BACHELOR

Cosetleader weight enumerator of the product code Cm ⊗ Cn

Geraerts, Lars J.M.

Award date:
2019

Cosetleader Weight Enumerator of the Product Code $C_m \otimes C_n$

L.J.M. Geraerts (0915372)
l.j.m.geraerts@student.tue.nl

May 24, 2019

Contents

1 Abstract

Error correction is an important part of coding theory. It ensures that, most of the time, the errors that a codeword has received get resolved. This paper is based on an article on the coset leader weight enumerator of a product code[1]. In the following paper[3], it is explained how these coset leaders can be used to correct errors. We focus on matrix embedding codes, which are codes that use syndromes to correct errors. These syndromes can then be used to find the coset leader of the specific coset that the codeword containing an error is an element of. However, finding the coset leader of a coset can be quite difficult, which is why in this paper it is explained how to find an estimate of the coset leader in a specific case using sequences. Our ultimate goal would be to find a formula that would give us the corresponding coset leader when plugging in a codeword, and the coset leader weight enumerator for the code. Hence we will mainly focus on finding the coset leader and the process on how we manage to find them.

2 Introduction

Coding theory has as goal to ensure that the message that is sent indeed is received correctly by the receiver. One of those methods is to add redundant information to the original message. To model this, we consider the Binary Symmetric Channel (BSC) that uses bits, that is 1's and 0's, to send information from a transmitter to a receiver. Usually bits are received correctly, and the receiver can read the message accordingly. However on a noisy channel, there is a probability p, that a bit gets flipped, meaning a 1 changes to a 0 or vice versa. BSC assumes that the same probability p holds for both switches. This probability is known as the "crossover probability".

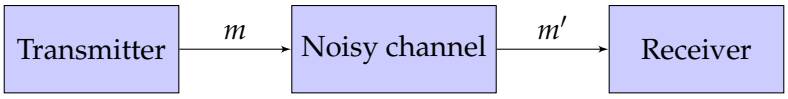


Figure 1: Change in message over a noisy channel.

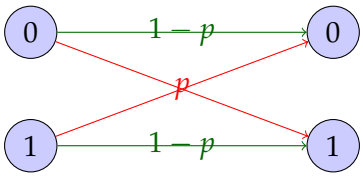


Figure 2: Crossover probability.

In figure 1 it is shown how the transmitter sends their original message m over a noisy channel to the receiver. However some bits may flip as seen in figure 2, which causes the message to be altered to m'. Obviously the receiver does not want to receive an altered message, that is not the original, sent by the transmitter. This would mean there should be a way that the devices can find out if the received message m' does, or does not equal the message m that has been sent, which is called error detection. If an error has been detected, it should also be able to change m' back into the original message m, which is called error correction.
A solution to this problem is to introduce encoding and decoding in this way of messaging.

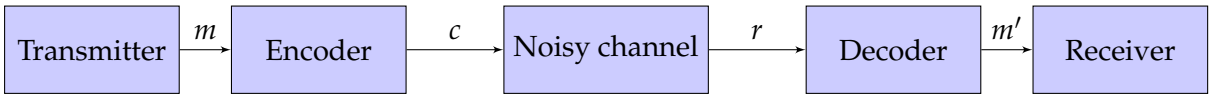


Figure 3: Noisy channel problem solved.

In figure 1 it is shown that a message m is sent through an encoder. The encoded message $c = E(m)$ is then sent over a noisy channel, where the bits may get flipped, therefore c will be changed into r. The decoder then decodes the received message r, and outputs $m' = D(r)$. What we obviously want, is that $m' = m$, since then the original message is received by the receiver.

3 Pre-knowledge

Definition 3.1. *An alphabet Q is a set containing all known symbols. Q^n is the set that contains all words of a set length n.*

Definition 3.2. *A (block) code is a non empty subset $C \subseteq Q^n$*

Example definition 3.2
A realistic comparison would be the dictionary. Here $Q = \{a, b, c, ..., x, y, z\}$ obviously, and C would be the words of length n. However Q^n contains strings of length n that are not real words.

Definition 3.3. *Let $Q = \mathbb{F}_q$, then it follows that $Q^n = \mathbb{F}_q^n$ is a vector field over $\mathbb{F}_q$. Let $C \subseteq \mathbb{F}_q^n$ be a linear subspace, then C is called a linear (block) code.*

Definition 3.4. *A codeword $\boldsymbol{x} \in \mathbb{F}_q^n$ is a string of digits in $\mathbb{F}_q^n$ or $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$, if p is prime, in the form $(x_1, x_2, \ldots, x_{n-1}, x_n)$.*

Definition 3.5. *The weight of a codeword $x \in \mathbb{F}_q^n$ is the number of non-zero entries, noted down as:*

$$\begin{aligned} wt(x) &= |supp(x)| \\ &= |\{i : x_i \neq 0\}| \end{aligned}$$

Definition 3.6. *The Hamming distance between two words $x, y \in Q^n$ is defined:*

$$d(x, y) = |\{i | x_i \neq y_i\}| .$$

Remark 3.1. For the Hamming distance on all $x, y \in Q^n$, several statements hold:

1. 1) $d(x, y) \geq 0$;
   2) $d(x, y) = 0 \Leftrightarrow x = y$;
2. $d(x, y) = d(y, x)$;
3. $d(x, y) \leq d(x, z) + d(z, y)$.

We also state that the distance of a code $C \subseteq Q^n$ is the minimal distance between two different codewords, or the minimum weight of the codewords, excluding the all-zero word.

$$\begin{aligned} &d(C) = \min\{d(x, y) | x, y \in C, x \neq y\}, \text{ or} \\ &d(C) = \min\{wt(x) | x \in C, x \neq 0\} \text{ if } C \text{ is linear.} \end{aligned}$$

Definition 3.7. *An arithmetical system in which every element, that is not equal to 0, has a multiplicative inverse, is called a field [2]. Addition, subtraction and division have to be defined on this set.*

Example definition 3.7
$Q = \{0, 1\} = \mathbb{F}_2$ is a finite field.
Since $\forall x, y \in Q : x + y, x - y, x \cdot y \in Q$ and $\forall x \in Q, y \in Q \setminus \{0\} : \frac{x}{y} \in Q$.
$\mathbb{F}_q$ is the finite field of size q, a power of a prime, and $\mathbb{F}_q = \mathbb{Z}/q\mathbb{Z}$ if q is a prime.

4 code

Definition 4.1. *Let C be a linear code of dimension k and length n, then it has a basis existing out of $g_1, g_2, ..., g_k$, that all have length n, which together form the rows of the generator matrix G of the code C, of size $k \times n$.*

$$G = \begin{bmatrix} —g_1— \\ —g_2— \\ \vdots \\ —g_{k-1}— \\ —g_k— \end{bmatrix}$$

Remark 4.1. For the generator matrix G of a code C the following must hold:

$$x \in C \Leftrightarrow x = aG, \text{ for } a \in \mathbb{F}_q^k.$$

Since a has dimensions $1 \times k$ and G has dimensions $k \times n$, x indeed has dimension n, and is therefore indeed an element of $\mathbb{F}_q^n$.

Definition 4.2. *The linear code $[n, k, d]_q$, is a $\mathbb{F}_q$-linear subspace of $\mathbb{F}_q^n$, of dimension k and minimum distance d.*

Definition 4.3. *The $\mathbb{F}_q$-linear code C_n, is defined by the property $\forall c_i \in C_n \Leftrightarrow \sum_{c_i \in C_n} c_i = \underline{0}$. A second property of C_n is that it would be the $[n, n-1, 2]_q$ code.*

Remark 4.2. *For $q = 2$, C_n becomes the $[n, n-1, 2]_2$ code, which is called the "even weight code", since $c \in C \Leftrightarrow c$ has an even weight.*

Example definition 4.1
The code C_n, has a generator matrix G_n, of size $(n-1) \times n$, that has the following form:

$$G_n = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & -1 \\ 0 & 1 & \ddots & 0 & 0 & -1 \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 1 & 0 & -1 \\ 0 & 0 & \cdots & 0 & 1 & -1 \end{bmatrix}$$

Remark 4.3. A $[n, k, d]_q$ code has q^k different codewords, and all of those codewords are given by means of the generator matrix G, for some $a \in \mathbb{F}_q^k$.

$$\begin{aligned} c &= aG \\ &= \begin{bmatrix} a_1 & a_2 & \cdots & a_{k-1} & a_k \end{bmatrix} \begin{bmatrix} g_{1,1} & g_{1,2} & \cdots & \cdots & g_{1,n-1} & g_{1,n} \\ g_{2,1} & g_{2,2} & \cdots & \cdots & g_{2,n-2} & g_{2,n} \\ \vdots & \vdots & \ddots & \iddots & \vdots & \vdots \\ \vdots & \vdots & \iddots & \ddots & \vdots & \vdots \\ g_{k-1,1} & g_{k-1,2} & \cdots & \cdots & g_{k-1,n-1} & g_{k-1,n} \\ g_{k,1} & g_{k,2} & \cdots & \cdots & g_{k,n-1} & g_{k,n} \end{bmatrix} \end{aligned}$$

Lemma 4.1. *If $q = 2$, then every codeword $x \in C_n \Leftrightarrow wt(x)$ is even.*

Proof. We know what the generator matrix G of C_n looks like, and see that every row vector $g_i \in G$ has two ones and $n-2$ zeros. Every codeword in C_n is a linear combination of these g_i's where each g_i adds a 1 at a different position of the codeword, and a 1 at the last position.

Therefore if there is an even amount of distinct g_i's that create the codeword x, there are an even amount of 1's in distinct positions, and an even amount of 1's that add up to 0 in $\mathbb{F}_2$ in the last position, hence the codeword is of even weight. Now when the codeword exists of an odd amount of distinct g_i's, there are an odd amount of 1's in distinct positions, and an odd amount of 1's in the last position, which in $\mathbb{F}_2$ add up to 1. Together this shows that all codewords have an even weight in C_n.

□

Definition 4.4. *A parity check matrix H, of an $\mathbb{F}_q$-linear $[n,k]_q$ code C, is an $(n-k) \times n$ matrix for which the following holds:*

1. $xH^T = 0$ *in* $\mathbb{F}_q$, $\forall x \in C$;
2. *all the rows of H must be independent;*
3. $rank(H) = n - k$.

Here points 2 and 3 are equivalent, and follow from 1 and the size of H.

Remark 4.4. *In the case of C_n, H is the all 1 vector of length n, by the definition of C_n.*

Example definition
We start with the linear code C_4, and note that the generator matrix therefore is

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

Next we will create the parity check matrix H by noting that all the codewords have an even weight, therefore the parity check matrix H of C, is equal to the all one vector $(1,1,1,1)$. And we note that $\text{rank}(H) = 4 - 3 = 1$, hence this row is independent, and it holds that $\forall x \in C : xH^T = 0$, since all codewords have an even weight.

5 Coset leader weight enumerator

Definition 5.1. *Let G be an abelian group with $+$ as group operator, H a subgroup of G and $g \in G$. Then:*

$$g + H = \{g + h : h \in H\} \text{ is the coset of } H \text{ in } G \text{ with respect to } g.$$

When bits get flipped, codewords originating from the code C, end up in some other coset $x + C$. Clearly we want to find the original codeword, and we do this by removing a codeword $c' \in x + C$, such that c' has the lowest possible weight in that coset. Such an element is called a "coset leader". It is possible that a coset contains several different codewords of minimum weight, but then it does not matter which one becomes the coset leader.

Definition 5.2. *We let $C \in \mathbb{F}_q^n$. Then we define the following:*
$\alpha_i(C)$ is the number of coset leaders of C having weight i;
$\alpha_C(Z)$ is the coset leader weight enumerator of C and is given by:

$$\alpha_C(Z) = \sum_{i=0}^{n} \alpha_i(C) Z^i$$

Remark 5.1. In the case of $C = [n, n-1, 2]_q$ it would follow that:

$$\begin{aligned} \alpha_{C_n}(Z) &= \alpha_0(C_n) + \alpha_1(C_n) Z \\ &= 1 + (q-1)\, Z \end{aligned}$$

Later on we will also use the 2-variable homogeneous coset leader weight enumerator, which is defined by:

$$\alpha_{C_n}(X, Y) = \sum_{i=0}^{n} \alpha_i(C) X^{n-i} Y^i$$

Now we have all the information we need, we will have a look at the cosets of a binary code C_n. In the following example, we will use $n = 4$ for the binary case.

Example definition 5.1 applied to $\mathbb{F}_2^4$
We will look at C_4, and note that:

$$C_4 = \{0000, 1001, 0101, 0011, 1100, 1010, 0110, 1111\}.$$

While in fact there are $2^4 = 16$ different codewords $x \in \mathbb{F}_2^4$. Hence we will compute the elements in the coset $C_4 + 1000$, and find:

Coset	Codewords
$C_4 + \underline{0}$	0000, 0101, 0011, 0110, 1001, 1100, 1010, 1111
$C_4 + 1000$	1000, 1101, 1011, 1110, 0001, 0100, 0010, 0111

All of the elements in $\mathbb{F}_2^4$ have now been listed, hence there are indeed 2 cosets.

Remark 5.2. We now note that for every set $\mathbb{F}_2^n$ it is possible to divide the set in two distinct cosets, namely C, and $C + g'$, where g' is an element of $\mathbb{F}_2^n \setminus C$.

Definition 5.3. *The coset leader is a word of minimum weight in any particular coset. If several words have the minimum weight, you can pick any of these words*

Example definition 5.3
We will look at C_4 again in the binary case, and conclude:

Coset	Codewords	Coset leader
$C_4 + \underline{0}$	0000, 0101, 0011, 0110, 1001, 1100, 1010, 1111	0000
$C_4 + 1000$	1000, 1101, 1011, 1110, 0001, 0100, 0010, 0111	0100

1000, 0010 or 0001 are also coset leaders of $C_4 + 1000$, but as said before we are allowed to simply choose one. It is clear that each coset leader would result in the same coset.

Remark 5.3. All C_n have indeed 2 cosets in the binary case. This is easy to see when we look at the generator matrix existing of row vectors g_i's. We create the cosets $C_n + \underline{0}$ and $C_n + 100...00$, and see:

Coset	Codewords	Coset leader
$C_n + \underline{0}$	all linear combinations with the g_i's	$\underline{0}$
C_n + 100...00	all linear combinations with the g_i's+100...00	100...00

As stated before C_n has 2^{n-1} codewords, so $C_n + 100...00$ must also have 2^{n-1} codewords, and $\mathbb{F}_2^n$ has $2^n = 2^{n-1} + 2^{n-1}$ codewords, hence all codewords are contained in these two cosets.
In the case of the $[n, n-1, 2]_q$ code C_n, we note that there are q^n codewords in F_q^n, while C only has q^{n-1} codewords. It follows there are q cosets namely the following:

$$C + \lambda(1, 0, 0, ..., 0, 0), \text{ with } \lambda \in \mathbb{F}_q.$$

All cosets are distinct and have size q^{n-1}, so the total amount of elements in the cosets is $q \cdot q^{n-1} = q^n$. Hence all codewords are shown exactly once.

Definition 5.4. *Let C be a linear code in $\mathbb{F}_q^n$. Then we define the distance of any codeword $x \in \mathbb{F}_q^n$ to C as follows:*

$$d(x, C) = min\{d(x, c) | c \in C\}.$$

Definition 5.5. *Let C be a linear code in $\mathbb{F}_q^n$. Then the covering radius $\rho(C)$ is defined in the following two ways:*

$$\begin{aligned} \rho(C) &= max\left\{d(x, C) | x \in \mathbb{F}_q^n\right\} \\ &= max\left\{i | \alpha_i(C) \neq 0\right\} \end{aligned}$$

Proof. We note that $d(x, C)$ is the minimum distance between an arbitrary codeword x, and a codeword $c \in C$. We know that for the code C, it holds that any linear combination of codeword in C, must also be in C. Hence if x is in some coset, $x - c$ must be in the same coset. When we would compute $x - c, \forall c \in C$, one of these would give us the coset leader of this coset. So if we do this for every $x \in \mathbb{F}_q^n$, this would give us every single coset leader, and it follows that the maximum over the weight of the coset leaders equals the maximum over the distances from each codeword x to the code C.
Therefore $\max\left\{d(x, C) | x \in \mathbb{F}_q^n\right\} = \max\left\{i | \alpha_i(C) \neq 0\right\}$

□

Proposition 5.1. *Let C be an $[n,k,d]_q$ code. Then:*

$$\alpha_i(C) = \binom{n}{i}(q-1)^i, \text{ if } i \leq \frac{d-1}{2}.$$

Proof. Right away we note that C only contains codewords with a weight greater or equal to d, except for the all zero codeword. From this it follows that whenever we calculate the weight of a coset $x + C$, we simply add this element x to all of the codewords in C, and note that if the weight of x is less or equal to $\frac{d-1}{2}$, there will be exactly one element of that weight, since all others will have a weight of at least $d - i$. Therefore if $\text{wt}(x) = i \leq \frac{d-1}{2}$, we can choose i places out of the total n choices where we put a number not equal to zero, and this number can equal $q - 1$ different numbers. Therefore it follows:

$$\alpha_i(C) = \binom{n}{i}(q-1)^i, \text{ if } i \leq \frac{d-1}{2}.$$

□

One, very tedious, way to encode bits, is by repeating them several times. here the bit 0 would be encoded as 00...00 and 1 as 11...11. We represent the repetition code that repeats bits n times, as S_n. For each n, S_n has only 2 elements, namely 00...00 and 11...11, therefore the amount of cosets is equal to 2^{n-1}, all with 2 elements. Now we want to find all of the $\alpha_i(S_n)$. First we note that the weight of the coset leaders can never exceed $\frac{n}{2}$, cause either one of the elements in $x + S_n$ will always have at most $\frac{n}{2}$ 1's.
To find the number of cosets with weight i, we note that this is equal to simply choosing i locations in the codeword to be 1's while all others are 0's. Therefore for $n > 2i$ it must hold that $\alpha_i(S_n) = \binom{n}{i}$.
Example $\alpha_2(S_5)$
$S_5 = \{00000, 11111\}$

Coset	Codewords
$11000 + S_5$	11000, 00111
$10100 + S_5$	10100, 01011
$10010 + S_5$	10010, 01101
$10001 + S_5$	10001, 01110
$01100 + S_5$	01100, 10011
$01010 + S_5$	01010, 10101
$01001 + S_5$	01001, 10110
$00110 + S_5$	00110, 11001
$00101 + S_5$	00101, 11010
$00011 + S_5$	00011, 11100

Therefore $\alpha_2(S_5) = \binom{5}{2} = 10$.

Next up we note that if n is an even number, $\alpha_{\frac{n}{2}}(S_n)$ must equal $\alpha_{\frac{n}{2}-1}(S_{n-1})$. This is not too difficult to understand, for we can again take the first $\frac{n}{2}$ positions to be 1's, hence the last $\frac{n}{2}$ positions must be 0's. However the codeword which looks like 00...0011...11 is also in that particular coset. Hence each mirrored codeword will instantly be added to that coset. Therefore we can reduce the problem by claiming that the first position must always be a 1, for the mirrored codeword will also be in the coset. We reduced the problem by one bit, and one 1, therefore it is equal to $\alpha_{\frac{n}{2}-1}(S_{n-1}) = \binom{n-1}{\frac{n}{2}-1}$.

Now we will put these results in a table to find:

	S_1	S_2	S_3	S_4	S_5	S_6	S_7
α_0	$\binom{1}{0}$	$\binom{2}{0}$	$\binom{3}{0}$	$\binom{4}{0}$	$\binom{5}{0}$	$\binom{6}{0}$	$\binom{7}{0}$
α_1	0	$\alpha_0(S_1)$	$\binom{3}{1}$	$\binom{4}{1}$	$\binom{5}{1}$	$\binom{6}{1}$	$\binom{7}{1}$
α_2	0	0	0	$\alpha_1(S_3)$	$\binom{5}{2}$	$\binom{6}{2}$	$\binom{7}{2}$
α_3	0	0	0	0	0	$\alpha_2(S_5)$	$\binom{7}{3}$

And we conclude:

	S_1	S_2	S_3	S_4	S_5	S_6	S_7
α_0	1	1	1	1	1	1	1
α_1	0	1	3	4	5	6	7
α_2	0	0	0	3	10	15	21
α_3	0	0	0	0	0	10	35

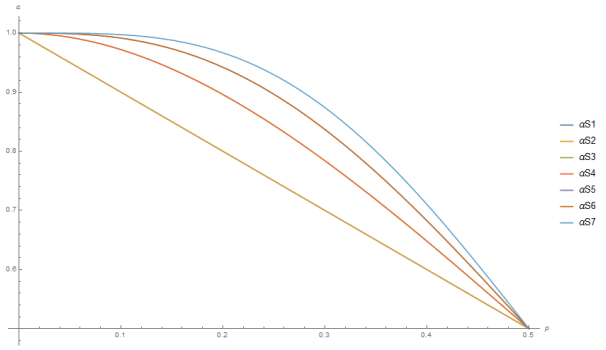
In general:

$$\alpha_i(S_n) = \begin{cases} 0 & \text{, if } i > \lceil \frac{n-1}{2} \rceil \\ \binom{n-1}{i-1} & \text{, if } i = \frac{n}{2} \\ \binom{n}{i} & \text{, if } i < \lfloor \frac{n+1}{2} \rfloor \end{cases}$$

Now we can compute the coset leader weight enumerators for any S_n. First we change this to a bivariate function:

$$\alpha_{S_n}(X, Y) = \sum_{i=0}^{n} \alpha_i(S_n) X^{n-i} Y^i$$

We then say $X = 1 - p$ and $Y = \frac{p}{q-1}$, and plot the first few coset leader weight enumerators.



We note that we plotted 7 functions, even though only 4 curves visible, The reason for this is that for $q = 2$, and for all $n = 1, 2, ...$, it holds that $\alpha_{S_{2n}}(1 - p, p) = \alpha_{S_{2n-1}}(1 - p, p)$.

Claim: $\alpha_{S_{2n}}(1-p,p) = \alpha_{S_{2n-1}}(1-p,p)$.
Proof:
For $n = 1$ we get the very easy equations:

$$\begin{aligned}
\alpha_{S_1}(1-p,p) &= \sum_{i=0}^{1} \alpha_i(S_1)(1-p)^{1-i}p^i \\
&= 1 \cdot (1-p)^1 \\
&= 1-p \\
\alpha_{S_2}(1-p,p) &= \sum_{i=0}^{2} \alpha_i(S_2)(1-p)^{2-i}p^i \\
&= 1 \cdot (1-p)^2 + 1 \cdot (1-p)^1 p^1 \\
&= 1 - 2p + p^2 + p - p^2 \\
&= 1-p
\end{aligned}$$

So for $n = 1$ it holds. However it turns out that this will quickly become quite hard to calculate by hand. Therefore this is only a setup of the full proof.
We note that for $\alpha_{S_{2n}}$ it holds that $\alpha_i = \binom{2n}{i}, \forall i \leq n-1$, $\alpha_n = \frac{\binom{2n}{n}}{2} = \binom{2n-1}{n-1}$, and zero for all $i > n$. Therefore:

$$\begin{aligned}
\alpha_{S_{2n}}(X,Y) &= \sum_{i=0}^{n} \alpha_i(S_{2n}) X^{2n-i}Y^i \\
&= \sum_{i=0}^{n-1} \binom{2n}{i} X^{2n-i}Y^i + \binom{2n-1}{n-1} X^n Y^n \\
\alpha_{S_{2n-1}}(X,Y) &= \sum_{i=0}^{n-1} \alpha_i(S_{2n-1}) X^{2n-1-i}Y^i \\
&= \sum_{i=0}^{n-1} \binom{2n-1}{i} X^{2n-1-i}Y^i
\end{aligned}$$

Now it is shown that we would have to deal with different binomial numbers, together with polynomials that are supposed to add up to zero, while we are unable to solve this any further than this. We will end this "proof" here for now.

We now want to use the coset leader weight enumerator $\alpha_C(X,Y)$ to show the probability of correcting an error to the correct codeword. Hence our variables are p and q, where p is the crossover probability.

Definition 5.6. *Let C be a code, then the error probability is defines as follows:*

$$P(p) = \alpha_C(1-p, \frac{p}{q-1})$$

6 Product codes

Now it is known what a code is, we move on to the so called "product code". Codewords of a product code, can be represented in two different ways. One way being the already known string vector, and the other being in an array. The product code of two codes C and D will be denoted as $C \otimes D$, and when the codewords would be denoted in the array form, then each column of that array is a codeword in C, while each row is a codeword in D. To go from the array form to the string form of a codeword, we simply write down all of the rows behind each other in descending order.

Example a codeword in $C_3 \otimes C_4$

$$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} \Leftrightarrow \begin{bmatrix} 1010 & 0011 & 1001 \end{bmatrix}$$

Hence we prefer to work with the array notation of codewords.

Definition 6.1. *Let A be an $m \times n$ matrix and B be a $k \times l$ matrix. We define the Kronecker product, with the symbol $\otimes$ as follows:*

$$A \otimes B = \begin{bmatrix} a_{1,1}B & \cdots & a_{1,j}B & \cdots & a_{1,n}B \\ \vdots & & \vdots & & \vdots \\ a_{i,1}B & \cdots & a_{i,j}B & \cdots & a_{i,n}B \\ \vdots & & \vdots & & \vdots \\ a_{m,1}B & \cdots & a_{m,j}B & \cdots & a_{m,n}B \end{bmatrix}$$

Hence $A \otimes B$ becomes an $mk \times nl$ matrix.

Example definition 6.1

$$\begin{bmatrix} 1 & 3 \\ 4 & 2 \end{bmatrix} \otimes \begin{bmatrix} 0 & 5 & 4 \\ 7 & 1 & 0 \\ 3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 \begin{bmatrix} 0 & 5 & 4 \\ 7 & 1 & 0 \\ 3 & 2 & 1 \end{bmatrix} & 3 \begin{bmatrix} 0 & 5 & 4 \\ 7 & 1 & 0 \\ 3 & 2 & 1 \end{bmatrix} \\ 4 \begin{bmatrix} 0 & 5 & 4 \\ 7 & 1 & 0 \\ 3 & 2 & 1 \end{bmatrix} & 2 \begin{bmatrix} 0 & 5 & 4 \\ 7 & 1 & 0 \\ 3 & 2 & 1 \end{bmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 5 & 4 & 0 & 15 & 12 \\ 7 & 1 & 0 & 21 & 3 & 0 \\ 3 & 2 & 1 & 9 & 6 & 3 \\ 0 & 20 & 16 & 0 & 10 & 8 \\ 28 & 7 & 0 & 14 & 2 & 0 \\ 12 & 8 & 4 & 6 & 4 & 2 \end{bmatrix}$$

Remark 6.1. *The generator matrix, $G_{A\otimes B}$, of a product code, is equal to the Kronecker product of the generator matrices of those codes G_A and G_B.*

$$G_{A\otimes B} = G_A \otimes G_B.$$

Proof. We let A be the $[n_1, k_1, d_1]_q$ code, and B be the $[n_2, k_2, d_2]_q$ code. We know that, when representing codewords of $A \otimes B$ in the array form, each row must be an element of B, while each column is an element of A. For visualisation we define:

$$G_A = \begin{bmatrix} a_{1,1} & \cdots & a_{1,n_1} \\ \vdots & & \vdots \\ a_{k_1,1} & \cdots & a_{k_1,n_1} \end{bmatrix} \qquad G_B = \begin{bmatrix} b_{1,1} & \cdots & b_{1,n_2} \\ \vdots & & \vdots \\ b_{k_2,1} & \cdots & b_{k_2,n_2} \end{bmatrix}$$

From definition 6.1 we also know that:

$$G_A \otimes G_B = \begin{bmatrix} a_{1,1}G_B & a_{1,2}G_B & \cdots \\ a_{2,1}G_B & a_{2,2}G_B & \cdots \\ \vdots & \vdots & \end{bmatrix}$$

$$= \begin{bmatrix} a_{1,1}b_{1,1} & a_{1,1}b_{1,2} & \cdots & a_{1,1}b_{1,n_2} & a_{1,2}b_{1,1} & \cdots \\ a_{2,1}b_{1,1} & a_{2,1}b_{1,2} & \cdots & a_{2,1}b_{1,n_2} & a_{2,2}b_{1,1} & \cdots \\ \vdots & \vdots & & \vdots & \vdots & \end{bmatrix}$$

W.L.O.G. we look at an arbitrary row i (where $i = k_2 i_1 + i_2$) of this matrix, and note that after each n_2 numbers, the following n_2 numbers will be placed in the row below. Therefore this singular row will add the following to the codeword that would be created:

$$\begin{bmatrix} a_{i_1,1}b_{i_2,1} & a_{i_1,1}b_{i_2,2} & \cdots & a_{i_1,1}b_{i_2,n_2} \\ a_{i_1,2}b_{i_2,1} & a_{i_1,2}b_{i_2,2} & \cdots & a_{i_1,2}b_{i_2,n_2} \\ \vdots & \vdots & & \vdots \\ a_{i_1,n_1}b_{i_2,1} & a_{i_1,n_1}b_{i_2,2} & \cdots & a_{i_1,n_1}b_{i_2,n_2} \end{bmatrix}$$

From this representation we see that each row of $G_A \otimes G_B$ adds exactly a codeword from A, multiplied by a number to the columns. Therefore when we choose any linear combination of $G_A \otimes G_B$, each row adds only codewords of A to the columns, and therefore each column in $A \otimes B$ is a codeword of A. This same argument holds for the rows, and can also be seen in the representation

□

7 The product code $C_m \otimes C_n$

Definition 7.1. *We let $x \in \mathbb{F}_q^{m\times n}$. Define r in $\mathbb{F}_q^m$ and c in $\mathbb{F}_q^n$ as follows:*

$$r_i(x) = \sum_{j=1}^{n} x_{ij}, \text{ the syndrome of the i-th row of } x.$$
$$c_j(x) = \sum_{i=1}^{m} x_{ij}, \text{ the syndrome of the j-th column of } x.$$

Lemma 7.1. *The sum of the rows of x equals the sum of the columns of x. Hence:*

$$\sum_{j=1}^{n} c_j(x) = \sum_{j=1}^{n}\sum_{i=1}^{m} x_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} = \sum_{i=1}^{m} r_i(x)$$

In the binary case, C_m and C_n are the even weight codes of length m and n respectively. For each code $x \in \mathbb{F}_2^{m\times n}$ we only have to look at $I_1(x)$ and $J_1(x)$, hence we will define the following:

$$I(x) := I_1(x) = \{i| \text{ the number of 1's in row i of x is odd}\};$$
$$J(x) := J_1(x) = \{j| \text{ the number of 1's in column j of x is odd}\}.$$

Proposition 7.1. *Let $x \in \mathbb{F}_q^{m\times n}$, then:*

$$\sum_{\alpha\in\mathbb{F}_q} |I_\alpha(x)|\alpha = \sum_{\beta\in\mathbb{F}_q} |J_\beta(x)|\beta$$

Proof. W.L.O.G. we can assume that $|I_0(x)| = |J_0(x)| = 0$. We start off by using lemma 7.1.

$$\sum_{i=1}^{m} r_i(x) = \sum_{i\in\mathbb{F}_q} r_i(x)$$

Next we note that the order in which we add each c_j does not change the sum, and that there are $|I_\alpha|$ rows with a syndrome of α. Therefore we can add all these values α in order to get:

$$\sum_{j\in\mathbb{F}_q} r_i(x) = \sum_{\alpha\in\mathbb{F}_q} |I_\alpha(x)|\alpha$$

Now the same reasoning for the columns holds and we get:

$$\begin{aligned}\sum_{j=1}^{n} c_j(x) &= \sum_{j\in\mathbb{F}_q} c_j(x)\\ &= \sum_{\beta\in\mathbb{F}_q} |J_\beta(x)|\beta\end{aligned}$$

And we conclude:

$$\sum_{\alpha\in\mathbb{F}_q} |I_\alpha(x)|\alpha = \sum_{j\in\mathbb{F}_q} r_i(x) = \sum_{j=1}^{n} c_j(x) = \sum_{\beta\in\mathbb{F}_q} |J_\beta(x)|\beta$$

□

Lemma 7.2. *For each $x \in \mathbb{F}_2^{m\times n}$, it holds that*

$$wt(x) \equiv |I(x)| \equiv |J(x)| \text{ (mod 2)}.$$

Proof. We will look at x row wise, and note that there are two cases.
Case 1: Row i has an odd amount of 1's. In this case, since we work in modulus 2, row i with an odd amount of 1's adds 1 to $\text{wt}(x)$, and the syndrome of this row is 1, hence there is also an element added to $I(x)$;
Case 2: Row i has an even amount of 1's. In this case, row i adds an even amount to $\text{wt}(x)$, which is reduced to 0 in modulus 2. Row i also has an even syndrome, which is also reduced to 0 in modulus 2, hence row i is not added to $I(x)$.
Therefore all rows adds an equal amount to the left and right hand side of the equation. The proof for $|J(x)|$ is similar.

□

Definition 7.2. *Let $x \in \mathbb{F}_q^{m\times s}$ and $\alpha, \beta \in \mathbb{F}_q$. Then we define the following:*

$$I_\alpha(x) = \{i | r_i = \alpha\}$$
$$J_\beta(x) = \{j | c_j = \beta\}$$

Lemma 7.3. *Let $x \in \mathbb{F}_q^{m\times n}$, then:*

$$\left\{I_\alpha | \alpha \in \mathbb{F}_q\right\} \text{ is a partition of } \{1, ..., m\};$$
$$\left\{J_\beta | \beta \in \mathbb{F}_q\right\} \text{ is a partition of } \{1, ..., n\}.$$

Proof. We let $S = \{\alpha_1, \alpha_2, ..., \alpha_k\}$ be the set of row syndromes of x. Every row i in x has a unique syndrome $r_i(x)$, and hence $\forall \alpha_i \neq \alpha_j \in S \Rightarrow I_{\alpha_i} \cap I_{\alpha_j} = \emptyset$. Furthermore it is clear that every row has a syndrome that is an element of $\mathbb{F}_q$, hence

$$\bigcup_{\alpha\in S} I_\alpha(x) = \{1, ..., m\}.$$

From this it follows that $\{I_\alpha | \alpha \in S\}$ is a partition of $\{1, ..., m\}$, and thus $\left\{I_\alpha | \alpha \in \mathbb{F}_q\right\}$ is also a partition of $\{1, ..., m\}$.

□

We now want to say something about the weights of cosets $x + C_m \otimes C_n$. However this proves to be quite difficult, for there are $q^{(m-1)(n-1)}$ elements in each coset of length mn. Which means there are $q^{mn-(m-1)(n-1)} = q^{m+n-1}$ cosets. All that we know for sure, is that the weight of each coset, not equal to $C_m \otimes C_n$, must be at least 1.

Remark 7.1. Let the $\mathbb{F}_2^m$-linear code C have generator matrix G_m, and the $\mathbb{F}_2^n$-linear code D have generator matrix G_n. Then the generator matrix of the product code $C \otimes D$ is given by the Kronecker product of the generator matrices G_m and G_n.

$$G_m \otimes G_n = \begin{bmatrix} G_n & 0 & \cdots & 0 & 0 & G_n \\ 0 & G_n & \ddots & 0 & 0 & G_n \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & G_n & 0 & G_n \\ 0 & 0 & \cdots & 0 & G_n & G_n \end{bmatrix}$$

From here we note that $G_m \otimes G_n = [mn, (m-1)(n-1), 4]_2$. The codewords that will be generated are words of length mn, but we can cut this up in m codewords of length n and put them in a matrix where the first row would be the first n digits, the second row would be the second n digits, etc.

Lemma 7.4. *We let $x \in \mathbb{F}_q^{m\times n}$. Then for the weight of the coset leader of $x + C_m \otimes C_n$, it holds that:*

$$wt(x + C_m \otimes C_n) \geq max\left\{\sum_{\alpha\in\mathbb{F}_q^*} |I_\alpha(x)|, \sum_{\beta\in\mathbb{F}_q^*} |J_\beta(x)|\right\}$$

Proof. For each $\alpha, \beta \in \mathbb{F}_q^*$, we note that according to lemma 7.3 $I_\alpha(x)$ and $J_\beta(x)$ are elements of partitions of $\{1, ..., m\}$ and $\{1, ..., n\}$ respectively. Then for each $\alpha \in \mathbb{F}_q^*$, there are $|I_\alpha(x)|$ rows that need at least one position to be changed such that the row syndrome becomes zero. Also for each $\beta \in \mathbb{F}_q^*$ there are $|J_\beta(x)|$ columns that need at least one position to change such that the column syndrome becomes zero.
From these two conditions it follows that at least $\max\left\{\sum_{\alpha\in\mathbb{F}_q^*} |I_\alpha(x)|, \sum_{\beta\in\mathbb{F}_q^*} |J_\beta(x)|\right\}$ positions need to be changed such that both the row and column syndromes of x are equal to zero. Or in other words the weight of the coset leader is at least $\max\left\{\sum_{\alpha\in\mathbb{F}_q^*} |I_\alpha(x)|, \sum_{\beta\in\mathbb{F}_q^*} |J_\beta(x)|\right\}$.
□

7.1 Binary case

Proposition 7.2. *Let $x \in \mathbb{F}_q^{m\times n}$, so not necessarily in $C_m \otimes C_n$. Let $r_i = r_i(x)$ and $c_j = c_j(x)$. We now let $y \in \mathbb{F}_q^{m\times n}$ as follows:*

$$\begin{bmatrix} y_{11} & c_2 & c_3 & \cdots & c_n \\ r_2 & 0 & 0 & \cdots & 0 \\ r_3 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_m & 0 & 0 & \cdots & 0 \end{bmatrix}$$

Where

$$y_{11} = c_1 + r_1 - \sum_{i=1}^{m} r_i$$

Then x and y have the same coset of $C_m \otimes C_n$, and:

$$wt(y) = \begin{cases} wt(r) + wt(c) & \text{, if } r_1 \equiv 0 \wedge c_1 \equiv 0, \text{ and } y_{11} \equiv 0; \\ wt(r) + wt(c) + 1 & \text{, if } r_1 \equiv 0 \wedge c_1 \equiv 0, \text{ and } y_{11} \not\equiv 0; \\ wt(r) + wt(c) - 1 & \text{, if } r_1 \equiv 0 \vee\!\vee\, c_1 \equiv 0, \text{ and } y_{11} \equiv 0; \\ wt(r) + wt(c) & \text{, if } r_1 \equiv 0 \vee\!\vee\, c_1 \equiv 0, \text{ and } y_{11} \not\equiv 0; \\ wt(r) + wt(c) - 2 & \text{, if } r_1 \not\equiv 0 \wedge c_1 \not\equiv 0, \text{ and } y_{11} \equiv 0; \\ wt(r) + wt(c) - 1 & \text{, if } r_1 \not\equiv 0 \wedge c_1 \not\equiv 0, \text{ and } y_{11} \not\equiv 0. \end{cases}$$

Where $\vee\!\vee$ is the exclusive or symbol.

To simplify this, we will define $r' = \begin{bmatrix} r_2 \\ r_3 \\ \vdots \\ r_m \end{bmatrix}$ and $c' = \begin{bmatrix} c_2 & c_3 & \cdots & c_n \end{bmatrix}$ and note that for y it holds that

$$y = \begin{bmatrix} y_{11} & c' \\ r' & 0 \end{bmatrix}$$

And hence

$$wt(y) = wt(r') + wt(c') + wt(y_{11})$$

Proof. First we note that

$$\begin{aligned} r_1(y) &= y_{11} + \sum_{j=2}^{n} c_j \\ &= c_1 + r_1 - \sum_{i=1}^{m} r_i + \sum_{j=2}^{n} c_j \\ &= r_1 - \sum_{i=1}^{m} r_i + \sum_{j=1}^{n} c_j \\ &= r_1 \end{aligned} \qquad \begin{aligned} c_1(y) &= y_{11} + \sum_{j=2}^{m} r_i \\ &= c_1 + r_1 - \sum_{i=1}^{m} r_i + \sum_{j=2}^{m} r_i \\ &= c_1 - \sum_{i=1}^{m} r_i + \sum_{j=1}^{m} r_i \\ &= c_1 \end{aligned}$$

It follows that $r_i(x-y) \equiv c_j(x-y) \equiv 0 \pmod{q}$, and hence $x - y \in C_m \otimes C_n$. From this it follows that x and y have the same coset, else they would be in a different coset of $\mathbb{F}_q^{m\times n}$.
For the weight of y, we know that it is equal to the weight of r and c so starting off with the equation $wt(y) = wt(r) + wt(c)$ is a good start. However we note that at the location of y_{11} both r_1 and c_1 are taken into account, as well as the sum $\sum_{i=1}^{n} r_i$. It is not too hard to see we are able to create all of these three variables however we want, and hence we can create $2^3 = 8$ different cases since all three of them either do, or do not, equal zero. We now look at what these three variables add to the weights of r, c and y, for each case, and note that in some cases only adding the weights of c and y is not enough, or too much. It is not hard to see we end up at the given answer.

□

When we have a product code $C_m \otimes C_n$, and a codeword $x \in \mathbb{F}_q^{m\times n}$, it is in fact not so hard to figure out, if x is also in $C_m \otimes C_n$. The only thing that you'll have to check, is if all rows of x are elements of C_n, and if all columns of x are elements of C_m.

Lemma 7.5. *In the binary case it holds that for each code $x \in C_m \otimes C_n$,*

$$r_i(x) = c_j(x) = 0,\ \forall i \in \mathbb{F}_m,\ j \in \mathbb{F}_n.$$

Proof. $C_m \otimes C_n$ has generator matrix G, where all row vectors g_i have weight 4, but syndrome of $1+1+1+1 = 0$, in $\mathbb{F}_2$. The rows of x are linear combinations of these g_i's, therefore must have $r_i(x) = 0$ in $\mathbb{F}_2$. For the columns of this code it holds that each of the g_i's add the same n digits to the last row of the codeword, as it adds to any other row. The last column of C_n's makes sure of that. Hence every g_i that is added, adds either 0 or $1+1=$ in $\mathbb{F}_2$ to the column syndromes. Hence $c_j(x) = 0$ in $\mathbb{F}_2$.

□

Remark 7.2. From the generated words of $C_m \otimes C_n$, we can see that in all of the m rows, the syndrome must be 0. Hence all digits in the rows combined must add up to 0. This results in the first m rows of the parity check matrix. We also note that the column syndrome of the codeword must equal 0, and this results in the final n rows of the parity check matrix. This all results in a parity check matrix of size $mn \times (m+n)$, in the form:

$$\begin{bmatrix} \underbrace{1\cdots1}_{n} & 0 & \cdots & 0 & 0 \\ 0 & \underbrace{1\cdots1}_{n} & \ddots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \ddots & \underbrace{1\cdots1}_{n} & 0 \\ 0 & 0 & \cdots & 0 & \underbrace{1\cdots1}_{n} \\ I_n & I_n & \cdots & I_n & I_n \end{bmatrix}$$

Lastly we note that the last row of this matrix is dependant of all of the other $m+n-1$ rows, so we can reduce this to:

$$H = \begin{bmatrix} \underbrace{1\cdots1}_{n} & 0 & \cdots & 0 & 0 \\ 0 & \underbrace{1\cdots1}_{n} & \ddots & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \ddots & \underbrace{1\cdots1}_{n} & 0 \\ 0 & 0 & \cdots & 0 & \underbrace{1\cdots1}_{n} \\ \begin{bmatrix} I_{n-1} & 0 \end{bmatrix} & \begin{bmatrix} I_{n-1} & 0 \end{bmatrix} & \cdots & \begin{bmatrix} I_{n-1} & 0 \end{bmatrix} & \begin{bmatrix} I_{n-1} & 0 \end{bmatrix} \end{bmatrix}$$

Where the 0 in the last row, is the 0 column-vector of length $n-1$.

Lemma 7.6. *Every coset leader of $x + C_m \otimes C_n$ is in one-to-one correspondence with a pair (I, J), such that $I \subseteq \{1, ..., m\}$, $J \subseteq \{1, ..., n\}$, and $|I| \equiv |J|$ mod 2.*

Proof. We let $x, y \in \mathbb{F}_2^{m\times n}$ with the same syndromes such that $I(x) = I(y)$ and $J(x) = J(y)$. From this it follows that all row and column syndromes of $x - y$ are zero, hence $x - y \in C_m \otimes C - n$, or in other words, x and y are in the same coset.
Now we suppose we have subsets I and J of $\{1, ..., m\}$ and $\{1, ..., n\}$ respectively, where $|I| \equiv |J|$ mod 2. We assume that the following holds for some $r, s \in \mathbb{N}$:

$$I = \{i_1, i_2, ..., i_r\}.$$
$$J = \{j_1, j_2, ..., j_s\}.$$

By lemma 7.2 we know that $|I| \equiv |J|$ mod 2, and hence that $|I| - |J| = r - s \equiv 0$ mod 2. Now we note that there are several different cases, and for each we create a different $e \in \mathbb{F}_2^{m\times n}$, such that for the chosen $x \in \mathbb{F}_2^{m\times n}$ it holds that $x + e \in C_m \otimes C_n$. The first case is easy: if both r and s are zero, all syndromes are zero, hence the corresponding codewords x are in the coset $C_m \otimes C_n$ already, therefore $e = 0$. For all other cases we will define:

$$e = \begin{cases} e_{i_k,j_k} = 1 \text{ for } k = 1, ..., r,\ e_{i_1,j_l} = 1 \text{ for } l = r+1, ..., s & \text{, if both } r, s \geq 1, \text{ and } r < s; \\ e_{i_k,j_k} = 1 \text{ for } k = 1, ..., s,\ e_{i_l,j_1} = 1 \text{ for } l = r+1, ..., s & \text{, if both } r, s \geq 1, \text{ and } r > s; \\ e_{i_k,j_k} = 1 \text{ for } k = 1, ..., r & \text{, if both } r, s \geq 1, \text{ and } r = s; \\ e_{1,j_k} = 1 \text{ for } k = 1, ..., s & \text{, if } r = 0 \text{ and } s \neq 0; \\ e_{i_k,1} = 1 \text{ for } k = 1, ..., r & \text{, if } r \neq 0 \text{ and } s = 0. \end{cases}$$

We note that since $r - s \equiv 0$ mod 2 for all cases it holds that $I(e) = I$ and $J(e) = J$. Hence e is in the same coset as $x + C_m \otimes C_n$, and therefore $x + e \in C_m \otimes C_n$.

□

Proposition 7.3. *The weight of the coset $x + C_m \otimes C_n$ is equal to $max\{|I(x)|, |J(x)|\}$.*

Proof. From lemma 7.6, it follows that for two pairs (I_1, J_1), and(I_2, J_2), that both satisfy the conditions, it holds that the weight of their corresponding cosets must be equal, if $|I_1| = |I_2|$ and $|J_1| = |J_2|$.
We note that for the coset leader weight enumerator, the cosets themselves are not important, but only their weights. Thus we are allowed to interchange rows and columns of e, to find

different cosets with the same weight. We can then choose to exchange rows and columns in such a way that $I = \{1, ..., r\}$, and $J = \{1, ..., s\}$. Then e will have the following form:

$$\begin{cases} \begin{array}{|c|c|} \hline 1 & 1 \ \cdots \ 1 \\ \ddots & \\ 1 & \\ \hline \end{array} & \text{, if } |I| \leq |J| \\ \\ \begin{array}{|c|} \hline 1 \\ \ddots \\ 1 \\ \hline 1 \\ \vdots \\ 1 \\ \hline \end{array} & \text{, if } |I| > |J|. \end{cases}$$

Lastly we note that e is of minimal weight, since either each row or each column in e has at most one 1. It is also not possible to have less 1's, for that would alter the sets I and J.
e is the coset leader of $x + C_m \otimes C_n$, and it's weight is $\max\{|I(x)|, |J(x)|\}$.

□

Remark 7.3.

$$\rho(C_m \otimes C_n, \mathbb{F}_2) = max\{m, n\}$$

Proof. From definition 5.5, this remark follows quite easily.

$$\rho(C_m \otimes C_n, \mathbb{F}_2) = \max\{i | \alpha_i(C_m \otimes C_n) \neq 0\}$$

Now we note that from proposition 7.3 it follows that there are indeed cosets with weight m and n, but no cosets with weight more than $\max\{m, n\}$. Therefore:

$$\begin{aligned} \rho(C_m \otimes C_n, \mathbb{F}_2) &= \max\{i | \alpha_i(C_m \otimes C_n) \neq 0\} \\ &= \max\{m, n\} \end{aligned}$$

□

Proposition 7.4. *The number of coset leaders of the product code $C_m \otimes C_n$ of weight t, is given as follows:*

$$\alpha_t(x + C_m \otimes C_n) = \sum_{\substack{r \equiv t \bmod 2 \\ 0 \leq r \leq t}} \binom{m}{r}\binom{n}{t} + \sum_{\substack{s \equiv t \bmod 2 \\ 0 \leq s < t}} \binom{m}{t}\binom{n}{s}$$

Proof. By proposition 7.3, we know that the weight of a coset $x + C_m \otimes C_n$ is equal to $\max\{|I(x)|, |J(x)|\}$. From this it simply follows that if a coset $x + C_m \otimes C_n$ has weight t, at least one or both of $|I(x)|$ and $|J(x)|$ must be equal to t, while not exceeding t. When keeping in mind the conditions of lemma 7.6, we can then choose the elements in I and J. Therefore there are several cases:
Case 1: $|I| = |J| = t$. In this case we have to choose t elements out of m for the set I, and also choose t elements out of n for the set J. Therefore the total amount of possibilities becomes:

$$\binom{m}{t}\binom{n}{t}$$

Case 2: $|I| = t$, $|J| < t$. Choosing t elements for the set I goes in the same way as before, but for the set J we now have to keep in mind it can only have sized that are equivalent to t mod 2. Therefore the total amount of possibilities becomes:

$$\binom{m}{t}\binom{n}{t-2} + \binom{m}{t}\binom{n}{t-4} + ... = \sum_{\substack{s \equiv t \bmod 2 \\ 0 \leq s < t}} \binom{m}{t}\binom{n}{s}$$

Case 3: $|I| < t$, $|J| = t$. This is the same case as case 2, but the other way around. Therefore the total amount of possibilities becomes:

$$\binom{m}{t-2}\binom{n}{t} + \binom{m}{t-4}\binom{n}{t} + ... = \sum_{\substack{r \equiv t \bmod 2 \\ 0 \leq r < t}} \binom{m}{r}\binom{n}{t}$$

Next we note that case 1 can be added to the summation in either cases, but we choose to add it to case 3, such that it results in:

$$\begin{aligned}\binom{m}{t}\binom{n}{t} + \sum_{\substack{s \equiv t \bmod 2 \\ 0 \leq s < t}} \binom{m}{t}\binom{n}{s} + \sum_{\substack{r \equiv t \bmod 2 \\ 0 \leq r < t}} \binom{m}{r}\binom{n}{t} \\ = \sum_{\substack{s \equiv t \bmod 2 \\ 0 \leq s < t}} \binom{m}{t}\binom{n}{s} + \sum_{\substack{r \equiv t \bmod 2 \\ 0 \leq r \leq t}} \binom{m}{r}\binom{n}{t}\end{aligned}$$

□

7.2 Ternary case

In this section we have a look at the case when we work in the field $\mathbb{F}_3^{m \times n}$. We no longer only look at i_1 and j_1, but also i_2 and j_2, since we are only interested in the non-zero syndromes.

Lemma 7.7. *Let $x \in \mathbb{F}_3^{m \times n}$, such that either both $i_1 \leq j_1$ and $i_2 \leq j_2$, or $j_1 \leq i_1$ and $j_2 \leq i_2$. Then*

$$wt(x + C_m \otimes C_n) = \begin{cases} j_1 + j_2 & \text{, if } i_1 \leq j_1 \wedge i_2 \leq j_2; \\ i_1 + i_2 & \text{, if } j_1 \leq i_1 \wedge j_2 \leq i_2. \end{cases}$$

Proof. We will create a matrix $e \in \mathbb{F}_3^{m \times n}$, in the same was as we did in proposition 7.3, but now with the sets I_1, I_2, J_1 and J_2. W.L.O.G. we choose to prove the first case, and hence $i_1 \leq j_1$ and $i_2 \leq j_2$ must hold. For I_1 and J_1 we run the algorithm of creating e once, and then run it again with the sets I_2 and J_2, but now we fill in 2's on the designated locations. After doing this, we can rearrange the rows and columns Note that the coset will change when we rearrange these, but the weight will not be changed since the position of numbers has nothing to do with the weight of the coset. We end up with a matrix of the form:

1 $\ddots$ 1	$1 \cdots 1$		$2 \cdots 2$
		2 $\ddots$ 2	

Lemma 7.1 now gives us the equivalent relation $i_1 + 2i_2 \equiv j_1 + 2j_2$, and thus $i_1 - j_1 + 2i_2 - 2j_2 \equiv 0$.
What we want is that there are i_1 rows with syndrome 1, i_2 rows with syndrome 2, etc. So

we say that the upper left identity matrix has i_1 1's, and to its right must therefore be the remaining $j_1 - i_1$ 1's. Also the identity matrix with entries 2, will be of size i_2, therefore the horizontal line of 2's will be of length $j_2 - i_2$. And it follows that the first row syndrome equals to $1 + (j_1 - i_1) + 2(j_2 - i_2) \equiv 1$. Hence the syndromes are correct. Hence the weight of this matrix is $j_1 + j_2$. The proof is similar for the case where $j_1 \leq i_1 \wedge j_2 \leq i_2$.

□

However it is not hard to find an $x \in \mathbb{F}_3^{m \times n}$ such that it is not in one of these 2 cases. Therefore we have to continue with other cases.
For example we note that we still miss the cases where $i_1 = j_2 = 0$ or $i_2 = j_1 = 0$.

Lemma 7.8.

$$wt(x + C_m \otimes C_n) = \begin{cases} j_1 & , \text{if } i_1 = j_2 = 0 \ \wedge \ 2i_2 \leq j_1; \\ j_2 & , \text{if } i_2 = j_1 = 0 \ \wedge \ 2i_1 \leq j_2; \\ i_2 & , \text{if } i_1 = j_2 = 0 \ \wedge \ 2j_1 \leq i_2; \\ i_1 & , \text{if } i_2 = j_1 = 0 \ \wedge \ 2j_2 \leq i_1. \end{cases}$$

Proof. W.L.O.G. We focus on only the first case, since all others have a similar proof.From lemma 7.1 it follows directly that $2i_2 \equiv j_1 \bmod 3$.
We now suppose $2i_2 \leq j_1$, and note that $j_1 > j_2 = 0$, since we would be in a case in lemma 7.7 if $j_1 = 0$. We also know from lemma 7.4 that the weight of this coset is at least j_1. Since $j_1 \geq 2i_2$ and $j_1 \equiv 2i_2 \bmod 3$, the following must hold:

$$\begin{aligned} &j_1 \equiv 2i_2 \bmod 3; \\ &j_1 = 2i_2 + 3k, \text{ for some } k \in \mathbb{N}^+. \end{aligned}$$

k must be greater or equal to 0, since the conditions state that $j_1 \geq 2i_2$. This makes sense, for with these numbers it is possible to create a matrix e that again satisfies the conditions for being a codeword. All we have to do is create a matrix the same syndromes, and we note that once again the position does not matter since we only want to find the exact weight, and not the codeword. Therefore we create e as follows:
$e_{i,2i-1} = 1$, and $e_{i,2i} = 1$, for $1 \leq i \leq i_2$, and $e_{i_2,j} = 1$, for $2i_2 < j \leq j_1$. It follows that e has the following form:

$$\left[\begin{array}{cccccc|ccc} 1 & 1 & & & & & & & \\ & & 1 & 1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & 1 & 1 & 1 \ \cdots & 1 \end{array}\right]$$

There are indeed j_1 columns with exactly one 1's, $i_2 - 1$ rows with exactly two 1's, and one row with $2 + j_1 - 2i_2 = 2 + 3k$ 1's, and hence also this syndrome equals 2. We conclude that the weight of this coset is indeed j_1.

The proof is similar for the three other cases.

□

Lemma 7.9.

$$wt(x + C_m \otimes C_n) = \begin{cases} \frac{2}{3}(i_2 + j_1) & , \text{if } i_1 = j_2 = 0, \ j_1 \leq 2i_2 \text{ and } i_2 \leq 2j_1; \\ \frac{2}{3}(i_1 + j_2) & , \text{if } i_2 = j_1 = 0, \ j_2 \leq 2i_1 \text{ and } i_1 \leq 2j_2. \end{cases}$$

Proof. We only proof the first case. Let $e \in \mathbb{F}_3^{m \times n}$ be an element of minimal weight in its coset with respect to the set (I_2, J_1). W.L.O.G. we can claim $|I_2| = m$ and $|J_1| = n$, for there are only rows with syndrome 0 or 2, hence we can add rows of syndrome 0 to any other row without

changing anything, and the same holds for the columns.
It is impossible to have both a 1 and a 2 in the same row or column. If that would be the case, we note that underneath the 2 there must either be at least a 1 or a 2, since it is not possible to have a column syndrome of 2 in this case. We now take the original row with a 1 and a 2, and the row that contains the 1 or the 2 and see it has the following form:

$$\boxed{\begin{matrix}1 & 2\\ * & 1\end{matrix}} \text{ or } \boxed{\begin{matrix}1 & 2\\ * & 2\end{matrix}}$$

Where $*$ can be any number. We can add the codeword $\boxed{\begin{matrix}2 & 1\\ 1 & 2\end{matrix}}$ to get a strictly lower weight, while we don't change the syndromes, and therefore the coset. Hence there can only be 0's and 1's, or 0's and 2's in the same row or column. Now we define:

$$\begin{aligned}
I_{2,1} &= \{i|\text{row } i \text{ has only 0's and 1's}\};\\
I_{2,2} &= \{i|\text{row } i \text{ has only 0's and 2's}\};\\
J_{1,1} &= \{j|\text{column } j \text{ has only 0's and 1's}\};\\
J_{1,2} &= \{j|\text{column } j \text{ has only 0's and 2's}\}.
\end{aligned}$$

From this it follows that all above sets are distinct, and therefore it holds that $I_2 = I_{2,1} \bigsqcup I_{2,2}$ and $J_1 = J_{1,1} \bigsqcup J_{1,2}$. Here $\bigsqcup$ denotes the union of two disjoint sets.
We see that for all $i \in I_{2,1}$ there must be two distinct numbers $j, k \in J_{1,1}$ for which we can claim that $j < k$, for which it holds that there must be a 1 on the coordinates (i, j) and (i, k). Hence the size of the set $\{j|x_{i,j} = 1\}$ is equal to 2 mod 3. We also know, that for all $j \in J_{1,1}$ the size of the set $\{i|x_{i,j} = 1\}$ is equal to 1 mod 3. And we note that for e it is also impossible to have more than one 1 in a column, for if it did it must have at least four in the column, and 2 in each of those four corresponding rows. These four rows do not have to equal each other, and can therefore be, for example of the form:

$$\boxed{\begin{matrix}1 & 1 & & & \\ 1 & * & 1 & & \\ 1 & & & 1 & \\ 1 & & & ** & 1\end{matrix}}$$

Here $*$ and $**$ can be any number, but need to be reminded later on, and all empty spots can be any number as well. Now we want to reduce this to its minimal form, and for that we will focus on a smaller part of this submatrix, namely: $\boxed{\begin{matrix}1 & 1\\ 1 & *\end{matrix}}$. We will add $\boxed{\begin{matrix}1 & 2\\ 2 & 1\end{matrix}}$ to remain in the same coset, while decreasing its weight. Therefore in this case, what will happen is the following:

$$\boxed{\begin{matrix}1 & 1 & & & \\ 1 & * & 1 & & \\ 1 & & & 1 & \\ 1 & & & ** & 1\end{matrix}} + \boxed{\begin{matrix}1 & 2 & & \\ 2 & 1 & & \\ 1 & & & 2\\ 2 & & & 1\end{matrix}} = \boxed{\begin{matrix}2 & & 0 & & & \\ 0 & *+1 & 1 & & & \\ 2 & & & & 0 & \\ 0 & & & & **+1 & 1\end{matrix}}$$

And we see that we have reduced the weight, without changing the coset. Depending on $*$ and $**$ we now have rows and columns that are either elements of $I_{2,1}$, $I_{2,2}$, $J_{1,1}$ or $J_{1,2}$, or we have to again add a submatrix with only syndromes that equal 0 to change this as seen before. Hence each column $j \in J_{1,1}$ only has one 1. And since the rows of $I_{2,1}$ must have at least two 1's, it holds that $|J_{1,1}| \geq 2|I_{2,1}|$. Similarly it also holds that $|I_{2,2}| \geq 2|J_{1,2}|$. And since there can be no 1's and 2's in the same row or column, it also must hold that $wt(e) \geq |J_{1,1}| + |I_{2,2}|$.
From the above conditions it also must follow that

$$\sum e_{i,j} = |J_{1,1}| + 2|I_{2,2}|$$

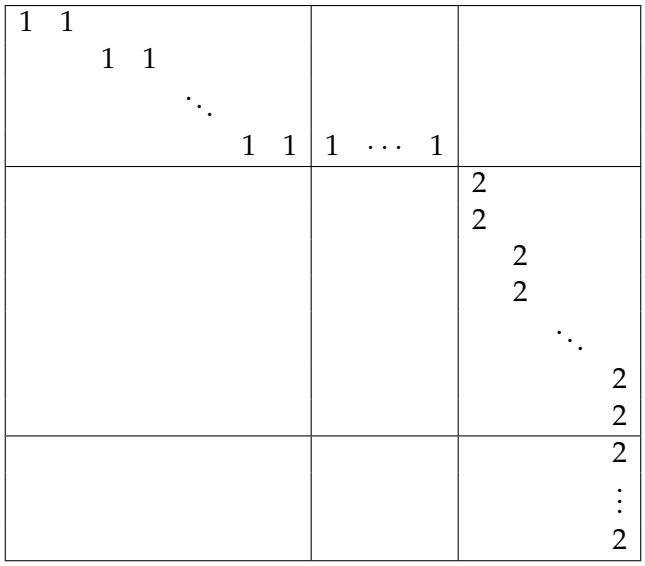
And therefore, when only counting the weight, it follows that:

$$\begin{aligned} wt(e) &= |J_{1,1}| + |I_{2,2}| \\ &= 2|I_{2,1}| + |I_{2,2}| \\ &= |J_{1,1}| + 2|J_{1,2}| \end{aligned}$$

Thus we have that $|J_{1,1}| = 2|I_{2,1}|$ and $|I_{2,2}| = 2|J_{1,2}|$. The following two equations arise:

$$\begin{aligned} |J_{1,1}| - 2|I_{2,1}| &\equiv 0 \bmod 3; \\ |I_{2,2}| - 2|J_{1,2}| &\equiv 0 \bmod 3. \end{aligned}$$

We now now the form of e, and it will be shown in the following figure:



We now let $x = |I_{2,2}|$, and $y = |J_{1,1}|$, and we recall that $|I_2| = m = |I_{2,1}| + |I_{2,2}|$ and $|J_1| = n = |J_{1,1}| + |J_{1,2}|$. Then the following holds:

$$\begin{aligned} y = |J_{1,1}| \geq 2|I_{2,1}| = 2|I_2| - 2|I_{2,2}| = 2m - 2x; \\ x = |I_{2,2}| \geq 2|J_{1,2}| = 2|J_1| - 2|J_{1,1}| = 2n - 2y. \end{aligned}$$

Now we have to find the values of x and y, such that the weight of e is minimal. Hence we want to find values of x and y, such that we are solving the following linear programming problem:

$$min\ x + y$$

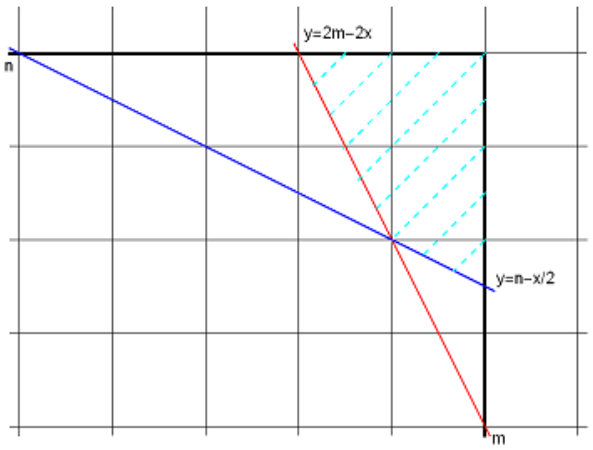
with the constraints:

$$\begin{aligned} &2m \leq y + 2x,\ 0 \leq y \leq n; \\ &2n \leq x + 2y,\ 0 \leq x \leq m; \\ &x, y \in \mathbb{Z}. \end{aligned}$$

We can rewrite these constraints to equations, such that they will result in two different lines.

$$\begin{aligned} y &= 2m - 2x; \\ y &= n - \frac{1}{2}x. \end{aligned}$$

Next up we note that the area we are working with, has the form shown in the following figure:



It is visible that the easiest corner of the area is located at $x = m, y = n$, and therefore this has weight $m + n$. We also note that for $x = m$ there is a second point at a different value for y. In order to find this, we simply fill in $x = m$ in the constraint $2n \leq x + 2y$, and we get:

$$\begin{aligned} 2n &\leq x + 2y \\ 2n &\leq m + 2y \\ 2y &\geq 2n - m \\ y &\geq n - \frac{1}{2}m \end{aligned}$$

Hence the corner point $x = m, y = n - \frac{1}{2}m$ has weight $\frac{1}{2}m + n$. Analogue it follows that $x = m - \frac{1}{2}n, y = n$ is a corner point with weight $m + \frac{1}{2}n$. For the fourth corner point we note that it is the intersection of the two lines, and therefore we simply equal them.

$$\begin{aligned} 2m - 2x &= n - \frac{1}{2}x \\ 2m - n &= \frac{3}{2}x \\ x &= \frac{2}{3}(2m - n) \\ x &= \frac{4}{3}m - \frac{2}{3}n \end{aligned}$$

Filling this in, in either one of the equations, gives us: $y = \frac{4}{3}n - \frac{2}{3}m$. This point therefore has weight: $\frac{4}{3}m - \frac{2}{3}n + \frac{4}{3}n - \frac{2}{3}m = \frac{2}{3}(m + n)$.
Now we have found the four corner points, it is clear that the point (m, n) has a higher weight than all other three points, and is therefore not the answer. Next we note that since $m \leq 2n$, the following holds:

$$\begin{aligned} m &\leq 2n \\ \frac{1}{6}m &\leq \frac{1}{3}n \\ \frac{2}{3}m &\leq \frac{1}{2}m + \frac{1}{3}n \\ \frac{2}{3}(m + n) &\leq \frac{1}{2}m + n \end{aligned}$$

Similarly from $n \leq 2m$ it follows that $\frac{2}{3}(m + n) \leq m + \frac{1}{2}n$. Hence $\frac{2}{3}(m + n)$ is the minimum value for this problem.
Lastly we note that $2i_2 \equiv j_1 \bmod 3$, and hence $2m - n \equiv 0 \bmod 3$. Therefore it must hold that

$\frac{2}{3}(2m-n) = \frac{4}{3}m - \frac{2}{3}n$ is indeed also an integer. The same argument holds that since $m \equiv 2n \mod 3$, also $\frac{4}{3}n - \frac{2}{3}m$ is an integer. Therefore the summation of these two integers gives us a new integer, and therefore we conclude that for any codeword in $\mathbb{F}_3^{m\otimes n}$ that fits the conditions, it indeed is in a coset with weight $\frac{2}{3}(i_2 + j_1)$.
The proof for the second case is analogue.

□

Example lemma 7.9
We start off with a submatrix for which the conditions hold.

1	1	1	1	1	*Syndromes*
0	1	2	0	0	0
0	1	0	1	0	2
0	0	2	0	0	2
1	2	0	0	2	2
0	0	0	0	2	2

$i_1 = j_2 = 0$, $i_2 = 4$ and $j_1 = 5$, therefore the coset leader should have weight $\frac{2}{3}(4+5) = 6$, and we note that we can indeed add codewords from $C_5 \otimes C_5$ to get the coset leader as follows.
Step 1:

1	1	1	1	1	*Syndromes*
0	1	2	0	0	0
0	1	0	1	0	2
0	0	2	0	0	2
1	2	0	0	2	2
0	0	0	0	2	2

$+$

0	0	0	0	0	*Syndromes*
0	2	1	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	1	2	0	0	0
0	0	0	0	0	0

$=$

1	1	1	1	1	*Syndromes*
0	0	0	0	0	0
0	1	0	1	0	2
0	0	2	0	0	2
1	0	2	0	2	2
0	0	0	0	2	2

Step 2:

1	1	1	1	1	*Syndromes*
0	0	0	0	0	0
0	1	0	1	0	2
0	0	2	0	0	2
1	0	2	0	2	2
0	0	0	0	2	2

$+$

0	0	0	0	0	*Syndromes*
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
2	0	0	0	1	0
1	0	0	0	2	0

$=$

1	1	1	1	1	*Syndromes*
0	0	0	0	0	0
0	1	0	1	0	2
0	0	2	0	0	2
0	0	2	0	0	2
1	0	0	0	1	2

And we have found the coset leader of weight 6, and not a single rows or columns contains both a 1 and a 2.

We have now seen several cases of how we can create the coset leader in $\mathbb{F}_3^{m\times n}$, but still there is not one closed answer on what the weight of the coset $x + C_m \otimes C_n$ is, for any arbitrary x. However all previous lemmas gave us all of the options of "building blocks" that we can use to build our coset leader v of minimal weight. We continue by creating a theorem that gives us the weight of the specific coset $x + C_m \otimes C_n$, for any element $x \in \mathbb{F}_3^{m\times n}$.

Theorem 7.1. *Let $x \in \mathbb{F}_3^{m \times n}$. We now define the following*

$$\begin{aligned} y_1 &= |i_1 - j_1| \\ y_2 &= |i_2 - j_2| \\ s &= min\left\{\left\lfloor \frac{y_1}{2} \right\rfloor, y_2\right\} \\ t &= min\{y_1 - 2s, y_2 - s\} \end{aligned}$$

Then

$$wt(x + C_m \otimes C_n) = \begin{cases} i_1 + i_2 & , \text{ if } i_1 \geq j_1 \text{ and } i_2 \geq j_2; \\ j_1 + j_2 & , \text{ if } i_1 \leq j_1 \text{ and } i_2 \leq j_2; \\ i_2 + j_1 - s - t & , \text{ if } i_1 < j_1 \text{ and } i_2 > j_2; \\ i_1 + j_2 - s - t & , \text{ if } i_1 > j_1 \text{ and } i_2 < j_2. \end{cases}$$

Proof. W.L.O.G. we can assume that $i_0 = j_0 = 0$, and therefore $i_1 + i_2 = m$ and $j_1 + j_2 = n$. From the above lemmas, we know what form the minimum weight element v must have in order to actually be the minimum weight element.
We will create an array of size $m \times n$, that has the following form:

$$\begin{array}{|cccccc|} \hline A & & & & G & H \\ & B & & & & \\ & & C & & & \\ & & & D & & \\ & & & & E & \\ & & & & & F \\ K & & & & & \\ L & & & & & \\ \hline \end{array}$$

Where A through L are all submatrices. Each of their sizes is dependant on i_1, i_2, j_1 and j_2. In order to find these sizes, we will first define each separate submatrix as follows:

$$A = \begin{array}{|ccc|} \hline 1 & & \\ & \ddots & \\ & & 1 \\ \hline \end{array} \qquad B = \begin{array}{|ccc|} \hline 2 & & \\ & \ddots & \\ & & 2 \\ \hline \end{array}$$

$$C = \begin{array}{|cccccc|} \hline 1 & 1 & & & & \\ & & 1 & 1 & & \\ & & & & \ddots & \\ & & & & & 1 \quad 1 \\ \hline \end{array}$$

$$D = \begin{array}{|cccccc|} \hline 1 & & & & & \\ 1 & & & & & \\ & 1 & & & & \\ & 1 & & & & \\ & & \ddots & & & \\ & & & & 1 & \\ & & & & 1 & \\ \hline \end{array}$$

$$E = \begin{array}{|cccccc|} \hline 2 & 2 & & & & \\ & & 2 & 2 & & \\ & & & & \ddots & \\ & & & & & 2 \quad 2 \\ \hline \end{array}$$

$$F = \begin{array}{|cccccc|} \hline 2 & & & & & \\ 2 & & & & & \\ & 2 & & & & \\ & 2 & & & & \\ & & \ddots & & & \\ & & & & 2 & \\ & & & & 2 & \\ \hline \end{array}$$

$$G = \begin{array}{|ccccc|} \hline 1 & 1 & \cdots & 1 & 1 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & & & \vdots \\ 0 & 0 & \cdots & 0 & 0 \\ \hline \end{array}$$

$$H = \begin{array}{|ccccc|} \hline 2 & 2 & \cdots & 2 & 2 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & & & \vdots \\ 0 & 0 & \cdots & 0 & 0 \\ \hline \end{array}$$

$$K = \begin{array}{|ccccc|} \hline 1 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 1 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ \hline \end{array}$$

$$L = \begin{array}{|ccccc|} \hline 2 & 0 & \cdots & 0 \\ 2 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 2 & 0 & \cdots & 0 \\ 2 & 0 & \cdots & 0 \\ \hline \end{array}$$

First we will denote their sizes with the corresponding letter, hence:

$$\begin{aligned}
&A \text{ has size } a \times a\\
&B \text{ has size } b \times b\\
&C \text{ has size } c \times 2c\\
&D \text{ has size } 2d \times d\\
&E \text{ has size } e \times 2e\\
&F \text{ has size } 2f \times f\\
&G \text{ has size } s \times g\\
&H \text{ has size } t \times h\\
&K \text{ has size } k \times u\\
&L \text{ has size } l \times v
\end{aligned}$$

Where s, t, u and v depend on the first submatrix that is not empty. For example, if A is empty, G and H move one spot down such that they arrange with B. The same holds for K and L.
We now want to minimise the total weight, and with our previous knowledge we can find a through l such that this works. We want to put as many numbers in either different rows, or different columns. Therefore we will put as many 1's in A as allowed, and hence $a = min\{i_1, j_1\}$. This same reasoning works for B, and hence $b = min\{i_2, j_2\}$. To now find c, we have to keep in mind what submatrix C adds to the syndromes. C only has rows of syndrome 2, and columns of syndrome 1. If we first look at the columns that C contributes to, $2c$ could equal at most the remaining $j_1 - a$ columns of syndrome 1, however we want to find c, and hence $j_1 - a$ must be halved, and rounded down to the nearest integer. We now look at the rows C contributes to, and note that here c can be at most $i_2 - b$. Therefore $c = min\left\{\left\lfloor \frac{|j_1-a|}{2} \right\rfloor, i_2 - b\right\}$. This same reasoning works for D, and hence $d = min\left\{\left\lfloor \frac{|i_1-a|}{2} \right\rfloor, j_2 - b\right\}$. For the value of e, we note that E contributes to both i_1, and to j_2. Following the same arguments as before we know that $e = min\{i_1 - a - 2d, j_2 - b - d\}$. This same reasoning works for F, and hence $f = min\{i_2 - b - c, j_1 - a - 2c\}$. Lastly for G, H, K and L it holds that they must top off all the remaining syndromes for their corresponding numbers, therefore all of the sizes in a list are:

$$\begin{aligned}
a &= min\{i_1, j_1\}\\
b &= min\{i_2, j_2\}\\
c &= min\left\{\left\lfloor \frac{|j_1 - a|}{2} \right\rfloor, i_2 - b\right\}\\
d &= min\left\{\left\lfloor \frac{|i_1 - a|}{2} \right\rfloor, j_2 - b\right\}\\
e &= min\{i_1 - a - 2d, j_2 - b - d\}\\
f &= min\{i_2 - b - c, j_1 - a - 2c\}\\
g &= j_1 - a - 2c - f\\
h &= j_2 - b - d - 2e\\
k &= i_1 - a - 2d - e\\
l &= i_2 - b - c - 2f
\end{aligned}$$

The problem that now arises, is that the row that contains G and H, and the column that contains K and L, obviously should not change syndromes. Therefore we will have to prove that $g + 2h \equiv k + 2l \equiv 0 \bmod 3$. We continue by only proving $g + 2h \equiv 0 \bmod 3$, since the proof goes similar for $k + 2l$.

$$\begin{aligned}
g + 2h &= j_1 - a - 2c - f + 2(j_2 - b - d - 2e)\\
&= j_1 + 2j_2 - a - 2b - 2c - 2d - 4e - f
\end{aligned}$$

From here we will simply define several cases:
Case 1: either $i_1 = j_1$, or $i_2 = j_2$.
If either one of these two holds, we can choose a or b however we like. Then $c = d = e = f = 0$, and it simply follows that:

$$\begin{aligned} g + 2h &= j_1 + 2j_2 - a - 2b \\ &= j_1 + 2j_2 - j_1 - 2j_2 \\ &= 0 \end{aligned}$$

Case 2: $i_1 < j_1$ and $i_2 < j_2$.
Here $a = i_1$ and $b = i_2$. Then $c = d = e = f = 0$ and through the same reasoning as in case 1 $g + 2h = 0$.
Case 3: $i_1 > j_1$ and $i_2 > j_2$.
Here $a = j_1$ and $b = j_2$. Then $c = d = e = f = 0$ and through the same reasoning as in case 1 and 2 $g + 2h = 0$.
Case 4: $i_1 < j_1$ and $i_2 > j_2$.
Here $a = i_1$ and $b = j_2$. Then $d = e = 0$ and $c \neq 0 \neq f$. However since both c and f could be two different numbers, we can't fill this in further and we have to work with subcases. Before we do this we note that f is dependant on c.
Case 4.1: $c = i_2 - j_2$.

$$\begin{aligned} f &= min\{i_2 - j_2 - c, j_1 - i_1 - 2c\} \\ &= min\{i_2 - j_2 - (i_2 - j_2), j_1 - i_1 - 2(i_2 - j_2)\} \\ &= min\{0, j_1 - i_1 - 2(i_2 - j_2)\} \\ &= 0 \end{aligned}$$

And from this it follows that

$$\begin{aligned} g + 2h &= j_1 - a - 2c - f + 2(j_2 - b - d - 2e) \\ &= j_1 - i_1 - 2(i_2 - j_2) \\ &= j_1 + 2j_2 - (i_1 + 2i_2) \\ &= \sum_{j=1}^{n} c_j(x) - \sum_{i=1}^{n} r_i(x) \\ &\equiv 0 \bmod 3 \end{aligned}$$

Case 4.2: $c = \left\lfloor \frac{j_1 - i_1}{2} \right\rfloor$.
In this case it is very difficult to find the value of f, since it uses a rounded down number. We also do not know anything about the values $i_2 - j_2$ and $j_1 - i_1$. However there is one last thing we have to see. f could equal $j_1 - i_1 - 2\left\lfloor \frac{j_1 - i_1}{2} \right\rfloor$, which, in turn, can only equal 0 or 1. And hence two different subcases arise.
Case 4.2.1: $f = 0$.
For f to equal 0, it must hold that $j_1 - i_1 = 2\left\lfloor \frac{j_1 - i_1}{2} \right\rfloor$, which only holds if $\left\lfloor \frac{j_1 - i_1}{2} \right\rfloor = \frac{j_1 - i_1}{2}$. Therefore it follows:

$$\begin{aligned} g + 2h &= j_1 - a - 2c - f + 2(j_2 - b - d - 2e) \\ &= j_1 - i_1 - 2\left\lfloor \frac{j_1 - i_1}{2} \right\rfloor \\ &= j_1 - i_1 - 2 \cdot \frac{j_1 - i_1}{2} \\ &= 0 \end{aligned}$$

Case 4.2.2: $f = 1$.
For f to equal 1, it must hold that $\left\lfloor \frac{j_1 - i_1}{2} \right\rfloor = \left\lfloor \frac{j_1 - i_1 - 1}{2} \right\rfloor = \frac{j_1 - i_1 - 1}{2}$. Therefore it follows:

$$\begin{aligned} g + 2h &= j_1 - a - 2c - f + 2(j_2 - b - d - 2e) \\ &= j_1 - i_1 - 2\left\lfloor \frac{j_1 - i_1}{2} \right\rfloor - 1 \\ &= j_1 - i_1 - 2 \cdot \frac{j_1 - i_1 - 1}{2} - 1 \\ &= 0 \end{aligned}$$

And therefore it holds that for the entire case 4 $g + 2h \equiv 0 \bmod 3$.
Case 5: $i_1 > j_1$ and $i_2 < j_2$.
This proof is analogue to case 4.
Therefore $g + 2h \equiv 0 \bmod 3$ in every single case.
Now we can move on to the weight of v. From the shown matrices before, we can easily find that the weight of v equals:

$$\begin{aligned} wt(r) &= a + b + 2c + 2d + 2e + 2f + g + h + k + l \\ &= i_1 + i_2 + j_1 + j_2 - a - b - c - d - e - f \end{aligned}$$

To solve this problem, we have to create separate cases.
Case 1: $i_1 < j_1$.
From this condition it immediately follows that $a = i_1$. It then follows that $d = e = 0$, and hence $wt(r) = i_2 + j_1 + j_2 - b - c - f$. In two subcases we will reduce this problem further.

Case 1.1: $i_2 \leq j_2$.
From this condition it immediately follows that $b = i_2$. It then follows that $c = f = 0$, and hence $wt(r) = j_1 + j_2$.

Case 1.2: $i_2 > j_2$.
From this condition it immediately follows that $b = j_2$. We note that both c and f are not equal to 0. So first we reduce the weight by filling in the value for b to get $wt(r) = i_2 + j_1 - c - f$. Before we fill in the values of c and f, we will first define $y_1 = |i_1 - j_1|$, $y_2 = |i_2 - j_2|$, $s = min\left\{\left\lfloor \frac{y_1}{2} \right\rfloor, y_2\right\}$, and $t = min\left\{y_2 - s, y_1 - 2s\right\}$, since that will make the notation a lot easier. We now continue calculating the weight:

$$\begin{aligned} wt(r) &= i_2 + j_1 - min\left\{\left\lfloor \frac{j_1 - i_1}{2} \right\rfloor, i_2 - j_2\right\} - min\left\{i_2 - j_2 - s, j_1 - i_1 - 2s\right\} \\ &= i_2 + j_1 - min\left\{\left\lfloor \frac{y_1}{2} \right\rfloor, i_2 - j_2\right\} - min\left\{y_2 - s, y_1 - 2s\right\} \\ &= i_2 + j_1 - s - t \end{aligned}$$

Case 2: $i_1 = j_1$.
From this condition it immediately follows that $a = i_1 = j_1$. It then follows that $c = d = e = f = 0$, and hence $wt(r) = i_1 + i_2 + j_1 + j_2 - a - b$. We do not choose the value for a just yet. In two subcases we will reduce this problem further.

Case 2.1: $i_2 < j_2$.
From this condition it immediately follows that $b = i_2$ and hence $wt(r) = i_1 + j_1 + j_2 - a$. We chose $a = i_1$, and get $wt(r) = j_1 + j_2$.

Case 2.2: $i_2 \geq j_2$.
From this condition it immediately follows that $b = j_2$, and hence $wt(r) = i_1 + i_2$. We chose $a = j_1$, and get $wt(r) = i_1 + i_2$.

Case 3: $i_1 > j_1$.
From this condition it immediately follows that $a = j_1$. It then follows that $c = f = 0$, and hence $wt(r) = i_1 + i_2 + j_2 - b - d - e$. In two subcases we will reduce this problem further.

Case 3.1: $i_2 < j_2$.
From this condition it immediately follows that $b = i_2$. We note that both d and e are not equal to 0. So first we reduce the weight by filling in the value for b to get $wt(r) = i_1 + j_2 - d - e$. Using the same y_1 and y_2 from before, we continue.

$$\begin{aligned} wt(r) &= i_1 + j_2 - min\left\{\left\lfloor \frac{i_1 - j_1}{2} \right\rfloor, j_2 - i_2\right\} - min\left\{j_2 - i_2 - s, i_1 - j_1 - 2s\right\} \\ &= i_1 + j_2 - min\left\{\left\lfloor \frac{y_1}{2} \right\rfloor, y_2\right\} - min\left\{y_2 - s, y_1 - 2s\right\} \\ &= i_1 + j_2 - s - t \end{aligned}$$

Case 3.2: $i_2 \geq j_2$.
From this condition it immediately follows that $b = j_2$. It then follows that $d = e = 0$, and hence $wt(r) = i_1 + i_2$.

We now have the following weight with the corresponding conditions:

$$wt(x + C_m \otimes C_n) = \begin{cases} j_1 + j_2 & \text{, if } i_1 < j_1 \text{ and } i_2 \leq j_2; \\ i_2 + j_1 - s - t & \text{, if } i_1 < j_1 \text{ and } i_2 > j_2; \\ j_1 + j_2 & \text{, if } i_1 = j_1 \text{ and } i_2 < j_2; \\ i_1 + i_2 & \text{, if } i_1 = j_1 \text{ and } i_2 \geq j_2; \\ i_1 + j_2 - s - t & \text{, if } i_1 > j_1 \text{ and } i_2 < j_2; \\ i_1 + i_2 & \text{, if } i_1 > j_1 \text{ and } i_2 \geq j_2. \end{cases}$$

And lastly we note the following:

$$(i_1 = j_1 \wedge i_2 \geq j_2) \wedge (i_1 > j_1 \wedge i_2 \geq j_2) = i_1 \geq j_1 \wedge i_2 \geq j_2$$

And we know that for $i_1 < j_1 \wedge i_2 < j_2$ the weight of the coset is $j_1 + j_2$, but also for $i_1 = j_1 \wedge i_2 < j_2$ and $i_1 < j_1 \wedge i_2 = j_2$. Therefore we conclude in the final answer:

$$wt(x + C_m \otimes C_n) = \begin{cases} i_1 + i_2 & \text{, if } i_1 \geq j_1 \text{ and } i_2 \geq j_2; \\ j_1 + j_2 & \text{, if } i_1 \leq j_1 \text{ and } i_2 \leq j_2; \\ i_2 + j_1 - s - t & \text{, if } i_1 < j_1 \text{ and } i_2 > j_2; \\ i_1 + j_2 - s - t & \text{, if } i_1 > j_1 \text{ and } i_2 < j_2. \end{cases}$$

Only one last thing remains. It is not difficult to see that A through F will be of minimum weight, since in any case at least two of the values c through f will equal zero. In the case that $i_1 \leq j_1$ it follows that $d = e = 0$, and hence all remaining columns in C and F have a syndrome of 1, while their row syndromes are different. Hence there is no way to distribute these numbers such that the weight becomes less. Now we note that if $g > 0$, it must hold that $a = i_1$, and therefore $k = 0$. The same holds the other way around, and for h and l. Now, to see what g means for l, we will assume that $g > 0$ and $h = 0$, since we obviously do not want l to equal zero. From these conditions we know that $g \equiv l \equiv 0 \bmod 3$. We also know that

$g + 2h \equiv k + 2l \equiv 0 \bmod 3$, and hence $g \equiv 2l \equiv 0 \bmod 3$. From these two conditions it follows that both g and l must be multiples of 3. To check if both g and l could be greater than zero at the same time, we simply run our algorithm again for the following array:

$*$	1	1	1	*syndromes*
$*$	1	1	1	$*$
2				2
2				2
2				2

Here $*$ could be any number, since G and L would be put in the same row or column of the first nonempty submatrix. However, if we now run our algorithm on this element, it would output the following matrix v:

$*$	1	1	1	*syndromes*
$*$				$*$
	1	1		2
			2	2
			2	2

And hence it will never be the case that both G and L would be nonempty. The same holds for H and K.

□

8 $\alpha_{C_3 \otimes C_3}(Z)$

We have looked at several cases so far, and noted it is very difficult to compute the general case where q can be arbitrarily chosen. Hence we will now look at a general case for the product code $C_3 \otimes C_3$ in $\mathbb{F}_q^{3\times 3}$. We say that q is a prime number bigger than 2, otherwise we would be back at the binary case. We now want to find $\alpha_{C_3\otimes C_3}(Z)$, for which we obviously have to find all of the $\alpha_i(C_3 \otimes C_3)$. Before we start we note that the weight of each coset is at most 5, since each coset must contain an element of the form described in proposition 7.2.
The easiest case is $i = 0$, since $\alpha_0(C_3 \otimes C_3) = 1$ for each q.
$\alpha_1(C_3 \otimes C_3)$: We can only change one of the nine 0's to a number between 1 and $q-1$, each of the nine positions giving us a different set of syndromes, and hence according to lemma 7.6 each one giving us a different coset. Therefore it holds:

$$\alpha_1(C_3 \otimes C_3) = 9(q-1).$$

$\alpha_2(C_3 \otimes C_3)$: Since we now have to change two different positions into numbers, λ_1 and λ_2, different than 0, it gets a bit more tricky. We split the problem in several cases.

Case 1: $\lambda_1 \neq \lambda_2$.
Case 1.1: λ_1 and λ_2 are neither in the same row, nor in the same column.
For both the row, and the column syndromes, there are 6 ways of choosing the positions for 0, λ_1 and λ_2. Therefore there are $6 \cdot 6 = 36$ different ways to distribute λ_1 and λ_2. There are $q-1$ choices for λ_1, and hence $q-2$ choices left for λ_2. We conclude that in this case there are a total of $6 \cdot 6(q-1)(q-2)$ different cosets.

Case 1.2: λ_1 and λ_2 are in the same row, and $\lambda_1 + \lambda_2 = 0$.
We can rearrange 0, λ_1 and λ_2 in 3 different ways, since we can put 0 in any of the three locations and then the λ's are put in the remaining places. We note that for the row syndrome to equal zero, we can choose λ_1 in $(q-1)$ ways, which then defines λ_2 instantly. It also does not matter in which row we place these numbers, since they will end up in the same cosets. We conclude that in this case there are a total of $3(q-1)$ different cosets.

Case 1.3: λ_1 and λ_2 are in the same row, and $\lambda_1 + \lambda_2 \neq 0$.
The column syndromes can be rearranged in 3 different ways, but this time there is one row syndrome unequal to 0, which therefore can be placed in 3 different positions. Also $\lambda_1 + \lambda_2$ should not equal zero, so if we choose λ_1 with $q-1$ possibilities, there is exactly one number, that is unequal to λ_1 since q is prime, that would add up to 0. Therefore λ_2 can be chosen from $q-3$ numbers. We conclude that in this case there are a total of $3 \cdot 3(q-1)(q-3)$ different cosets.

Case 1.4: λ_1 and λ_2 are in the same column, and $\lambda_1 + \lambda_2 = 0$.
This proof is analogous to case 1.2, therefore we conclude that in this case there are a total of $3(q-1)$ different cosets.

Case 1.5: λ_1 and λ_2 are in the same column, and $\lambda_1 + \lambda_2 \neq 0$.
This proof is analogous to case 1.3, therefore we conclude that in this case there are a total of $9(q-1)(q-3)$ different cosets.

Case 2: $\lambda_1 = \lambda_2 = \lambda$.
Case 2.1: λ_1 and λ_2 are neither in the same row, nor in the same column.
In the following cases the order of λ_1 and λ_2 do not matter, and therefore there are only 3 ways to distribute the row, and column syndromes. Again λ can be chosen from $q-1$ numbers. We conclude that in this case there are a total of $3 \cdot 3(q-1)$ different cosets.

Case 2.2: λ_1 and λ_2 are in the same row.
Again the column syndromes can be chosen in 3 different ways. We note that there is one row syndrome that equals 2λ. This syndrome will never equal 0, since it is a multiple of 2, while q is a prime greater than 2. Therefore it has a fixed value when choosing λ from its $q-1$ possibilities, and can be put on 3 different locations. We conclude that in this case there are a total of $3 \cdot 3(q-1)$ different cosets.

Case 2.3: λ_1 and λ_2 are in the same column.
This proof is analogous to case 2.2, therefore we conclude that in this case there are a total of $9(q-1)$ different cosets.

$$\begin{aligned}\alpha_2(C_3 \otimes C_3) &= 36(q-1)(q-2) + 3(q-1) + 9(q-1)(q-3) \\ &\quad + 3(q-1) + 9(q-1)(q-3) + 9(q-1) \\ &\quad + 9(q-1) + 9(q-1) \\ &= (q-1)\left(18(q-3) + 36(q-2) + 33\right)\end{aligned}$$

$\alpha_3(C_3 \otimes C_3)$: Now we have to fill in 3 positions with the numbers $1 \leq \lambda_1, \lambda_2, \lambda_3 \leq q-1$. The first problem here would be the many different locations of the λ's. However there is another problem that we have to keep in mind. Not only do we have to make sure that all of the different cases we will keep track of are of a different coset, but we also have to keep in mind that we do not actually create a coset with a smaller weight, since this is the first instance that we can place 3 elements in the matrix, but actually create an element that is in the coset with weight 2 or 1.
Example elements of $\mathbb{F}_5^{3\times 3}$ of different weights, but the same coset

0	3	0	*syndromes*
1	3	0	4
4	0	0	4
0	0	0	0

0	3	0	*syndromes*
0	4	0	4
0	4	0	4
0	0	0	0

0	2	0	*syndromes*
3	2	0	0
2	0	0	2
0	0	0	0

0	2	0	*syndromes*
0	0	0	0
0	2	0	2
0	0	0	0

This shows that the difficulty of the problem drastically increases. We will start by noting a few situations that would create cosets that do not have weight 3.

- $\exists! c_i, r_j \neq 0 : c_i = r_j$, this coset has weight 1.
- $\exists! c_i \neq 0, r_j = 0 : c_i = \sum_{j=1}^{3} r_j$, this coset has weight 2.
- $\exists! r_j \neq 0, c_i = 0 : r_j = \sum_{i=1}^{3} c_i$, this coset has weight 2.

We now continue by creating situations that indeed give us a weight of 3.
Case 1: $\forall c_i \neq 0$.
First off we note that each of the following cases, we can also mirror the rows to the columns and vise verse, and therefore these cases have to be counted twice.
All following coset leaders will be, for example, of the following form:

λ_1	λ_2	λ_3	*syndromes*
λ_1	λ_2	λ_3	$\lambda_1+\lambda_2+\lambda_3$
0	0	0	0
0	0	0	0

From here, there arise two sub-cases:
Case 1.1: $\lambda_1 + \lambda_2 + \lambda_3 = 0$.
Clearly λ_1 has $q-1$ options. Next we note that λ_2 can be any number, except for $q - \lambda_1$, since then $\lambda_1 + \lambda_2 = 0$ and λ_3 can not equal 0. Therefore λ_2 has a total of $q-2$ options. Lastly λ_3 must equal $-(\lambda_1 + \lambda_2)$ for the total sum to equal zero, and hence has only 1 option. We conclude that in this case there are a total of $(q-1)(q-2)$ different cosets, since it also does not matter in which row we place these λ's.

Case 1.2: $\lambda_1 + \lambda_2 + \lambda_3 \neq 0$.
Again, for λ_1 there are $q-1$ choices, but now this also holds for λ_2. The only thing we have to keep in mind, is that λ_3 does not equal $-(\lambda_1 + \lambda_2)$, for then the summation of all the λ's would equal zero. This summation $\lambda_1 + \lambda_2 + \lambda_3$ can also be put on all three of the rows. We conclude that in this case there are a total of $3(q-1)^2(q-2)$ different cosets.

Case 2: $\forall c_i \neq 0$, $\exists! r_j = 0$ and $\exists! i, j : c_i = r_j$.
In order to make sure this is a coset of weight 3, all coset leaders will be, for example, of the following form:

λ_1	λ_2	λ_3	*syndromes*
λ_1	λ_2	0	$\lambda_1 + \lambda_2$
0	0	λ_3	λ_3
0	0	0	0

Again we note this coset leader could be mirrored so we need to count every case twice. Obviously $\lambda_1 + \lambda_2$ may not equal zero, otherwise we end up in case 1.2. Several cases occur now.
Case 2.1: $\lambda_1 \neq \lambda_2 \neq \lambda_3 \neq \lambda_1$, and $\lambda_3 \neq \lambda_1 + \lambda_2$.
We pick λ_1 from its $q-1$ choices, then it follows that λ_2 can not equal it, nor can it add up to zero, and hence has $q-3$ options. It can not equal λ_1, λ_2, nor the sum of these, and therefore has $q-4$ options. We can also rearrange the rows in 6 different ways, and the columns in 6 different ways. We conclude that in this case there are a total of $6 \cdot 6(q-1)(q-3)(q-4)$ different cosets.

Case 2.2: $\lambda_1 \neq \lambda_2 \neq \lambda_3 \neq \lambda_1$, and $\lambda_3 = \lambda_1 + \lambda_2$.
Since $\lambda_3 = \lambda_1 + \lambda_2$ is a condition, we can rearrange the rows in 3 ways, by choosing the zero syndrome. λ_1 has $q-1$ options, and that gives λ_2 $q-3$ options, since the sum may also not equal zero. This fixes λ_3. We do know that λ_3 does not equal either of the two other λ's, and hence we can rearrange the columns in 6 different ways. We conclude that in this case there are a total of $3 \cdot 6(q-1)(q-3)$ different cosets.

Case 2.3: $\lambda_1 = \lambda_2 \neq \lambda_3$, and $\lambda_3 \neq \lambda_1 + \lambda_2$.
λ_1 gets chosen from $q-1$ choices, which fixes λ_2. λ_3 can not equal this number, nor its summation, hence there are $q-3$ options. Now the columns can only be switched in 3 different ways, but the rows in 6 ways. We conclude that in this case there are a total of $3 \cdot 6(q-1)(q-3)$ different cosets.

Case 2.4: $\lambda_1 = \lambda_2 \neq \lambda_3$, and $\lambda_3 = \lambda_1 + \lambda_2$.
λ_1 gets chosen from $q-1$ options, and this fixes λ_2 and λ_3. It follows that both the rows and columns can only be rearranged in 3 ways each. We conclude that in this case there are a total of $3 \cdot 3(q-1)$ different cosets.

Case 2.5: $\lambda_1 \neq \lambda_2 = \lambda_3$. And we immediately note that it does not matter if we choose λ_1 or λ_2 to equal λ_3, since we shuffle them around right away.
λ_1 has $q-1$ options, and it follows that λ_2 has $q-3$ options left. The columns can be rear-

ranged in 3 ways, while the rows can be rearranged in 6 ways, since λ_3 can never equal the sum of the other two λ's, since then one of them has to equal zero. We conclude that in this case there are a total of $3 \cdot 6(q-1)(q-3)$ different cosets.

Case 2.6: $\lambda_1 = \lambda_2 = \lambda_3$.
Obviously all λ get chosen at once, from a selection of $q-1$ choices. There is only one way to distribute the columns, while there are 6 ways to rearrange the rows, since λ_3 will never equal the sum $\lambda_1 + \lambda_2$. We conclude that in this case there are a total of $6(q-1)$ different cosets.

Case 3: $\exists! r_j, c_i = 0$.
When we create such a coset, the coset leader will have the following form:

$\lambda_1 + \lambda_3$	λ_2	0	*syndromes*
λ_1	λ_2	0	$\lambda_1 + \lambda_2$
λ_3	0	0	λ_3
0	0	0	0

We have to keep several things in mind now:
If one of the following holds, $\lambda_2 = \lambda_3$ OR $\lambda_1 + \lambda_2 = 0$ OR $\lambda_1 + \lambda_3 = 0$, it would give us a coset of weight 2. In this case it is not possible to mirror the row and column syndromes, since that's taken care of in this specific case already.
First we note that, even though the λ's and their order may differ, it could result in the same coset:

2	2	*syndromes*
4	2	1
3	0	3

2	2	*syndromes*
0	1	1
2	1	3

The following subcases arise:
Case 3.1:
To ensure the previous noted conditions do not occur, we start by choosing λ_1 from its $q-1$ options. It instantly follows that λ_2 can not equal $-\lambda_1$, and therefore has $q-2$ options. Lastly λ_3 can not equal $-\lambda_1$, nor λ_2, and therefore has $q-3$ options.

We will stop our computation here due to a lack of time, and since the goal of showing the difficulty of this problem should be shown by now.

$$\begin{aligned}\alpha_3(C_3 \otimes C_3) = 2 \cdot \big((q-1)(q-2) + 3(q-1)^2(q-2)\big) \\ + 2 \cdot (36(q-1)(q-3)(q-4) + 18(q-1)(q-3) \\ + 18(q-1)(q-3) + 9(q-1) + 18(q-1)(q-3) + 6(q-1)) \\ + \ldots\end{aligned}$$

From all these different cases and calculations, we note that this is not just a simple problem, even for such a "simple" situation as a 3×3 array. Therefore we will now stop the calculations here without computing $\alpha_4(C_3 \otimes C_3)$ and $\alpha_5(C_3 \otimes C_3)$. However these would be calculated in a somewhat same way, and hence we encourage the reader to solve this problem for themselves.

9 Epilogue

We will have to end our paper here due to a lack of time. However after feedback and some further inspection, we conclude that there are still some errors contained in this paper. Some are minor errors, some are a big worse. It it not hard to see this paper has room for improvement, for example extending it over all q, instead of only the prime numbers.
At the time of writing this paper, the article from Putranto Utomo and Ruud Pellikaan still contained some errors, however all of the errors we have found are corrected now. One of these errors was their proposition 2.17, on which our theorem 7.1 is based. Now this is corrected we can try to see if they give us the same results, by filling in all of the cases described in proposition 2.17 in our theorem. We find that indeed, these two are not equivalent. The reason for this is that we used a specific order of submatrices in our proof. This would need to be corrected. However, as described before, due to a lack of time this task will be postponed to a later moment.

References

[1] Putranto Utomo & Ruud Pellikaan. *The coset leader weight enumerator of the product code* $[m, m-1, 2]_q \otimes [n, n-1, 2]_q$.

[2] Arjeh M. Cohen, Hans Cuypers, & Hans Sterk. (2012-2013). *Sets, Logic and Algebra*.

[3] Hsi-Yuan Chang, Jyun-Jie Wang, Chi-Yuan Lin, & Chin-Hsing Chen. 2018, 'Development of low-complexity matrix embedding with an efficient iterative strategy', *ICIC Express Letters*, vol 9, no. 7, pp. 707-713.