MULTIPLICATIVE CODES OF REED-MULLER TYPE

BY

JIASHUN SHEN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Mathematics
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2014

Urbana, Illinois

Doctoral Committee:

Professor Bruce Hajek, Chair
Professor Bruce Reznick,Co-Chair
Professor Iwan Duursma, Director of Research
Professor Henry Schenck

Abstract

This is a comprehensive study of multiplicative codes of Reed-Muller type and their applications. Our codes apply to the fields of cryptography and coding theory, especially to multiparty computation and secret sharing schemes. We also study the AB method to analyze the minimum distance of linear codes. The multiplicative codes of Reed-Muller type and the AB method are connected when we study the distance and dual distance of a code and its square. Generator matrices for our codes use a combination of blocks, where a block consists of all columns of a given weight. Several interesting linear codes, which are best known linear codes for a given length and dimension, can be constructed in this way.

To my beloved daughter Chenyu and wife Chen Chen.

Acknowledgments

Many people gave me support to complete this thesis. Many thanks to my thesis adviser, Iwan Duursma, who gave me much useful advice for my Ph.D life and research. Also thanks to my committee members, Bruce Hajek, Bruce Reznick, and Henry Schenck, who gave me useful guidance and comments. Thanks to the department of mathematics of University of Illinois, Urbana Champaign, which gave me financial support and position as a teaching assistant. Thanks to my parents, who give me support for completing the degree. And finally, thanks to my wife and daughter, who brought me a lot of happiness for the later years of my time as a Ph.D candidate.

Table of Contents

List of Abbreviations

BKLC	Best Known Linear Codes.
LSSS	Linear Secret Sharing Schemes
MPC	Multiparty Computation.

List of Symbols

$\mathbf{1}$	Vector where every entry is 1
$\mathbf{0}$	The zero vector
A_i	Matrix where each column is of weight i
C	Linear code C
$C^{(t)}$	The t-th order of linear code C
$\widehat{C}_i^{(t)}$	The t-th order of linear code with blocks of weight i without $\mathbf{1}$
$D_i^{(t)}$	The t-th order of linear code which is generated by the multiplication of t different rows in A_i
n, k, d	Length, dimension and minimum distance of linear code
$RM(r, m)$	The rth order of Reed-Muller code
$RM_I(r, m)$	The linear code which is generated on blocks I.
$RM_I^*(r, m)$	The linear code which is generated on blocks I without $\mathbf{1}$.

Chapter 1

Introduction and Background of Multiparty Computation and Linear Secret Sharing Schemes

1.1 Multiparty Computation

In real life, people may often come together to use their own data or results to calculate some other results. The final results calculated by people are announced to every one of them, while each input should be kept secret. For example, someone wants to be voted as a president of some committee. If she receives more than half of the votes, then she will be elected as the president. However, people should not know each person's vote and only the final result is announced. Hence, it will be very important that we can invent a way of computing a final result from different parties but keep each party's input secret.

Of course, we can invite another trusted party to receive the information from everyone and then compute the result. However, in our real life, if the benefit of cheating or selling each party's information is high enough, the third party may not be so reliable. Hence, paying the trusted party a lot of money in order to keep the secret may be necessary.

Indeed, some methods are used to solve these problems. We need to introduce some protocols in order to find the correct computation result and keep each person's input secret. The basic multiparty computation consists of two calculations, one is addition, the other one is multiplication. In the following, we will introduce these two basic multiparty computations. In the book draft [2], multiparty computation and secret sharing schemes are discussed in detail. From Section 1.1 to Section 1.4, some introduction is given based on the material from the book draft [2].

1.1.1 Protocol Secure Addition

Assume three people have their own input and they want to find the sum of the inputs but keep their own input secret. Let us use P_1, P_2, P_3 for the three parties and their corresponding inputs are $a, b, c \bmod p$, where p is a prime number known to everybody. They want to find the sum $s = a + b + c \bmod p$ without knowing other people's input. The protocol is the following:

1. P_1 chooses two random numbers a_1, a_2 from $\{0, 1, 2, ...p-1\}$, and $a_3 = a - a_1 - a_2 \bmod p$. P_2 chooses two random numbers b_1, b_2 from $\{0, 1, 2, ...p-1\}$, and $b_3 = b - b_1 - b_2 \bmod p$. P_3 chooses two random numbers c_1, c_2 from $\{0, 1, 2, ...p-1\}$, and $c_3 = c - c_1 - c_2 \bmod p$.

2. P_1 distributes a_2, a_3 to P_1, a_1, a_3 to P_2 and a_1, a_2 to P_3. (Of course, P_1 tells a_2, a_3 to himself seems a little bit redundant). P_2 does the same thing and distributes b_2, b_3 to P_1, b_1, b_3 to P_2 and b_1, b_2 to P_3. P_3 distributes c_2, c_3 to P_1, c_1, c_3 to P_2 and c_1, c_2 to P_3.

3. P_1 computes $s_2 = a_2 + b_2 + c_2$ and $s_3 = a_3 + b_3 + c_3$. P_2 computes $s_1 = a_1 + b_1 + c_1$ and $s_3 = a_3 + b_3 + c_3$. P_3 computes $s_1 = a_1 + b_1 + c_1$ and $s_2 = a_2 + b_2 + c_2$. P_1, P_2, P_3 announce their results in this step to all parties.

4. The result $s = s_1 + s_2 + s_3 \pmod p$.

Let us see that each person's input is still kept secret. In Step 2, each person only receives two random numbers from other two people and no further information can be found. In Step 3, P_1 gets additional information s_1 and finds the final answer s. Assume P_1 knows s_1 and knows other information m which can not be derived from s. Notice that knowing s is equivalent to knowing s_1 to P_1 because $s_1 = s - s_2 - s_3$. If s_1 can imply other information m, then s can also imply m, which is a contradiction. Hence, in Step 3, P_1 knows s_1 only helps him to know the final result s and nothing more than that. The secret inputs b and c from P_2 and P_3 are not discovered by P_1.

1.1.2 Protocol Secure Multiplication

Let p be a prime number, assume P_1 has input $a \in \{0, 1, ..., p-1\}$ and P_2 has input $b \in \{0, 1, ..., p-1\}$. They want to compute $ab \pmod p$ securely. A third party with no input P_3 is invited. The following are the steps for Protocol Secure Multiplication

1. P_1 makes shares a_1, a_2, a_3 such that $a_1 + a_2 + a_3 = a$, where a_1, a_2 are random and $a_3 = a - a_1 - a_2$. P_2 makes shares b_1, b_2, b_3 such that $b_1 + b_2 + b_3 = b$, where b_1, b_2 are random and $b_3 = b - b_1 - b_2$.

2. P_1 distributes a_2, a_3 to P_1, $a_1, a_3$to P_2 and a_1, a_2 to P_3. P_2 does the same thing and distributes b_2, b_3 to P_1, b_1, b_3 to P_2 and b_1, b_2 to P_3.

3. P_1 computes $u_1 = a_2b_2 + a_2b_3 + a_3b_2 \mod p$, P_2 computes $u_2 = a_3b_3 + a_1b_3 + a_3b_1 \mod p$, P_3 computes $u_3 = a_1b_1 + a_1b_2 + a_2b_1 \mod p$.

4. P_1, P_2, P_3 use Protocol Secure Addition to compute the sum $s = u_1 + u_2 + u_3$.

The argument why the multiplication is secure is similar to the argument for addition. A good example for secure multiplication may happen in the real life. Assume A is a female and B is a male and they may be interested in each other. If A is interested in B, then her input $a = 1$, and if B is interested in A, then his input $b = 1$. They want to know whether they are both interested in each other and the value $s = ab$ should be computed securely. If the value $s = 1$, then it means that they are both interested in each other. If B is interested in A but A does not like B, without secure multiplication, A and B may feel embarrassed later on. Hence, keeping each other's input secret may avoid embarrassment later on. If A is not interested in B, then she would definitely know that the value of computation is 0 and she does not know whether B likes her or not. In this case, B knows that A is not interested in him but he should not be worried too much because A does not know his choice.

Here is another example for multiparty: Assume there are two secrets a_0 and b_0 and people want to compute a_0b_0 without disclosing a_0 and b_0. Let $f(x) = a_1x + a_0$, $g(x) = b_1x + b_0$. a_1 and b_1 are random numbers. The way to do this is the following:

Three parties P_1, P_2 and P_3 are invited, and each party P_i holds input x_i, which is known to everyone. Assume party P_i knows $f(x_i), g(x_i)$. If they announce their results to other people, any two of them can discover the secrets. To avoid this, parties compute $f(x_i)g(x_i)$ and announce it to other people. Notice that $f(x)g(x) = a_1b_1x^2 + (a_0b_1 + b_0a_1)x + a_0b_0$. Three points can determine a quadratic function's coefficients, which gives us the result a_0b_0. Since a_1 and b_1 are random numbers, we can not recover secrets a_0 and b_0.

1.2 Secret Sharing and Shamir Schemes

Assume we have n parties and some secret s is shared among them. Unfortunately, not all parties are honest. To protect the secret, we need several requirements. We may require that any t or fewer parties can not recover the secret and any l or more parties can recover the secret. A well-known method for secret sharing is the Shamir Scheme.

In Shamir Schemes, a secret s is involved with a polynomial $f(x) = s+s_1x+s_2x^2+...+s_{n-1}x^{n-1}$. Notice that there are n coefficients in this polynomial. Each party P_i has their own x_i and receives their own share $f(x_i)$ privately. Hence, any set of parties of size n or more would be able to recover the secret and parties of size less than n can not recover any information about s. When the n parties share their own outputs in order to recover the secret, they mainly use the method of Lagrange Interpolation:

The Langrange Interpolation Method is this. Let $f(x)$ be a polynomial of degree $n-1$ and S is a set of size $|S| = n$. Then

$$f(x) = \sum_{i \in S} f(x_i) f_i(x)$$

where x_i is the input for some individual party and $f(x_i)$ is the corresponding share. $f_i(x)$ should have the property that $f_i(x_i) = 1$ and $f_i(x_j) = 0$ when $j \neq i$. A good approach for $f_i(x)$ is that

$$f_i(x) = \prod_{j \in C, j \neq i} \frac{x - x_j}{x_i - x_j}.$$

Then the coefficients of $f(x)$ can be derived from the expression $\sum_{i \in S} f(x_i) f_i(x)$. This method saves time for solving the system of linear equations when people put their own x_i and $f(x_i)$ to the original polynomial. A quick example will be the following:

Let the prime number $p = 13$ and the secret $s = 3$. Assume there are five parties P_1, P_2, P_3, P_4, P_5 and their inputs are $1, 2, 3, 4, 5$. We need that three parties can recover the secret, so the degree of the polynomial is two. We randomly choose $s_1 = 2$ and $s_2 = 7$ so the polynomial is $f(x) = 3+2x+7x^2$. Hence, we send $(1, 12)$ to P_1, $(2, 9)$ to P_2 and $(3, 7)$ to P_3. Hence

$f_1(x) = \frac{x-2}{1-2} \cdot \frac{x-3}{1-3} = 7(x-2)(x-3) = 7x^2 + 4x + 3$

$f_2(x) = \frac{x-1}{2-1}\frac{x-3}{2-3} = 12x^2 + 4x + 10$

$f_3(x) = \frac{x-1}{3-1}\frac{x-2}{3-2} = 7x^2 + 5x + 1$

Then P_1, P_2, P_3 announce their own shares to each other and find out $f(x) = f(1)f_1(x) + f(2)f_2(x) + f(3)f_3(x) = 12(7x^2 + 4x + 3) + 9(12x^2 + 4x + 10) + 7(7x^2 + 5x + 1) = 7x^2 + 2x + 3$. Hence, the secret 3 is recovered by parties P_1, P_2, P_3.

Assuming any two parties in this group are corrupted, they can not use their shares to recover the secret. Moreover, no information about the secret is obtained by these two parties.

1.3 Linear Secret Sharing Schemes

According to [2], an adversary structure $\mathcal{A}$ is a family of subsets of $P = \{P_1, P_2, ..., P_n\}$ in which every set in $\mathcal{A}$ can be corrupted by an adversary. A simplicial adversary structure $\mathcal{A}$ requires $B \subseteq A, A \in \mathcal{A}$ implies $B \in \mathcal{A}$. This means that if some set A of parties are corrupted, then any subset of A can also be corrupted.

An adversary structure $\mathcal{A}$ is called Q_2 if for any $A_1, A_2 \in \mathcal{A}$, $A_1 \cup A_2 \neq P$. An adversary structure $\mathcal{A}$ is called Q_3 if for any $A_1, A_2, A_3 \in \mathcal{A}, A_1 \cup A_2 \cup A_3 \neq P$.

A linear secret sharing scheme $\mathcal{S}$ over a field $\mathbf{F}$ for n players consists of a matrix M. M is the matrix for the scheme $\mathcal{S}$. Player $P_{\phi(i)}$ owns the i th row of M. The players that know row i in M are given by the subset $\phi(i) \subseteq 1, 2, \ldots, n$, where ϕ is a function from $\{1, 2, \ldots, m\}$ to $\{1, 2, \ldots, n\}$. Let M_A be the matrix consisting of the rows from M owned by the set of parties A. Subset $A \subseteq 1, 2, \ldots, n$ knows row i if $\phi(i) \cap A \neq \emptyset$, i.e. if some party in A knows row i.

If $s \in \mathbf{F}$ is the secret, let s be the first coordinate of some column vector $\mathbf{r}_s$. We multiply M with $\mathbf{r}_s$ to distribute the shares to other people. Parties in $\phi(i)$ receive the shares $(M\mathbf{r}_s)_i$. Similarly, $M_A\mathbf{r}_s$ will be the shares of the set of parties A.

The adversary structure $\mathcal{S}$ consists of some family of subsets of P. If the distribution of $M_A\mathbf{r}_s$ is independent of s, then the set A is called adversary structure. If the secret s is uniquely determined by $M_A\mathbf{r}_s$, then the set A is called qualified.

1.4 Multiplicative Linear Secret-Sharing Scheme and Strongly Multiplicative Linear Secret-Sharing Scheme

A Linear Secret Sharing Scheme is multiplicative if each party $i \in P$ can compute a value c_i from his shares a_i for a and b_i for b. The product ab is a linear combination of all the c_i, $i \in P$. Notice that the Shamir Linear Secret Sharing Scheme is multiplicative when the number of parties $n > 2t$. If $a_i = f(x_i)$, $b_i = f(y_i)$, and $c_i = a_i b_i$, if we want to have $f(0)g(0)$ based on the values $c_1, c_2, ..., c_n$, we need n is greater than the degree of the polynomial $f(x)g(x)$, which means $n > 2t$.

Assume that the adversary can control at most t parties from $\{P_1, P_2, ..., P_n\}$.

Definition 1.4.1. A multiplicative Linear Secret Sharing Scheme is strongly multiplicative if the output ab can be computed as a linear combination of $n - t$ values of c_i.

For the Shamir Linear Secret Sharing Scheme, in order to be strongly multiplicative, we need $n - t > 2t$, which means $n > 3t$.

Our work is to construct multiplicative LSSS and strongly multiplicative LSSS from linear codes. A linear code C is self-orthogonal if for any $\mathbf{a} \cdot \mathbf{b} = (a_1, a_2, ..., a_n) \in C$ and $\mathbf{b} = (b_1, b_2, ..., b_n) \in C$, we have $a = a_1 b_1 + a_2 b_2 + ... + a_n b_n = 0$. A linear code C is strongly self-orthogonal if for any $\mathbf{a} = (a_1, a_2, ..., a_n) \in C$, $\mathbf{b} = (b_1, b_2, ..., b_n) \in C$ and $\mathbf{c} = (c_1, c_2, ..., c_n) \in C$, we have $(\mathbf{a} * \mathbf{b}) \cdot \mathbf{c} = a_1 b_1 c_1 + a_2 b_2 c_2 + ... + a_n b_n c_n = 0$.

In [6], the connection between multiplicative LSSS and strongly multiplicative LSSS with linear codes is the following:

Proposition 1.4.1. *(a) If a linear code is self-orthogonal then the corresponding LSSS is multiplicative.*
(b) If a linear code is strongly self-orthogonal then the corresponding LSSS is strongly multiplicative.

Hence, in order to achieve multiplicative LSSS and strongly multiplicative LSSS, we focus on self-orthogonal linear codes and strongly self-orthogonal linear codes. Some methods, such as Reed-Solomon codes, Algebraic Geometry codes, BCH codes and some cyclic codes have already been used to construct self-orthogonal codes and strongly self-orthogonal codes. However, we develop a

different linear code which is the so called "spherically punctured Reed-Muller codes" to achieve the goal. Moreover, since the spherically punctured Reed-Muller codes have a very good combinatorial structure, many interesting properties have also been found. Another good advantage is that our codes are more flexible compared with standard Reed-Muller codes. We can construct codes with good multiplicity with different lengths, dimensions and minimum distances.

1.5 Summary of the results

In the next chapter, we will show our results. First, we prove the condition for a linear code to have multiplicity t when we choose blocks of certain weights and the results are Lemma 2.3.1, Theorem 2.3.1 (page 12), Theorem 2.3.2 (page 14) , Theorem 2.3.3 (page 18), and some results are shown in [5]. In Theorem 2.4.1 (page 19), we also show a general condition when a linear code has multiplicity t. A probability argument is given in Theorem 2.4.3 (page 22). Next, we prove Theorem 2.5.1 (page 22) about the dimension of linear code C for any t-th power when there is only one single block. In Theorem 2.5.2 (page 25), we prove the dimension formula when the blocks are consecutive starting from 1. We also have some conjectures for the dimension in two other cases. The results are shown in Conjecture 1 (page 27) and Conjecture 2 (page 28). Condition one is that there are two blocks. Condition two is that the blocks are congruent to some number modulo 4. Later on, we do some analysis on the dimension when the dimension profile is symmetric. The reason why symmetric dimension is important is because it relates to dual codes which are similar to the condition of Reed-Muller codes. The next section is about the minimum distance. Before we start, we use some background from association schemes [15]. The result when $t = 1$ for a single block is proven in [4]. In Conjecture 3 (page 34), we claim a more general result for general t and how the minimum distance occurs. For multiple blocks, in Theorem 2.6.2 (page 36), we use linear forms to get upper bounds for the minimum distances. In Theorem2.6.3 (page 37), we give results for the minimum distance in some special cases. Next, regular partitions are discussed and partition graphs are created based on partitions and dual partitions. Next, we construct some good linear codes which have good parameters. In the section on Generalized Roos bounds of Chapter 3, we

introduce generalized Roos bounds in Theorem 3.4.1 (page 49) and Theorem 3.4.2 (page 50).

Chapter 2

Multiplicative codes of Reed-Muller type

2.1 Introduction of Reed-Muller codes

We begin this chapter by introducing Reed-Muller codes. According to the references [11][12][15], the Reed-Muller codes can be constructed recursively in the following way.

Definition 2.1.1. For integer $m \geqslant 1$, the first order Reed-Muller codes $RM(1,m)$ are binary codes defined recursively as follows:

(i) $RM(1,1) = \mathbf{F}_2^2 = \{00, 01, 10, 11\}$

(ii) For $m \geqslant 1$, $RM(1,m+1) = \{(\mathbf{u},\mathbf{u}) : \mathbf{u} \in RM(1,m)\} \cup \{(\mathbf{u},\mathbf{u}+\mathbf{1}) : \mathbf{u} \in RM(1,m)\}$

For $r \geqslant 0$, the r-th order of Reed-Muller code $RM(r,m)$ is defined as follows:

Definition 2.1.2. (i) The Reed-Muller codes $RM(0,m)$, for $m \geqslant 1$, are defined to be the all zero vector and $\mathbf{1}$ of length 2^m.

(ii) The first order Reed-Muller codes $RM(1,m)$ for $m \geqslant 1$ are in Definition 2.1.1.

(iii) For any $r \geqslant 2$, the r-th order Reed-Muller codes $RM(r,m)$ is defined for $m \geqslant r-1$, recursively by

$$RM(r,m+1) = \begin{cases} \mathbf{F}_2^{2^r} & m = r-1 \\ \{(\mathbf{u},\mathbf{u}+\mathbf{v}) : \mathbf{u} \in RM(r,m), \mathbf{v} \in RM(r-1,m)\} & m > r-1 \end{cases}$$

An alternative way to view Reed-Muller code is the following:

Let G be a matrix of length 2^m defined as follows:

(i) The first row of G is $\mathbf{1}$.

(ii) By taking off the first row of G, the remaining part is an m by 2^m matrix where the columns of this matrix are all the possible different columns of length m with entries $0, 1$.

Definition 2.1.3. Assume matrix G is defined as above, the r-th order binary Reed-Muller code $RM(r, m)$ is the set of vectors which are the evaluation of polynomials of degree at most r in the variables $x_1, x_2, \ldots, x_m$ evaluated in the 2^m $0, 1$-columns of length m.

For example, the first order Reed-Muller code $RM(1, 3)$ has generating matrix

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

and the second order Reed-Muller code $RM(2, 3)$ has generating matrix

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

The main parameters of a linear code are its length n, dimension k, and minimum distance d. The parameters of a Reed-Muller code are as follows: The following is the property is of Reed-Muller code.

Theorem 2.1.1. *If the r-th Reed Muller code $RM(r, m)$ has parameters $[n, k, d]$, then*

$$[n, k, d] = [2^m, \sum_{i=0}^{r} \binom{m}{i}, 2^{m-r}].$$

The dual code of the Reed-Muller code is the code $RM(m-r-1,m)$.

Every codeword except $\mathbf{0}$ and $\mathbf{1}$ in the first order Reed-Muller code has weight 2^{m-1}. This property makes the minimum distance of the first order Reed-Muller code relatively high. Since the Reed-Muller code has a very strong combinatorial structure, several algorithms are known to decode Reed-Muller codes efficiently.

2.2 Our basic construction

In [5], we introduced the following construction, and some of the results of section 2.2 and 2.3 are from [5]. In addition to the usual vector addition on $\mathbf{F}_2^n$, we define another operation called "coordinatewise multiplication" (Hadamard product). It is defined as follows:
For $(a_1,a_2,...,a_n),(b_1,b_2,...,b_n) \in \mathbf{F}_2^n$,$(a_1,a_2,...,a_n) * (b_1,b_2,...,b_n) = (a_1b_1,a_2b_2,...,a_nb_n)$. Based on this definition, we will construct codes that can be used for secure computation. The code construction is similar to that of Reed-Muller codes. In [4], they are called spherically punctured Reed-Muller codes. First of all, we define a code $\widehat{C}$ generated by a matrix G. The codes were introduced independently in [5][3].

$$G = [A_{i_1}A_{i_2}...A_{i_m}], i_1 \leqslant i_2 \leqslant i_3 \leqslant ... \leqslant i_m$$

where A_i is a k by $\binom{k}{i}$ matrix such that the columns of A_i consist of all different binary vectors of length k and weight i.

Also, we define the linear code $C = \widehat{C} + \mathbf{1}$, where $\mathbf{1}$ is the all-one vector. In other words, C is generated by the above G and the all-one vector $\mathbf{1}$.

For given m, and for $I \subset \{0,1,\ldots,m\}$, we define the Reed-Muller codes $RM_I(r,m)$ inductively via

(1) $RM_I(1,m)$ is generated by the $\mathbf{1}$ and the rowspan of the matrix $(\cdots|A_i|\cdots)_{i\in I}$. The latter is the matrix of all m-vectors with weight $i \in I$.

(2) $RM_I(r,m) = \langle \mathbf{a} * \mathbf{b} | \mathbf{a} \in RM_I(r-1,m), \mathbf{b} \in RM_I(1,m) \rangle$

Let $RM_I^*(r,m)$ be the linear code generated similar as $RM_I(r,m)$ but without $\mathbf{1}$.

In fact, $RM_I(r,m)$ is the linear code C when the blocks form the set I. $RM_I^*(r,m)$ is the linear code $\widehat{C}$ when the blocks form the set I.

Notice that the weight of each row of A_i is $i \cdot \binom{k}{i} \cdot \frac{1}{k} = \binom{k-1}{i-1}$. Under the above construction, we have the following result:

Lemma 2.2.1. *For a single matrix A_i, for $t \geqslant 1$, if we take t different rows from A_i and multiply them together, we will always get a vector of weight $\binom{k-t}{i-t}$.*

Proof. For the weight of the row, if the row is the product of rows $\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_t$, to count the weight, for a certain column j, we have to make sure that all $\mathbf{r}_1, \mathbf{r}_2, ...\mathbf{r}_t$ have 1 in column j and since we have put all weight i columns in the matrix, we know that the total choice for such kind of columns is $\binom{k-t}{i-t}$ (this is because for the remaining $k-t$ positions, we can freely chose $i-t$ $1s$.) □

Example 2.2.1. If $k = 5, i = 3$, then

$$\mathrm{A}_3 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}.$$

If we multiply any 2 different rows in A_3, we will always get a vector of weight $\binom{5-2}{3-2} = 3$.

Next, we define the t-th generation of C or $\widehat{C}$ denoted by $C^{(t)}$ or $\widehat{C}^{(t)}$, to be as follows: $\widehat{C}^{(t)} = \{\mathbf{c}_1 * \mathbf{c}_2 \cdots \mathbf{c}_t |\ \mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_t$ are rows in $G\}$. $C^{(t)} = \{\mathbf{c}_1 \cdot \mathbf{c}_2 \cdots \mathbf{c}_t |\ \mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_t$ are either $\mathbf{1}$ or rows in $G\}$. Thus, it is quite easy to see that for $\mathbf{v}_j \in C^{(j)}$, if $j_1 + j_2 + ... + j_m \leqslant t$, then $\mathbf{v}_{j_1} * \mathbf{v}_{j_2} ... * \mathbf{v}_{j_m} \in C^{(t)}$. For example, if $t = 6$, then for $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 \in C^{(2)}$, we have that $\mathbf{v}_1 * \mathbf{v}_2 * \mathbf{v}_3 \in C^{(6)}$.

Remark 2.2.1. If C_i is generated by block A_i and C_{k-i} is generated by block A_{k-i}, then the two

linear codes C_i and C_{k-i} are equal. This is because a row in C_i can be found by adding $\mathbf{1}$ to a row in C_{k-i} and vice versa.

Our motivation is to choose $k, i_1, i_2, ..., i_m$ such that every vector in $C^{(t)}$or $\widehat{C}^{(t)}$ has even weight for some t.

2.3 The Combination when all the codes in $\widehat{C}^{(t)}$ and $C^{(t)}$are of even weight

2.3.1 When $\widehat{C}^{(t)}$ are of even weights

Next, we take any different t ($t \leqslant i$) rows $\mathbf{c}_1, \mathbf{c}_2 \cdot \cdots \cdot \mathbf{c}_t$ from A_i, and we get the following vector $\mathbf{c}_1 \cdot \mathbf{c}_2 \ldots \cdot \mathbf{c}_t$. There are $\binom{k}{t}$ such vectors. Then we define $A_i^{(t)}$ to be the matrix consisting of all these vectors. So $A_i^{(t)}$ is a $\binom{k}{t}$ by $\binom{k}{i}$ matrix. We have the following lemma:

Lemma 2.3.1. *For $1 \leqslant t \leqslant i$, $A_i^{(t)}$ is a $\binom{k}{t} \times \binom{k}{i}$ matrix, the weight of each column is $\binom{i}{t}$ and the weight of each row is $\binom{k-t}{i-t}$.*

Proof. First, we need to show that each row is different. To show this, consider two different sets of rows $\{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_t\} \neq \{\mathbf{r}'_1, \mathbf{r}'_2, ..., \mathbf{r}'_t\}$. If $\mathbf{r}_1\mathbf{r}_2...\mathbf{r}_t = \mathbf{r}'_1\mathbf{r}'_2...\mathbf{r}'_t$, then after multiplication by $i-t$ rows $\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_{i-t}$, we will get the same answer, say $\mathbf{r}_1\mathbf{r}_2...\mathbf{r}_t\mathbf{s}_1\mathbf{s}_2...\mathbf{s}_{i-t} = \mathbf{r}'_1\mathbf{r}'_2...\mathbf{r}'_t\mathbf{s}_1\mathbf{s}_2...\mathbf{s}_{i-t}$. But $\{\mathbf{r}_1, \mathbf{r}_2, ...\mathbf{r}_t, \mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_{i-t}\} \neq \{\mathbf{r}'_1, \mathbf{r}'_2, ..., \mathbf{r}'_t, \mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_{i-t}\}$. However, any product of i rows is a weight 1 vector of length $\binom{k}{i}$. For any set of i rows, there is a unique column such that all the rows contain 1 at that column. So different sets of i rows will give different products. This gives a contradiction. And we see that for different sets of rows $\{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_t\} \neq \{\mathbf{r}'_1, \mathbf{r}'_2, ..., \mathbf{r}'_t\}$, the product $\mathbf{r}_1\mathbf{r}_2...\mathbf{r}_t \neq \mathbf{r}'_1\mathbf{r}'_2...\mathbf{r}'_t (1 \leqslant t \leqslant i)$. So the matrix $A_i^{(t)}$ is a $\binom{k}{t} \times \binom{k}{i}$ matrix with different rows.

Next, we need to show the weight of the column is $\binom{i}{t}$. This is obvious, because we take all the possible products of t rows of the original matrix and so the weight of each column is $\binom{i}{t}$. If two columns in $A_i^{(t)}$ are the same, then we would see that the corresponding two columns in A_i are the same, contradiction. So the columns of the matrix $A_i^{(t)}$ are different. According to Lemma

2.3.1, we have the desired result. We can check that the total number of $1s$ in the matrix A_i is $\binom{k}{t}\binom{k-t}{i-t} = \binom{k}{i}\binom{i}{t}$, and this equation is indeed the subcommittee identity. □

Definition 2.3.1. Let $D_i^{(t)}$ be the linear code generated by the rows in $A_i^{(t)}$. Let $\widehat{C}_i^{(t)}$ be the linear code defined as $\widehat{C}_i^{(t)} =< S_1, S_2, .., S_t >$, where

$$S_j = \{\mathbf{c}_1 \cdot \mathbf{c}_2 \cdots \mathbf{c}_j | \mathbf{c}_1, \mathbf{c}_2 ..., \mathbf{c}_j \text{ are different rows in} A_i\}$$

Let $C_i^{(t)} = \mathbf{1} + \widehat{C}_i^{(t)}$. It is easy to see that $\widehat{C}_i^{(t)} = D_i^{(1)} + D_i^{(2)} + ... + D_i^{(t)}$. Also, $\widehat{C}_i^{(t)} =< \mathbf{c}_1 \cdot \mathbf{c}_2 \cdots \mathbf{c}_t | \mathbf{c_i}$ is a row in A_i. $>$

Let us consider the linear code generated by the matrix $G = [A_{i_1} A_{i_2} ... A_{i_m}]$.
Let $G^{(t)} = \left[A_{i_1}^{(t)} A_{i_2}^{(t)} ... A_{i_m}^{(t)}\right]$, and let $D^{(t)}$ be the linear code generated by the rows of $G^{(t)}$. We consider a vector $\mathbf{v} = (v_1, v_2, ..., v_k) \in \mathbf{F}_2^k$, and let A_s be in G if and only if $\mathbf{v}_s = 0$. Vector $\mathbf{v}$ gives us information whether some block A_i is chosen or not. For example, if $k = 10$ and $\mathbf{v} = (1, 0, 1, 0, 0, 1, 1, 0, 1, 1)$, then $G = [A_1 A_3 A_6 A_7 A_9 A_{10}]$. For fixed k and t, we want to find all the possible vectors $\mathbf{v} \in \mathbf{F}_2^{\mathrm{k}}$ such that the weight of each row of $G^{(t)}$ is even.

Proposition 2.3.1. *The set of $\mathbf{v} \in \mathbf{F}_2^k$ such that each row of $G^{(t)}$ has even weight is a subspace of $\mathbf{F}_2^k$ with dimension $k - 1$.*

Proof. First of all, we want to prove that it forms a vector space. Assume G_1 corresponds to $\mathbf{v}_1$ with weight $w_1 + w$ and G_2 corresponds to $\mathbf{v}_2$ with weight $w_2 + w$, where w is the weight of the common part of G_1 and G_2. If G_3 corresponds to $\mathbf{v}_1 + \mathbf{v}_2$, then each row of G_3 has weight $w_1 + w_2 - 2w$. Since $w_1 + w_2$ are even, w_3 is also even. Since $\mathbf{v} = \mathbf{0}$ also gives us even weight, we see that the set of vectors which make the weight of each row of $G^{(t)}$ even is a vector space. We know from Lemma 2.3.1 that the weight of each row of $A_i^{(t)}$ is $\binom{k-t}{i-t}$. Consider the binomial coefficients $\binom{k-t}{0}, \binom{k-t}{1}, ..., \binom{k-t}{k-t}$. If $\binom{k-t}{s_1}, \binom{k-t}{s_2}, ..., \binom{k-t}{s_j}$ are even, then we define the set of vectors $E = \{\mathbf{e}_{i+t-1}^k = (0, 0, ..0, 1, 0..0) \in \mathbf{F}_2^k | t \in \{s_1, s_2, ..., s_j\}$, and 1 occurs at position $i+t-1$ in $\mathbf{e}_{i+t-1}^k\}$. So E has cardinality j. Suppose $\binom{k-t}{t_1}, \binom{k-t}{t_2}, ..., \binom{k-t}{t_l}$ are odd, then we define the following set of vectors $F = \{\mathbf{f}_{u+t-1, v+t-1}^k = (0, 0, .., 1, 0..0, 1, 0..0) \in \mathbf{F}_2^k | u < v$ are pairs of consecutive numbers in

$\{t_1, t_2, ..., t_l\}$, the two 1s occur at position $u+t-1, v+t-1\}$, and this set has cardinality $l-1$. It is also obvious that $j+l = k-t+1$. We also define a set of vectors $G = \{\mathbf{e}_r^k \in \mathbf{F}_2^k | 1 \leqslant r \leqslant t-1\}$. So it is obvious that $E \cup F \cup G$ gives us a basis for our desired vector space. Since $|E| = j$, $|F| = l-1$, $|G| = t-1$ and E, F, G are mutually disjoint, we have that $|E \cup F \cup G| = j+l-1+t-1 = k-1$. So our theorem holds. □

Remark 2.3.1. The above theorem not only tells us that half of the vectors in $\mathbf{F}_2^k$ give even weight vectors, but also tells us how to find a basis of such vector space. Thus, we have characterized all the possible cases of vectors in $\mathbf{F}_2^k$ such that for fixed k, t, any row in $G^{(t)}$ has even weight.

Next, for a fixed value k, we wish to find those vectors in $\mathbf{F}_2^k$ which make the rows of $G^{(t)}$ even for several choices of t at the same time. For example, if $k = 10$, we may consider this problem: for what vectors in $\mathbf{F}_2^{10}$, are the rows of $G^{(t)}$ even for $t = 1, 2, 3, 4$? Furthermore, we see that the cardinality of S, which is a subset of $\{1, 2, 3, ..., k\}$, can determine the number of choices of $\mathbf{v} \in \mathbf{F}_2^k$. In the following, we prove a relation between the set S and the number of choices for $\mathbf{v}$.

Theorem 2.3.1. *If $|T|$=m, the vectors which make any matrix $G^{(t)}$, for $t \in T$, have all even weight rows form a vector space of dimension $k-m$, and so the number of choices for $\mathbf{v} \in \mathbf{F}_2^k$ is 2^{k-m}.*

Proof. The fact that vectors which make G^S is a vector space is proved similarly as above. Consider a set $S = \{a_1, a_2, ..., a_m\} \subseteq \{1, 2, 3, ..., k\}$, with $a_1 > a_2 > a_3 > ... > a_m$. From what we did above, for $G^{(a_1)}$ to have all even weight rows, we know that for the last $k - a_1 + 1$ positions, we have that the dimension is $k - a_1$. So the dimension of vector space which make $G^{(a_1)}$ to have all even weight rows is $a_1 + k - a_1 + 1 = k - 1$. Now consider the $a_2, a_2 + 1, ..., a_1 - 1$ positions, if a vector makes $G^{(a_1)}, G^{(a_2)}$ to have all even weight rows, when the last $k - a_1 + 1$ positions are fixed, the positions $a_2, a_3, ..., a_1 - 1$ should rise to dimension $a_1 - a_2 - 1$ (because half of the vectors with length $a_1 - a_2 - 1$ give even weight and half odd weight). So for $G^{(a_1)}, G^{(a_2)}$ to all have even weight rows, the dimension is $k - a_1 + (a_1 - a_2 - 1) + a_2 - 1 = k - 2$. Similarly, we can repeat this argument, for fixed $a_2, a_2+1, ..., a_1, a_1+1..., k$ positions, consider the positions $a_3, a_3+1, ..., a_2-1$, half of them will make that $G^{(a_1)}, G^{(a_2)}, G^{(a_3)}$ all have even weight rows. So the dimension is $k-a_1+(a_1-a_2-1)+(a_2-a_3-1)-(a_3-1) = k-3$. Therefore, we can repeat this argument to see that

the dimension of the vector space is $k-a_1+(a_1-a_2-1)+(a_2-a_3-1)-...-(a_{m-1}-a_m-1)+(a_m-1) = k-m$. So the number of choices for $\mathbf{v} \in \mathbf{F}_2^k$ is 2^{k-m}. $\square$

Example 2.3.1. For $k = 10$, $S = \{1, 3, 5, 8\}$, we can first choose the last $10 - 8 + 1 = 3$ positions, for example, we can let the last three positions to be $(1, 0, 1)$ in order to make any matrix in $G^{(8)}$ to have all even rows. In this case, A_8 and A_{10} are chosen and the rows of $G^{(8)}$ have weight $\binom{10-8}{8-8} + \binom{10-8}{10-8} = 2$, which is even. For $t = 5$, $k - t = 5$, so $(1, 0, 1)$ in the last three positions will give rise to weight $\binom{10-5}{8-5} + \binom{10-5}{10-5} = 11$, which is odd, so we can pick positions 5,6,7 to be (1,0,1). Next, $t = 3$, $10 - t = 7$, so for the last six positions $(1, 0, 1, 1, 0, 1)$, it results in a total weight of $\binom{10-3}{5-3} + \binom{10-3}{7-3} + \binom{10-3}{8-3} + \binom{10-3}{8-3}$ = even. So we can pick positions 3,4 to be $(1, 1)$. For $t = 1$, since the last 8 positions are $(1, 1, 1, 0, 1, 1, 0, 1)$, it gives a total weight of $\binom{10-1}{3-1} + \binom{10-1}{4-1} + \binom{10-1}{5-1} + \binom{10-1}{7-1} + \binom{10-1}{8-1} + \binom{10-1}{10-1}$ =odd, so we can pick the first 2 positions to be $(1, 0)$, therefore, the vector $(1, 0, 1, 1, 1, 0, 1, 1, 0, 1)$ makes any matrix in G^S have all even weight rows.

Next, in our construction, we are more interested in $T = [t] = \{1, 2..., t\}$ for $t \leqslant k$. If each row in $G^{(t)}$ has even weight for every $t \leqslant n$, then for every $t \leqslant n$, every vector in $D^{(t)}$ has even weight and thus every vector in $\widehat{C}^{(n)}$ has even weight. As is shown in the above results, for fixed $n \leqslant k$, since $[n]$ has cardinality n, the vector space U which makes $\widehat{C}^{(n)}$ to have all even weight vectors has dimension $k - n$. Let us give a way to find a basis for the vector space U.

First, we need to construct a matrix M_k, where the $s - t$ entry of M_k is $m_{st} = \binom{s-1}{t-1}$ for $s \geqslant t$, for $s < t$, $m_{st} = 0$. Here, we denote $\binom{0}{0} = 1$. We want to have a matrix N_k such that $M_k N_k = T_k$, where T_k is a k by k matrix with $t_{uv} = 1$ for $u \leqslant v$ and $t_{uv} = 0$ for $u > v$. The first column in matrix N_k denotes the choice for which $\widehat{C}^{(1)} \subseteq \widehat{C}^{(2)} \subseteq ... \subseteq \widehat{C}^{(k-1)}$ are all even weight codes but $\widehat{C}^{(k)}$ has some odd weight vector. Similarly, the j-th column in matrix N_k denotes the choice for which $\widehat{C}^{(1)} \subseteq \widehat{C}^{(2)} \subseteq ... \subseteq \widehat{C}^{(k-j)}$ have all even weight codes but $\widehat{C}^{(k-j+1)} \subseteq \widehat{C}^{(k-j+2)} \subseteq ... \subseteq \widehat{C}^{(k)}$ have some odd weight vectors. We have the results below:

Lemma 2.3.2. *If M_k is defined as above, then $M_k^{-1} = M_k$.*

Proof. We just consider M_k^2, when $u < v$, the uv entry for M_k^2 is 0 because M_k is a lower triangular

matrix. When $u \geqslant v$, the $u-v$ entry for M_k^2 is

$$m_{uv}^{(2)} = \sum_{i=v}^{u} \binom{u-1}{i-1}\binom{i-1}{v-1} = \sum_{i=v}^{u} \frac{(u-1)!}{(u-i)!(i-v)!(v-1)!}.$$

Hence, $m_{uv}^{(2)} = \sum_{i=v}^{u} \binom{u-v}{i-v}\binom{u-1}{v-1} = \binom{u-1}{v-1}2^{u-v}$. When $u > v$, we see that m_{uv} is even, which is 0 modulo 2. When $u = v$, we see that $m_{uv} = 1$. So $M_k^2 = I_{kk}$. □

Proposition 2.3.2. *If M_k, N_k, T_k are defined as above, then*

$$N_k = \begin{pmatrix} 1 & 1 & . & . & . & 1 \\ & & & & & 0 \\ & & & & & 0 \\ & & M_{k-1} & & & . \\ & & & & & . \\ & & & & & 0 \end{pmatrix}.$$

Proof. By Lemma 2.3.2, we see that $N_k = M_k T_k$. The uv entry for N_k is

$$n_{uv} = \sum_{i=1}^{v} \binom{u-1}{i-1}.$$

For $u > 1$, we show that $n_{uv} - m_{(u-1)v}$ is a even number. Indeed,

$$n_{uv} - m_{(u-1)v} = \sum_{i=1}^{v} \binom{u-1}{i-1} - \binom{u-1-1}{v-1} = 2\sum_{i=0}^{v-1} \binom{u-2}{i},$$

which is even. So n_{uv} is congruent to $m_{(u-1)v}$ modulo 2. This shows why M_{k-1} is a submatrix in N_k. □

Example 2.3.2. For example, if $k = 6$, then

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

and

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

By what we proved above,

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

From what we did above, we have the following theorem:

Theorem 2.3.2. *If $(a_1, a_2, \ldots, a_k)^T$ is the vector generated by the first m columns in matrix M_k, then the corresponding linear code has characteristic vector is $(a_k, a_{k-1}, \ldots, a_1)$ for blocks $1, 2, ..., k$. And it has multiplicity $k - m$.*

2.3.2 The combination when all the codes in $C^{(t)}$ are of even weight

The above shows the combination of blocks A_i such that the codewords in $\widehat{C}^{(t)}$ all have even weight. Since $C^{(t)} = \widehat{C}^{(t)} + \mathbf{1}$, we just need to make sure that the length of the codes are even. There are two ways to make it happen. One way is easy, if the length is already even, we can just add $\mathbf{1}$. If the length is odd, we can add $\mathbf{1}$ and an additional column with 1 at the top and zeros below.

Definition 2.3.2. A linear code C has multiplicity t if $C^{(t)}$ has all even weight codewords but $C^{(t+1)}$ does not.

Example 2.3.3. Consider $k = 6$, we take the sum of the second and the third column in M_6 as the vector, which corresponds to block 1,4,5. This linear code C has multiplicity 3 and length $\binom{6}{1} + \binom{6}{4} + \binom{6}{5} = 27$. To make $C^{(3)}$ even, we add $\mathbf{1}$ on top and add an additional column:

The original code has generator matrix

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

After adjustment, it becomes:

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

In terms of $[n, k, d]$, a $[27, 6, 12]$ code becomes $[28, 7, 12]$ code. Both codes have the biggest minimum distance for given length and dimension.

There is another way to make $C^{(t)}$ even without adding an additional column.

Theorem 2.3.3. *Let V be the vector space corresponding to the characteristic vector which make $C^{(t)}$ even. For fixed k, in the matrix M_k, if $\binom{k}{r}$ is even, then the $r+1$ th column in M_k is a vector in the basis of the space V . If $\binom{k}{r}$ is odd, then the sum of the first column and the r th column in M_k is a vector in the basis of the space V. The dimension is equal to $k-t-1$.*

Proof. First, we would like to see that

$$\sum_{j=0}^{k} \binom{k-j}{r}\binom{k}{k-j} = \begin{cases} 0 & r \neq k \\ 1 & r = k \end{cases},$$

where the answer is taken modulo 2. This is an easy result from the fact that $M_k^2 = I_k$. For $r \neq k$, $0 = \sum_{j=0}^{k} \binom{k-j}{r}\binom{k}{k-j} = \binom{k}{r} + \sum_{j=1}^{k} \binom{k-j}{r}\binom{k}{k-j}$, notice that $\sum_{j=1}^{k} \binom{k-j}{r}\binom{k}{k-j}$ has the same parity as the length of the code when we choose the r th column as the characteristic vector for blocks. Hence, for a fixed t, if $C^{(t)}$ has all even weight codes, among the first t columns in M_k, for $r \leqslant k-t$, if the value $\binom{k}{r}$ is even, then the r-th column in M_k is a vector in the basis. Since the first column has $r = 0$, $\binom{k}{0} = 1$, the first column will always give us an odd weight. Therefore, for some r th column in M_k, if $\binom{r}{k}$ is odd, then the sum of this column with the first column is a vector in the basis. The dimension is therefore equal to $k-t-1$. □

The above result gives us a chance to pick up even length codes in $\widehat{C}^{(t)}$ so that $C^{(t)}$ will be even after adding the all one vector.

2.4 Characterization of linear codes with multiplicity t

Let us consider linear codes C which contain $\mathbf{1}$ and are of dimension $k+1$ such that $C^{(t)}$ is an even weight code. Assume that the generator matrix of C is a $k+1$ by n matrix with $\mathbf{1}$ in the first row and no repeated columns, so that C is a punctured code of the Reed-Muller code $RM(1,k)$. If some position of $RM(1,k)$ is chosen, then we denote 1 at that position, if else, we denote 0. Hence, we have a characteristic vector of length 2^k. Let $\mathbf{c}_t$ be the product of t vectors in C. The weight of

$\mathbf{c}_t$ is just the weight of the unpunctured code in $RM(1,k)$ multiplied with the characteristic vector $\mathbf{v}$. In other words,

$$wt(\mathbf{c}_t) = wt(\mathbf{v} * \mathbf{w}).$$

where $\mathbf{w}$ is a codeword in $RM(t,k)$

For $\mathbf{c}_t$ to have even weight, we need $\mathbf{v}\cdot\mathbf{w}$ to have even weight, hence, $\mathbf{v}$ must be in $RM(k-t-1,k)$. And indeed the reverse also holds.

Theorem 2.4.1. *For linear codes C with $\mathbf{1}$ (also called self complimentary) and dimension $k+1$ and no repeated columns in the generating matrix, $C^{(t)}$ is even if and only if the characteristic vector v is in $RM(k-t-1,k)$.*

Hence, we can characterize all the possible linear codes with no repeated columns and have $\mathbf{1}$.

Theorem 2.4.2. *If v is in $RM(k-t-1,k)$, which is a $[2^k, s, 2^{t+1}]$ code, then v can be written as a sum of N vectors in $RM(k-t-1,k)$ of the minimum weight 2^{t+1}.*

Proof. Since any codeword in $RM(r,k)$ is an evaluation of polynomial of degree less than or equal to r. As $x_1x_2 = x_1x_2(1+x_3) + x_1x_2x_3$, we can rewrite any codeword in $RM(r,k)$ into a sum of some vectors where each vector is a multiplication of r components. Since any multiplication of r components in $RM(r,k)$ has weight 2^{k-r}, so the characteristic vector is a sum of weight 2^{t+1} codes. □

According to this observation, we can add some repeated blocks to the original generating matrix such that the extended generating matrix divides into N blocks where each block is $RM(1,t+1)$.

For example, a (6,3) code has generating matrix

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

After adding 4 columns

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

we can make the generating matrix of C as

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

After some permutation, we can arrange the generating matrix of C as

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

Each block is a $RM(1,3)$ code.

Moreover, we write the $RM(1,5)$ in the following way:

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

The characteristic vector is

$$0000001111111111111111111111000000$$

and it is a sum of the following 3 vectors:

$$0010000010000101101010000100000100$$

$$0010000001110000000001111000000100$$

$$0000001100001010010101000011000000$$

Applying the same idea, for a (10,5) code, which has $t = 2$, we can view this code as a punctured linear code of $RM(1,9)$. The length of this code is 252 and we can add 28 repeated columns so that it becomes a length 280 code and this code can be divided into 35 $RM(1,3)$ codes.

Also, for a (14,7) code, which has $t = 6$, this linear code can be added 1048 columns to make it a length 4480 code. This code can be divided into 35 $RM(1,7)$ codes.

Let us consider a probability problem:

Theorem 2.4.3. *Given a linear code C of dimension $k+1$ with even length and* **1**, *the probability of this linear code to have multiplicity t is*

$$P(b) = \frac{2^{k_t} - 1}{2^{2^k - 1} - 1}$$

where k_t is the dimension of $RM(k-t-1)$, which is $\sum_{i=0}^{k-t-1} \binom{k}{i}$.

Proof. The proof is obvious, the answer is the number of characteristic vectors which has multiplicity t over the total number of choices. The characteristic vector is not **0**, so we minus 1 on both sides. □

2.5 Dimension of $C^{(t)}$

The dimension of $C^{(t)}$ when there is only one block is shown in the following result:

Theorem 2.5.1. *For $t \leqslant i$, if $C^{(t)}$ has only one block A_i, then we have that Dimension$(C^{(t)}) = \binom{k}{t}$.*

Proof. In order to prove the dimension, we need to choose a basis and show that the number of vectors in the basis is $\binom{k}{t}$. Here is how we choose the basis:

When $t = 0$, we choose the all-one vector.

When $t = 1$, we delete the first row x_1 from the matrix A_i.

When $t = 2$, in addition to the above deletion, from $A_i^{(2)}$, we delete 2-tuples which contain x_1 or x_2x_3.

When $t = 3$, in addition to the above deletion, from $A_i^{(3)}$, we delete 3-tuples which contain $x_1, x_2x_3, x_2x_4x_5$ or $x_3x_4x_5$.

When $t = 4$, in addition to the above deletion, from $A_i^{(4)}$, we delete 4-tuples which contain $x_1, x_2x_3, x_2x_4x_5$,$x_3x_4x_5, x_2x_3x_6x_7, x_2x_4x_6x_7, x_2x_5x_6x_7, x_3x_4x_6x_7, x_3x_5x_6x_7$.

For general t, in addition to the deletion in the previous generation, from t tuples, we apply the deletion to the previous codes that contain x_1, x_2x_3 etc.). Next, we delete $x_{b_1}x_{b_2}...x_{b_{t-2}}x_{2t-2}x_{2t-1}$ where $b_1 < b_2 < ... < b_{t-2} \in \{2, 3, ..., 2t-3\}$ and $x_{b_1}x_{b_2}...x_{b_{t-2}}$ is not in the previous delete combination. Let a_t be the number of "extra" deletions from t tuples in the t generation. So $a_1 = a_2 = 1$, $a_3 = 2$, $a_4 = 5$, $a_5 = 14$... Hence, we have a recurrence for a_t :

$$a_t = \binom{2t-4}{t-2} - \sum_{i=2}^{t-2} a_i \binom{2t-2i-2}{t-i-2}. \tag{2.1}$$

It turns out that $a_{t+1} = \frac{1}{n+1}\binom{2n}{n}$, so a_t is a Catalan number. In the following we will prove this result. The reference for Catalan number is in [9]. First, we show that

$$C_t = \binom{2t-2}{t-1} - \sum_{i=2}^{t-1} C_{i-1} \binom{2t-2i}{t-i-1}.$$

To prove this, we consider an integer grid of size $t-1$ by $t-1$. We count the paths from $(0,0)$ to (t,t) which start with a horizontal move to $(1,0)$ and end with a vertical move from $(t,t-1)$. The number of possible movements is $\binom{2t-2}{t-1}$.

On the other hand, let the very first point on the path that is above the line $y = x$ be $(i-1, i)$, $(2 \leqslant i \leqslant t-1)$. From $(0,0)$ to $(i-1,i)$, we have C_{i-1} ways and from then on, there are $\binom{2t-1-2i+1}{t-1-i}$ ways. So the total number is $\sum_{i=2}^{t-1} C_{i-1}\binom{2t-2i}{t-i-1}$. If no such situation occurs, there are C_t ways. Hence $C_t + \sum_{i=2}^{t-1} C_{i-1}\binom{2t-2i}{t-i-1} = \binom{2t-2}{t-1}$.

Then we will use induction to prove that

$$a_i = C_{i-1} = \frac{1}{i}\binom{2i-2}{i-1}. \tag{2.2}$$

For the first three cases, the result holds. If for $i \leqslant t-1$, $a_i = C_{i-1}$, then for t, $a_t = \binom{2t-4}{t-2} - \sum_{i=2}^{t-2} a_i\binom{2t-2i-2}{t-i-2} = \binom{2t-4}{t-2} - \sum_{i=2}^{t-2} C_{i-1}\binom{2t-2i-2}{t-i-2}$. Hence, from the above result, we know that $a_t = C_{t-1}$.

With the thing we have proved, we will show that the following recurrence holds:

For $t \leqslant i$ and $t \leqslant k-i$, we have

$$a_t = \binom{k}{t-1} - \sum_{j=1}^{t-1} a_j \binom{k-2j+1}{t-j}. \tag{2.3}$$

To prove this, we also use a combinatorial argument. Consider a path from $(0,0)$ to $(t-1, k-t+1)$. The total number of choices is $\binom{k-t+1+t-1}{t-1} = \binom{k}{t-1}$. On the other hand, let j be the first number where the path is from $(j-1, j-1)$ to $(j-1, j)$. The range of j is $1 \leqslant j \leqslant t$. The number of paths from $(j-1, j)$ to $(t-1, k-t+1)$ is $\binom{k-2j-1}{t-j}$. Hence, we have the result that $\binom{k}{t-1} = \sum_{j=1}^{t-1} C_{j-1}\binom{k-2j-1}{t-j} + C_{t-1}$. Therefore, by the above result, we have that $a_t = \binom{k}{t-1} - \sum_{j=1}^{t-1} a_j \binom{k-2j+1}{t-j}$.

Now, we will show that the vectors we delete are redundant. In general, we claim the following statement:

Lemma 2.5.1. $\sum x_{a_1}x_{a_2}, \ldots, x_{a_j} \in < 1, \sum x_{u_1}, \ldots, \sum x_{v_1}x_{v_2}, \ldots, \sum x_{w_1}x_{w_2}...x_{w_{j-1}} >$ *where the summation of* $a_1, a_2, ..., a_j$ *are all positive integers in* $\{1, 2, ...k\}\backslash\{i_1, i_2, ...i_s\}$ *and* $u_1, v_1, v_2, ...w_1, ...w_{j-1}$ *are all positive integers in* $\{i_1, i_2, ...i_s\}$ Before we prove the lemma, we have an example: if $k = 6$ and $i = 3$; we have that $x_1x_3x_5 + x_1x_3x_6 + x_1x_5x_6 + x_3x_5x_6 \in < 1, x_2 + x_4, x_2x_4 > +I$, and I is some ideal.

Proof. To prove this lemma, consider the vector space $< 1, \sum x_{u_1}, \sum x_{v_1}x_{v_2}, \ldots, \sum x_{w_1}x_{w_2}w_{j-1} >$. We just need to consider the weight of the vector $(x_{i_1}, x_{i_2}, ..., x_{i_{k-j}})$. Indeed, the vector space is spanned by the following generating matrix: let $s = k - j$

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

We know that the rank of this matrix is $s + 1$. Indeed, at certain positions, the summation

$\sum x_{a_1}x_{a_2}...x_{a_j}$ is determined by how many 1's in $(x_{i_1}, x_{i_2}, ..., x_{i_s})$, which is exactly the weight of that vector. Hence, there are only 2^{s+1} possible answers for $\sum x_{a_1}x_{a_2}...x_{a_j}$. As the rank of the above matrix is $s+1$, we can see that

$$\sum x_{a_1}x_{a_2}...x_{a_j} \in < 1, \sum x_{u_1}, \sum x_{v_1}x_{v_2}, ... \sum x_{w_1}x_{w_2}...x_{w_{j-1}} >$$

Therefore, by this result, we see that the extra vectors we take out are indeed redundant.

The final task is to show that the remaining vectors are linearly independent. To prove this, we may just consider the special case when $t = i$. In this special case, the number of redundant vectors in t tuples is $a_t + \sum_{j=1}^{t-1} a_j \binom{k-2j+1}{t-j}$, which is equal to $\binom{k}{t-1}$ by (3). Hence, the total number of redundant vectors in step t is $\binom{k}{t-1}$ and if we sum them, we will have to delete $\sum_{j=1}^{t-1} \binom{k}{j-1}$ vectors. Since the number of rows is $\sum_{j=0}^{t} \binom{k}{j}$, we see that the dimension at level t is at most $\binom{k}{t}$. On the other hand, since at level t, we have an identity matrix $I_{\binom{k}{i}}$, the dimension has to be $\binom{k}{i}$. This shows that the rest of the rows are linearly independent after our deletion and this completes the proof. □

2.5.1 Dimension of $C^{(t)}$when the blocks are consecutive starting from 1

We find a pattern as follows:

Theorem 2.5.2. *Assume the blocks for C are $\{A_i | 1 \leqslant i \leqslant n\}$, which means consecutive i's starting from 1. If the dimension of $C^{(t)}$ is $dim(C^{(t)})$, then*

$$\dim(C^{(t)}) = \begin{cases} \sum_{i=0}^{t} \binom{k}{i} & t < n \\ \sum_{i=0}^{n} \binom{k}{i} - 1 & t \geqslant n \end{cases}$$

Proof. For a fixed k, we start with block A_1. It is obvious that when $t \geqslant 1$, the dimension is k.

Next, we consider blocks A_1 and A_2. When $t = 1$, since $\mathbf{1}$ can be put into the basis due to the existence of A_2, the dimension is $k + 1 = \sum_{i=0}^{1} \binom{k}{i}$. When $t = 2$, A_1 produces a zero matrix of size $\binom{k}{2} \times k$ below the matrix A_1. There will be an identity matrix of size $\binom{k}{2} \times \binom{k}{2}$ next to the

zero matrix produced by A_1. However, if we add all the rows up, we can get 0, because the weight in the A_1 part will be $1+1=2$ and the weight in the A_2 part is equal to $\binom{2}{0}+\binom{2}{1}+\binom{2}{2}=2^2$, which is even. This shows that $\mathbf{1}$ should be taken out from the basis. Hence, the dimension is $\binom{k}{1}+\binom{k}{2}=\sum_{i=0}^{2}\binom{k}{i}-1$. For $t \geqslant 3$, it will be the same as the situation with $t=2$.

Now, we use the same idea in the above argument, for a general $A_1, A_2, ...A_n$, when $t < n$. We see that the generating matrix G looks like:

$$\begin{pmatrix}
1 & 1 & 1 & . & . & & & & & & & 1 & 1 \\
 & I_k & & & & & & & & & & & \\
 & & & I_{\binom{k}{2}} & & & & & & & & & \\
 & & & & & I_{\binom{k}{3}} & & & & & & & \\
 & & & & & & . & & & & & & \\
 & & & & & & & . & & & & & \\
 & & & & & & & & . & & & & \\
 & & & & & & & & & . & & & \\
 & & & & & & & & & & & & \\
 & & & & & & & & & & I_{\binom{k}{t}} & &
\end{pmatrix}$$

where I_m is a m by m identity matrix.

Since $n > t$, if we add all the rows up to t th order, then the weight for the A_{t+1} part will be $\sum_{i=0}^{t}\binom{t+1}{i}=2^{t+1}-1$, which is odd. So $\mathbf{1}$ is not redundant and the dimension will be equal to $\sum_{i=0}^{t}\binom{k}{i}$.

For $t \geqslant n$, we can stop when t hits n. Since when we add all rows up for the t th order, we would get the weight for block A_i is $\sum_{j=0}^{i}\binom{i}{j}=2^i$, which is even. Hence, $\mathbf{1}$ is redundant. By the above

matrix, we can see that the remaining rows are linearly independent. The dimension is therefore equal to $\sum_{i=0}^{n} \binom{k}{i} - 1$. □

We can add a matrix A_0 which is a $k \times 1$ matrix with all entries equal to 0. If $\mathbf{1}$ is added on the top, because of A_0, $\mathbf{1}$ is always not redundant. Hence, for $t \geqslant n$, the dimension formula can be changed to $\sum_{i=0}^{n} \binom{k}{i}$ for consecutive i starting from 0.

Remark 2.5.1. For the dimension part, we know that a single block gives us dimension equal to $\binom{k}{t}$. For a Reed-Muller code $RM(t,k)$, the dimension is equal to $\sum_{i=0}^{t} \binom{k}{t}$, which is greater than the dimension for any single block. Indeed, if we want to have this dimension $\sum_{i=0}^{t} \binom{k}{i}$, we can just use $A_0, A_1, A_2, ...A_t$ instead of the whole $RM(t,k)$.

2.5.2 Dimension of $C^{(t)}$ when there are two blocks

Now we only consider the case in which there are two blocks in the generating matrix. Let us denote i_1 and i_2 for the two different i's. From the program we ran, it appears that the longer block is dominating the dimension. Indeed, the dimension of the combination minus the dimension of the longer block is a shift of the dimension of the short block except for some cases.

Let us assume $\binom{k}{i_1} \geqslant \binom{k}{i_2}$. Hence, i_1 is the longer block. Let k_{12} be the dimension of the combination, k_1 be the dimension of block i_1, and k_2 be the dimension of block i_2. Assume 2^m divides $i_1 - i_2$ but 2^{m+1} does not. There are several cases below.

Conjecture 1. i) $m = 0$, which means that one of i_1, i_2 is even and the other is odd. In this case, $k_{12}^{(t)} - k_1^{(t)} = k_2^{(t-1)}$, for $t \geqslant 1$. We denote $k_2^{(0)} = 1$.

ii) $m = 1$, in this case 2 divides $i_1 - i_2$ and 4 does not, The dimension $k_{12}^{(t)} - k_1^{(t)} = k_2^{(t-2)}$, for $t \geqslant 1$.We denote $k_2^{(0)} = 1$ and $k_2^{(-1)} = 0$.

When $m \geqslant 2$, the behavior of k_{12} is irregular. One of the cases is that $k = 11$ and $i_1 = 4$, $i_2 = 8$. It is a $[495, 10, 240]$ code. The dimensions of $C^{(t)}$ for $t = 1, 2, 3..., 11$ are

$$[11, 55, 165, 331, 441, 485, 495, 495, 495, 495, 495].$$

The difference $k_{12} - k_1$ is $[0, 0, 0, 1, 111, 155, 165, 165, 165, 165, 165]$, while the dimensions of block $i_2 = 8$ are $[11, 55, 165, 165, 165, 165, 165, 165, 165, 165, 165]$.

2.5.3 Dimension of $C^{(t)}$ when blocks are congruent to some number modulo 4

Conjecture 2. Let $k^{(t)}$ be the dimension of the linear code $C^{(t)}$ which has blocks $i, i+4, i+8, ...$ for some $i = 0, 1, 2, 3$. Let S be the set of indices of the above block numbers. Let l' be the number in S such that $\binom{k}{l}$ is the largest. Let $l = \min\{l', k - l'\}$. Then $k^{(t)} = \sum_{n \geqslant 0} \binom{k}{t-4n}$ for $t \leqslant l$. For $l < t \leqslant l + m + 1$, $k^{(t)} = k^{(t-1)} + k_{(t)}$, where $k_{(t)} = k^{(m-(t-l-1))} - k^{(m-(t-l-1)-1)}$. $m = \min\{m', k - m'\}$ and m' is defined as the number in S such that $\binom{k}{m'}$ is the second largest. For $l + m + 1 < t \leqslant k$, $k^{(t)}$ is equal to the length of the code. Here, we denote $k^{(0)} = 1, k^{(-1)} = 0$ when $i \equiv 1, 2, 3 \pmod 4$,$\mathrm{k}^{(0)} = k^{(-1)} = 0$ when $i \equiv 0 \pmod 4$.

Remark 2.5.2. Finally, when t is large enough, the dimension will be the full rank, which is the length of the code, and we can predict the smallest t which gives us the full rank. Indeed, it is equal to $l + m + 1 = k - 3$.

The data for $k = 11, 12, 13, 14$ are listed in the following tables, the j th position in the vector is the dimension when $t = j$.

$k = 11$	Dimension of the linear codes $C^{(t)}$
$i = 1, 5, 9$	[11, 55, 165, 331, 473, 517, 527, 528, 528, 528, 528]
$i = 2, 6, 10$	[11, 55, 165, 331, 473, 517, 527, 528, 528, 528, 528]
$i = 3, 7, 11$	[11, 55, 165, 331, 441, 485, 495, 496, 496, 496, 496]
$i = 4, 8$	[11, 55, 165, 331, 441, 485, 495, 495, 495, 495, 495]

$k = 12$	Dimension of the linear codes $C^{(t)}$
$i = 1, 5, 9$	[12, 66, 220, 496, 804, 958, 1012, 1023, 1024, 1024, 1024, 1024]
$i = 2, 6, 10$	[12, 66, 220, 496, 804, 990, 1044, 1055, 1056, 1056, 1056, 1056]
$i = 3, 7, 11$	[12, 66, 220, 496, 804, 958, 1012, 1023, 1024, 1024, 1024, 1024]
$i = 4, 8, 12$	[12, 66, 220, 496, 772, 926, 980, 991, 991, 991, 991, 991]

$k = 13$	Dimension of the linear codes $C^{(t)}$
$i = 1, 5, 9, 13$	[13, 78, 286, 716, 1300, 1730, 1938, 2003, 2015, 2016, 2016, 2016, 2016]
$i = 2, 6, 10$	[13, 78, 286, 716, 1300, 1794, 2002, 2067, 2079, 2080, 2080, 2080, 2080]
$i = 3, 7, 11$	[13, 78, 286, 716, 1300, 1794, 2002, 2067, 2079, 2080, 2080, 2080, 2080]
$i = 4, 8,12$	[13, 78, 286, 716, 1300, 1730, 1938, 2003, 2015, 2015, 2015, 2015, 2015]

$k = 14$	Dimension of the linear codes $C^{(t)}$
$i = 1, 5, 9, 13$	[14, 91, 364, 1002, 2016, 3030, 3668, 3941, 4018, 4031, 4032, 4032, 4032, 4032]
$i = 2, 6, 10, 14$	[14, 91, 364, 1002, 2016, 3094, 3732, 4005, 4082, 4095, 4096, 4096, 4096, 4096]
$i = 3, 7, 11$	[14, 91, 364, 1002, 2016, 3094, 3796, 4069, 4146, 4159, 4160, 4160, 4160, 4160]
$i = 4, 8,12$	[14, 91, 364, 1002, 2016, 3094, 3732, 4005, 4082, 4095, 4095, 4095, 4095, 4095]

2.5.4 An Algebraic Geometry approach to the dimension problem

Let k be the integer mentioned above. For general combinations $I = \{i_1, i_2, ...i_m\}$, up to the level t, what is the dimension of $C^{(t)}$? We may think of using an algebraic geometry approach to work it out. Let $R = \mathbf{F}_2[x_1, x_2, ..., x_k]$, let I be the ideal vanishing on every column of the generating matrix, then I is of the form $J =< x_1^2 - x_1, x_2^2 - x_2, \ldots, x_k^2 - x_k, f(x_1, x_2, ..., x_k) >$. Since we work over $\mathbf{F}_2$, we see that $x_i^2 - x_i = 0$ always. Now the problem is to decide what $f(x_1, x_2, ..., x_k)$ is. First, we see that if there is no $f(x_1, x_2, ..., x_k)$, then the linear code becomes a Reed-Muller code, which means we choose all the blocks. In another way, the choice of $f(x_1, x_2, ..., x_k)$ is somehow to reflect the choice of $I = \{i_1, i_2, ..., i_m\}$. The usage of this ideal J is the following. We take the Hilbert series of it, the coefficient of each term is just the increase of the dimension when t is increased. For

example, when $k = 7$ and $i = 1, 2, 5, 6$, the ideal is $J =< x_1^2 - x_1, x_2^2 - x_2, \ldots, x_7^2 - x_7, 1 + s_1 + s_2 >$, where s_1 is the first order symmetric function, $s_1 = x_1 + x_2 + ... + x_7$ and s_2 is the second order symmetric function, $s_2 = x_1x_2 + x_1x_3 + ... + x_6x_7$. After running the program in Macaulay2, we find that the Hilbert Series is $1 + 7T^2 + 20T^2 + 20T^3 + 7T^4 + 1$, which means that $C^{(0)}$ has dimension 1, C has dimension $1 + 7 = 8$, $C^{(2)}$ has dimension 28, $C^{(3)}$ has dimension 48, $C^{(4)}$ has dimension 55, $C^{(5)}$ has dimension 56, which is the full rank. The way to choose the function $f(x_1, x_2, ..., x_k)$ is the following, $f(x_1, x_2, ..., x_k)$ must vanish on the chosen blocks and not on other blocks. In the above example,

i	1	2	5	6	i	3	4	7
s_1	$\binom{1}{1} = 0$	$\binom{2}{1} = 0$	$\binom{5}{1} = 1$	$\binom{6}{1} = 0$	s_1	$\binom{3}{1} = 1$	$\binom{4}{1} = 0$	$\binom{7}{1} = 1$
s_2	$\binom{1}{2} = 0$	$\binom{2}{2} = 1$	$\binom{5}{2} = 0$	$\binom{6}{2} = 1$	s_2	$\binom{3}{2} = 1$	$\binom{4}{2} = 0$	$\binom{7}{2} = 1$

From the table, we see that $1+s_1+s_2$ vanishes on $1, 2, 5, 6$ and not on $3, 4, 7$. Hence, $f(x_1, x_2, ..., x_k) = 1 + s_1 + s_2$ is the polynomial we want.

Assume we have blocks $i_1, i_2, ..., i_m$. Let v be the characteristic function for those blocks, which means that the output of v is 1 on those blocks and 0 on the other blocks.

We can first find out the polynomials for which only one block i is 1 and the rest blocks are 0. Let v_i be the corresponding polynomial, we have the following:

Lemma 2.5.2. *For* $0 \leqslant i \leqslant k$, $v_i = \sum_{j=1}^{k} \binom{j}{i} s_j$

Proof. This is true because of the fact we mentioned above, $M_k^2 = I_k$. Let c_j be the $j+1$ th column in M_k, so $M_k c_j$ is almost a zero vector except at the j th position, which exactly means that block j is 1 and the rest blocks are 0. □

From the above lemma, we know that for $I = \{i_1, i_2, ..., i_m\}$, if we want the elements in I to have 1 and the rest have 0, we just choose the characteristic vector to be $v_{i_1} + v_{i_2} + ... + v_{i_m}$. Then the polynomial we need is $f(x_1, x_2, ..., x_k) = 1 + v_{i_1} + v_{i_2} + ... + v_{i_m}$.

The characteristic vector v is written in terms of symmetric polynomials $s_1, s_2, ...s_k$ and $\mathbf{1}$. In the Reed-Muller code $RM(1, k)$, if v is computed, then $v_n = 1$ if and only if the position n is in

the blocks $i_1, i_2, ..., i_m$. Hence, v is the characteristic vector for blocks $i_1, i_2, ...i_m$ punctured from Reed-Muller code $RM(1,k)$. The biggest r appeared as s_r in the expression of v will decide that v is in $RM(r,k)$. Hence, the corresponding linear codes formed by blocks $i_1, i_2, ...i_m$ has multiplicity $k-r-1$.

Example 2.5.1. For $k=7$ and $i=1,2,5,6$, the characteristic function $v = v_1+v_2+v_5+v_6 = s_1+s_2$. As 2 is the largest number appearing in v, we conclude that the linear code formed by blocks $1,2,5,6$ has multiplicity $7-2-1=4$.

2.5.5 Symmetric dimensions

The motivation of symmetric dimensions is due to the understanding that the dimension of Reed-Muller codes are symmetric.

For example, when $k=7$, $RM(r,7)$ has dimension 1,8,29,64,99,120,127,128, the increase of dimensions are $1,7,21,35,35,21,7,1$, which are symmetric. The symmetric properties are corresponding to the properties such that $RM(r,m)$ is dual to $RM(m-r-1,m)$. If the dimension is symmetric, then the summation of the dimension of the r th order linear code $C^{(r)}$ with the dimension of the l th order linear code $C^{(l)}$is equal to the total length. This is a necessary condition for which $C^{(r)}$ is dual to $C^{(l)}$. Furthermore, if the multiplicity $t \geqslant l+r$, then $C^{(r)}$ is dual of $C^{(l)}$. For certain blocks combinations, the dimension of the linear code $C^{(t)}$ will increase in a symmetric way. For example, when $k=7$ and $i=1,2,5,6$, the dimension for $C^{(0)}, C^{(1)}, ..., C^{(5)}$ are $1,8,28,48,55,56$, the increasement is symmetric, which is $1,7,20,20,7,1$.

In the following, we used the Macaulay2 program to test those combinations of linear codes such that the increase of dimension is symmetric as t is increasing. If the largest number in the second bracket is m, then the characteristic vector is in $RM(m,k)$, so the linear code has multiplicity $k-m-1$. For example, the Reed-Muller code has $m=0$, so it has multiplicity $k-1$. If we take only the odd blocks, then we would see that $m=1$, so the linear code has multiplicity $k-2$.

Blocks	Multiplicity	Dimension	Length
0246	5	1,6,15,20,15,6,1	64
1357	5	1,6,15,20,15,6,1	64
1256	4	1,7,20,20,7,1	56
012456	4	1,7,21,34,21,7,1	92
15	3	1,6,14,6,1	28
26	3	1,6,14,6,1	28
123567	3	1,7,21,34,21,7,1	92
0167	2	1,7,7,1	16
17	1	1,6,1	8
06	1	1,6,1	8
012567	0	1,7,21,21,7,1	58
07	0	1,1	2
7	0	1	1
0124567	0	1,7,21,35,21,7,1	93
0157	0	1,6,15,6,1	30
01567	0	1,7,21,7,1	37
067	0	1,7,1	9
017	0	1,7,1	9
01267	0	1,7,21,7,1	37
0123567	0	1,7,21,35,21,7,1	93
026	0	1,6,15,6,1	29
0	0	1	1

2.6 Minimum distance of $C_i^{(t)}$ and $Dual(C_i^{(t)})$

2.6.1 Backgrounds from association schemes

The theory of association schemes are useful to describe the weight of the code. The main reference for the association schemes are from the book [1], and the definition based on the book is the following:

Definition 2.6.1. An association scheme consists of a set X with a partition of the set of 2 element subsets of X into d non-empty classes $\Gamma_1, \Gamma_2, ..., \Gamma_d$ which satisfy

(a) Given $x \in X$, the number $n_i(x)$ of points $y \in X$ with $\{x, y\} \in \Gamma_i$ depends only on i, not on x.

(b) Given $x, y \in X$ with $\{x, y\} \in \Gamma_k$, the number $p_{ij}^k(x, y)$ of points $z \in X$ with $\{x, z\} \in \Gamma_i$ and $\{z, y\} \in \Gamma_j$ depends only on i, j, k, not on x and y. Points x and y are called i^{th} associates if $\{x, y\} \in \Gamma_i$.

Example 2.6.1. Let $H(n, q)$ be the set of all ordered n tuples of elements from a set A of q elements. Two n tuples are i associates if they differ in i coordinates. Indeed, $H(n, q)$ is an association scheme, which is called a Hamming scheme.

In our case, $q = 2$, so $H(n, 2)$ is a binary Hamming scheme. Given length n and an n tuple x, the number of $y \in \mathbf{F}_2^n$ which gives x, y are i^{th} associates is $n_i(x) = \binom{n}{i}$. Given $x, y \in \mathbf{F}_2^n$ and x, y are k associates, the number of $z \in \mathbf{F}_2^n$ which gives x, z are i associates and y, z are j associates is $p_{ij}^k = \binom{n-k}{\frac{i+j-k}{2}}\binom{k}{\frac{i-j+k}{2}}$ when $i + j - k$ is divisible by 2. If $i + j - k$ is odd, then $p_{ij}^k = 0$.

2.6.2 The minimum distance for one single block

In some cases it was not possible to describe the parameters for the general case. But we have observations that are true for all cases that we considered. With further effort, we believe those patterns can be proved. We treat those cases as conjectures rather than as theorems.

Here are the results of the minimum distance for one single block A_i:

Conjecture 3. Let $x_1, x_2, ..., x_k$ be the rows in the matrix A_i. Assume $t \leqslant i$ and $i \leqslant k - t$.

(a) For $k > 2i$, the distance of $C^{(t)}$ is $\binom{k-t}{i-t}$, and the minimum weight is achieved by $w = x_1x_2...x_t$.

(b) For $k = 2i$, the distance of $C^{(t)}$ is $2\binom{k-t-1}{i}$, and the minimum weight is achieved by $w = (1 + x_1 + x_2)(1 + x_1 + x_3)...(1 + x_1 + x_{t+1})$

(c) For $k < 2i$, the distance of $C^{(t)}$ is $\binom{k-t}{i}$, and the code w which has the minimum weight is achieved by $w = (1 + x_1)(1 + x_2)...(1 + x_t)$

For $t > i$ or $i > k - t$, $D(C^{(t)}) = 1$ because $C^{(t)} = \mathbf{F}_2^{\binom{k}{i}}$.

When $t = 1$, the minimum distance is proved in [4].

Conjecture 4. The dual code $Dual(C^{(t)})$ has minimum distance $D(Dual(C^{(t)})) = 2^{t+1}$

When $t = 1$, $D(Dual(C)) = 4$ because in the generating matrix of C, any 3 columns are linearly independent and there are 4 columns which are dependent. Hence, the dual distance is equal to 4. We introduce the following lemma first:

Lemma 2.6.1. *Let w_u be defined as the weight of a codeword obtained by adding u rows from A_i. Then w_u is a fixed number if u is fixed (i.e any sum of u rows will give us the same weight) and*

$$w_u = \sum_j \binom{u}{2j-1}\binom{k-u}{i-2j+1}$$

We have that $\binom{a}{b} = 0$ if $b < 0$ or $b > a$.

Proof. Let us add u rows of $A_i^{(1)}$ and get a vector, say $\mathbf{v} = \mathbf{r}_1 + \mathbf{r}_2 + ... + \mathbf{r}_u = (v_1, v_2, ..., v_{\binom{k}{i}})$. If $v_m = 1$, then $\{r_{1m}, r_{2m}, ..., r_{um}\}$ has odd number of 1s.

Let $\{r_{1m}, r_{2m}, ..., r_{um}\}$ contains only one 1 and the rest are 0. There are $\binom{u}{1}\binom{k-1}{i-1}$ possible choices. So by this combination, $\binom{u}{1}\binom{k-1}{i-1}$ 1s will appear in the vector $\mathbf{v}$.
Similarly, if $\{r_{1m}, r_{2m}, ..., r_{um}\}$ contains three 1s and the rest are 0. There are $\binom{u}{3}\binom{k-1}{i-3}$ such choices. Therefore, we can follow this argument so that the weight of any sum of u rows of $A_i^{(1)}$ is

$$w_u = \sum_j \binom{u}{2j-1}\binom{k-u}{i-2j+1}$$

.

However, there are 4 boundary conditions: $2j-1 \geqslant 0$, $i-2j+1 \geqslant 0$, $i-2j+1 \leqslant k-u$ and $2j-1 \leqslant u$. So the range of j is $max(1, \frac{u+1+i-k}{2}) \leqslant j \leqslant min(\frac{i+1}{2}, \frac{u+1}{2})$.

Since adding all rows in $A_i^{(1)}$ will give $\mathbf{0}$ or $\mathbf{1}$, thus, we have the following result: □

Theorem 2.6.1. *The minimum distance of C_1 is $min\{w_u | 1 \leqslant u \leqslant k-1\}$, where w_u is the weight of the codeword obtained by adding u rows.*

Definition 2.6.2. The Krawtchouk polynomial is defined as $K_i(x) = \sum_{j=0}^{i} (-1)^j (q-1)^{i-j} \binom{x}{j} \binom{n-x}{i-j}$, where q is a prime power and n is some positive integer.

Remark 2.6.1. The formula can also be written using Krawtchouk polynomial. For the binary fields, since w_u is taken over the odd binomial entries and the Krawtchouk polynomial is alternating, we can write

$$w_u = \frac{1}{2}\left[\sum_{j=0}^{i}\binom{u}{j}\binom{k-u}{i-j} - \sum_{j=1}^{i}(-1)^j\binom{u}{j}\binom{k-u}{i-j}\right]$$

where $\sum_{j=0}^{i}\binom{u}{j}\binom{k-u}{i-j} = \binom{k}{i}$ is an easy combinatorial argument: If among k people, there are u males and $k-u$ females. To select i people, we select j males and $i-j$ females for $j = 0, 1, \ldots, i$.

From above, we see that

$$w_u = \frac{1}{2}\left[\binom{k}{i} - \sum_{j=1}^{i}(-1)^j\binom{u}{j}\binom{k-u}{i-j}\right] = \frac{1}{2}\left[\binom{k}{i} - K_i(u)\right]$$

2.6.3 Minimum Distance for multiple blocks

For general combinations of blocks, instead of predicting a precise result, we may only approximate the minimum distance. The way to do this is the following. For a certain t, we take the multiplication of t vectors where each vector is a sum of u rows and the rows are all distinct. Hence, we require that $ut \leqslant k$. Assuming the blocks are $i_1, i_2, ...i_m$, the total weight of the vector after multiplying t

vectors is equal to

$$\sum_{1\leqslant j\leqslant m}\sum_{p_1,\ldots,p_t}\binom{u}{p_1}\binom{u}{p_2}\ldots\binom{u}{p_t}\binom{k-tu}{i_j-p_1-p_2-\ldots-p_t},$$

where $p_1, p_2, ..., p_t$ are taken over all odd numbers. The above sum is determined by u only. Hence we can possibly get an upper bound for the minimum distance for multiple blocks for general t.

$$d\leqslant \min_{1\leqslant u\leqslant \frac{k}{t}}\{\sum_{1\leqslant j\leqslant m}\sum_{p_1,\ldots,p_t}\binom{u}{p_1}\binom{u}{p_2}\ldots\binom{u}{p_t}\binom{k-tu}{i_j-p_1-p_2-\ldots-p_t}\}.$$

For example, $k = 10$, $i = 3, 6$. When $t = 2$, we have the weight in the following:

$u = 1$, $w_1 = \binom{1}{1}\binom{1}{1}\binom{8}{1} + \binom{1}{1}\binom{1}{1}\binom{8}{4} = 78$

$u = 2$, $w_2 = \binom{2}{1}\binom{2}{1}\binom{6}{1} + \binom{2}{1}\binom{2}{1}\binom{6}{4} = 84$

$u = 3$, $w_3 = \binom{3}{1}\binom{3}{1}\binom{4}{1} + \binom{3}{3}\binom{3}{1}\binom{4}{2} + \binom{3}{1}\binom{3}{3}\binom{4}{2} + \binom{3}{3}\binom{3}{3}\binom{4}{0} = 82$

$u = 4$, $w_4 = \binom{4}{1}\binom{4}{1}\binom{2}{1} + \binom{4}{3}\binom{4}{1}\binom{2}{2} + \binom{4}{1}\binom{4}{3}\binom{2}{2} + \binom{4}{3}\binom{4}{3}\binom{2}{0} = 80$

$u = 5$, $w_5 = \binom{5}{5}\binom{5}{1}\binom{0}{0} + \binom{5}{1}\binom{5}{5}\binom{0}{0} + \binom{5}{3}\binom{5}{3}\binom{0}{0} = 110$

A more general formula is the following.

Theorem 2.6.2. *Assume we take the multiplication of t vectors where each vector is a sum of rows and the rows are all different, say $c = \mathbf{v}_1\cdot\mathbf{v}_2...\mathbf{v}_t$, where $\mathbf{v}_j$ is a sum of r_j different rows. The weight of c is $wt(c) = \sum_{1\leqslant j\leqslant m}\sum_{p_1,\ldots,p_t}\binom{v_1}{p_1}\binom{v_2}{p_2}\cdots\binom{v_t}{p_t}\binom{k-v_1-v_2-\ldots v_t}{i-p_1-p_2-\ldots-p_t}$,where $p_1, p_2, \ldots, p_t$ are taken all over odd numbers.*

So an upper bound for the minimum distance can be determined by taking the minimum value of $wt(c)$, with the minimum taken over all $v_1 + v_2 + ... + v_t \leqslant k$.

For the same example, $k = 10$, $t = 2$, $i = 3, 6$. If we take linear forms v_1 and v_2 with different

parameters r_1 and r_2. The restriction is $r_1 + r_2 \leqslant 10$.

	v_1	1	2	3	4	5	6	7	8	9
v_2										
1		78	84	78	80	85	84	70	64	126
2		84	84	80	88	92	76	56	112	
3		78	80	82	88	78	56	98		
4		80	88	88	80	64	104			
5		85	92	78	64	110				
6		84	76	56	104					
7		70	56	98						
8		64	112							
9		126								

As shown above, an upper bound for the minimum distance d for $C^{(2)}$ where $k = 10$, $i = 3, 6$ is that $d \leqslant 56$.

For multiple blocks, we have two results:

Theorem 2.6.3. *Let k be a positive integer.*

(a) If the blocks $i_1, i_2, ..., i_m$ chosen are all less than $\frac{k}{2}$, then the minimum distance of $C^{(t)}$ is equal to

$$\sum_{1 \leqslant j \leqslant m} \binom{k-t}{i_j - t},$$

and the codeword which has the minimum nonzero weight is achieved by multiplying t different rows.

(b) Assume there are only two blocks i_1 and i_2, one of them is even and the other is odd. For simplicity, let $\binom{k}{i_1} \geqslant \binom{k}{i_2}$. Then the minimum distance of $C^{(t)}$ is equal to $\binom{k-t+1}{i_2-t+1}$ for $i_2 < \frac{k}{2}$ and equal to $\binom{k-t+1}{i_2}$ for $i_2 > \frac{k}{2}$. In other words, the minimum distance for i_1, i_2 under multiplicity t is equal to the minimum distance for the shorter block under multiplicity $t-1$.

Proof. (a) We assume the previous Conjecture 3. Since for one block, the minimum distance is achieved by $x_1x_2...x_t$ or $(1+x_1)(1+x_2)...(1+x_t)$, we can therefore conclude that if we combine them together, by using the same method, the minimum distance is achieved.

(b) Since i_1 and i_2 have different parities, we can construct a vector which has 0 on the support of i_2 and 1 on the support of i_1. Let $C_2^{(t)}$ be the linear code corresponding to block i_2. Then we just need to multiply this vector with the vector which has the minimum nonzero weight in $C_2^{(t-1)}$ and the minimum distance for blocks i_1 and i_2 is achieved. □

Remark 2.6.2. Using the above method, we can always conclude that the minimum distance of $C^{(t)}$ can never exceed $\frac{n}{2^t}$. This is because we can always split the blocks into odd blocks and even blocks, the minimum distance is therefore bounded by the $t-1$ distance of the short sides. When $t=1$, the distance is bounded by $\frac{n}{2}$. Then, by applying the iteration, we conclude that the minimum distance of $C^{(t)}$ is always less than or equal to $\frac{n}{2^t}$.

2.6.4 Regular partitions

The background for such interesting partitions is based on commutative algebra and Hilbert functions. The book [14] contains many useful illustrations about background materials.

Given a certain k, we can make some partition $\mathcal{P}$ of set $\{1,2,...,k\}$ to the blocks i. For simplicity, we do not include $\mathbf{1}$ here. Let $\mathcal{P}=\{C_1,C_2,...C_m\}$, let w_{ui} be the weight of summing u distinct vectors for blocks C_i $1 \leqslant i \leqslant m$. If for some u, all the w_{ui} are the same, then we put those u's together to a set, by applying this procedure, we can get another partition $\mathcal{P}'$, which may be different from the original partition $\mathcal{P}$. We have the following:

$|\mathcal{P}| \leqslant |\mathcal{P}'|$. If the equality holds, then if we start with $\mathcal{P}'$, we would get the corresponding partition be $\mathcal{P}$, which means that $\mathcal{P}''=\mathcal{P}$. Of course, if we have bad $\mathcal{P}$ such that $\mathcal{P}'$ is of size k, then this case is trivial and not interesting at all. For a non-trivial example, if $k=7$, let

$\mathcal{P} = \{\{1,2,5,6\},\{3,4\},\{7\}\}$. We have the following table:

	1,2,5,6	3,4	7
w_1	28	35	1
w_2	24	40	0
w_3	28	35	1
w_4	32	32	0
w_5	28	35	1
w_6	24	40	0
w_7	28	35	1

According to this table, the partition $\mathcal{P}'$ should be $\{\{1,3,5,7\},\{2,6\},\{4\}\}$.

Now, if we do the same procedure with respect to $\mathcal{P}'$, we would get

	$1,3,5,7$	$2,6$	4
w_1	32	12	20
w_2	32	12	20
w_3	32	16	16
w_4	32	16	16
w_5	32	12	20
w_6	32	12	20
w_7	64	0	0

we can see that $\mathcal{P}''$ is again $\mathcal{P}$. The two partitions are in duality.

In the following, we introduce some of the conjugate partitions for k. We list those partitions in the graph, if B is a subpartition of A, which means that B is gotten from A by breaking some

partitions of A, then we connect A to B and B is sitting below A. The following are the partition graphs for $5 \leqslant k \leqslant 12$. There are four patterns for the graphs, according to the remainder of k divided by 4. The graphs for $13 \leqslant k \leqslant 16$ are not listed because the graph for $k = m$ and $k = m+4$ are the same for $m = 9, 10, 11, 12$.

More examples are shown in the appendix. In the graphs which are shown in the appendix, in each node, the above is the partition, and the number below the partition is the distribution of the biggest degree of characteristic vectors.

Each graph is symmetric, in the way that symmetric points must be dual to each other.

Let us take $k = 9$, we want to find out the dual partitions from the graph. First of all, we list a table of weights for $k = 9$. Let w_{ui} be the weight of the vector which is obtained by adding u different vectors in block i.

The weight of w_u is given in this table:

	i	1	2	3	4	5	6	7	8	9
u										
1		1	8	28	56	70	56	28	8	1
2		2	14	42	70	70	42	14	2	0
3		3	18	46	66	60	38	18	6	1
4		4	20	44	60	60	44	20	4	0
5		5	20	40	60	66	44	16	4	1
6		6	18	38	66	66	38	18	6	0
7		7	14	42	70	56	42	22	2	1
8		8	8	56	56	56	56	8	8	0
9		9	0	84	0	126	0	36	0	1

From the above data, after evaluating the sum, we can conclude that the following partitions are dual to each other:

(12345678)(9) and (13579)(2468)

(1256)(3478)(9) and (2468)(159)(37)

(1278)(3456)(9) and (13579)(46)(28)

(1458)(2367)(9) and (13589)(26)(48)

(1357)(2468)(9) and (1357)(2468)(9) (self dual)

(12)(34)(56)(78)(9) and (37)(46)(28)(19)(5)

(45)(36)(18)(27)(9) and (13579)(2)(4)(6)(8)

(35)(17)(28)(46)(9) and (35)(17)(28)(46)(9) (self dual)

(15)(26)(37)(48)(9) and (15)(26)(37)(48)(9) (self dual)

The dual partitions are symmetric in the graph. The remaining partitions are self-dual.

Moreover, in some level, fix a partition of k. Let the number of parts be l, so that we can construct $2^l - 1$ different linear codes by counting classes in the partition. We set up a program to evaluate the multiplicity of these $2^l - 1$ linear codes. We use a vector $\mathbf{v}_l$ of length l to represent the information. In v_l, the first coordinate is 0. If the n th position in $\mathbf{v}_l$ is a_n, then there are 2^{n-1} codes which have characteristic vector with highest degree a_n, which means the multiplicity is $\max(k - a_n - 1, 0)$. (Sometimes, when $a_n = k$, the multiplicity is also 0).

Example 2.6.2. For $k = 9$, in the graph, there is a partition $(0)(45)(36)(18)(27)(9)$, and the vector $\mathbf{v}_6$ is $(0, 2, 4, 6, 8, 9)$. There are 63 possible linear codes formed by blocks $0, 45, 36, 18, 27, 9$. Among them, 32 have degree 9, which means multiplicity 0. 16 have degree 8, which also has multiplicity 0. 8 have multiplicity 2, 4 have multiplicity 4, 2 have multiplicity 6, 1 has multiplicity 8.

In detail,

n	a_n	multiplicity	combinations
6	9	0	$(0), (9), (459), (369), (189), (279), (34569), (14589), (24579)$
			$(13689), (23679), (12789), (1345689), (2345679)$
			$(1245789), (1236789), (123456789)$
			$(045), (036), (018), (027), (03456), (01458), (02457)$
			$(01368), (02367), (01278), (0134568), (0234567)$
			$(0124578), (0123678), (012345678)$
5	8	0	$(18), (1458), (1368), (1278), (134568)$
			$(124578), (123678), (12345678)$
			$(09), (0459), (0369), (0279), (034569)$
			$(024579), (023679), (02345679)$
4	6	2	$(0189), (27), (36), (01236789), (2457)$
			$(3456), (01245789), (01345689)$
3	4	4	$(45),\ (234567), (012789), (013689)$
2	2	6	$(2367), (014589)$
1	0	8	(0123456789)

2.7 Good linear codes

Good linear codes can be generated by combining suitable blocks A_i. For certain multiplicity t, from what we did above, we know that in order to make $\widehat{C}^{(t)}$ even, we need to choose characteristic vectors in the span of the first $k-t$ columns in the matrix M_k. Hence, there are 2^{k-t} linear codes with this property. One approach is that for small $k-t$, we list all the possible choices. After we have the $[n, k, d]$, In some cases, to guarantee that $C^{(t)}$ has even length, we add a parity check symbol. In those cases, we add all zero column of length k and then add $\mathbf{1}$ to the code.

Let $k = 7$, $t = 4$. We list all the possible combinations of blocks with multiplicity $t \geqslant 4$. In M_7, the blocks corresponding to the first columns are $1, 2, 3, 4, 5, 6, 7, 2, 4, 6$ and $1, 4, 5$. So the 8

combinations of blocks are listed below:

Combinations of blocks	$[n,k,d]$	Adjusted $[n,k,d]$	BKLC with same n,k
1234567	$[127,7,64]$	$[128,8,64]$	$[128,8,64]$
246	$[63,6,32]$	$[64,7,32]$	$[64,7,32]$
1357	$[64,7,32]$	$[64,7,32]$	$[64,7,32]$
145	$[63,7,28]$	$[64,8,28]$	$[64,8,29]$
2367	$[64,7,28]$	$[64,8,28]$	$[64,8,29]$
1256	$[56,7,24]$	$[56,8,24]$	$[56,8,24]$
347	$[71,7,32]$	$[72,8,32]$	$[72,8,32]$

The table shows that a Reed-Muller Code $RM(1,7)$ can be decomposed in three different ways: 2,4,6 and 1,3,5,7, 1,4,5 and 2,3,6,7, 1,2,5,6 and 3,4,7. The key is that we always put two congruent classes together and then the other two together. This must be true because the first three columns are periodic modulo 4 and any combinations of them is also periodic modulo 4.

Among all these linear codes, we pick up those which have shortest length. Indeed, many of them are good linear codes. Here are some of the examples,

Linear codes with multiplicity t which are shortest among possible combinations of blocks

	Columns	Blocks	$[n,k,d]$	Adjusted $[n,k,d]$	Best Known Linear codes with same n,k
$t=8$	1	123456789	$[511,9,256]$	$[512,10,256]$	$[512,10,256]$
$t=7$	2	2468	$[255,8,128]$	$[256,9,128]$	$[256,9,128]$
$t=6$	3	2367	$[240,9,112]$	$[240,10,112]$	$[240,10,114 \leqslant d \leqslant 116]$
$t=5$	34	37	$[120,9,56]$	$[120,9,56]$	$[120,9,56]$
$t=4$	145	12789	$[91,9,32]$	$[92,10,32]$	$[92,10,41 \leqslant d \leqslant 42]$
$t=3$	1256	79	$[46,9,16]$	$[46,9,16]$	$[46,9,19 \leqslant d \leqslant 20]$
$t=2$	1357	189	$[19,9,4]$	$[20,10,4]$	$[20,10,6]$
$t=1$	12345678	19	$[10,9,2]$	$[10,9,2]$	$[10,9,2]$

Chapter 3

AB methods and applications

3.1 Dimension-length profiles

The dimension-length profile of a linear code was introduced by Forney [8]. It is an important tool for describing the state trellis complexity of a linear code [16][10]. The dimension-length profile is determined by the higher weights of a linear code. Partial information about the higher weights gives bounds for the dimension-length profile. In the next section we will derive a lower bound for the minimum distance of a code C from the dimension-length profiles of codes A and B that satisfy $C \perp A * B$. An example of a bound of this type is the classical Roos bound. That bound uses only partial information about the higher weights of A and B (namely the minimum distance of A and the dual minimum distance of B). By casting the Roos bound in the setting of dimension-length profiles we get both a much more general theorem and a shorter and more transparent proof.

Let G be a generator matrix for the linear code C of length n and dimension k. Define $p_i(C)$ as the minimal rank of any i columns in the generator matrix of C. Thus $0 \leq p_i(C) \leq i$. For an MDS code, any k columns are linearly independent and

$$\text{(MDS)} \quad p_i(C) = \min\{i, k\}.$$

For an arbitrary code, the minimum distances of the code and its dual provide lower bounds for $p_i(C)$. For a code with dual distance $d(C^\perp)$, any $d(C^\perp) - 1$ columns are linearly independent and

$$p_i(C) \geq \min\{i, d(C^\perp) - 1\}, \quad \text{for all } i. \tag{3.1}$$

For a code with minimum distance $d(C)$, any $n-d(C)+1$ columns are of full rank k, and thus any $i = n - d(C) + 1 - j$ columns of rank at least $k - j$.

$$p_i(C) \geq \min\{k, k - (n - d(C) + 1 - i)\}, \quad \text{for all } i. \tag{3.2}$$

Note that the lower bounds reduce to the same bounds $p_i(C) \geq \min\{i, k\}$ and $p_i(C) \geq \min\{k, i\}$ for an MDS code. On the other hand, for a general code the lower bounds are independent. As an example, for the first order Reed-Muller code $n = 16, k = 5, d(C) = 8$, and $d^{\perp}(C) = 4$. The lower bounds become

$$p_i(C) \geq \min\{i, 3\} \quad \text{and} \quad p_i(C) \geq \min\{5, i - 4\}.$$

For $i = 0, 1, \ldots, 16$, $p_i(C)$ has lower bounds

$$(0, 1, 2, 3,\ 3, 3, 3, 3,\ 4, 5, 5, 5,\ 5, 5, 5, 5,\ 5).$$

The functions $p_i(C)$ and $p_i(C^{\perp})$ satisfy

$$k - p_i(C) + p_{n-i}(C^{\perp}) = n - i. \tag{3.3}$$

This follows by comparing the dimensions of two dual codes of length $n-i$. Shortening a code C in i positions of minimal rank gives a code of length $n - i$ and dimension $k - p_i(C)$. The dual of this code is the punctured version of the dual code, which has dimension $p_{n-i}(C^{\perp})$. Using the relation (3.3), we recover (3.2) applied to C from (3.1) applied to $C^{\perp}$.

$$\begin{aligned} & p_{n-i}(C^{\perp}) \geq \min\{n - i, d(C) - 1\} \\ \Leftrightarrow\ & p_i(C) \geq \min\{n - i, d(C) - 1\} - (n - i) - k \\ \Leftrightarrow\ & p_i(C) \geq \min\{k, k - (n - d(C) + 1 - i)\} \end{aligned}$$

Using the notation $d_1^\perp = d(C^\perp)$ and $g_1 = n + 1 - k - d(C)$, the bounds (3.1) and (3.2) become

$$p_i(C) \geq \min\{i, d_1^\perp - 1\} \quad \text{and} \quad p_i(C) \geq \min\{k, i - g_1\}. \tag{3.4}$$

3.2 The Roos bound

In [13], Roos derives the *Roos bound for cyclic codes* [id., Theorem 2] from a more general theorem [id., Theorem 1].

A code is nondegenerate if its generator matrix has no zero columns. In that case, for any position there exists a codeword that is nonzero in that position.

Theorem 3.2.1 (Roos bound for linear codes [13, Theorem 1])**.** *Let A, B and C be nondegenerate linear codes such that $C \perp (A * B)$. Let $g(A) = n + 1 - k(A) - d(A)$. If $g(A) < d(B^\perp) - 1$ then $d(C) \geq d(B^\perp) + k(A) - 1$.*

Proof. Let $c \in C$ be a nonzero word of weight w. The restrictions of A and B to the w coordinates of c are of total dimension $k(c * A) + k(c * B) \leq w$. With (3.5),

$$k(c * A) \geq \min\{k(A), w - g(A)\}.$$
$$k(c * B) \geq \min\{w, d(B^\perp) - 1\}.$$

We show that the minima are attained in $d(B^\perp) - 1$ and in $k(A)$. The minimum for $k(c * B)$ can not be attained in w, for then $k(c * A) = 0$ would violate that A is nondegenerate. Next, with $k(c * B) \geq d(B^\perp) - 1$, $k(c * A) \geq w - g(A)$ would violate the assumption $g(A) \leq d(B^\perp) - 1$. Thus $k(c * A) \geq k(A)$ and $w \geq k(A) + d(B^\perp) - 1$. □

The following theorem uses a condition $g(A) < k(B)$ that is weaker than the condition $g(A) < d(B^\perp) - 1$ of the Roos bound but adds to that a condition $g(B) < k(A)$. Thus the theorems apply in general under different conditions and supplement each other. While Theorem 3.2.1 gives a lower bound for the minimum distance, the next theorem excludes certain weights. It can be used to

improve lower bounds obtained with other methods, or it can be used multiple times so that the total set of excluded weights provides a lower bound for the minimum distance. For examples of both situations we refer to [7].

Theorem 3.2.2 (Symmetric Roos bound [7]). *Let A, B and C be nondegenerate linear codes such that $C \perp (A * B)$. If $g(A) < k(B)$ and $g(B) < k(A)$ then, for a nonzero word $c \in C$ of weight w, either $w \leq g(A) + g(B)$ or $w \geq k(A) + k(B)$.*

Proof. As in the previous proof we have that $k(c * A) + k(c * B) \leq w$. With (3.5),

$$k(c * A) \geq \min\{k(A), w - g(A)\}.$$
$$k(c * B) \geq \min\{k(B), w - g(B)\}.$$

The minima are not attained in $k(A)$ and $w - g(B)$ for then $k(A) + w - g(B) \leq w$ would violate the assumption $g(B) < k(A)$. By symmetry the minima are attained either in $k(A)$ and $k(B)$, resulting in $w \geq k(A) + k(B)$, or in $w - g(A)$ and $w - g(B)$, resulting in $w \leq g(A) + g(B)$. □

3.3 Higher weights

We return to the first order Reed-Muller code of length $n = 16$. The actual values for $p_i(C)$ are

$$(0, 1, 2, 3,\ 3, 4, 4, 4,\ 4, 5, 5, 5,\ 5, 5, 5, 5,\ 5).$$

The lower bounds obtained with $d(C) = 8$ and $d(C^\perp) = 4$ only give that the values are at least

$$(0, 1, 2, 3,\ 3, 3, 3, 3,\ 4, 5, 5, 5,\ 5, 5, 5, 5,\ 5).$$

The actual values $p_i(C) = 4$ at $i = 5, 6, 7$ are missed by the combined lower bounds

$$p_i(C) \geq \min\{i, 3\} \quad \text{and} \quad p_i(C) \geq \min\{5, i - 4\}.$$

To describe the actual values requires knowledge of the higher weights of the code. The higher weight $d_r(C)$ is defined as the smallest number of coordinates that support r independent codewords. The higher weights of the first order Reed-Muller code are $d_1 = 8, d_2 = 12, d_3, = 14, d_4 = 15, d_5 = 16$. The actual values show that the profile increases at $i = 1, 2, 3, 5, 9$. The increases correspond to the higher weights via $\{n + 1 - d_r : r = 1, 2, 3, 4, 5\} = \{1, 2, 3, 5, 9\}$. In this case, knowledge of the second weight $d_2 = 12$ suffices to explain the increase of $p_i(C)$ at $i = 5$ and thus the values $p_i(C) = 4$ at $i = 5, 6, 7$ that were missed by the other lower bounds.

We used the minimum distance and dual distance to obtain the lower bounds

$$p_i(C) \geq \min\{i, d_1^\perp - 1\} \quad \text{and} \quad p_i(C) \geq \min\{k, i - g_1\}. \tag{3.5}$$

Here $g_1(C) = n + 1 - k - d(C)$. We generalize these lower bounds, for $r = 1, 2, \ldots, k$ to

$$p_i(C) \geq \min\{i + 1 - r, d_r^\perp - r\} \quad \text{and} \quad p_i(C) \geq \min\{k + 1 - r, i - g_r\}. \tag{3.6}$$

Together the lower bounds for $r = 1, 2, \ldots, k$ describe the actual profile $p_i(C)$. The lower bounds implicitly define the parameters $d_r^\perp$ and g_r. The higher dual distance is the largest integer $d_r^\perp$ such that

$$p_i(C) \geq \min\{i + 1 - r, d_r^\perp - r\}, \quad \text{for all } i.$$

The higher genus is the smallest integer g_r such that

$$p_i(C) \geq \min\{k + 1 - r, i - g_r\}, \quad \text{for all } i.$$

The higher genus $g_r(C)$ relates to the higher weight $d_r(C)$ via $n + r = k + d_r + g_r$.

For the Reed-Muller code $d_1(C^\perp) = 4, d_2(C^\perp) = 6$ and $d_1(C) = 8, d_2(C) = 12$, so that $g_1 =$

$4, g_2 = 1$. This yields the four lower bounds

$$(r = 1) \quad p_i(C) \geq \min\{i, 3\} \quad \text{and} \quad p_i(C) \geq \min\{5, i-4\},$$
$$(r = 2) \quad p_i(C) \geq \min\{i-1, 4\} \quad \text{and} \quad p_i(C) \geq \min\{4, i-1\}.$$

In this case, the lower bounds under $(r = 2)$ coincide because of duality.

$$\begin{array}{lllllllllllll}
\min\{i,3\} & \geq (& 0, & 1, & 2, & 3, & 3, & 3, & 3, & 3, & 3, & 3, & \cdots & 3). \\
\min\{5,i-4\} & \geq (& -4, & -3, & -2, & -1, & 0, & 1, & 2, & 3, & 4, & 5, & \cdots & 5). \\
\min\{i-1,4\} & \geq (& -1, & 0, & 1, & 2, & 3, & 4, & 4, & 4, & 4, & 4, & \cdots & 4). \\
p_i(C) & = (& 0, & 1, & 2, & 3, & 3, & 4, & 4, & 4, & 4, & 5, & \cdots & 5).
\end{array}$$

3.4 Generalized Roos bounds

The known bounds in Theorem 3.2.1 and Theorem 3.2.2 apply to codes A, B, C with $C \perp A * B$. They use only the minimum distance and the dual minimum distance of the codes A and B. The proof that we gave for both theorems is the same in each case and makes use of the profiles $p_i(A)$ and $p_i(B)$. Using the same proof but with more information about A and B, ideally knowledge of the full profiles $p_i(A)$ and $p_i(B)$, we improve the bounds of Theorem 3.2.1 and Theorem 3.2.2.

Theorem 3.4.1. *Let A, B and C be nondegenerate linear codes such that $C \perp (A * B)$. Let $d(A^\perp) > s$ and $g_r(A) < d_s(B^\perp) - s$. Then the weight w of a codeword $c \in C$ satisfies $w \leq s - 1$ or $w \geq d_s(B^\perp) + k(A) + 1 - r - s$.*

Proof. Let $c \in C$ be a nonzero word of weight w. The restrictions of A and B to the w coordinates of c are of total dimension $k(c * A) + k(c * B) \leq w$. With (3.5),

$$k(c * A) \geq \min\{k(A) + 1 - r, w - g_r(A)\}.$$
$$k(c * B) \geq \min\{w + 1 - s, d_s(B^\perp) - s\}.$$

We show that the minima are attained in $d_s(B^\perp)-s$ and in $k(A)+1-r$. The minimum for $k(c*B)$ can not be attained in $w+1-s$, for then $k(c*A) \leq s-1$ would violate that $d(A^\perp) > s$ (i.e. that any s positions in A are independent). Next, with $k(c*B) \geq d_s(B^\perp)-s$, $k(c*A) \geq w-g_r(A)$ would violate the assumption $g_r(A) < d_s(B^\perp)-s$. Thus $k(c*A) \geq k(A)+1-r$ and $w \geq k(A)+d_s(B^\perp)-s$. □

For the first order Reed-Muller code, the conditions are met for $(r,s) = (2,1), (2,2)$.

$$(r=2, s=1) \quad w \leq 0 \text{ or } w \geq 7.$$
$$(r=2, s=2) \quad w \leq 1 \text{ or } w \geq 8.$$

Thus the code has no words of weight $w \in (0,8)$. This is best possible, since the code has minimum distance $d(C) = 8$. Since the conditions are not met for $(r,s) = (1,1)$ the original Roos bound Theorem 3.2.1 does not apply in this case.

Theorem 3.4.2. *Let A, B and C be nondegenerate linear codes such that $C \perp (A*B)$. If $g_r(A) < k(B)+1-s$ and $g_s(B) < k(A)+1-r$ then, for a nonzero word $c \in C$ of weight w, either $w \leq g_r(A)+g_s(B)$ or $w \geq k(A)+k(B)+2-r-s$.*

Proof. As in the previous proof we have that $k(c*A)+k(c*B) \leq w$. With (3.5),

$$k(c*A) \geq \min\{k(A)+1-r, w-g_r(A)\}.$$
$$k(c*B) \geq \min\{k(B)+1-s, w-g_s(B)\}.$$

The minima are not attained in $k(A)+1-r$ and $w-g_s(B)$ for then $k(A)+1-r+w-g_s(B) \leq w$ would violate the assumption $g_s(B) < k(A)+1-r$. By symmetry the minima are attained either in $k(A)+1-r$ and $k(B)+1-s$, resulting in $w \geq k(A)+k(B)+2-r-s$, or in $w-g_r(A)$ and $w-g_s(B)$, resulting in $w \leq g_r(A)+g_s(B)$. □

For the first order Reed-Muller code, the conditions are met for $(r, s) = (1, 1), (2, \leq 4), (\geq 3, \leq 5)$.

$$(r = 1, s = 1) \quad w \leq 8 \text{ or } w \geq 10.$$

$$(r = 2, s = 2) \quad w \leq 2 \text{ or } w \geq 8.$$

$$(r = 2, s = 3) \quad w \leq 1 \text{ or } w \geq 7.$$

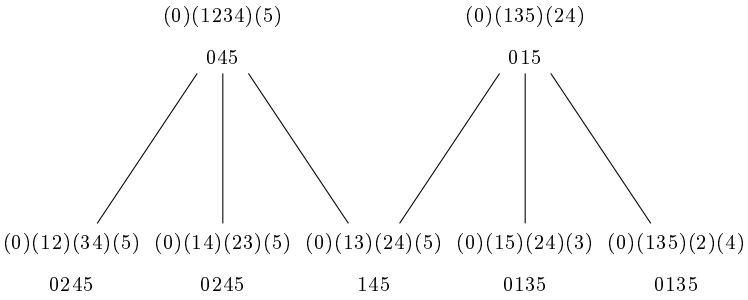
$$(r = 3, s = 3) \quad w \leq 0 \text{ or } w \geq 6.$$

Thus all weights $w \in (0, 10)$ are excluded except $w = 8$. In this case, the conditions are met for $(r, s) = (1, 1)$ and Theorem 3.2.2 applies. It excludes the weight $w = 9$ but does not give information about weights $w \leq 8$, and does not provide information about the minimum distance of the code.

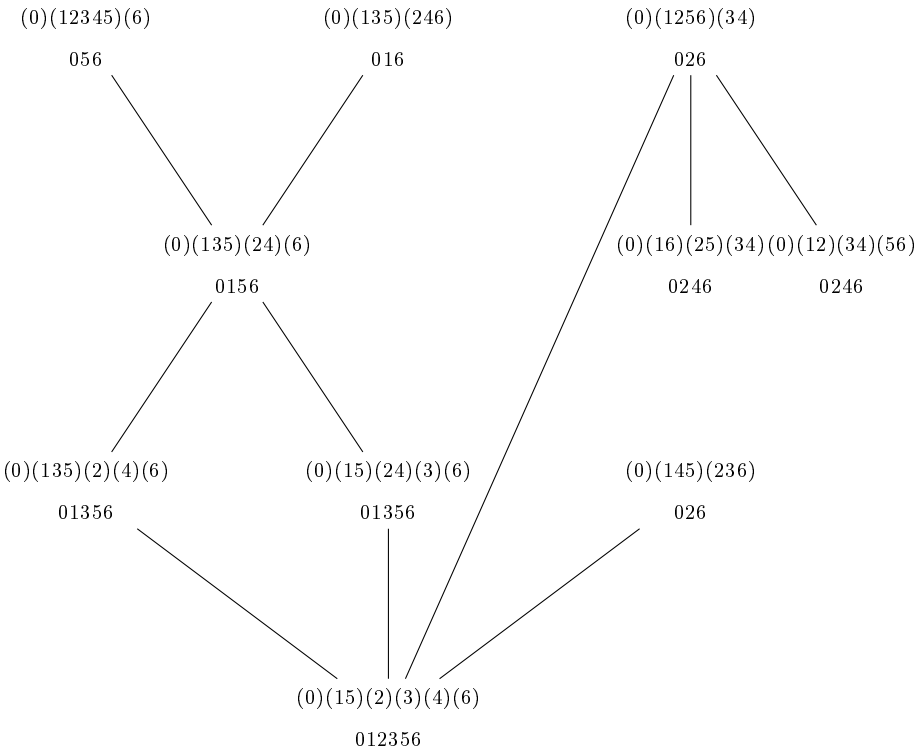
Appendix

The partition graphs for $k = 5, 6, 7, 8, 9$ are shown below. For every node, the top is the partition, the bottom is the distribution of the biggest degree of the characteristic vector corresponding to the combinations of the blocks.

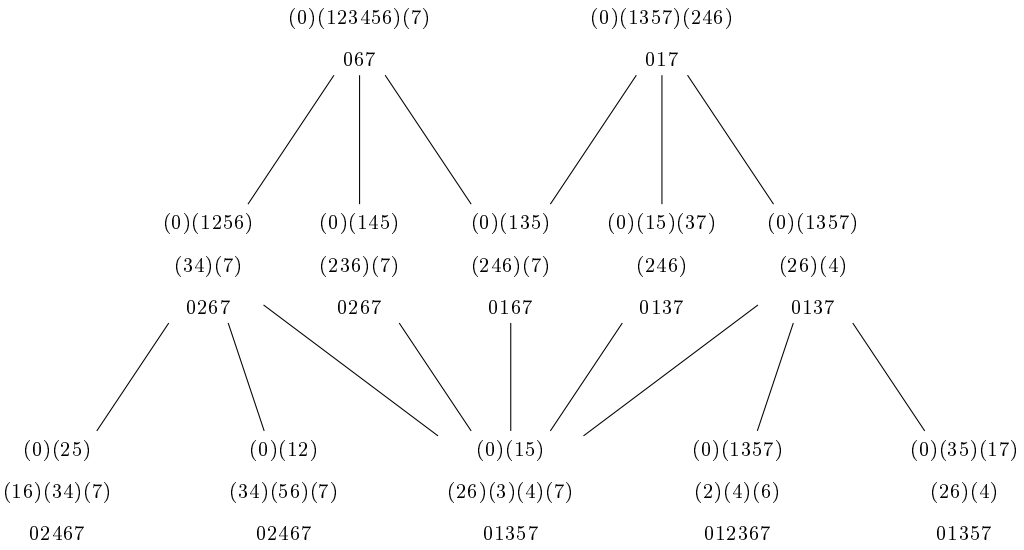
$k = 5$



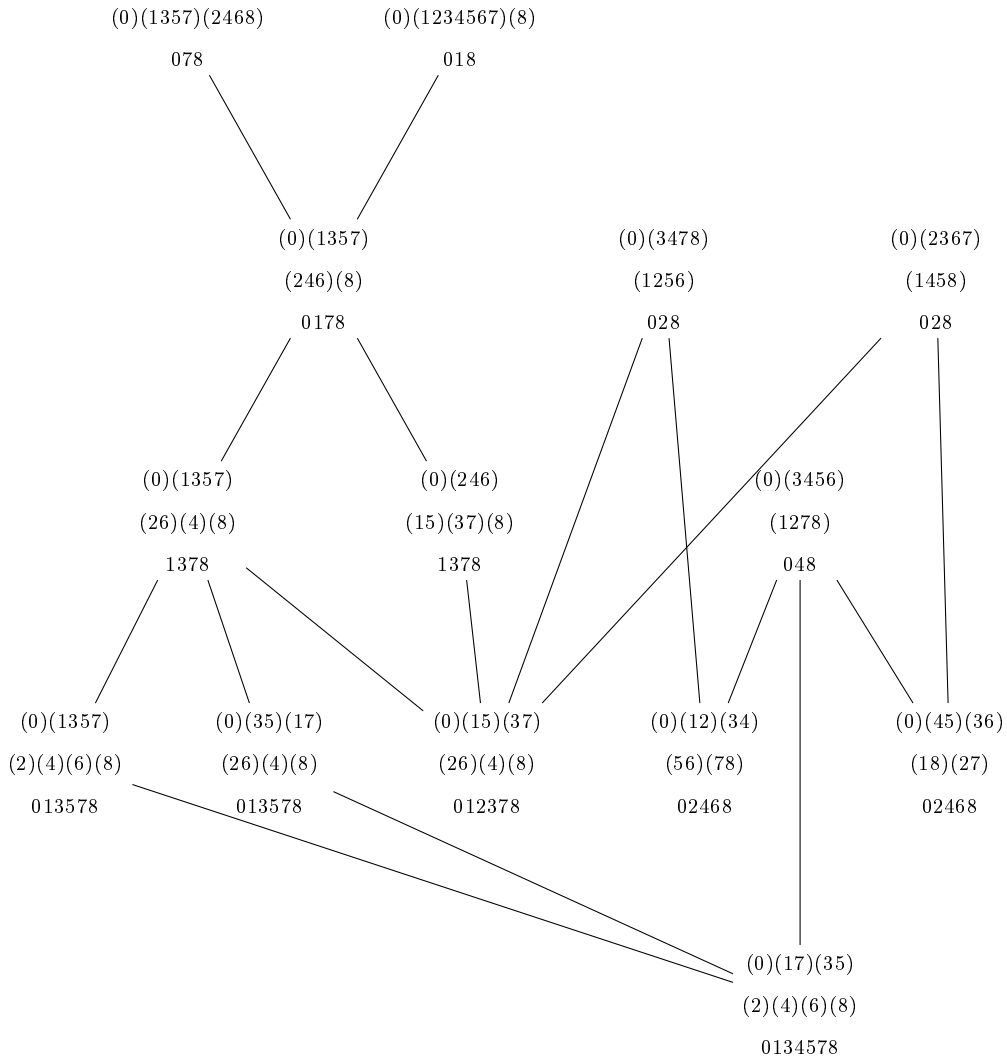
$k = 6$


(0)(12345)(6)
056
(0)(135)(246)
016
(0)(1256)(34)
026
(0)(135)(24)(6)
0156
(0)(16)(25)(34)
0246
(0)(12)(34)(56)
0246
(0)(135)(2)(4)(6)
01356
(0)(15)(24)(3)(6)
01356
(0)(145)(236)
026
(0)(15)(2)(3)(4)(6)
012356

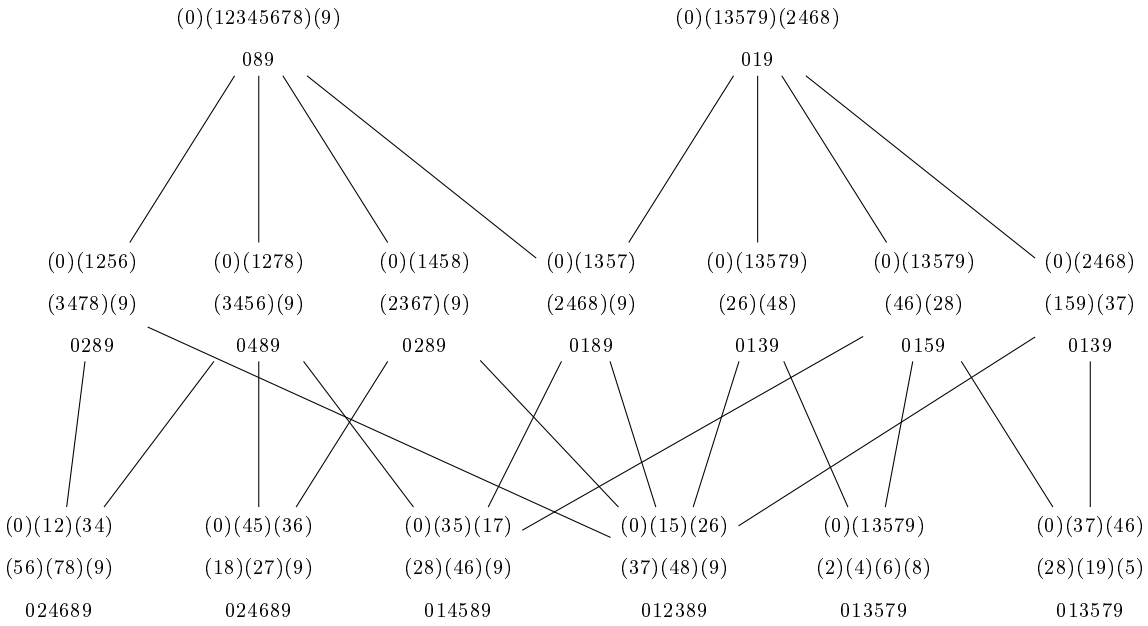
$k = 7$


(0)(123456)(7)
067
(0)(1357)(246)
017
(0)(1256)
(34)(7)
0267
(0)(145)
(236)(7)
0267
(0)(135)
(246)(7)
0167
(0)(15)(37)
(246)
0137
(0)(1357)
(26)(4)
0137
(0)(25)
(16)(34)(7)
02467
(0)(12)
(34)(56)(7)
02467
(0)(15)
(26)(3)(4)(7)
01357
(0)(1357)
(2)(4)(6)
012367
(0)(35)(17)
(26)(4)
01357

$k = 8$

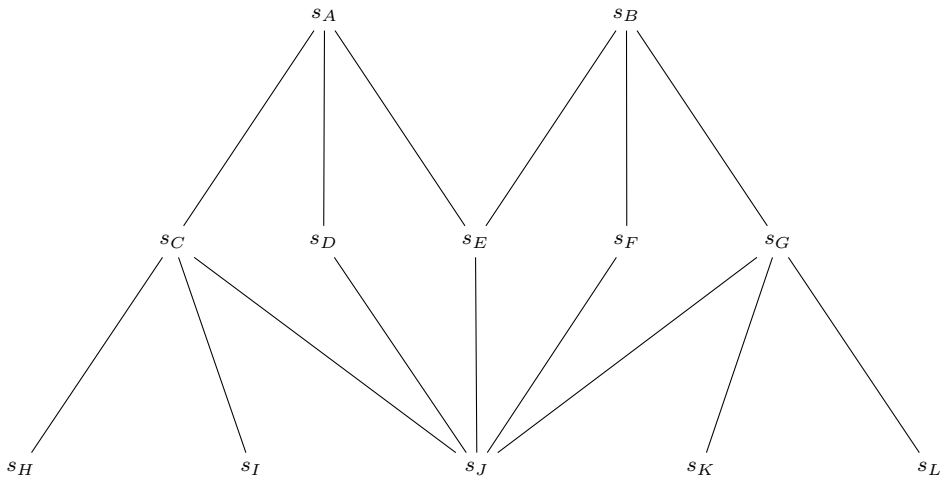

(0)(1357)(2468)
078
(0)(1234567)(8)
018
(0)(1357)
(246)(8)
0178
(0)(3478)
(1256)
028
(0)(2367)
(1458)
028
(0)(1357)
(26)(4)(8)
1378
(0)(246)
(15)(37)(8)
1378
(0)(3456)
(1278)
048
(0)(1357)
(2)(4)(6)(8)
013578
(0)(35)(17)
(26)(4)(8)
013578
(0)(15)(37)
(26)(4)(8)
012378
(0)(12)(34)
(56)(78)
02468
(0)(45)(36)
(18)(27)
02468
(0)(17)(35)
(2)(4)(6)(8)
0134578

$k = 9$


(0)(12345678)(9)
089
(0)(13579)(2468)
019
(0)(1256)
(3478)(9)
0289
(0)(1278)
(3456)(9)
0489
(0)(1458)
(2367)(9)
0289
(0)(1357)
(2468)(9)
0189
(0)(13579)
(26)(48)
0139
(0)(13579)
(46)(28)
0159
(0)(2468)
(159)(37)
0139
(0)(12)(34)
(56)(78)(9)
024689
(0)(45)(36)
(18)(27)(9)
024689
(0)(35)(17)
(28)(46)(9)
014589
(0)(15)(26)
(37)(48)(9)
012389
(0)(13579)
(2)(4)(6)(8)
013579
(0)(37)(46)
(28)(19)(5)
013579

$k = 7$

The graph for characteristic vectors



$s_A = (s_1 + s_2 + s_3 + s_4 + s_5 + s_6, s_7)$

$s_B = (s_1, s_2 + s_3 + s_4 + s_5 + s_6 + s_7)$

$s_C = (s_1 + s_2, s_3 + s_4 + s_5 + s_6 + s_7)$

$s_D = (s_1 + s_3 + s_4 + s_5 + s_6 + s_7, s_2 + s_7, s_7)$

$s_E = (s_1 + s_7, +s_2 + s_3 + s_4 + s_5 + s_6 + s_7, s_7)$

$s_F = (s_3, s_2 + s_3 + s_4 + s_5 + s_6 + s_7, s_1 + s_3)$

$s_G = (s_1, s_2 + s_3, s_4 + s_5 + s_6, s_7)$

$s_H = (s_2 + s_3 + s_5 + s_6, s_1 + s_3 + s_5 + s_6, s_3 + s_4 + s_5 + s_6, s_7)$

$s_I = (s_1 + s_2 + s_5 + s_6, s_3 + s_4 + s_5 + s_6, s_5 + s_6, s_7)$

$s_J = (s_1 + s_3, s_2 + s_3, s_3 + s_7, s_4 + s_5 + s_6 + s_7, s_7)$

$s_K = (s_1, s_2 + s_3 + s_6 + s_7, s_4 + s_5 + s_6 + s_7, s_6 + s_7)$

$s_L = (s_3 + s_5, s_1 + s_3 + s_5, s_2 + s_3, s_4 + s_5 + s_6 + s_7)$

References

[1] P. J. Cameron and J. H. van Lint. *Designs, graphs, codes and their links*, volume 22 of *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 1991.

[2] R. Cramer, I. Damgard, and B. Nielsen, Jesper. *Secure Multiparty Computation and Secret Sharing An Information Theoretic Approach*. Book Draft. 2013.

[3] I. Dumer and O. Kapralova. Spherically punctured biorthogonal codes. In *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*, pages 259–263, July 2012.

[4] I. Dumer and O. Kapralova. Spherically punctured biorthogonal codes. *IEEE Trans. Inform. Theory*, 59(9):6010–6017, 2013.

[5] I. Duursma and J. Shen. Multiplicative secret sharing schemes from reed-muller type codes. In *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*, pages 264–268, July 2012.

[6] I. M. Duursma. Algebraic geometry codes: general theory. In *Advances in algebraic geometry codes*, volume 5 of *Ser. Coding Theory Cryptol.*, pages 1–48. World Sci. Publ., Hackensack, NJ, 2008.

[7] I. M. Duursma and R. Pellikaan. A symmetric Roos bound for linear codes. *J. Combin. Theory Ser. A*, 113(8):1677–1688, 2006.

[8] G. D. Forney, Jr. Dimension/length profiles and trellis complexity of linear block codes. *IEEE Trans. Inform. Theory*, 40(6):1741–1752, 1994.

[9] T. Koshy. *Catalan numbers with applications*. Oxford University Press, Oxford, 2009.

[10] A. Lafourcade and A. Vardy. Lower bounds on trellis complexity of block codes. *IEEE Trans. Inform. Theory*, 41(6, part 2):1938–1954, 1995.

[11] S. Ling and C. Xing. *Coding theory*. Cambridge University Press, Cambridge, 2004. A first course.

[12] F. J. MacWilliams and N. J. A. Sloane. *The theory of error-correcting codes. I.* North-Holland Publishing Co., Amsterdam-New York-Oxford, 1977. North-Holland Mathematical Library, Vol. 16.

[13] C. Roos. A new lower bound for the minimum distance of a cyclic code. *IEEE Trans. Inform. Theory*, 29(3):330–332, 1983.

[14] H. Schenck. *Computational algebraic geometry*, volume 58 of *London Mathematical Society Student Texts*. Cambridge University Press, Cambridge, 2003.

[15] J. H. van Lint. *Introduction to coding theory*, volume 86 of *Graduate Texts in Mathematics*. Springer-Verlag, Berlin, third edition, 1999.

[16] A. Vardy. Trellis structure of codes. In *Handbook of coding theory, Vol. I, II*, pages 1989–2117. North-Holland, Amsterdam, 1998.