapsto \hat{g}$.

Now, we want to show further that the map is injective. Suppose $g_1 \neq g_2$ are distinct elements of G. We want to show $\hat{g}_1 \neq \hat{g}_2$ i.e. there exists some character χ on G such that

$$\hat{g}_1(\chi) = \chi(g_1) \neq \chi(g_2) = \hat{g}_2(\chi),$$

this follows from Proposition A.0.4. □

Now, the main result:

Theorem A.0.6. *Let G be a finite abelian group. Then,*

$$|G| = |\hat{G}|.$$

Proof. Label the elements of $G = \{a_1, a_2, \ldots, a_n\}$. So, $|G| = n$. For each character $\chi \in \hat{G}$ define an element of $\mathbb{C}^n$

$$v(\chi) = (\chi(a_1), \chi(a_2), \ldots, \chi(a_n)).$$

We claim that each of these vectors is linearly independent. Using the usual (Hermitian) inner product on $\mathbb{C}^n$ we see

$$\langle v(\chi), v(\phi)\rangle = \sum_{i=1}^{n} \chi(a_i)\overline{\phi(a_i)} = \sum_{g\in G} \chi(g)\overline{\phi(g)} = \begin{cases} 1 & \chi = \phi \\ 0 & \chi \neq \phi, \end{cases}$$

for all $\chi, \phi \in G$, by the Orthogonality Relations A.0.1. Since, any linearly independent subset in $\mathbb{C}^n$ can have at most n elements we have $|\hat{G}| \leq n = |G|$.

Since, by Proposition A.0.5, $|G| \leq |\hat{\hat{G}}|$ we can put these results together to find $|G| \leq |\hat{\hat{G}}| \leq |\hat{G}|$. Hence, $|G| = |\hat{G}|$. □