Using the First Order Reed-Muller Code for Channels With Unknown Offset

Over het gebruiken van eerste orde Reed-Muller codes voor kanalen met onbekende offset

B. van Prooijen

Technische Universiteit Delft

TU Delft

Using the First Order Reed-Muller Code for Channels With Unknown Offset

Over het gebruiken van eerste orde Reed-Muller codes voor kanalen met onbekende offset

by

B. van Prooijen

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Thursday August 25, 2022 at 2:00 PM.

Student number:	4937740	
Project duration:	April 18, 2022 – August 10, 2022	
Thesis committee:	Dr. Ir. J.H. Weber,	TU Delft, supervisor
	Dr. J.A.M. de Groot,	TU Delft

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Abstract

While the Minimum Euclidean Distance detection is known to be optimal for channels affected by Gaussian noise, it has been shown that Minimum Pearson Distance detection (MPD) may perform better when the channel is also affected by an unknown offset, though for a good performance some adaptations for classical binary block codes are necessary [9]. It is shown for cosets of first order Reed-Muller codes $\mathcal{R}(1, m)$ containing words of weight $d/2$, where d is the code's distance, that the minimum Pearson distance is always low for $m \leq 4$. However, it is possible to find cosets where the minimum Pearson distance is higher for $m \geq 5$.

Contents

1

Introduction

As anyone who has interacted with other humans in the past two and a half years knows, both data transmission and storage are essential to our modern lives. In real time, we are able to speak to, connect and work together with people all over the globe with almost no delay. However, while most people have some general understanding of computers storing information “as zeros and ones”, the average person may not know, perhaps, exactly *how* this works.

In this thesis we will consider the mathematical side of the equation through the lens of coding theory. Coding theory is an application of information theory and it finds its origins in 1948 in the paper (and later book) *The Mathematical Theory of Communication* by Claude Shannon[8][10], and in the years since the field has grown to the point that we cannot live without its applications. It is fundamental to our ability to reliably transmit and store large amounts of data despite the chances of error produced by the physical circumstances all this must happen in. There are many different types of errors, like noise and burst errors among others, and the one central to this project: offset.

This last type of error is a problem in flash drives and other flash-based storage devices like solid state drives. In this kind of storage device information is stored bit by bit in floating-gate transistors called “Flash cells”. Information is written into a cell by adding an amount of charge into it corresponding to the value(s) of the particular bit(s) which will be stored in this cell. Later, the information can be read by measuring this charge and translating it back to bits [7].

However, the problem is that Flash cells experience charge leakage, that is, over a longer period of time electrons may leak out and thereby lower the charge in a cell. If the charge is lowered too much, it might become closer to a charge corresponding to another bit than the original [7]. For example, to simplify things say that a one is stored as a high charge, e.g. one as well, which then starts leaking. We later measure the charge and find 0.4, which is closer to zero than one, so we read the bit as zero. In reality the situation is a bit more complicated, but this simplified version is how we model offset.

The conventional way of detecting errors, Minimum Euclidean Distance detection, does not perform well for channels with unknown offset. However, in the article *Minimum Pearson Distance Detection for Multilevel Channels With Gain and/or Offset Mismatch* by Schouhamer Immink and Weber it was shown that Minimum *Pearson* Distance detection is in fact immune to offset entirely [4]. Later, Weber, Bu, Cai and Immink proposed three adaptations of binary block codes for which using Minimum Pearson Distance detection would show a good performance for channels affected by both noise and offset: cosets of linear codes, constant weight codes, and unordered codes [9].

In the Bachelor’s thesis *Codes for Noisy Channels with unknown offset* of Guus van Hemert each of these options was applied to a different known code and evaluated [3]. However, in this thesis we will only focus on the first option: using cosets of linear codes, specifically of the Reed-Muller code.

We will attempt to answer the following question: would cosets of Reed-Muller codes be suitable to use for channels with unknown offset? To do that, we will first introduce both the basics of Coding Theory and the findings of Weber et al. in Chapter 2, after which the knowledge needed about Reed-Muller codes will be introduced in Chapter 3. Finally, in Chapter 4 a number of statements about the Reed-Muller codes and the possible distances of the cosets are first proven in Sections 4.1 through 4.4. To conclude, the results are then summarized in Section 5.1, and evaluated, and afterwards some recommendations for further research proposed, in Section 5.2.

2

Background

To provide some insight into the workings of coding theory, the basics will first be covered in Section 2.1, which will be built upon in Section 2.2 by introducing some concepts that will be used in later chapters.

2.1. Basics of Coding Theory

As was mentioned in the Introduction, coding theory deals with the problem of transmission and storage of information through a channel, during which errors may occur in the form of noise and offset. Figure 2.1 contains a diagram that roughly shows how this transmission of information happens: A message is encoded into a codeword, which is then transmitted or stored through a channel where errors may occur, after which the receiver must decode the word they received for them to be able to read the original message.

2.1.1. Block Codes

The first step of information transmission is to encode a message into a *codeword*. These codewords belong to a set C, called a *code*. In the case that all codewords have the same length n, C is called a *block code*. All codewords in C consist of n digits, which might be zeros and ones if C is a binary code, but in general a block code is a subset of $[q]^n$, the n-dimensional vectorspace over the *alphabet* $[q]$ or $\mathbb{F}_q$, where $q = 2$ for a binary code. The number of codewords $|C|$ is called the *cardinality* or *size* of the code.

When encoding a message, one or more bits are added in a way to make error correction possible.

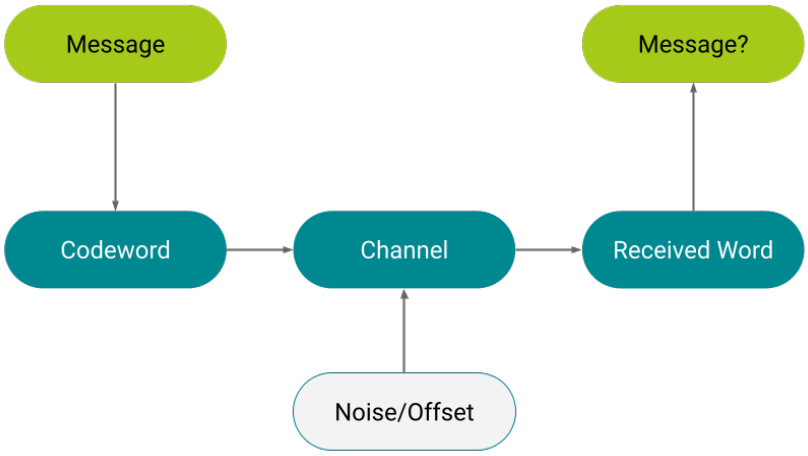

Figure 2.1: Diagram showing a model of information transmission as it is considered in coding theory.

This adds protection from errors, but also redundancy. The *information rate*

$$\frac{\log^q |C|}{n}$$

represents how much of a code contains actual information. When there are no redundant bits, the information rate will be equal to one. Now note that $|C| \leq q^n$. If we add two (redundant) bits to make error correction possible, we'll find $|C| = q^{n-2} < q^n$. So a code where error correction is possible, will never have an information rate equal to one. The importance of correcting any errors needs to be weighed up against the efficiency needed to make a code practical. In practice, available storage size and time used to send, receive and decode transmissions will provide physical limitations for how much redundancy is practical.

2.1.2. Channel

Information is transmitted through, or stored in, a channel. Some examples of channels are phone networks, CDs or DVDs, radio transmissions or hard disks and solid state drives. During transmission (or storage) noise may cause errors to occur. For example, a zero may be received when a one was sent or vice versa, or the zeros and ones are switched around in a group of digits (a burst error).

There are many ways to model noise. One such model is the Binary Symmetric Channel (BSC), where it is assumed that a. no burst errors occur, and b. each digit has the same probability p to flip from a zero to a one or vice versa. In this case the codewords received after transmission are still elements of $[q]^n$, where the code $C \subseteq [q]^n$, and an error will be detected when the codeword received is not an element of C, and, as was mentioned before, in some cases it is possible to correct the error.

However, if one were to model the noise as being normally distributed (which is then called Gaussian noise), it would be better to consider the codewords as elements of $\mathbb{R}^n$ instead. Then if we send a codeword $\mathbf{x} \in C \subseteq \mathbb{R}^n$, we might receive the word $\mathbf{r} = \mathbf{x} + \boldsymbol{\nu} \in \mathbb{R}^n$, where $\boldsymbol{\nu} = (\nu_1, \dots, \nu_n)$ and $\nu_i \sim \mathcal{N}(0, \sigma)$ for all $i = 1, \dots, n$ and some standard deviation σ. Again, if $\mathbf{r} \notin C$, the error is detected and could potentially be corrected.

When considering the code as a subset of $\mathbb{R}^n$, it is also possible to add the offset as a problem in addition to the noise. Then, if $\mathbf{x}$ is sent,

$$\mathbf{r} = \mathbf{x} + \boldsymbol{\nu} + b\mathbf{1}$$

may be received, where $\mathbf{1} \in \mathbb{R}^n$ is the all one-vector and $b\mathbf{1} \in \mathbb{R}^n$ is then the vector representing the offset, where $b \in \mathbb{R}$. Note that where noise could have a different value in each digit, the offset is defined to be the same across the entire word here. This is not always the case in practice, but it is the only case that will be considered in this project. However, there has also been other research done into coding techniques that can be used when noise and offset are modeled differently like in [1].

Remark. For both binary codewords $\mathbf{x} \in [q]^n$ and vectors $\mathbf{x} \in \mathbb{R}^n$ the same notation is used in this thesis. However, the latter is only used in the context of words affected by noise and/or offset, in particular to visualize the decoding of words in Figures 2.2 through 2.7. Whenever we speak of a "codeword" unaffected by noise or offset, it may be assumed to be an element of $[q]^n$.

2.1.3. Construction of Linear Codes

In this project a particular type of block code, the *linear* code, will be used:

Definition 2.1. "A code C is called a *linear code* if $\mathbf{v} + \mathbf{w}$ is a word in C whenever $\mathbf{v}$ and $\mathbf{w}$ are in C. That is, a linear code is a code which is closed under addition of words."(from p.27 [2])

The first thing to note is that linear codes always contain the all zero-word. The second is that a linear code C is a subspace of $[q]^n$, meaning that it has a dimension (which we will call k), and a set of linearly independent basis vectors that span C.

Definition 2.2. A *generator matrix* G for a code C is a $k \times n$ matrix whose rows form a basis for C.

A generator matrix is used to encode message words $\mathbf{m}$ of length k and turn them into codewords $\mathbf{x}$ of length n, where $\mathbf{x} = \mathbf{m}G$. That means that $C = \{\mathbf{m}G : \mathbf{m} \in [q]^k\}$, and $|C| = q^k$. So in a linear code $n - k$ bits are added, making the *information rate* $\log^q(q^k)/n = k/n$.

Note that two generator matrices generate the same code if they are row equivalent.

2.1.4. Error Correction

As has been mentioned before, adding (redundant) bits when encoding messages makes it possible for the recipient of a transmission to detect or even correct errors. The basic concept of error correction is that the recipient compares the word they have received with the codewords in the code, and conclude that the codeword that is "closest" to the word they have received, must have been the one that was sent.

Example 1. Suppose we have the code $C = \{00000, 01011, 10101, 11110\}$ and we send one word $\mathbf{x} \in C$ to the recipient through a noisy channel. They receive $\mathbf{r} = 11111$ and since they know that $\mathbf{r}$ is not a codeword, they have detected an error. They attempt to correct the error: if 00000 had been sent, five errors would have occured; for 01011 two errors, for 10101 also two errors, and for 11110 only one error would have occured. They conclude that it is *most likely* that the sender originally sent 11110, so they correct their word received to 11110.

Definition 2.3. Let $\mathbf{u}$ and $\mathbf{v}$ be words of length n. The *Hamming distance* between $\mathbf{u}$ and $\mathbf{v}$ is the number of positions in which $\mathbf{u}$ and $\mathbf{v}$ differ, denoted by $d(\mathbf{u}, \mathbf{v})$.

Furthermore, *the distance* of a code C is

$$d_{\min} = \min\{d(\mathbf{u}, \mathbf{v}) : \mathbf{u}, \mathbf{v} \in C, \mathbf{u} \neq \mathbf{v}\},$$

which is the shortest possible distance between any two codewords in C [2].

A linear code with length n, dimension k and distance $d_{\min}$ is also called a $[n, k, d_{\min}]$ code.

Example (1 cont.). $d(00000, 11111) = 5$, $d(01011, 11111) = 2$, $d(10101, 11111) = 2$, and $d(11110, 11111) = 1$. The recipient corrects the word they've received to the codeword to which it has the lowest Hamming distance. If they've received $\mathbf{r} = 11111$, they correct it to 11110.

Of course, when considering the code as a subset of $\mathbb{R}^n$, it would make more sense to use a continuous distance measure. After all, intuitively we would say that $(0,\ 0.2,\ 0)$ is closer to $(0,\ 0,\ 0)$ than it is to $(0,\ 1,\ 0)$, but the Hamming distance would not differentiate between the two options. It is better to use the squared Euclidean distance, hereafter referred to as the Euclidean distance for ease of reading, in this case; not just because it is continuous, but also because, when the noise is normally distributed (Gaussian noise), it is known that maximum likelihood decoding is achieved when using the Euclidean distance to determine the "closest" codeword for error correction[9].

Definition 2.4. Let $\mathbf{u}$ and $\mathbf{v}$ be words of length n in $\mathbb{R}^n$. Then the (squared) Euclidean distance between $\mathbf{u}$ and $\mathbf{v}$ is

$$\delta(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{n} (v_i - u_i)^2.$$

Similar to error correction using the Hamming distance, when having received the word $\mathbf{r}$, error correction using the Euclidean distance will return the codeword $\mathbf{x}$ that minimizes $\delta(\mathbf{r}, \mathbf{x})$.

The distance of a code C as defined in Definition 2.3, is a parameter of the code that measures how well error detection and correction of the code performs. Now while "*the*" distance of a code is defined using the Hamming distance, $d_{\min}$, it is not farfetched for us to also define

$$\delta_{\min} = \min_{\mathbf{u},\mathbf{v} \in C, \mathbf{u} \neq \mathbf{v}} \delta(\mathbf{u}, \mathbf{v})$$

as *the minimum Euclidean distance* of the code C, a measure for how well the code performs when using a Euclidean decoder; though note that in the binary case ($q = 2$) the Hamming distance and the

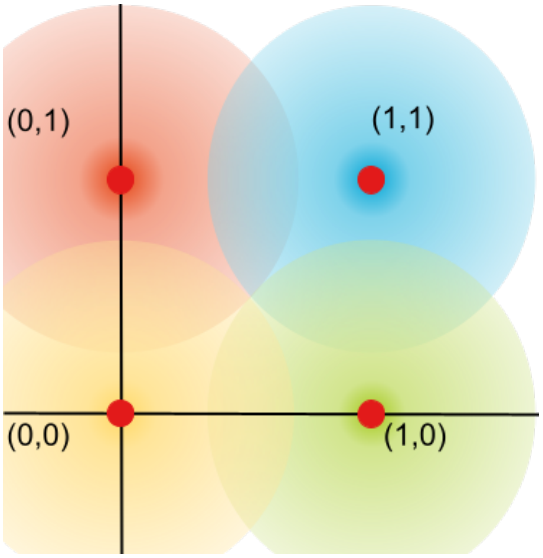

Figure 2.2: The code $C = \{00, 01, 10, 11\}$ pictured in $\mathbb{R}^2$ with "clouds" of noise around the codewords.

Euclidean distance of a code are the same.

In practice it is not always true that a code with a higher minimum distance will perform better than a code with a lower minimum distance. If for example a code has only one pair of words that have this low minimum distance between them, while all other words have a very high distance amongst themselves, the code may actually perform better than a code where all codewords have the same relatively low distance to each other.

For that reason a more suitable method for analysing the performance of a code is to use the theoretical *word error rate* or WER. The WER is defined as "the ratio of the number of incorrectly decoded words to the number of words transmitted"[1]. While it is possible to run many simulations to approximate this error rate, there are some well known upper bounds for the WER which we can use. In fact, it was shown in [4] that in channels only affected by Gaussian noise and for small σ (the noise standard deviation), the word error rate is approximately equal to:

$$\text{WER} \approx N_{\delta_{\min}} \times Q\left(\sqrt{\delta_{\min}}/(2\sigma)\right), \tag{2.1}$$

where $N_\alpha = \frac{1}{|C|}\sum_{\mathbf{u}\in C} |\{\mathbf{v} \in C : \delta(\mathbf{u}, \mathbf{v}) = \alpha\}|$ and $Q(z) = \frac{1}{\sqrt{2\pi}}\int_z^\infty e^{-u^2/2}du$.

As was mentioned before, a Euclidean decoder achieves maximum likelihood decoding when a channel is affected by Gaussian noise [9], this has been well-researched. However, the premise of this project is to research the possibilities of using cosets of linear codes to mitigate the problem of *unknown offset*, not just Gaussian noise. In the following section the effect of offset on a linear code will be shown, together with a method that may be used to circumvent the problem entirely.

2.2. Channels With Unknown Offset

Where the previous section focused on the basic definitions of coding theory, including the problem noise might pose, this section will focus on a problem specific to certain kinds of information storage: offset. But first recall that when considering the codewords from a code C as vectors in $\mathbb{R}^n$, it has been established that noise would cause a codeword $\mathbf{x}$ to change into a different word $\mathbf{r} = \mathbf{x} + \boldsymbol{\nu}$, where $\boldsymbol{\nu}$ is the result of the noise, which might be normally distributed in certain models, i.e. $\nu_i \sim \mathcal{N}(0, \sigma)$.

Example 2. An example of noise is pictured in Figure 2.2 for a code $C = \{00, 01, 10, 11\}$.

Recall that when using a Euclidean decoder, received words $\mathbf{r} \notin C$ are decoded to 00 when the

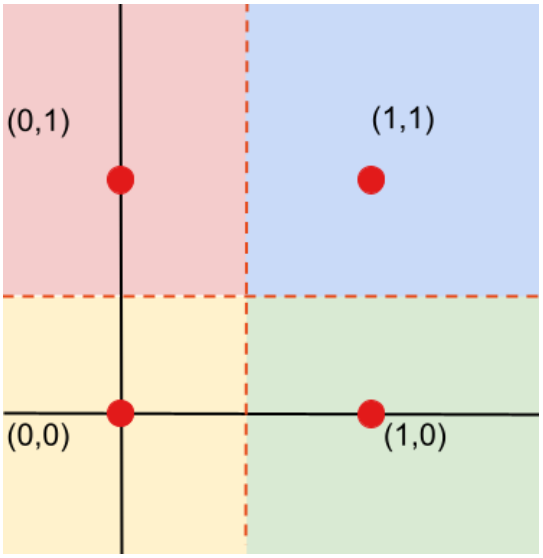

Figure 2.3: The code $C = \{00, 01, 10, 11\}$ pictured in $\mathbb{R}^2$ where each colored area around a codeword represents which words received will be corrected to that codeword when using a Euclidean decoder.

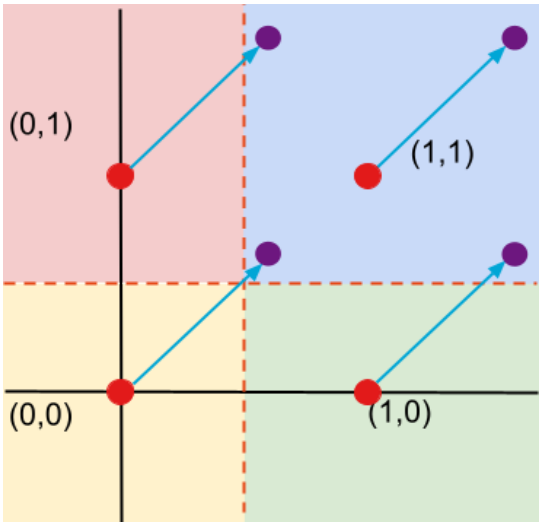

Figure 2.4: The code $C = \{00, 01, 10, 11\}$ pictured in $\mathbb{R}^2$ where each codeword has been affected by the same offset b.

distance $\delta(\mathbf{r}, 00) < \delta(\mathbf{r}, \mathbf{x})$ for $\mathbf{x} \in C$ and $\mathbf{x} \neq 00$. Each codeword has an area in $\mathbb{R}^2$ where words received will be corrected to that codeword. These areas can be seen in Figure 2.3.

2.2.1. Offset and the Modified Pearson Distance

In Section 2.1.2 we saw that in the channel model used for this project offset may turn a codeword $\mathbf{x}$ into $\mathbf{r} = \mathbf{x} + b \cdot \mathbf{1}$ after it passes through the channel, where $b \in \mathbb{R}$ is the offset which affects each digit of $\mathbf{x}$ equally; unlike noise, where each digit might change independent of the others. This b might also come from a certain probability distribution or even be a function depending on a physical parameter of some sort, but for now it will be left as an arbitrary scalar that's constant over the whole length n.

Example (2 cont.). An example of offset affecting codewords is pictured in Figure 2.4 for the code $C = \{00, 01, 10, 11\}$. Looking at the figure, it quickly becomes clear that an offset where $b > 0.5$ or $b < -0.5$ will result in error correction to the wrong word, 11 and 00 respectively, when using the Euclidean decoder.

Though it would be easy to find the word that was sent if the offset was known, that is not generally the case. Instead, a different distance measure will need to be used to find a better "closest" codeword

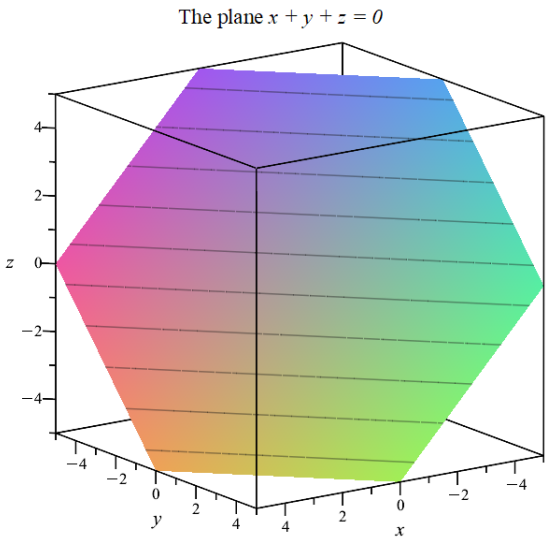

Figure 2.5: The plane $x + y + z = 0$ onto which words of length $n = 3$ are projected while determining the Pearson distance.

to correct the word received to when the channel is affected by offset.

Definition 2.5. The *modified Pearson distance* (hereafter just "Pearson distance") between two codewords $\mathbf{u}$ and $\mathbf{v}$ is

$$\delta^*(\mathbf{u}, \mathbf{v}) = \delta(\mathbf{u} - \bar{\mathbf{u}}\mathbf{1}, \mathbf{v} - \bar{\mathbf{v}}\mathbf{1})$$

where $\bar{\mathbf{u}}$ is the average of $\mathbf{u}$ [9].

We also define the minimum Pearson distance of a code C as

$$\delta^*_{\min} = \min\{\delta^*(\mathbf{u}, \mathbf{v}) : \mathbf{u}, \mathbf{v} \in C, \mathbf{u} \neq \mathbf{v}\}.$$

The Pearson distance can be seen as the distance between two "normalized" vectors, but to be a bit more precise: if the two vectors are of length n, the Pearson distance is actually the Euclidean distance between projections of the two vectors onto the plane $x_1 + x_2 + \cdots + x_n = 0$. So for $n = 3$, the words received are projected onto the plane $x + y + z = 0$, see Figure 2.5, and for $n = 2$ onto the line $x + y = 0$, see Figure 2.6. Since we only consider an offset constant over the whole vector of length n, any offset vector $b\mathbf{1}$ is orthogonal to this plane. That means that by projecting a vector affected by offset onto it, any changes caused by offset will be negated. A decoder that uses the Pearson distance is therefore immune to offset.

Example (**2 cont.**)**.** Figure 2.6 shows how words received will be corrected using the Pearson distance instead of the Euclidean distance like in Figure 2.4. The words received are projected onto the line $x + y = 0$ and then corrected to the codeword for which the projection is closest.

Looking at the example, a new problem immediately presents itself. While sending 01 and 10 will now never result in problems due to offset, any word originating from either 00 or 11 will be projected back onto 00. After all, the Pearson distance between the two is $\delta^*(00, 11) = 0$.

Recall that any linear code includes the all zero-word by definition, and many also include the all one-word. Unfortunately, that means that only for linear codes without the all one-word, the Pearson distance can be used for decoding. However, it was shown in [9] that it is possible to use *cosets* of these otherwise unsuitable linear codes.

2.2.2. Coset Codes as a Solution

Using cosets of otherwise unsuitable linear codes will remove any problems caused by the presence of both the all one-word and the all zero-word when using the Pearson distance, while still allowing us

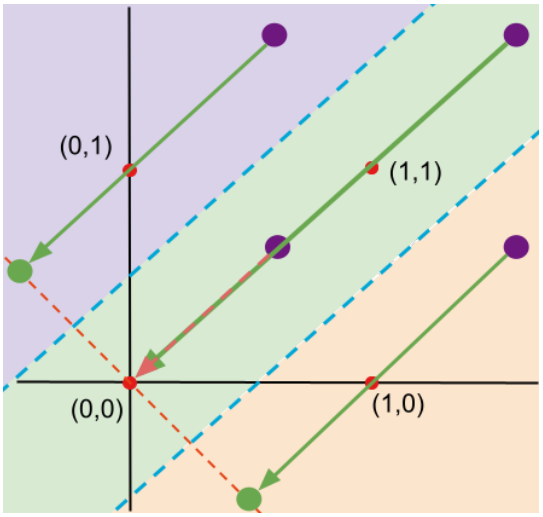

Figure 2.6: The code $C = \{00, 01, 10, 11\}$ pictured in $\mathbb{R}^2$ where each codeword has been affected by the same offset, and is projected onto the line $y = -x$ in the process of being corrected using the Pearson distance.

to use known parameters like the minimum Hamming distance of the original code. This means that the many codes that contain both the all one- and all zero-words will not need to be discounted when dealing with offset.

Definition 2.6. "If C is a linear code of length n, and if $\mathbf{u}$ is any word of length n, we define the *coset of* C *determined by* $\mathbf{u}$ to be the set of all words of the form $\mathbf{v} + \mathbf{u}$ as $\mathbf{v}$ ranges over all the words in C. We denote this coset by $C_\mathbf{u}$ or $C + \mathbf{u}$."(from p.54 [2]) So

$$C_\mathbf{u} = C + \mathbf{u} = \{\mathbf{v} + \mathbf{u} \mid \mathbf{v} \in C\}$$

Note that since linear codes are closed under addition, we have that if $\mathbf{u} \in C$, $C_\mathbf{u} = C$. As a result, the number of cosets of a given code C with length n and dimension k, is equal to 2^{n-k}, where the code itself is counted as one of the cosets. Furthermore, the Hamming distance between two words is invariant under addition, so each coset has the same minimum distance as the original code.

Example 3. Let $C = \{00, 11\}$. Then $n = 2$ and $k = 1$, so there are two cosets, including C itself. The other coset is $C_{01} = C_{10} = \{01, 10\}$. The distance of C is two, and so is the distance of C_{01}. The Pearson distance $\delta^*(00, 11) = 0$ is the minimum Pearson distance of C, so $\delta^*_{\min,C} = 0$, and for C_{01} we have $\delta^*(01, 10) = \delta^*_{\min,C_{01}} = 2$. In Figure 2.7 this coset code C_{01} is shown in $\mathbb{R}^2$ with each codeword once again affected by the same offset, only this time the modified pearson distance is able to correct each received word to its proper corresponding codeword.

Now that it has been established that it is possible to use cosets to create codes immune to offset, we may discuss the performance of these codes. Previously in Section 2.1.4 it was established that the minimum distance, be it the Hamming distance or the Euclidean distance, could be used as a measure for a code's performance. In the same way it will now be important to consider the minimum (modified) Pearson distance, $\delta^*_{\min}$ and the word error rate when using the Pearson distance (WER*). Similarly to the approximation of the WER, in [4] the WER* was shown to be approximately equal to

$$\text{WER}^* \approx N^*_{\delta^*_{\min}} \times Q\left(\sqrt{\delta^*_{\min}}/(2\sigma)\right) \tag{2.2}$$

for channels affected by Gaussian noise for small σ and by unknown offset.

However, whereas the minimum Hamming distance is generally known for a code, and thus also its cosets, the minimum Pearson distance is not. Indeed, it may even differ between cosets depending on the vectors they are determined by. To be able to say more about the minimum Pearson distance of a given coset code, we use two theorems from [9].

Firstly, a simpler way to compute the Pearson distance between two codewords:

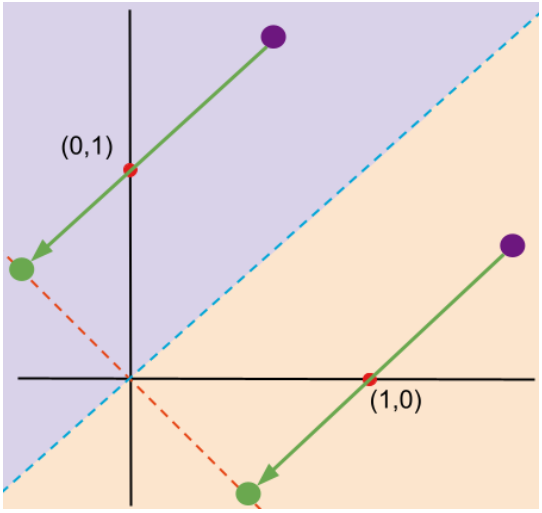

Figure 2.7: The coset code $C_{01} = \{01, 10\}$ pictured in $\mathbb{R}^2$ where each codeword has been affected by the same offset, corrected using the Pearson distance.

Theorem 2.7 (Theorem 1 in [9]). *"For any binary vectors* $\mathbf{u}$ *and* $\mathbf{v}$ *of length* n,

$$\delta^*(\mathbf{u}, \mathbf{v}) = d(\mathbf{u}, \mathbf{v}) - (N(\mathbf{v}, \mathbf{u}) - N(\mathbf{u}, \mathbf{v}))^2/n,$$

where $N(\mathbf{v}, \mathbf{u}) = |\{i : v_i = 0 \wedge u_i = 1\}|$."

Which leads us to:

Corollary 2.8 (Corollary 1 in [9]).

$$\delta^*(\mathbf{u}, \mathbf{v}) = d(\mathbf{u}, \mathbf{v}) - (d(\mathbf{u}, \mathbf{v}) - 2m(\mathbf{u}, \mathbf{v}))^2/n,$$

where $m(\mathbf{u}, \mathbf{v}) = \min\{N(\mathbf{v}, \mathbf{u}), N(\mathbf{u}, \mathbf{v})\}$.

And then secondly, a theorem that gives us a lower bound for $\delta^*_{\min}$ in cosets determined by a vector of a certain weight.

Theorem 2.9 (Theorem 2 in [9]). *"Let C be a binary* $[n, k, d_{\min}]$ *code with* $d_{\min} \geq 2$, *which contains the all-one vector, i.e.* $\mathbf{1} \in C$. *Then, for any binary vector* $\boldsymbol{\alpha}$ *of length* n *with weight* $\lfloor \frac{d_{\min}}{2} \rfloor$, $\lceil \frac{d_{\min}}{2} \rceil$, $n - \lfloor \frac{d_{\min}}{2} \rfloor$, *or* $n - \lceil \frac{d_{\min}}{2} \rceil$, *it holds that*

$$\delta^*(\mathbf{u}, \mathbf{v}) \geq d_{\min}\left(1 - \frac{d_{\min}}{n}\right) \tag{2.3}$$

for all $\mathbf{u}, \mathbf{v} \in C_{\boldsymbol{\alpha}}, \mathbf{u} \neq \mathbf{v}$."

Weber et al. use this second theorem to determine lower bounds for the minimum Pearson distance for cosets of three different families of codes: the repetition code, codes with a single parity bit and (shortened) Hamming codes [9]. In Chapter 4 the same will be done for first order Reed-Muller codes, but first the Reed-Muller code itself and its properties will be introduced in Chapter 3.

3

Reed-Muller Codes

In the article by Weber et al. the use of cosets to create codes where the Pearson distance may be used, is applied to the Repetition code, codes with a single parity bit and (shortened) Hamming codes, where lower bounds for the minimum Pearson distance is found for each of them [9]. Continuing in that direction, the same will be done for first order Reed-Muller codes in Chapter 4. In this chapter the Reed-Muller codes and their properties will be introduced.

Firstly, the construction of these codes is explained in Section 3.1. Secondly, some properties of the Reed-Muller code will be discussed in Section 3.2, so as to provide a clear image of how well the code performs normally, before venturing into coset and offset territory. Finally, we will consider what happens to the same properties when considering cosets of our code in Section 3.3.

Before that, however, it is necessary to introduce a base idea of what Reed-Muller codes are. One definition of a Reed-Muller code is:

Definition 3.1. "The r^{th} order binary Reed-Muller (or RM) code $\mathcal{R}(r,m)$ of length $n = 2^m$, for $0 \leq r \leq m$, is the set of all vectors $\mathbf{f}$, where $f(v_1, \dots, v_m)$ is a Boolean function which is a polynomial of degree at most r."[6]

Remark. A Boolean function (or B.F.) is any function that only takes on the values 0 and 1, where logical operations are translated into binary form in the following way:

$$f \text{ XOR } g = f + g$$
$$f \text{ AND } g = fg$$
$$f \text{ OR } g = f + g + fg$$
$$\text{NOT } f = \bar{f} = 1 + f$$

Furthermore it is known that any Boolean function $f(v_1, \dots, v_m)$ can be expressed as the sum of the 2^m functions

$$1,\ v_1,\ v_2, \dots,\ v_m,\ v_1v_2,\ v_1v_3, \dots,\ v_{m-1}v_m, \dots,\ v_1v_2 \cdots v_m$$

Then a Boolean function $f(v_1, \dots, v_m) = 1 \cdot v_1v_2 \cdots v_m + \ \dots$ is a polynomial of degree m [6].

Example 4. An RM-code of order $r = 2$ and $m = 3$ would have codewords of the form

$$a_0\mathbf{1} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + a_3\mathbf{v_3} + a_{1,2}\mathbf{v_1}\mathbf{v_2} + a_{1,3}\mathbf{v_1}\mathbf{v_3} + a_{2,3}\mathbf{v_2}\mathbf{v_3}, \quad a_i, a_{i,j} = 0 \text{ or } 1.$$

3.1. Construction

From the above it becomes clear that an RM-code has $k = 1 + \binom{m}{1} + \binom{m}{2} + \cdots + \binom{m}{r}$ basis vectors. Though the exact order does not matter for the final result, in general the first $m + 1$ vectors, which are then used to calculate the others, are always of the form: all digits one; the first half is zero, the second half one; four quarters, alternately all zero or one; etc. until the groups of zeros and ones cannot be divided further. See also Example 5.

Example 5. The first $m+1$ basis vectors for $m=2$ are

$$\begin{array}{ll} \mathbf{1} & 1\ 1\ 1\ 1 \\ \mathbf{v_1} & 0\ 0\ 1\ 1 \\ \mathbf{v_2} & 0\ 1\ 0\ 1 \end{array} \tag{3.1}$$

and for $m=3$

$$\begin{array}{ll} \mathbf{1} & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\ \mathbf{v_1} & 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1 \\ \mathbf{v_2} & 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1 \\ \mathbf{v_3} & 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1 \end{array} \tag{3.2}$$

The generator matrix for $\mathcal{R}(1,2)$ contains the vectors $\mathbf{1}$, $\mathbf{v_1}$, $\mathbf{v_2}$ as given in (3.1). Indeed, since $k = 1 + \binom{2}{1} = 3$, we know that the generator matrix should have three rows.

For Example 4, the different rows of the generator matrix of $\mathcal{R}(2,3)$ are:

$$\begin{array}{ll} \mathbf{1} & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1 \\ \mathbf{v_1} & 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1 \\ \mathbf{v_2} & 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1 \\ \mathbf{v_3} & 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1 \\ \mathbf{v_1v_2} & 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1 \\ \mathbf{v_1v_3} & 0\ 0\ 0\ 0\ 0\ 1\ 0\ 1 \\ \mathbf{v_2v_3} & 0\ 0\ 0\ 1\ 0\ 0\ 0\ 1 \end{array} \tag{3.3}$$

The vectors $\mathbf{a} = (a_0,\ a_1, \ldots, a_{2,3})$ that we see in Example 4, are the original message words of length seven.

In the next theorem an alternative way of constructing Reed-Muller codes is given, for which we use the following notation: for $\mathbf{u} = 0101$ and $\mathbf{v} = 1010$, we have $|\mathbf{u}|\mathbf{v}| = 01011010$.

Theorem 3.2. *[Theorem 13.2 in [6]]*

$$\mathcal{R}(r+1, m+1) = \{|\mathbf{u}|\mathbf{u}+\mathbf{v}| : \mathbf{u} \in \mathcal{R}(r+1,m), \mathbf{v} \in \mathcal{R}(r,m)\},$$

where

$$\mathcal{R}(1,1) = \{00, 01, 11, 10\}, \qquad \mathcal{R}(0,1) = \{00, 11\}.$$

Or equivalently, if $G(r,m)$ is the generator matrix of $\mathcal{R}(r,m)$,

$$G(r+1, m+1) = \begin{pmatrix} G(r+1,m) & G(r+1,m) \\ 0 & G(r,m) \end{pmatrix},$$

where

$$G(1,1) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \qquad G(0,1) = \begin{pmatrix} 1 & 1 \end{pmatrix},$$

and for any $G(r,m)$ where $r > m$, we define $G(r,m) := G(m,m)$.

Example 6. To construct a generator matrix $G(2,3)$ for $\mathcal{R}(2,3)$ using Theorem 3.2, first generator matrices for $\mathcal{R}(2,2)$ and $\mathcal{R}(1,2)$, $G(2,2)$ and $G(1,2)$ resp., need to be known.

Using Theorem 3.2 we find

$$\begin{aligned} G(1,2) &= \begin{pmatrix} G(1,1) & G(1,1) \\ 0 & G(0,1) \end{pmatrix} \\ &= \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \end{aligned}$$

and using the same method to construct $G(2,2)$, we find that

$$G(2,2) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Then using Theorem 3.2, we find

$$G(2,3) = \left(\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right). \tag{3.4}$$

Comparing the vectors in (3.3) and the rows of (3.4), it is clear that the two different methods generate the same code $\mathcal{R}(2,3)$.

3.2. Properties of the Reed-Muller Code

One important thing to note is that a binary Reed-Muller code where $r = m$ is in fact equal to the whole vector-space $[2]^n$, where $n = 2^m$, instead of a proper subset. Indeed, computing the reduced row echelon form of $G(2,2)$ as given in the matrix (6), shows that it is equivalent to the identity matrix. That means that using cosets to create a code immune to offset will not work in the case $r = m$, since the code itself is the only coset and it contains both the all one- and all zero-word. Therefore only cases where $r < m$ will ever be considered going forward.

In Section 2.1.4 we already discussed the distance of a code as a basic measure for how well it performs. For Reed-Muller codes we know this distance for every possible value of r and m:

Theorem 3.3 (Theorem 13.3 in [6])**.** *A Reed-Muller code $\mathcal{R}(r,m)$ has distance $d_{\min} = 2^{m-r}$.*

Example 7. Recall from Example 6 that a generator matrix of the code $C = \mathcal{R}(1,2)$ is

$$G(1,2) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

Then if we multiply the eight message words $000, 001, 010, 011, 100, \ldots$ with the matrix we find the code $C = \{0000, 0101, 0011, 0110, 1111, 1010, 1100, 1001\}$. According to Theorem 3.3 the minimum Hamming distance we will find in this code is $d_{\min} = 2^{2-1} = 2$, and indeed, we can easily find a pair of words such that the distance between them is 2, but there is no pair of words such that the distance between them is smaller. In fact, each word has exactly one word to which the distance is *higher* than $d_{\min} = 2$, and that is the pairs $\mathbf{x}, \mathbf{x} + 1111 \in C$, where the distance is $d(\mathbf{x}, \mathbf{x} + 1111) = 4$. This result is pictured in Figure 3.1.

For the $\mathcal{R}(1,3)$ and $\mathcal{R}(2,3)$ codes the distances between the different codewords are pictured in Figure 3.2. The distance of an RM-code with order $r = 1$ and $m = 3$ is equal to $2^{3-1} = 4$, and for $r = 2$ and $m = 3$ the distance is again equal to 2. Figure 3.2 also shows how an RM-code of order two differs from a code of order one in structure. The distance between two words from an RM-code of order one, can only be equal to the length of the words 2^m or the distance of the code 2^{m-r}, while in a second order RM-code the distance between words varies between those two values.

In fact, the possible distances between words coincide with the possible weights of codewords [6], which in the second order RM-code's case are known to be of the form $2^{m-1} + \epsilon 2^l$, where $\epsilon = 0, -1, 1$ and $m/2 - 1 \leq l \leq m - 1$ [5]. So for the $\mathcal{R}(2,3)$ code that means that the possible distances are $2, 4, 6$ and 8, while for the $\mathcal{R}(1,3)$ code the only possible distances are 4 and 8, because (except for the all zero-word) all words have weights 4 or 8.

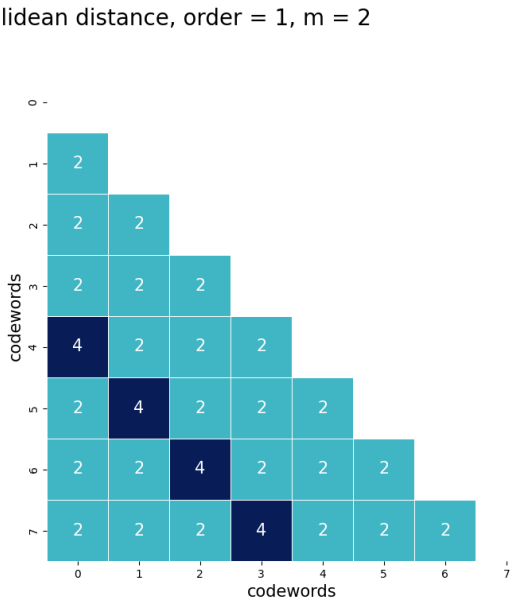

Figure 3.1: A heatmap showing the Euclidean distance between each pair of codewords in a first order Reed-Muller code with $m = 2$.

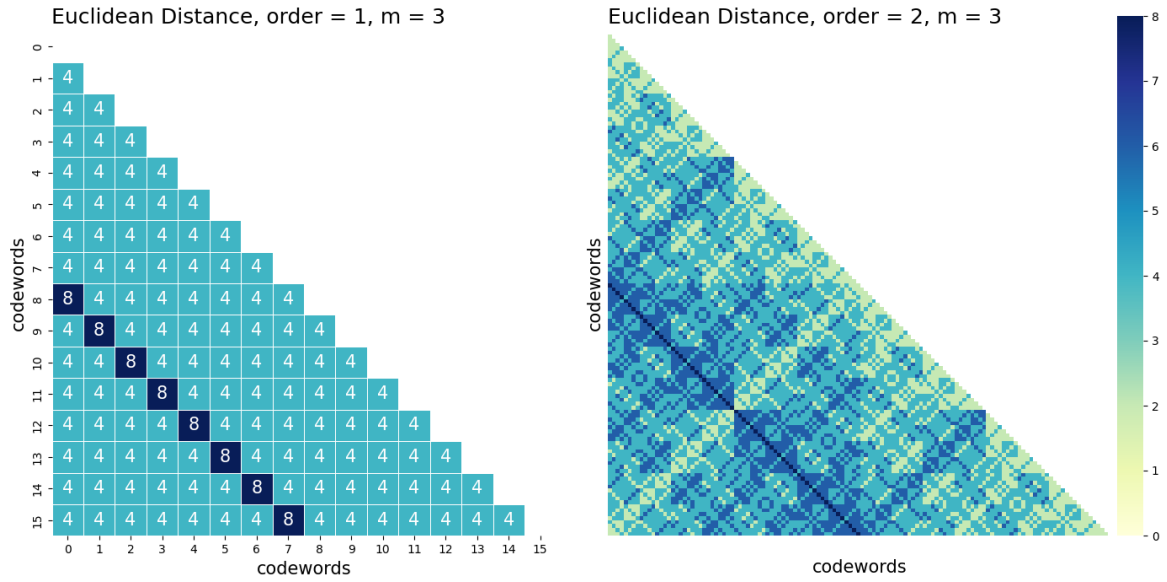

Figure 3.2: A heatmap showing the Euclidean distance between each pair of codewords in a first (left) and second (right) order Reed-Muller code with $m = 3$ (the latter contains 128 codewords).

If we make the same figures for the codes $\mathcal{R}(1,2)$, $\mathcal{R}(1,3)$ and $\mathcal{R}(2,3)$ but for the Pearson distance (3.3 (left) and 3.4), we see that the Pearson distance is often equal to the Hamming distance for many of the codewords, but for others the Pearson distance is lower. Now we also see that the Pearson distance between the all one- and all zero-word is indeed zero, meaning that we can't differentiate between the two words during decoding, so the next step is to look at the Reed-Muller codes' cosets.

3.3. Cosets

In the previous section we saw the problem that the all zero- and all one-words pose once again, though this time for the Reed-Muller codes. So as was explained in Section 2.2.2, to be able to use the Pear-

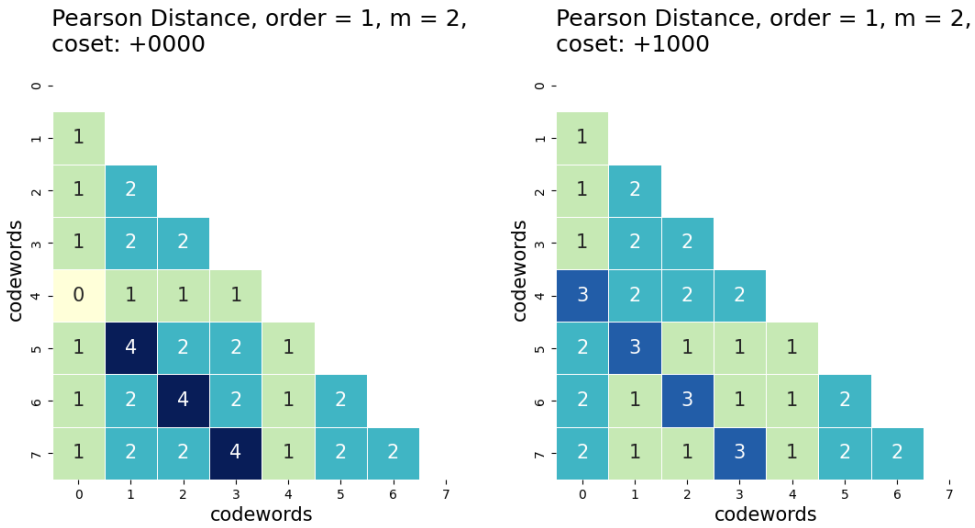

Figure 3.3: A heatmap showing the Pearson distance between the codewords in the first order RM-code with $m = 2$ (left) and its coset (right).

son distance in combination with Reed-Muller codes when a channel is affected by offset, we will need to investigate how well the RM-codes' cosets could perform. In this section the minimum Pearson distance for a small number of cosets is found to provide us with some level of intuition to use later when looking for a possibly better lower bound of this distance.

First, recall that the dimension of an RM-code $\mathcal{R}(r, m)$ is $k = \sum_{i=0}^{r} \binom{m}{i}$ and that the total number of cosets for any code with words of length n is equal to 2^{n-k}. For $\mathcal{R}(1, 2)$, which has dimension $k = 3$, that means that there are 2 cosets, including the code itself.

Example 8. Let $C = \mathcal{R}(1, 2)$ and $\boldsymbol{\alpha} = 0001$. Note that $\boldsymbol{\alpha} \notin C$, so the coset $C_{\boldsymbol{\alpha}} = \{\mathbf{u} + \boldsymbol{\alpha} | \mathbf{u} \in C\} \neq C$, so we have found its only coset. As was mentioned before, the Hamming and Euclidean distance between the codewords does not change from the original code to a coset, but we can see in Figure 3.3 on the right that the Pearson distance does change. The minimum Pearson distance of this coset is $\delta^*_{\min} = 1$.

We can see a similar thing happening for $\mathcal{R}(1, 3)$ and three of its cosets in Figure 3.5. For each coset (here determined by the vectors 1000.0000, 1100.0000, and 1110.0000) we can see a different pattern appear. It does seem like the words that originally had a Hamming distance of $n = 8$ to each other still have a relatively high Pearson distance to each other, regardless of the choice of the vector that determines the coset. For all three of these cosets the minimum Pearson distance found is equal to $\delta^*_{\min} = 2$, though note that these are not all the cosets of $\mathcal{R}(1, 3)$, there being $2^{(8-4)} = 16$ cosets in total.

Remark. At this point, we've found a minimum Pearson distance $\delta^*_{\min}$ equal to 1 for the coset of $\mathcal{R}(1, 2)$ and equal to 2 for three cosets of $\mathcal{R}(1, 3)$. Now recall what we know from Theorem 2.9: for a $[n, k, d_{\min}]$ code C with $d_{\min} \geq 2$ that contains the all one vector, for any $\boldsymbol{\alpha}$ of length n with weight $\lfloor \frac{d_{\min}}{2} \rfloor$, $\lceil \frac{d_{\min}}{2} \rceil$, $n - \lfloor \frac{d_{\min}}{2} \rfloor$, or $n - \lceil \frac{d_{\min}}{2} \rceil$, it holds that

$$\delta^*(\mathbf{u}, \mathbf{v}) \geq d_{\min} \left(1 - \frac{d_{\min}}{n}\right)$$

for all $\mathbf{u}, \mathbf{v} \in C_{\boldsymbol{\alpha}}, \mathbf{u} \neq \mathbf{v}$.

For $\mathcal{R}(1, 2)$ we have $d_{\min} = 2$ and $n = 4$. So for a vector $\boldsymbol{\alpha}$ of weight 1 or 3 that determines the coset, we should find $\delta^*_{\min} \geq 1$. And similarly for $\mathcal{R}(1, 3)$ we find $\delta^*_{\min} \geq 2$ for a coset $C_{\boldsymbol{\alpha}}$, since $d_{\min} = 4$

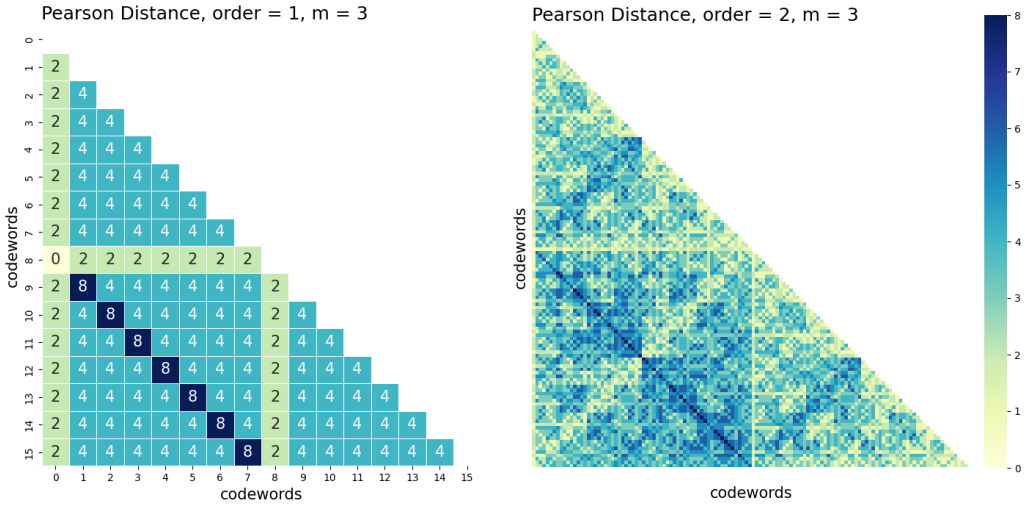

Figure 3.4: A heatmap showing the Pearson distance between the codewords in a first (left) and second (right) order RM-code with $m = 3$.

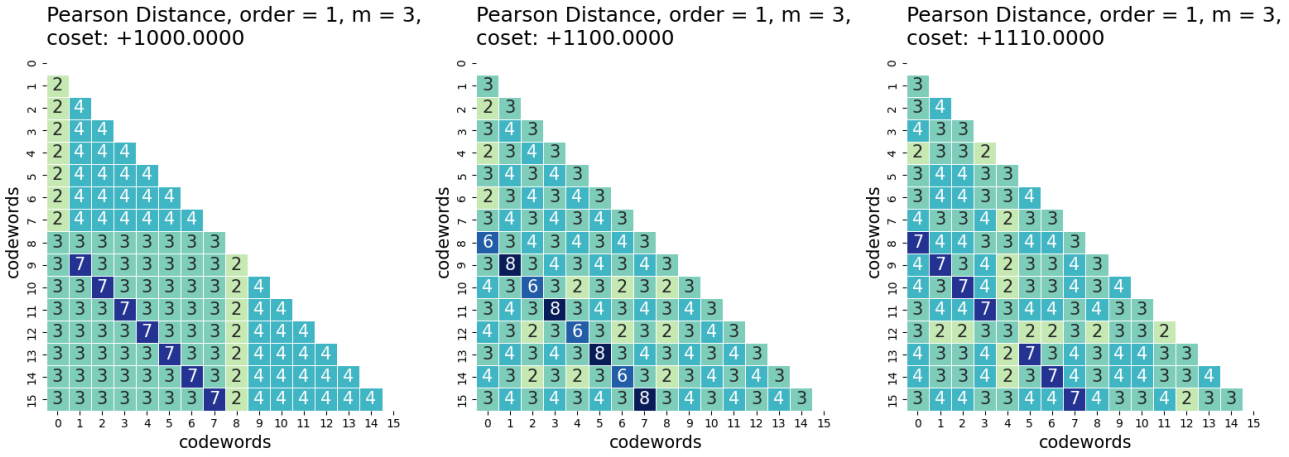

Figure 3.5: A heatmap showing the Euclidean distance and Pearson distance between all codewords in three cosets of the first order Reed-Muller code with $m = 3$.

and $n = 8$ and where $wt(\boldsymbol{\alpha}) = 2$ or $wt(\boldsymbol{\alpha}) = 6$. So in Example 8 we have in fact found cosets where there is at least one pair of words $\mathbf{u}, \mathbf{v} \in C_{\boldsymbol{\alpha}}$ such that

$$\delta^*(\mathbf{u}, \mathbf{v}) = d_{\min}\left(1 - \frac{d_{\min}}{n}\right),$$

namely for the cases $\boldsymbol{\alpha} = 0001$ for $\mathcal{R}(1,2)$ and $\boldsymbol{\alpha} = 1100.0000$ for $\mathcal{R}(1,3)$.

This raises the question whether this equality *always* holds for all m, r and all cosets determined by $\boldsymbol{\alpha}$'s of these weights — and potentially other weights if we consider the other two cosets where we found the same value. Unfortunately, the latter falls outside of the scope of this project, as we are basing our arguments on Theorem 2.9, but fortunately it turns out that the former is *not* true, as we will soon discover in Chapter 4.

4

Results

In the previous chapter the suspicion arose that cosets of Reed-Muller codes determined by $\boldsymbol{\alpha}$'s of the weights $\lfloor \frac{d_{\min}}{2} \rfloor$, $\lceil \frac{d_{\min}}{2} \rceil$, $n - \lfloor \frac{d_{\min}}{2} \rfloor$, and $n - \lceil \frac{d_{\min}}{2} \rceil$, might always have a minimum Pearson distance equal to the lower bound determined in Theorem 2.9. Fortunately, this will turn out not to be the case.

Remark. In this chapter all the various properties of the codes and their cosets will be written in terms of m and r. So for example $n = 2^m$ and $d_{\min} = 2^{m-r}$. If the value for m or r is known, this will also be substituted into the different properties; so for example for $r = 1$, we use $d_{\min} = 2^{m-1}$. Then Theorem 2.9 can be written as:

> Let $C = \mathcal{R}(r, m)$ where $r < m$. Then for any binary vector $\boldsymbol{\alpha}$ of length 2^m and weight $d_{\min}/2 = 2^{m-r-1}$ or $n - d_{\min}/2 = 2^m - 2^{m-r-1} = 2^m\left(1 - 2^{-(r+1)}\right)$, it holds that
>
> $$\delta^*(\mathbf{u}, \mathbf{v}) \geq 2^{m-r}\left(1 - 2^{-r}\right) \tag{4.1}$$
>
> for all $\mathbf{u}, \mathbf{v} \in C_{\boldsymbol{\alpha}}$, where $\mathbf{u} \neq \mathbf{v}$.

Furthermore, unless stated otherwise any $\boldsymbol{\alpha}$ may be assumed to be of length 2^m and the weights $wt(\boldsymbol{\alpha}) = 2^{m-r-1}$ or $wt(\boldsymbol{\alpha}) = 2^m\left(1 - 2^{-(r+1)}\right)$, though this will often be repeated for clarity's sake.

The goal of this chapter is to find lower bounds for the minimum Pearson distance of cosets of Reed-Muller codes determined by $\boldsymbol{\alpha}$'s of weight 2^{m-r-1} or $2^m\left(1 - 2^{-(r+1)}\right)$. While the first lemma that will be proven here holds for all orders r, the main results only hold for first order Reed-Muller codes.

First, two lemmas about the structure of Reed-Muller codes and of its codewords will be proven in Section 4.1. These are then used in Section 4.2 and 4.3 to show that 1. equality is always reached in equation (4.1) for cosets determined by $\boldsymbol{\alpha}$ of first order RM-codes where $m \leq 4$; and 2. for higher values of m it is possible to find a coset determined by some $\boldsymbol{\alpha}$ for which a *higher* minimum Pearson distance is found. Finally, we also find an upper bound for that minimum Pearson distance in Section 4.4.

4.1. The Structure of Reed-Muller Codes

The first thing we will show in this section is that the observation made in Example 8 is true: that codewords in the coset that have Hamming distance 2^m to each other also have a relatively high Pearson distance to each other.

Lemma 4.1. *Let $C = \mathcal{R}(r, m)$. Then for all $\boldsymbol{\alpha}$ and for all $\mathbf{u}, \mathbf{v} \in C_{\boldsymbol{\alpha}}$, if $d(\mathbf{u}, \mathbf{v}) = 2^m$,*

$$\delta^*(\mathbf{u}, \mathbf{v}) \geq \frac{3}{2} \cdot 2^{m-r}.$$

Proof. Let $\boldsymbol{\alpha}$ have weight 2^{m-r-1} or $2^m\left(1-2^{-(r+1)}\right)$, and let $\mathbf{u},\mathbf{v} \in C_{\boldsymbol{\alpha}}$. Suppose $d(\mathbf{u},\mathbf{v}) = 2^m$. Then $\mathbf{u}$ and $\mathbf{v}$ differ in all 2^m positions, so $N(\mathbf{u},\mathbf{v}) = wt(\mathbf{v})$ and since $\mathbf{u} = \mathbf{v} + \mathbf{1}$, we also have $N(\mathbf{v},\mathbf{u}) = wt(\mathbf{u}) = 2^m - wt(\mathbf{v})$.
We know that for any word $\mathbf{c} \in C_{\boldsymbol{\alpha}}$, the weight of $\mathbf{c}$ is $2^{m-r-1} \le wt(\mathbf{c}) \le 2^m\left(1-2^{-(r+1)}\right)$, so

$$m(\mathbf{u},\mathbf{v}) = \min\{N(\mathbf{v},\mathbf{u}), N(\mathbf{u},\mathbf{v})\} \ge 2^{m-r-1}.$$

Then we see that

$$\begin{aligned}\delta^*(\mathbf{u},\mathbf{v}) &= d(\mathbf{u},\mathbf{v}) - (d(\mathbf{u},\mathbf{v}) - 2m(\mathbf{u},\mathbf{v}))^2/2^m \qquad (4.2)\\ &\ge 2^m - \frac{(2^m - 2^{m-r})^2}{2^m}\\ &= 2^{m-r}(2-2^{-r})\\ &\ge \frac{3}{2}\cdot 2^{m-r}.\end{aligned}$$

Where equation (4.2) comes from Corollary 2.8, and the first inequality follows from the result $m(\mathbf{u},\mathbf{v}) \ge 2^{m-r-1}$. □

Then, since we know that $d_{\min} \ge \delta^*_{\min}$, we find that if $d(\mathbf{u},\mathbf{v}) = 2^m$, then $\delta^*(\mathbf{u},\mathbf{v}) \ge \frac{3}{2}d_{\min} > \delta^*_{\min}$.

Remark. For first order Reed-Muller codes the only possible distances between two words $\mathbf{u},\mathbf{v}$ are 2^m or 2^{m-1}. So from Lemma 4.1 we know that if $d(\mathbf{u},\mathbf{v}) = \delta^*_{\min}$, we must have $d(\mathbf{u},\mathbf{v}) = 2^{m-1}$.

The next lemma provides us with a more precise description of what the codewords in a first order RM-code look like. The idea is that, because RM-codes can be constructed using recursion (see Theorem 3.2), we need only consider the "rules" of the recursion and how its basis, $\mathcal{R}(1,2)$, looks.

Lemma 4.2. *A codeword* $\mathbf{u} \in \mathcal{R}(1,m)$ *(*$m > 1$*) is built up of* $2^m/4 = 2^{m-2}$ *"blocks" of length 4: either of* 2^{m-2} *blocks* $\mathbf{b}$ *and zero blocks* $\mathbf{b}+1111$*, or of* $2^{m-2}/2$ *blocks* $\mathbf{b}$ *and* $2^{m-2}/2$ *blocks* $\mathbf{b}+1111$*, where* $\mathbf{b}$ *may be any of the following:*

$$\begin{array}{cc} 0000 & 1111\\ 0011 & 1100\\ 0101 & 1010\\ 0110 & 1001\end{array}$$

Proof. Proof by induction over m.
The generator matrix for an order 1, $m = 2$ RM code is

$$G(1,2) = \begin{pmatrix}1&1&1&1\\0&0&1&1\\0&1&0&1\end{pmatrix},$$

so

$$\mathcal{R}(1,2) = \{0000, 0011, 0101, 0110, 1111, 1100, 1010, 1001\}.$$

For this code $2^{m-2} = 1$ and we see that each codeword is equal to one of the eight blocks proposed above.

For $m = 3$ we have the generator matrix

$$G(1,3) = \begin{pmatrix}1&1&1&1&1&1&1&1\\0&0&0&0&1&1&1&1\\0&0&1&1&0&0&1&1\\0&1&0&1&0&1&0&1\end{pmatrix},$$

and then the code is (with periods added for ease of reading):

$$\begin{aligned}\mathcal{R}(1,3) = \{&0000.0000, 0101.0101, 0011.0011, 0110.0110,\\ &0000.1111, 0101.1010, 0011.1100, 0110.1001,\\ &1111.1111, 1010.1010, 1100.1100, 1001.1001,\\ &1111.0000, 1010.0101, 1100.0011, 1001.0110\},\end{aligned}$$

which shows that a codeword is either built up of $2^{m-2} = 2$ of the same blocks, or $2^{m-2}/2 = 1$ block $\mathbf{b}$ and $2^{m-2}/2 = 1$ block $(\mathbf{b} + 1111)$.

Now suppose that the lemma holds for an $\mathcal{R}(1, m-1)$ code. From Theorem 3.2 we know that

$$\mathcal{R}(1,m) = \{|\mathbf{u}|\mathbf{u}+\mathbf{v}| : \mathbf{u} \in \mathcal{R}(1,m-1), \mathbf{v} \in \mathcal{R}(0,m-1)\}. \tag{4.3}$$

Recall that $\mathcal{R}(0, m-1) = \{00 \ldots 0, 11 \ldots 1\}$, so we can rewrite (4.3) to

$$\mathcal{R}(1,m) = \{|\mathbf{u}|\mathbf{u}| : \mathbf{u} \in \mathcal{R}(1,m-1)\} \cup \{|\mathbf{u}|\mathbf{u}+\mathbf{1}| : \mathbf{u} \in \mathcal{R}(1,m-1)\}.$$

By the induction hypothesis any $\mathbf{u} \in \mathcal{R}(1, m-1)$ is already of the correct form, with 2^{m-3} blocks $\mathbf{b}$, or $2^{m-3}/2$ blocks $\mathbf{b}$ and $(\mathbf{b} + 1111)$ each. Then $|\mathbf{u}|\mathbf{u}|$ either consists of $2 \cdot 2^{m-3} = 2^{m-2}$ blocks $\mathbf{b}$, or of $2 \cdot 2^{m-3}/2 = 2^{m-2}/2$ blocks $\mathbf{b}$ and $(\mathbf{b} + 1111)$ each. And similarly $|\mathbf{u}|\mathbf{u} + \mathbf{1}|$ consists of $2^{m-2}/2$ blocks $\mathbf{b}$ and $(\mathbf{b} + 1111)$. So all codewords in $\mathcal{R}(1, m)$ are of the form all $\mathbf{b}$, or half $\mathbf{b}$ and half $\mathbf{b} + 1111$. □

Now we will use these results to prove in the next section that the suspicion that equality is reached in equation (4.1) for cosets of first order Reed-Muller codes with $m \leq 4$ is true. In Section 4.3 we will see that this is not always the case for higher m.

4.2. The Minimum Pearson Distance for Order 1 $m \leq 4$ RM-Codes

At the end of Section 3.3 we found in Example 8 that for a number of cosets $C_{\boldsymbol{\alpha}}$ the minimum Pearson distance found was actually equal to the general lower bound found by Weber et al. in [9]. In this section we will show that this is in fact the case for all cosets $C_{\boldsymbol{\alpha}}$ of $\mathcal{R}(1, m)$, where $m \leq 4$ and $wt(\boldsymbol{\alpha}) = 2^{m-2}$ or $wt(\boldsymbol{\alpha}) = 3 \cdot 2^{m-2}$.

First, it is necessary to develop some intuition for when this equality is reached in general. By Corollary 2.8, the Pearson distance between two words $\mathbf{u}, \mathbf{v}$ is equal to

$$\delta^*(\mathbf{u},\mathbf{v}) = d(\mathbf{u},\mathbf{v}) - (d(\mathbf{u},\mathbf{v}) - 2m(\mathbf{u},\mathbf{v}))^2/n,$$

and by Theorem 2.9 the lower bound for the minimum Pearson distance in a coset $C_{\boldsymbol{\alpha}}$ where $\boldsymbol{\alpha}$ has the appropriate weights as stated above, is

$$\delta^*_{\min} \geq 2^{m-r}\left(1 - 2^{-r}\right).$$

When $d(\mathbf{u},\mathbf{v}) = 2^{m-r}$ or $d(\mathbf{u},\mathbf{v}) = 2^m\left(1 - 2^{-r}\right)$ (which is the same value for $r = 1$) and $m(\mathbf{u},\mathbf{v}) = 0$, we find $\delta^*(\mathbf{u},\mathbf{v}) = 2^{m-r}\left(1 - 2^{-r}\right)$, which is then equal to the minimum Pearson distance of that coset.

But what does it mean for us to find $m(\mathbf{u},\mathbf{v}) = 0$? Recall that $m(\mathbf{u},\mathbf{v}) = \min\{N(\mathbf{u},\mathbf{v}), N(\mathbf{v},\mathbf{u})\}$, where $N(\mathbf{u},\mathbf{v}) = |\{i : u_i = 0 \wedge v_i = 1\}|$. So when $N(\mathbf{u},\mathbf{v}) = 0$, it means that wherever $\mathbf{u}$ has the digit zero, $\mathbf{v}$ will also have the digit zero.

Example 9. Let

$$\mathbf{u} = 0011 \text{ and } \mathbf{v} = 0001.$$

Then $N(\mathbf{u},\mathbf{v}) = 0$ and $N(\mathbf{u},\mathbf{v}) = 1$. Indeed, wherever $u_i = 0$, we also have $v_i = 0$.

So now consider $C = \mathcal{R}(1, 2)$. Suppose that $\boldsymbol{\alpha} = 1000$, and $\mathbf{u} = 0011 \in C$. We find the coset $C_{\boldsymbol{\alpha}}$, such that $\boldsymbol{\alpha} \in C_{\boldsymbol{\alpha}}$ and $\mathbf{u} + \boldsymbol{\alpha} = 1011 \in C_{\boldsymbol{\alpha}}$. $N(\mathbf{u} + \boldsymbol{\alpha}, \boldsymbol{\alpha}) = 0$, since wherever a digit in $\mathbf{u} + \boldsymbol{\alpha}$ is zero, that same digit is zero in $\boldsymbol{\alpha}$. So $m(\mathbf{u} + \boldsymbol{\alpha}, \boldsymbol{\alpha}) = 0$ and furthermore, we have found a pair of codewords in the coset such that

$$\delta^*(\mathbf{u} + \boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-r}\left(1 - 2^{-r}\right) = 1,$$

so this coset has a minimum Pearson distance equal to the lower bound from Theorem 2.9.

While it is possible to look at all words $\mathbf{u}+\boldsymbol{\alpha}$ in the coset $C_{\boldsymbol{\alpha}}$ to see which have the property $(\mathbf{u}+\boldsymbol{\alpha})_i = 0 \Rightarrow \alpha_i = 0$, we can also rewrite that to

$$\begin{aligned} N(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 0 &\Leftrightarrow ((\mathbf{u}+\boldsymbol{\alpha})_i = 0 \Rightarrow \alpha_i = 0) \\ &\Leftrightarrow (\alpha_i = 1 \Rightarrow (\mathbf{u}+\boldsymbol{\alpha})_i = 1) \\ &\Leftrightarrow (\alpha_i = 1 \Rightarrow u_i = 0). \end{aligned} \tag{4.4}$$

Then we can show the following Lemma:

Lemma 4.3. *Let* $\mathbf{u} \in C = \mathcal{R}(1,m)$ *and let* $\boldsymbol{\alpha}$ *have weight* 2^{m-2} *or* $3 \cdot 2^{m-2}$*. Then if* $wt(\mathbf{u}) = 2^{m-1}$ *and if*

$$(\alpha_i = 1 \Rightarrow u_i = 0) \text{ for all } i = 1, \dots, 2^m$$

or

$$(\alpha_i = 0 \Rightarrow u_i = 0) \text{ for all } i = 1, \dots, 2^m,$$

then

$$\delta^*(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 2^{m-2}.$$

Proof. We know that $d(\mathbf{u},\mathbf{0}) = wt(\mathbf{u})$, so we find

$$d(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = wt(\mathbf{u}),$$

which is equal to 2^{m-1}. Then if we can show that $(\alpha_i = 1 \Rightarrow u_i = 0) \Rightarrow m(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 0$ and $(\alpha_i = 0 \Rightarrow u_i = 0) \Rightarrow m(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 0$, we find $\delta^*(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 2^{m-2}$.

The first follows from the equivalence $N(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 0 \Leftrightarrow (\alpha_i = 1 \Rightarrow u_i = 0)$ shown above in Example 9, since

$$N(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 0 \Rightarrow m(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 0.$$

For the second we have

$$\begin{aligned} N(\boldsymbol{\alpha},\mathbf{u}+\boldsymbol{\alpha}) = 0 &\Leftrightarrow (\alpha_i = 0 \Rightarrow (\mathbf{u}+\boldsymbol{\alpha})_i = 0) \\ &\Leftrightarrow (\alpha_i = 0 \Rightarrow u_i = 0), \end{aligned}$$

and again,

$$N(\boldsymbol{\alpha},\mathbf{u}+\boldsymbol{\alpha}) = 0 \Rightarrow m(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 0.$$

So since $d(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 2^{m-1}$ and $m(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 0$, the Pearson distance between $\mathbf{u}+\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}$ is

$$\delta^*(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 2^{m-2}.$$

□

And now this brings us to the following theorem:

Theorem 4.4. *Let* $C = \mathcal{R}(1,m)$ *with* $m \leq 4$*. Then for all* $\boldsymbol{\alpha}$ *of weight* 2^{m-2} *or* $3 \cdot 2^{m-2}$*, there exists a* $\mathbf{u} \in C$ *such that*

$$\delta^*(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 2^{m-2}$$

Proof. By Lemma 4.3 we only need to show that for all $\boldsymbol{\alpha}$, there exists a $\mathbf{u} \in C$ such that $wt(\mathbf{u}) = 2^{m-1}$ and such that for all $i = 1, \dots, 2^m (\alpha_i = 1 \Rightarrow u_i = 0)$ or that for all $i = 1, \dots, 2^m (\alpha_i = 0 \Rightarrow u_i = 0)$.

We have three cases: $m = 2, m = 3$ and $m = 4$. For each case we give one or more examples where we see that $\delta^*(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) = 2^{m-2}$, which are meant to illustrate how finding a suitable $\mathbf{u}$ can be done for *any* $\boldsymbol{\alpha}$ of weight 2^{m-2} or $3 \cdot 2^{m-2}$, since the number of possible $\boldsymbol{\alpha}$'s is too high to show it for every single one.

- $m = 2$: There are two possible weights for $\boldsymbol{\alpha}$: $wt(\boldsymbol{\alpha}) = 1$ or $wt(\boldsymbol{\alpha}) = 3$, and eight possible $\boldsymbol{\alpha}$'s:

$$\begin{array}{cc} 0001 & 1110 \\ 0010 & 1101 \\ 0100 & 1011 \\ 1000 & 0111 \end{array}$$

Recall that

$$\mathcal{R}(1,2) = \{0000, 0011, 0101, 0110, 1111, 1100, 1010, 1001\},$$

so for $\boldsymbol{\alpha} = 0001$ or $\boldsymbol{\alpha} = 1110$, we may take $\mathbf{u} = 0110, 1100$ or 1010. Indeed, we see that for $\boldsymbol{\alpha} = 0001$, $\boldsymbol{\alpha}_i = 1 \Rightarrow \mathbf{u}_i = 0$ and for $\boldsymbol{\alpha} = 1110$ we see that $\boldsymbol{\alpha}_i = 0 \Rightarrow \mathbf{u}_i = 0$. So for example, $\delta^*(0110 + 0001, 0001) = 2^{m-r}(1 - 2^{-r}) = 1$, making 1 the minimum Pearson distance of the coset C_{0001}.

Similarly, for the second row of possible $\boldsymbol{\alpha}$'s we could pick $\mathbf{u} = 1001, 1100$ or 0101, for the third row $\mathbf{u} = 1001, 0011,$ or 1010 and the fourth $\mathbf{u} = 0110, 0011$ or 0101. These results can also be seen in Table 4.1.

Table 4.1: Words $\mathbf{u}$ of length 4 such that $\alpha_i = 1 \Rightarrow u_i = 0$ ($\boldsymbol{\alpha}$ in first row) or $\alpha_i = 0 \Rightarrow u_i = 0$ ($\boldsymbol{\alpha}$ in second row).

$\boldsymbol{\alpha}$	0001 1110	0010 1101	0100 1011	1000 0111
$\mathbf{u}$	0110 1100 1010	1001 1100 0101	1001 0011 1010	0110 0011 0101

- $m = 3$: The possible weights for $\boldsymbol{\alpha}$ are 2 and 6 for a total word length of 8. We know from Lemma 4.2 that words $\mathbf{u} \in \mathcal{R}(1,3)$ are built up of two words from $\mathcal{R}(1,2)$. That means that if for example $\boldsymbol{\alpha} = 0001.0100$, we can use Table 4.1 to pick a $\mathbf{u}$; for example, $\mathbf{u} = 1010.1010$. In fact, for any $\boldsymbol{\alpha}$ that is a combination of two words of length 4 and weight 1, we can find a $\mathbf{u} = |\mathbf{u}'|\mathbf{u}'|$ such that $\alpha_i = 1 \Rightarrow u_i = 0$, where we pick $\mathbf{u}'$ from Table 4.1. Similarly, we can use the table to pick a $\mathbf{u}$ when $\boldsymbol{\alpha}$ is a combination of two length 4 words of weight 3.

In the case that $\boldsymbol{\alpha}$ is built up of a length 4 block of weight 2 and one of weight 0 or 4, e.g. 0011.0000 or 1100.1111, we have two options for $\mathbf{u}$: letting $\mathbf{u}$ be built up of 0000 and 1111 blocks, e.g. 0000.1111 for our example $\boldsymbol{\alpha}$'s, or letting $\mathbf{u}$ be built up of two blocks $\mathbf{b}$ with weight two. For the latter we can use Table 4.2 to find $\mathbf{u} = 1100.1100$ for both $\boldsymbol{\alpha}$'s.

Table 4.2: blocks $\mathbf{b}$ such that $\boldsymbol{\alpha}_i = 1 \Rightarrow \mathbf{b}_i = 0$ ($\boldsymbol{\alpha}$ in first row) or $\boldsymbol{\alpha}_i = 0 \Rightarrow \mathbf{b}_i = 0$ ($\boldsymbol{\alpha}$ in second row).

$\boldsymbol{\alpha}$	0011 1100	0101 1010	0110 1001	1001 0110	1010 0101	1100 0011
$\mathbf{b}$	1100	1010	1001	0110	0101	0011

- $m = 4$: The possible weights for $\boldsymbol{\alpha}$ are 4 and 12 for a total word length of 16. Finding suitable $\mathbf{u}$'s is similar to the previous two cases, so we will give some examples to illustrate the possible solutions, and then summarize them afterwards:

  1. If $\boldsymbol{\alpha} = 1111.0000.0000.0000$, or $0000.1111.1111.1111$, take $\mathbf{u} = 0000.0000.1111.1111$.

  2. Similarly, if $\boldsymbol{\alpha} = 1110.1000.0000.0000$, or $0001.0111.1111.1111$, take $\mathbf{u} = 0000.0000.1111.1111$.

3. If $\boldsymbol{\alpha} = 1110.0000.1000.0000$, or $1110.1111.0111.1111$, take $\mathbf{u} = 0000.1111.0000.1111$,

4. and if $\boldsymbol{\alpha} = 0011.0101.0000.0000$, take $\mathbf{u} = 0000.0000.1111.1111$, and so on for other $\boldsymbol{\alpha}$'s of weight 4 with blocks of weight 2, 3 or 4, and for $\boldsymbol{\alpha}$'s of weight 12 with blocks with 2, 3 or 4 zeros.

5. If $\boldsymbol{\alpha} = 0011.1000.0100.0000$, take $\mathbf{u} = 1100.0011.0011.1100$, where we pick a suitable block to build $\mathbf{u}$ with from Table 4.2 based on the block of weight 2 in $\boldsymbol{\alpha}$.

6. If $\boldsymbol{\alpha} = 0001.0010.0100.1000$, take for example $\mathbf{u} = 0110.1001.1001.0110$, where we pick a suitable block to build $\mathbf{u}$ with from Table 4.1.

The first four points show that if $\boldsymbol{\alpha}$ of weight 4 contains a block of weight 3 or 4, or at least two blocks of weight 2, $\mathbf{u}$ will be some combination of 0000 and 1111 (though note that $0000.1111.1111.1111$ is not a possible codeword, since the second half of the word should equal or "mirror" the first half of the word). This is always possible, since $\mathbf{u}$ may contain two blocks 0000 and two blocks 1111, while an $\boldsymbol{\alpha}$ with blocks of these weights will only contain either two blocks that are non-zero or two that are not of weight 4.

Similarly, for $\boldsymbol{\alpha}$ of weight 12 containing a block with 2, 3 or 4 zeros, the same $\mathbf{u}$ may be picked. In this case, an $\boldsymbol{\alpha}$ of this weight will only contain two blocks that are not all ones.

Note that for point 4 $\mathbf{u}$ is in fact the only possible codeword for which $(\boldsymbol{\alpha}_i = 1 \Rightarrow \mathbf{u}_i = 0)$, because $\boldsymbol{\alpha}$ contains two blocks of weight 2: $\boldsymbol{\alpha}_1 = 0011$ and $\boldsymbol{\alpha}_2 = 0101$ where $\boldsymbol{\alpha}_1 \neq \boldsymbol{\alpha}_2 + a \cdot 1111$ $(a = 0, 1)$. If we'd looked at $\boldsymbol{\alpha} = 0101.0000.1010.0000$ for example, we could also have picked $\mathbf{u} = 1010.0101.0101.1010$.

Point five and six show that if $\boldsymbol{\alpha}$ of weight 4 contains a block of weight 1, a suitable $\mathbf{u}$ may be found using Tables 4.1 and 4.2. When all the blocks $\boldsymbol{\alpha}$ contains are of weight 1, there are many options for $\mathbf{u}$, but when $\boldsymbol{\alpha}$ contains one block of weight 2, there is again only one possible $\mathbf{u}$ that can be picked, determined by that block of weight 2 and the position of the two blocks of weight 1.
Similarly, for $\boldsymbol{\alpha}$ of weight 12 containing a block with 1 zero, a suitable $\mathbf{u}$ may again be found using Tables 4.1 and 4.2. When all the blocks in $\boldsymbol{\alpha}$ contain 1 zero, there are again many options for $\mathbf{u}$, but again, if $\boldsymbol{\alpha}$ contains a block of weight 2, there are fewer options for $\mathbf{u}$, determined by that block.

So for $m = 2$, $m = 3$ and $m = 4$ we can always find a suitable $\mathbf{u} \in \mathcal{R}(1, m)$ for every possible $\boldsymbol{\alpha}$, such that $(\alpha_i = 1 \Rightarrow u_i = 0)$ or $(\alpha_i = 0 \Rightarrow u_i = 0)$.

So we find that for $C = \mathcal{R}(1, m)$ with $m \leq 4$, for all $\boldsymbol{\alpha}$ of weight 2^{m-2} or $3 \cdot 2^{m-2}$, there exists a $\mathbf{u} \in C$ such that

$$\delta^*(\mathbf{u} + \boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-2}$$

□

So for $r = 1$ and $m \leq 4$, the minimum Pearson distance of any coset $C_{\boldsymbol{\alpha}}$ is equal to

$$\delta^*_{\min} = 2^{m-2}.$$

An example of three different cosets of $\mathcal{R}(1, 3)$ can be seen in Figure 4.1, and there we see that the minimum Pearson distance is indeed equal to 2.

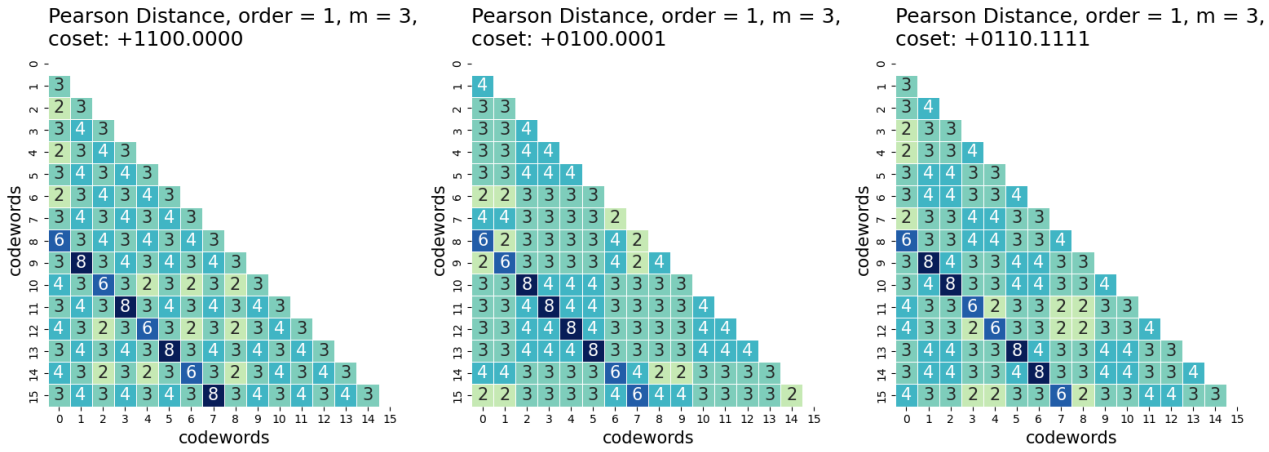

Figure 4.1: A heatmap showing the Pearson distance between all codewords in three cosets of the first order Reed-Muller code with $m = 3$.

4.3. A New Lower bound for $m > 4$

In the previous section we proved that the minimum Pearson distance of any coset $C_{\boldsymbol{\alpha}}$ of a $\mathcal{R}(1, m)$, where $m \leq 4$, is equal to the lower bound found in Theorem 2.9.

Now the question is what happens for higher values of m.

Example 10. Let $C = \mathcal{R}(1, 5)$. We know that the words in C have length 32 and the distance of the code is $d_{\min} = 16$. So, for example, (with periods and a slash added for readability) one of the codewords is

$$0110.1001.1001.0110/1001.0110.0110.1001 \in C.$$

We saw in the proof of Theorem 4.4 that there were fewer options to pick for a suitable $\mathbf{u} \in \mathcal{R}(1, 4)$ when $\boldsymbol{\alpha}$ contained blocks of weight 2 ("suitable" meaning a $\mathbf{u}$ such that $(\alpha_i = 1 \Rightarrow u_i = 0)$ or $(\alpha_i = 0 \Rightarrow u_i = 0)$).

In fact, using that there were fewer suitable $\mathbf{u} \in C$ when $\boldsymbol{\alpha}$ had multiple blocks of weight two, is precisely how an example could be found of an $\boldsymbol{\alpha}$ such that the minimum Pearson distance of the coset $C_{\boldsymbol{\alpha}}$ for $C = \mathcal{R}(1, 5)$ is strictly greater than our lower bound 2^{m-2}.

Example (10 cont.). Let

$$\boldsymbol{\alpha} = 1100.0100.0010.0000/0110.0000.0001.1000,$$

then by calculating all the Pearson distances between codewords we find that $\delta^*_{\min} = 11.5$ instead of $2^{m-2} = 8$! (See Figures 4.2 and 4.3).

```
RM(1,5) + 1100.0100.0010.0000.0110.0000.0001.1000

d_min =  16
Hamming:  Counter({16.0: 1984, 32.0: 32})

min Pearson:  11.5
Pearson:  Counter({15.875: 704, 15.5: 480, 16.0: 380, 14.875: 288, 14.0: 84, 12.875: 32,
 11.5: 16, 31.5: 16, 30.0: 8, 32.0: 7, 24.0: 1})
```

Figure 4.2: A screenshot of a Python output showing that the minimum Pearson distance for $\mathcal{R}(1, 5)_{\boldsymbol{\alpha}}$ where $\boldsymbol{\alpha} = 1100.0100.0010.0000/0110.0000.0001.1000$ is equal to 11.5.

Another example of an $\boldsymbol{\alpha}$ that leads to a higher minimum Pearson distance than the previous lower bound is

$$\boldsymbol{\alpha} = 1100.0000.0110.0000/0110.0000.0001.1000,$$

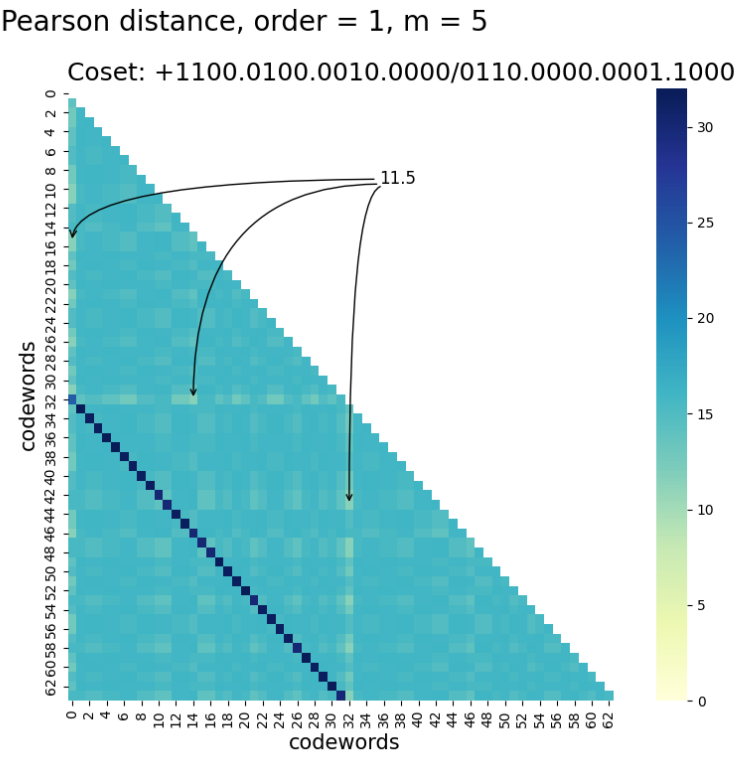

Figure 4.3: A heatmap showing the Pearson distance between all codewords for $\mathcal{R}(1,5)_{\boldsymbol{\alpha}}$ where $\boldsymbol{\alpha}$ = 1100.0100.0010.0000/0110.0000.0001.1000. A number of cells with value 11.5 are labelled with arrows, though not all 16 of them.

again for order 1 and $m = 5$, though this coset has minimum Pearson distance $9.875 > 8$ instead (see Figure 4.4 with the Python output below).

```
RM(1,5) + 1100.0000.0110.0000.0110.0000.0001.1000

d_min =  16
Hamming:  Counter({16.0: 1984, 32.0: 32})

min Pearson:  9.875
Pearson:  Counter({15.875: 748, 15.5: 422, 16.0: 416, 14.875: 228, 14.0: 112, 12.875: 44,
31.5: 14, 32.0: 11, 11.5: 10, 9.875: 4, 30.0: 4, 27.5: 2, 24.0: 1})
```

Figure 4.4: A screenshot of a Python output showing that the minimum Pearson distance for $\mathcal{R}(1,5)_{\boldsymbol{\alpha}}$ where $\boldsymbol{\alpha}$ = 1100.0000.0110.0000/0110.0000.0001.1000 is equal to 9.875.

At first glance these two values might appear somewhat random. However, we know that $d(\mathbf{u},\mathbf{v}) = 2^{m-1} = 16$ for whichever words $\mathbf{u},\mathbf{v} \in C_{\boldsymbol{\alpha}}$ such that they have Pearson distance $\delta^*(\mathbf{u},\mathbf{v}) = \delta^*_{\min}$, and by Corollary 2.8

$$\delta^*(\mathbf{u},\mathbf{v}) = d(\mathbf{u},\mathbf{v}) - (d(\mathbf{u},\mathbf{v}) - 2m(\mathbf{u},\mathbf{v}))^2/2^m.$$

So we find that for $\boldsymbol{\alpha} = 1100.0100.0010.0000/0110.0000.0001.1000$ (where $\delta^*_{\min} = 11.5$) these $\mathbf{u},\mathbf{v} \in C_{\boldsymbol{\alpha}}$ must have $m(\mathbf{u},\mathbf{v}) = 2$. For $\boldsymbol{\alpha} = 1100.0000.0110.0000/0110.0000.0001.1000$ where $\delta^*_{\min} = 9.875$, we find $m(\mathbf{u},\mathbf{v}) = 1$.

For every $\mathbf{u},\mathbf{v} \in C_{\boldsymbol{\alpha}}$ such that they have Pearson distance $\delta^*(\mathbf{u},\mathbf{v}) = \delta^*_{\min}$, we know that $d(\mathbf{u},\mathbf{v}) = 2^{m-1}$, and that $m(\mathbf{u},\mathbf{v})$ must be an integer, so we can conclude that there are no cosets of $\mathcal{R}(1,5)$ with a minimum Pearson distance between 8 and 9.875 or between 9.875 and 11.5. Then we can also say that *if* a coset exists with an even higher minimum Pearson distance, we know that it must correspond to some $\delta^*(\mathbf{u},\mathbf{v})$ where $m(\mathbf{u},\mathbf{v}) = 3, 4, \ldots$.

We will later show that this $m(\mathbf{u}, \mathbf{v})$ and thus $\delta^*_{\min}$ has an upper bound, but first we prove that the result from Example 10 can be extended to cosets of $\mathcal{R}(1, m)$ where $m > 5$.

4.3.1. Proof for all $m > 4$

The two $\boldsymbol{\alpha}$'s in Example 10 showed us that we can find cosets of RM-codes with $r = 1$ and $m = 5$ still determined by $\boldsymbol{\alpha}$'s of the weights 2^{m-2} and $3 \cdot 2^{m-2}$, such that the minimum Pearson distance of a coset determined by that $\boldsymbol{\alpha}$ is higher than the lower bound found by Weber et al. in Theorem 2.9. Of the two, the first resulted in a higher minimum Pearson distance than the other: 11.5 as opposed to 9.875. This leads us to the following theorem:

Theorem 4.5. *Let $C = \mathcal{R}(1, m)$, where $m > 4$. Then there exists an $\boldsymbol{\alpha}$ of weight 2^{m-2} or $3 \cdot 2^{m-2}$ such that for all $\mathbf{u}, \mathbf{v} \in C$*

$$\delta^*(\mathbf{u} + \boldsymbol{\alpha}, \mathbf{v} + \boldsymbol{\alpha}) \geq \frac{23}{64} \cdot 2^m.$$

In particular, for $m = 5$ and $\boldsymbol{\alpha} = 1100.0100.0010.0000/0110.0000.0001.1000$, we find $\delta^*_{\min} = 23/64 \cdot 2^5 = 23/2$ for $C_{\boldsymbol{\alpha}}$, where $C = \mathcal{R}(1, 5)$, and this result can be extended for all $m \geq 5$.

Proof. First we will prove that 1. for a certain $\boldsymbol{\alpha}$, namely $\boldsymbol{\alpha} = 1100.0100.0010.0000/0110.0000.0001.1000$, there is always a $\mathbf{u} \in C$ such that $\delta^*(\mathbf{u} + \boldsymbol{\alpha}, \boldsymbol{\alpha}) = \frac{23}{64} \cdot 2^m$, and then later that 2. there are no $\mathbf{u}, \mathbf{v} \in C$ such that $\mathbf{u} \neq \mathbf{v}$ and $\delta^*(\mathbf{u} + \boldsymbol{\alpha}, \mathbf{v} + \boldsymbol{\alpha}) < \frac{23}{64} \cdot 2^m$.

1. Let $C_5 = \mathcal{R}(1, 5)$ and $\boldsymbol{\alpha}_5 = 1100.0100.0010.0000/0110.0000.0001.1000$. From Example 10 we know that there exists a $\mathbf{u}_5 \in C_5$ such that

   $$\delta^*(\mathbf{u}_5 + \boldsymbol{\alpha}_5, \boldsymbol{\alpha}_5) = 11.5 = \frac{23}{64} \cdot 2^5,$$

   for example $\mathbf{u}_5 = 0110.0110.1001.1001/1001.1001.0110.0110$. Indeed, for that $\mathbf{u}_5$ we have $m(\mathbf{u}_5 + \boldsymbol{\alpha}_5, \boldsymbol{\alpha}_5) = 2$ and $wt(\mathbf{u}_5) = 16$, so

   $$\delta^*(\mathbf{u}_5 + \boldsymbol{\alpha}_5, \boldsymbol{\alpha}_5) = d(\mathbf{u}_5, \mathbf{0}) - \frac{(d(\mathbf{u}_5, \mathbf{0}) - 2m(\mathbf{u}_5 + \boldsymbol{\alpha}_5, \boldsymbol{\alpha}_5))^2}{32} = 11.5.$$

   We also know from Example 10 that there is no pair of words $\mathbf{v}_5, \mathbf{w}_5 \in C_5$ such that $\delta^*(\mathbf{v}_5 + \boldsymbol{\alpha}_5, \mathbf{w}_5 + \boldsymbol{\alpha}_5) < \frac{23}{64} \cdot 2^m$.

   Now let $\mathbf{u}_m = |\mathbf{u}_{m-1}|\mathbf{u}_{m-1}|$ and $\boldsymbol{\alpha}_m = |\boldsymbol{\alpha}_{m-1}|\boldsymbol{\alpha}_{m-1}|$ for $m > 5$, where $\boldsymbol{\alpha}_5$ and $\mathbf{u}_5$ are the same as the example defined above. Claim: then $m(\mathbf{u}_m + \boldsymbol{\alpha}_m, \boldsymbol{\alpha}_m) = 2^{m-4}$ and $wt(\mathbf{u}_m) = 2^{m-1}$.

   We've already seen for $m = 5$ that $m(\mathbf{u}_5 + \boldsymbol{\alpha}_5, \boldsymbol{\alpha}_5) = 2$ and $wt(\mathbf{u}_5) = 16$. Then the induction hypothesis is $m(\mathbf{u}_{m-1} + \boldsymbol{\alpha}_{m-1}, \boldsymbol{\alpha}_{m-1}) = 2^{m-5}$, and we find

   $$m(|\mathbf{u}_{m-1} + \boldsymbol{\alpha}_{m-1}|\mathbf{u}_{m-1} + \boldsymbol{\alpha}_{m-1}|, |\boldsymbol{\alpha}_{m-1}|\boldsymbol{\alpha}_{m-1}|) = 2 \cdot 2^{m-5} = 2^{m-4},$$

   and if $wt(\mathbf{u}_{m-1}) = 2^{m-2}$, then $wt(\mathbf{u}_m) = 2 \cdot 2^{m-2} = 2^{m-1}$.

   So for each $m \geq 5$ we can find

   $$\delta^*(\mathbf{u}_m + \boldsymbol{\alpha}_m, \boldsymbol{\alpha}_m) = 2^{m-1} - \frac{(2^{m-1} - 2 \cdot 2^{m-4})^2}{2^m} = \frac{23}{64} \cdot 2^m.$$

2. Now it remains to show that there are no $\mathbf{u}_m, \mathbf{v}_m \in C_m$ such that $\delta^*(\mathbf{u}_m + \boldsymbol{\alpha}_m, \mathbf{v}_m + \boldsymbol{\alpha}_m) < \frac{23}{64} \cdot 2^m$.

   Let $C_{m-1} = \mathcal{R}(1, m-1)$ and let $\boldsymbol{\alpha}_i = |\boldsymbol{\alpha}_{i-1}|\boldsymbol{\alpha}_{i-1}|$ for all $i > 5$, where $\boldsymbol{\alpha}_5 = 1100.0100.0010.0000/0110.0000.0001.1000$. We assume that $C_{\boldsymbol{\alpha}_{m-1}}$ has minimum Pearson distance $\delta^*_{\min,m-1} = \frac{23}{64} \cdot 2^{m-1}$.

Now let $C_m = \mathcal{R}(1, m)$ and let $C_{\boldsymbol{\alpha}_m}$ be determined by $\boldsymbol{\alpha}_m = |\boldsymbol{\alpha}_{m-1}|\boldsymbol{\alpha}_{m-1}|$.

Suppose there exist $\mathbf{u}_m, \mathbf{v}_m \in C_m$ such that $m(\mathbf{u}_m + \boldsymbol{\alpha}_m, \mathbf{v}_m + \boldsymbol{\alpha}_m) < 2^{m-4}$. Then $\delta^*_{\min,m} < \frac{23}{64} \cdot 2^m$ for $C_{\boldsymbol{\alpha}_m}$.

If $\mathbf{u}_m = |\mathbf{u}_{m-1}|\mathbf{u}_{m-1}|$ and $\mathbf{v}_m = |\mathbf{v}_{m-1}|\mathbf{v}_{m-1}|$, that means that

$$\begin{aligned} m(\mathbf{u}_m + \boldsymbol{\alpha}_m, \mathbf{v}_m + \boldsymbol{\alpha}_m) < 2^{m-4} \\ \Leftrightarrow\ & m(|\mathbf{u}_{m-1}|\mathbf{u}_{m-1}| + |\boldsymbol{\alpha}_{m-1}|\boldsymbol{\alpha}_{m-1}|, |\mathbf{v}_{m-1}|\mathbf{v}_{m-1}| + |\boldsymbol{\alpha}_{m-1}|\boldsymbol{\alpha}_{m-1}|) < 2^{m-4} \\ \Leftrightarrow\ & m(|\mathbf{u}_{m-1} + \boldsymbol{\alpha}_{m-1}|\mathbf{u}_{m-1} + \boldsymbol{\alpha}_{m-1}|, |\mathbf{v}_{m-1} + \boldsymbol{\alpha}_{m-1}|\mathbf{v}_{m-1} + \boldsymbol{\alpha}_{m-1}|) < 2^{m-4} \\ \Leftrightarrow\ & 2 \cdot m(\mathbf{u}_{m-1} + \boldsymbol{\alpha}_{m-1}, \mathbf{v}_{m-1} + \boldsymbol{\alpha}_{m-1}) < 2^{m-4}, \end{aligned}$$

which is a contradiction, since $m(\mathbf{u}_{m-1} + \boldsymbol{\alpha}_{m-1}, \mathbf{v}_{m-1} + \boldsymbol{\alpha}_{m-1}) \geq 2^{m-5}$. Similarly, if $\mathbf{u}_m = |\mathbf{u}_{m-1}|\mathbf{u}_{m-1}|$ and $\mathbf{v}_m = |\mathbf{v}_{m-1}|\mathbf{v}_{m-1} + \mathbf{1}|$, we find

$$\begin{aligned} m(\mathbf{u}_m + \boldsymbol{\alpha}_m, \mathbf{v}_m + \boldsymbol{\alpha}_m) < 2^{m-4} \Leftrightarrow\ & m(\mathbf{u}_{m-1} + \boldsymbol{\alpha}_{m-1}, \mathbf{v}_{m-1} + \boldsymbol{\alpha}_{m-1}) + \\ & m(\mathbf{u}_{m-1} + \boldsymbol{\alpha}_{m-1}, \mathbf{v}_{m-1} + \mathbf{1} + \boldsymbol{\alpha}_{m-1}) < 2^{m-4}, \end{aligned}$$

which is again a contradiction.

So we find that for $\boldsymbol{\alpha}_m = |\boldsymbol{\alpha}_{m-1}|\boldsymbol{\alpha}_{m-1}|$, $m > 5$, where $\boldsymbol{\alpha}_5 = 1100.0100.0010.0000/0110.0000.0001.1000$, the coset of $\mathcal{R}(1, m)$ determined by $\boldsymbol{\alpha}_m$ has minimum Pearson distance $\frac{23}{64} \cdot 2^m$. □

The result found in Theorem 4.5 is considerably higher than the previous lower bound of 2^{m-2}. Indeed,

$$2^{m-2} = \frac{16}{64} \cdot 2^m < \frac{23}{64} \cdot 2^m.$$

It is unclear whether this is the highest minimum Pearson distance we can possibly find, though there are a number of examples of $\boldsymbol{\alpha}$'s for which the Pearson distance is also higher than the lower bound like in Figure 4.4, but no higher value than the one presented in Theorem 4.5 has been found yet.

However, we can prove which value the minimum Pearson distance will never exceed.

4.4. An Upper bound for $m > 4$

In the previous section a new, higher, lower bound for the minimum Pearson value of cosets $C_{\boldsymbol{\alpha}}$ of first order Reed-Muller codes with $m > 4$ was found. However, it is unclear whether this minimum is the highest that could possibly be found. For that reason we will narrow down the interval of possible values for this minimum Pearson distance in this section, by finding an upper bound for the minimum Pearson distance of the cosets. This will be done by first showing some values the minimum Pearson distance *cannot* possibly be in Lemma 4.6. This will finally lead us to Theorem 4.7, in which we will prove that $\frac{7}{16} \cdot 2^m$ is an upperbound for the minimum Pearson distance of our cosets.

Lemma 4.6. *Let $C = \mathcal{R}(1, m)$, where $m > 4$ and let $\boldsymbol{\alpha}$ have weight 2^{m-2} or $3 \cdot 2^{m-2}$. Let $\mathbf{u} \in C$ with $wt(\mathbf{u}) = 2^{m-1}$. If for some $x \in \{0, 1, \dots, 2^{m-3}\}$*

$$m(\boldsymbol{\alpha}, \mathbf{u} + \boldsymbol{\alpha}) = 2^{m-2} - x,$$

then

$$m(\boldsymbol{\alpha}, (\mathbf{u} + \mathbf{1}) + \boldsymbol{\alpha}) = x.$$

Proof. Let $C = \mathcal{R}(1, m)$, where $m > 4$, and let $\boldsymbol{\alpha}$ have weight 2^{m-2} or $3 \cdot 2^{m-2}$. Now let $\mathbf{u} \in C$ such that $m(\boldsymbol{\alpha}, \mathbf{u} + \boldsymbol{\alpha}) = 2^{m-2} - x$ for some $x \in \{0, 1, \dots, 2^{m-3}\}$ and $d(\mathbf{u}, \mathbf{0}) = 2^{m-1}$. We know that $N(\boldsymbol{\alpha}, \mathbf{u} + \boldsymbol{\alpha}) + N(\mathbf{u} + \boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-1}$ so supposing that $N(\mathbf{u} + \boldsymbol{\alpha}, \boldsymbol{\alpha}) \leq N(\boldsymbol{\alpha}, \mathbf{u} + \boldsymbol{\alpha})$ we find $N(\boldsymbol{\alpha}, \mathbf{u} + \boldsymbol{\alpha}) =$

Table 4.3: A situation where $N(\boldsymbol{\alpha}, \mathbf{u}+\boldsymbol{\alpha}) = 2^{m-2} - x$ and $N(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-2} + x$ (digits reordered for ease of reading).

$\boldsymbol{\alpha}$	$00\ldots0$	$11\ldots1$	$\cdots$
$\mathbf{u}+\boldsymbol{\alpha}$	$\underbrace{11\ldots1}_{2^{m-2}+x}$	$\underbrace{00\ldots0}_{2^{m-2}-x}$	$\underbrace{\cdots}_{2^{m-1}}$

$2^{m-2} + x$ and $N(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-2} - x$ (see also Table 4.3).

Again, the 2^{m-1} remaining digits should be the same in $\boldsymbol{\alpha}$ and $\mathbf{u}+\boldsymbol{\alpha}$. If $\boldsymbol{\alpha}$ has weight 2^{m-2} that means that there are another x ones among those 2^{m-1} digits, and $2^{m-1} - x$ zeros. Then if we once again consider $(\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}$ as well, we see that $N((\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = x$ and $N(\boldsymbol{\alpha}, (\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}) = 2^{m-1} - x$ (see also Table 4.4). So $m(\boldsymbol{\alpha}, (\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}) = x$.

Table 4.4: A situation where $N(\boldsymbol{\alpha}, \mathbf{u}+\boldsymbol{\alpha}) = 2^{m-2} - x$, and $wt(\boldsymbol{\alpha}) = 2^{m-2}$ (digits reordered for ease of reading).

$\boldsymbol{\alpha}$	$00\ldots0$	$11\ldots1$	$11\ldots1$	$00\ldots0$
$\mathbf{u}+\boldsymbol{\alpha}$	$11\ldots1$	$00\ldots0$	$11\ldots1$	$00\ldots0$
$(\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}$	$\underbrace{00\ldots0}_{2^{m-2}+x}$	$\underbrace{11\ldots1}_{2^{m-2}-x}$	$\underbrace{00\ldots0}_{x}$	$\underbrace{11\ldots1}_{2^{m-1}-x}$

Note that if $N(\boldsymbol{\alpha}, \mathbf{u}+\boldsymbol{\alpha}) = 2^{m-2} - x$ and $N(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-2} + x$, it must be the case that $wt(\boldsymbol{\alpha}) = 2^{m-2}$; otherwise more than 2^{m-1} ones are needed where digits in $\boldsymbol{\alpha}$ and $\mathbf{u}+\boldsymbol{\alpha}$ are equal. We move on to the case $N(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) \leq N(\boldsymbol{\alpha}, \mathbf{u}+\boldsymbol{\alpha})$ where we must have $wt(\boldsymbol{\alpha}) = 3 \cdot 2^{m-2}$ instead. Looking at Table 4.5, we see that now $N((\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-1} - x$ and $N(\boldsymbol{\alpha}, (\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}) = x$. So also for $wt(\boldsymbol{\alpha}) = 3 \cdot 2^{m-2}$ we find that $m(\boldsymbol{\alpha}, (\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}) = x$.

Table 4.5: A situation where $N(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-2} - x$, and $wt(\boldsymbol{\alpha}) = 3 \cdot 2^{m-2}$ (digits reordered for ease of reading).

$\boldsymbol{\alpha}$	$00\ldots0$	$11\ldots1$	$11\ldots1$	$00\ldots0$
$\mathbf{u}+\boldsymbol{\alpha}$	$11\ldots1$	$00\ldots0$	$11\ldots1$	$00\ldots0$
$(\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}$	$\underbrace{00\ldots0}_{2^{m-2}-x}$	$\underbrace{11\ldots1}_{2^{m-2}+x}$	$\underbrace{00\ldots0}_{2^{m-1}-x}$	$\underbrace{11\ldots1}_{x}$

□

Now we can finally prove the new upper bound:

Theorem 4.7. *Let $C = \mathcal{R}(1, m)$, where $m > 4$. Then for all $\boldsymbol{\alpha}$ of weight 2^{m-2} or $3 \cdot 2^{m-2}$, the minimum Pearson distance $\delta^*_{\min}$ for the coset $C_{\boldsymbol{\alpha}}$ has the upper bound*

$$\delta^*_{\min} \leq \frac{7}{16} \cdot 2^m.$$

Proof. We use Lemma 4.6 to show that the minimum Pearson distance of a coset $C_{\boldsymbol{\alpha}}$ of $C = \mathcal{R}(1, m)$, where $m > 4$ and $wt(\boldsymbol{\alpha}) = 2^{m-2}$ or $wt(\boldsymbol{\alpha}) = 3 \cdot 2^{m-2}$, cannot exceed $7/16 \cdot 2^m$.

Let $\mathbf{u} \in C$ such that $d(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-1}$. Recall that by Corollary 2.8

$$\delta^*(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = d(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) - \frac{(d(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) - 2m(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}))^2}{2^m},$$

which increases as $m(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha})$ increases.

From Lemma 4.6 we know that if $m(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = 2^{m-2} - x$ for some $x \in \{0, 1, \ldots, 2^{m-3}\}$, we find $m((\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}, \boldsymbol{\alpha}) = x$. If x is low, we find that $\delta^*(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha})$ is high, but then $\delta^*((\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}, \boldsymbol{\alpha})$ is low again. The upper bound of the minimum Pearson distance must correspond to the value x where both $m(\mathbf{u}+\boldsymbol{\alpha}, \boldsymbol{\alpha})$ and $m((\mathbf{u}+\mathbf{1})+\boldsymbol{\alpha}, \boldsymbol{\alpha})$ are as high as possible, that is, where $2^{m-2} - x = x$, which holds for $x = 2^{m-3}$.

Then

$$\begin{aligned}\delta^*(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) &= d(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) - \frac{\left(d(\mathbf{u}+\boldsymbol{\alpha},\boldsymbol{\alpha}) - 2x\right)^2}{2^m} \\ &= 2^{m-1} - \frac{\left(2^{m-1} - 2\cdot 2^{m-3}\right)^2}{2^m} \\ &= \frac{7}{16}\cdot 2^m\end{aligned}$$

So $\frac{7}{16}\cdot 2^m$ is an upper bound for the minimum Pearson distance of cosets of first order Reed-Muller codes. □

Example (**10 cont.**). In Example 10 on page 23 we found the coset $C_{\boldsymbol{\alpha}}$ of $C = \mathcal{R}(1,5)$ for which $\delta^*_{\min} = 23/64\cdot 2^m = 23/2$, namely for $\boldsymbol{\alpha} = 1100.0100.0010.0000/0110.0000.0001.1000$. If we have $\delta^*(\mathbf{u},\mathbf{v}) = \delta^*_{\min}$ for some $\mathbf{u},\mathbf{v} \in C_{\boldsymbol{\alpha}}$, this corresponds to $m(\mathbf{u},\mathbf{v}) = 2$. This lead us to theorise that it might be possible to find cosets where the minimum Pearson distance value corresponds to a higher $m(\mathbf{u},\mathbf{v})$. We can now say that the upper bound for this is equal to

$$m(\mathbf{u},\mathbf{v}) = 2^{m-3} = 4,$$

for which we would find

$$\delta^*_{\min} = \frac{7}{16}\cdot 2^m = 14.$$

To conclude, in the previous sections we found that for cosets of first order Reed-Muller codes determined by $\boldsymbol{\alpha}$'s of weight 2^{m-2} or $3\cdot 2^{m-2}$, the minimum Pearson distance can't be higher than 2^{m-2} (the lower bound found in [9]) when $m \leq 4$, while for $m > 4$ it is possible to find cosets for which the minimum Pearson distance is equal to $23/64\cdot 2^m$, specifically when $\boldsymbol{\alpha}_5 = 1100.0100.0010.0000/0110.0000.0001.1000$ and $\boldsymbol{\alpha}_{m+1} = |\boldsymbol{\alpha}_m|\boldsymbol{\alpha}_m|$. It may be possible to find cosets for which the minimum Pearson distance is even higher, but it can never exceed $7/16\cdot 2^m$.

5

Conclusion and Recommendations

5.1. Conclusion

The goal of this project was to investigate the suitability of cosets of Reed-Muller codes for channels with unknown offset. For this purpose, two measures for a code's performance were introduced in the Sections 2.1.4 and 2.2: the distance of the code and the word error rate.

However, in Chapter 4 the focus only lay on finding a new lower bound for the minimum Pearson distance for cosets of first order Reed-Muller codes. The main results found in the chapter are:

§4.2 For $m \leq 4$, cosets of first order RM-codes determined by $\boldsymbol{\alpha}$'s of weights 2^{m-2} or $3 \cdot 2^{m-2}$, all have the minimum Pearson distance

$$\delta^*_{\min} = 2^{m-2} = \frac{16}{64} \cdot 2^m,$$

which is equal to the lower bound found by Weber et al. in [9].

§4.3 For $m > 4$, there exists an $\boldsymbol{\alpha}$ of weight 2^{m-2} or $3 \cdot 2^{m-2}$, such that the coset $C_{\boldsymbol{\alpha}}$ of the first order RM-code $C = \mathcal{R}(1, m)$ has minimum Pearson distance

$$\delta^*_{\min} \geq \frac{23}{64} \cdot 2^m.$$

In particular, one sequence of $\boldsymbol{\alpha}$'s for which the cosets $C_{\boldsymbol{\alpha}_m}$ have this minimum Pearson distance is $\boldsymbol{\alpha}_{m+1} = |\boldsymbol{\alpha}_m|\boldsymbol{\alpha}_m|$, where $\boldsymbol{\alpha}_5 = 1100.0100.0010.0000/0110.0000.0001.1000$.

§4.4 For $m > 4$, the minimum Pearson distance of all cosets of first order RM-codes determined by $\boldsymbol{\alpha}$'s of weights 2^{m-2} or $3 \cdot 2^{m-2}$, has upper bound

$$\delta^*_{\min} \leq \frac{7}{16} \cdot 2^m = \frac{28}{64} \cdot 2^m.$$

To conclude, if we only take cosets determined by $\boldsymbol{\alpha}$ of weight 2^{m-2} or $3 \cdot 2^{m-2}$, then for $m \leq 4$ we cannot find cosets of first order Reed-Muller codes that have a higher minimum Pearson distance than 2^{m-2}; for $m > 4$, we can definitely find a coset where the minimum distance is $23/16$ times greater than this lower bound 2^{m-2}: $23/64 \cdot 2^m$; however, the minimum Pearson distance cannot ever exceed $7/16 \cdot 2^m$.

5.2. Discussion and Recommendations

While the results put forward in this report are (presumably) all correct, their proofs are very much defined by the intuition used to reach them.

For example, in what would eventually become Example 10 in the report, we stumbled upon a particular coset of $\mathcal{R}(1, 5)$ determined by $\boldsymbol{\alpha} = 1100.0100.0010.0000/0110.0000.0001.1000$, for which the

minimum Pearson distance of the coset was higher than the earlier lower bound. The intuition used for this was the idea that first order Reed-Muller codes are built up of the eight blocks proposed in Lemma 4.2, and that finding a $\mathbf{u} \in \mathcal{R}(1, m)$ such that $m(\mathbf{u} + \boldsymbol{\alpha}, \boldsymbol{\alpha}) = 0$ might be more difficult if $\boldsymbol{\alpha}$ contained two blocks of weight two that don't occur together in codewords.

This later turned out to only be possible for $m \geq 5$ and thus the proofs for Lemma 4.2 and Theorems 4.4 and 4.5 are based on that intuition made precise. However, it may very well be that there is a more straightforward approach to reaching this result that could possibly be more easily extended to Reed-Muller codes of higher orders.

If we were to insist on using an extension of Lemma 4.2 for higher orders, we would need to develop a sense of how the codewords from higher order codes are built. While it is possible that there is a similar structure in higher order codes to the one found in first order codes, to the eye it does not seem to be all that similar to our eight simple blocks; considering for example the following codeword from $\mathcal{R}(2, 4)$:

$$\mathbf{u} = 0000.0101.0110.0011.$$

None of the blocks of four in this codeword $\mathbf{u}$ could ever occur in the same word in a first order code. However, the two halves 0000.0101 and 0110.0011 are both elements of the code $\mathcal{R}(2, 3)$. Perhaps a similar lemma to Lemma 4.2 could be proven with the 128 codewords from $\mathcal{R}(2, 3)$ as the base case, though whether this would then lead to a nice extension of the two Theorems 4.4 and 4.5 where Lemma 4.2 was (directly or indirectly) used, is a new question to be answered.

Another thing worth looking into is the word error rate, how it changes for different levels of noise (different σ's in the Q-function) and what that means for the suitability of using Reed-Muller codes in practical situations. While it was shown in this thesis that there are certainly cosets with higher minimum Pearson distances than the lower bound previously found by Weber et al., due to the constraints of time, we did not get to dive deeper into the word error rate of these cosets.

During the research phase of this project an interesting discovery was made for the coset of $\mathcal{R}(1, 5)$ determined by $\boldsymbol{\alpha}_5 = 1100.0100.0010.0000/0110.0000.0001.1000$: the number of occurences of the distance $23/64 \cdot 2^m = 11.5$ between two codewords was 16 (as shown in Example 10). For higher values of m, where $\boldsymbol{\alpha}_m = |\boldsymbol{\alpha}_{m-1}|\boldsymbol{\alpha}_{m-1}|$, this number stayed the same! This was admittedly unexpected, since between $\mathcal{R}(1, m)$ and $\mathcal{R}(1, m + 1)$ the total number of codewords increases twofold. A question that comes up is whether all cosets have the property that the number of occurances of the minimum Pearson distance between two words remains the same.

Finally, the results of this thesis only hold for cosets determined by vectors $\boldsymbol{\alpha}$ of weight 2^{m-2} or $3 \cdot 2^{m-2}$, because Theorem 2.9 does so as well. Cosets determined by vectors of different weights may have different minimum Pearson distances entirely. Indeed, if we consider $\mathcal{C} = \mathcal{R}(1, 4)$, which has distance $d_{\min} = 2^{m-1} = 8$, and calculate all the Pearson distances between words like was done in Example 10, we find that the coset $\mathcal{C}_{\boldsymbol{\alpha}}$ where $\boldsymbol{\alpha} = 1100.0100.0110.0000$, has minimum Pearson distance $\delta^*_{\min} = 5.75$ rather than $2^{m-2} = 4$, which, as we know, would be the result when using an $\boldsymbol{\alpha}$ of weight 2^{m-2} or $3 \cdot 2^{m-2}$!

It may be true that Weber's choice of weights for $\boldsymbol{\alpha}$ leads to the "best" lower bound for a general linear block code with the all one- and all zero-word, but it seems that this may not be true for the highly structured Reed-Muller codes. Before venturing into the territory of higher order Reed-Muller codes, it might be prudent to determine which $\boldsymbol{\alpha}$'s could produce better results than $\boldsymbol{\alpha}$'s of weight 2^{m-2} or $3 \cdot 2^{m-2}$.

Bibliography

[1] R. Bu. *Coding Techniques for Noisy Channels with Gain and/or Offset Mismatch*. PhD thesis, Delft University of Technology, Netherlands, 2021. URL https://doi.org/10.4233/uuid:18ab40eb-5336-4498-a20b-b5d9a1663003.

[2] D. R. Hankerson, D. G. Hoffman, D. A. Leonard, C. C. Lindner, K. T. Phelps, C. A. Rodger, and J. R. Wall. *Coding Theory and Cryptography The Essentials*. CRC Press, Boca Raton, 2nd edition edition, 2000.

[3] G. van Hemert. Codes for noisy channels with unknown offset. Bachelor thesis, Delft University of Technology, Netherlands, 2020.

[4] K. A. Schouhamer Immink and J. H. Weber. Minimum pearson distance detection for multilevel channels with gain and/or offset mismatch. *IEEE Transactions on Information Theory*, 60(10): 5966–5974, 2014. doi: 10.1109/TIT.2014.2342744.

[5] T. Kasami. *Weight distributions of BCH codes*. (Chapel Hill, NC) Univ. of North Carolina Press, 1969.

[6] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes*. North-Holland Publishing Company, New York, 2nd reprint edition, 1977.

[7] F. Sala, K. A. Schouhamer Immink, and L. Dolecek. Error control schemes for modern flash memories: Solutions for flash deficiencies. *IEEE Consumer Electronics Magazine*, 4(1):66–73, 2015. doi: 10.1109/MCE.2014.2360965.

[8] C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE mobile computing and communications review*, 5(1):3–55, 2001.

[9] J. H. Weber, R. Bu, K. Cai, and K. A. Schouhamer Immink. Binary block codes for noisy channels with unknown offset. *IEEE Transactions on Communications*, 68(7):3975–3983, 2020. doi: 10.1109/TCOMM.2020.2986200.

[10] R. W. Yeung. *The Science of Information*, pages 1–4. Springer US, Boston, MA, 2008. ISBN 978-0-387-79234-7. doi: 10.1007/978-0-387-79234-7_1. URL https://doi.org/10.1007/978-0-387-79234-7_1.