Constructing CSS-T Codes from Extended Quasi-Cyclic Codes

Ross George

Thesis presented to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Mathematics

Gretchen L. Matthews, Chair
Eduardo Camps-Moreno, Co-chair
Giuseppe Cotardo
Hiram H. López Valdez

20 May 2025
Blacksburg, Virginia

Keywords: Coding Theory, CSS-T Codes, Quasi-Cyclic Codes, Quantum Codes

Constructing CSS-T Codes from Extended Quasi-Cyclic Codes

Ross George

(ABSTRACT)

We analyze a construction of quasi-cyclic codes in an extension field based on the polynomials which generate the corresponding codes in the base fields. We establish a set of code operations on which trace relationships between extended quasi-cyclic codes are the same as the operations taken on the original codes. We also discuss how these relationships can be used to construct CSS-T codes for quantum-error correction.

Constructing CSS-T Codes from Extended Quasi-Cyclic Codes

Ross George

(GENERAL AUDIENCE ABSTRACT)

We study a construction of classical error-correcting codes which is unaffected by shifting actions and how those codes can be used to construct codes useful for error-correction in quantum computing.

Dedication

To my muse, Madeline, and to my parents.

Acknowledgments

This document would not have come together without the expertise and guidance of Dr. Gretchen Matthews and Dr. Eduardo Camps-Moreno. Besides introducing the author to the problem, their subject knowledge and patience proved vital in the development of this work.

Contents

Chapter 1

Introduction

1.1 Preliminaries

In this section, we recall some basic concepts and operations to be used throughout this thesis. For additional background, we refer the reader to [5].

The notation $\mathbb{F}_q$ will be used to represent the unique (up to isomorphism) finite field on q elements, where q is a power of a prime number. We use $\mathbb{F}_q^n$ to mean the set of n-tuples with entries in $\mathbb{F}_q$, i.e.

$$\mathbb{F}_q^n := \{(a_1, \ldots, a_n) : a_i \in \mathbb{F}_q\}.$$

In this thesis, we will write vectors horizontally, i.e. for $\mathbf{c} \in \mathbb{F}_q^n$, $\mathbf{c} = (c_1, \cdots, c_n)$. This differs from the vertically expressed vectors often used in algebra and linear algebra and is done for compatibility with other works on coding theory. Moreover, given $\mathbf{c} \in \mathbb{F}_q^n$ and $i \in [n]$, c_i denotes the i-th coordinate of $\mathbf{c}$ (where $[n] := \{1, \ldots, n\}$). We use the notation $R^{m \times n}$ to mean the set of $m \times n$ matrices with entries in some ring R.

Definition 1.1 (Matrix Determinant)**.** For a matrix $M \in R^{n \times n}$ over some ring R, the *determinant* is a map $R^{n \times n} \to R$ recursively defined as

$$\det(M) := \begin{cases} \sum_{i=0}^{n-1} (-1)^i m_{0,i} \det(M_{0,i}), \ n > 1 \\ m_{0,0}, \ n = 1 \end{cases},$$

where $m_{i,j}$ is the entry in the i-th row and j-th column of M and $M_{i,j}$ is the matrix obtained by removing the i-th row and j-th column from M. The determinant is sometimes written $|M| := \det(M)$.

Example 1.2. The determinant of

$$A := \begin{pmatrix} 1 & 1 \\ 2 & 0 \end{pmatrix} \in \mathbb{F}_3^{2\times 2}$$

is

$$\det(A) = (-1)^0 \cdot 1 \cdot \det((0)) + (-1)^1 \cdot 1 \cdot \det((2)) = 1.$$

Definition 1.3 (Linear Code)**.** An $[n,k]_q$ *linear code* is an $\mathbb{F}_q$-vector space $C \leq \mathbb{F}_q^n$ with dimension k. An element $\mathbf{c} \in C$ is called a *codeword*.

For the remainder of this thesis, *code* can be taken to mean *linear code*.

Definition 1.4 (Generator Matrix)**.** If C is an $[n,k]_q$ code, then any matrix $G \in \mathbb{F}_q^{n\times k}$ such that $C = \{\mathbf{v}G : \mathbf{v} \in \mathbb{F}_q^k\}$ is called a *generator matrix* of C. We will often write *that G generates C* or *that C is generated by G*. Furthermore, any matrix G whose rows form a basis for C is a generator matrix for C.

Proposition 1.5. *An $[n,k]_q$ code C has a generator matrix (up to permutation of columns) of the form $[I_k \ A]$, where I_k is the $k \times k$ identity matrix in $\mathbb{F}_q$.*

Proof. Since C is a vector space over $\mathbb{F}_q$, it has a basis and therefore a generator matrix G. Because $G \in \mathbb{F}_q^{n\times k}$ where $\mathbb{F}_q$ is a field, we can put G in row-echelon form G'. If P is a permutation matrix which orders the pivot columns of G' in in the first k columns (and permutes the rest of the columns as desired), then $G = [I_k \ A]P$. □

Definition 1.6 (Hamming weight)**.** For a vector $\mathbf{c} \in \mathbb{F}_q^n$, the *Hamming weight* of $\mathbf{c}$ is

$$|\mathbf{c}| := |\{i : c_i \neq 0\}|,$$

where c_i is the i-th entry of the codeword $\mathbf{c}$.

Example 1.7. Working in $\mathbb{F}_2^3$, $|(1,0,1)| = 2$.

Definition 1.8 (Hamming Distance)**.** For two vectors $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{F}_q^n$, the *Hamming distance* between them is

$$\Delta(\mathbf{c}_1, \mathbf{c}_2) := |\mathbf{c}_1 - \mathbf{c}_2|.$$

Example 1.9. Given $\mathbf{c} = (1,1,0,2), \mathbf{c}' = (1,2,0,2) \in \mathbb{F}_3^4$,

$$\Delta(\mathbf{c}, \mathbf{c}') = |(1,1,0,2) - (1,2,0,2)| = |(0,2,0,0)| = 1.$$

Definition 1.10 (Minimum Distance)**.** For an $[n,k]_q$ code C, the *minimum distance* of C is

$$d(C) := \min_{\mathbf{c} \in C/\{\mathbf{0}\}} |\mathbf{c}|.$$

If C is an $[n,k]_q$ code with minimum distance d, then we say that C is an $[n,k,d]_q$ code. It is easy to see that this is equivalent to the minimum distance between two different codewords in C, meaning

$$d = \min_{\substack{\mathbf{c}_1, \mathbf{c}_2 \in C \\ \mathbf{c}_1 \neq \mathbf{c}_2}} \Delta(\mathbf{c}_1, \mathbf{c}_2).$$

Example 1.11. The code

$$C := \{(0,0,0), (1,1,0), (0,1,1), (1,0,1)\} \leq \mathbb{F}_2^3$$

has minimum distance 2.

Theorem 1.12 (Singleton Bound)**.** *For any $[n, k, d]_q$ code, $d \leq n - k + 1$.*

Proof. Because C has minimum distance d, any two distinct codewords in C differ in at least d entries. Therefore, if we create a new code C' by removing any set of $d - 1$ fixed entries from each codeword in C (removing the same entries from each codeword), every codeword remains distinct. In particular, the minimum distance of the resulting code is at least 1. Therefore, there can be no more codewords in C than codewords in the punctured version of C. We also see that C' can have no more than q^{n-d+1} codewords (the number of codewords in $\mathbb{F}_q^{n-d+1}$), so $|C| = |C'|$, which implies $q^k \leq q^{n-d+1}$ and finally $k \leq n - d + 1$. □

Remark 1.13. The *support* of $\mathbf{w} \in \mathbb{F}_q^n$ is

$$\text{supp}(\mathbf{w}) := \{i \in [n] : w_i \neq 0\}.$$

Definition 1.14 (Even Code)**.** A code $C \leq \mathbb{F}_q^n$ is an *even code* if every codeword in C has even Hamming weight.

Example 1.15. The code $C \leq \mathbb{F}_2^3$ generated by

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

is even. We can enumerate

$$C = \{(1, 0, 1), (0, 1, 1), (1, 1, 0), (0, 0, 0)\}$$

to verify that every nonzero codeword has weight 2.

Definition 1.16 (Dual Code)**.** For a code $C \leq \mathbb{F}_q^n$, the *dual code* $C^\perp \leq \mathbb{F}_q^n$ is the code

$$C^\perp := \{\mathbf{d} \in \mathbb{F}_q^n : \mathbf{c} \cdot \mathbf{d} = 0 \quad \forall \mathbf{c} \in C\},$$

where $\mathbf{c} \cdot \mathbf{d}$ is the Euclidean dot product given by

$$\mathbf{c} \cdot \mathbf{d} := \sum_{i=0}^{n} c_i d_i \in \mathbb{F}_q.$$

Proposition 1.17. *If C is an $[n,k]_q$ code, then $C^\perp$ is an $[n, n-k]_q$ code.*

Proof. By Proposition 1.5, we may write a generator matrix for C (up to permutation of columns) as $[I_k \ A]$ where $A \in \mathbb{F}_q^{(n-k)\times k}$. We will show that $[-A^T \ I_{n-k}]$ is a generator matrix for $C^\perp$. To begin,

$$[I_k \ A][-A^T \ I_{n-k}]^T = I_k(-A^T)^T + A\, I_{n-k}^T = -A + A = 0.$$

Therefore, $\text{row}([-A^T \ I_{n-k}]) \leq C^\perp$. Suppose we have some vector $\mathbf{v} \in C^\perp$, which we split into components $\mathbf{v}_1 \in \mathbb{F}_q^k$ and $\mathbf{v}_2 \in \mathbb{F}_q^{n-k}$. It follows that

$$(\mathbf{v}_1 + \mathbf{v}_2 A^T)^T = \mathbf{v}_1^T + A\mathbf{v}_2^T = I_k \mathbf{v}_1^T + A\mathbf{v}_2^T = [I_k \ A][\mathbf{v}_1 \ \mathbf{v}_2]^T = [I_k \ A]\mathbf{v}^T = 0.$$

Therefore, $\mathbf{v}_1 = \mathbf{v}_2(-A^T)$. In other words, $\mathbf{v} = \mathbf{v}_2[-A^T \ I_{n-k}]$. Thus, $C^\perp \leq \text{row}([-A^T \ I_{n-k}])$. Therefore, $C^\perp = \text{row}([-A^T \ I_{n-k}])$, so $\dim(C^\perp) = n - k$. □

Example 1.18. The $[3,2]_2$ code

$$C := \{(0,0,0), (1,0,0), (0,1,0), (1,1,0)\} \leq \mathbb{F}_2^3$$

has dual code

$$C^{\perp} = \{(0,0,0),(0,0,1)\}.$$

Definition 1.19 (Self-Orthogonal Code)**.** A code $C \leq \mathbb{F}_q^n$ is *self-orthogonal* if $C \leq C^{\perp}$.

Example 1.20. The code given by

$$C := \{(0,0,0,0),(1,0,1,0)\} \leq \mathbb{F}_2^4$$

is self-orthogonal. This follows from the fact that for called any $\mathbf{a}, \mathbf{b} \in C$, $\mathbf{a} \cdot \mathbf{b} = 0$. In fact, any even code C such that $|\mathrm{supp}(\mathbf{c}) \cap \mathrm{supp}(\mathbf{c}')|$ is even for all $\mathbf{c}, \mathbf{c}' \in C$, is self-orthogonal.

Definition 1.21 (Self Dual Code)**.** A code $C \leq \mathbb{F}_q^n$ is *self-dual* if $C = C^{\perp}$.

Example 1.22. The code $C \leq \mathbb{F}_3^4$ generated by

$$G := \begin{pmatrix} 1 & 2 & 0 & 1 \\ 1 & 0 & 1 & 2 \end{pmatrix}$$

is self-dual.

Definition 1.23 (Parity Check Matrix)**.** For a code $C \leq \mathbb{F}_q^n$, any matrix which generates $C^{\perp}$ is called a *parity check matrix* of C. For a matrix G, the notation $G^{\perp}$ will be used to denote *a* parity check matrix of the code generated by G.

Example 1.24. Let $C \leq \mathbb{F}_3^4$ be generated by

$$G := \begin{pmatrix} 1 & 1 & 2 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Then the dual code $C^\perp$ is generated by

$$H := \begin{pmatrix} 0 & 1 & 1 & 2 \\ 1 & 2 & 0 & 0 \end{pmatrix}.$$

To confirm, note that $\dim C^\perp = 2$, and

$$GH^T = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

Definition 1.25 (Sum Code)**.** For two codes $C_1, C_2 \leq \mathbb{F}_q^n$, the *sum code* $C_1 + C_2$ is

$$C_1 + C_2 := \langle \mathbf{c}_1 + \mathbf{c}_2 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle \leq \mathbb{F}_q^n.$$

Furthermore, if C_1 is an $[n, k_1, d_1]_q$ code and C_2 is an $[n, k_2, d_2]_q$ code, then $C_1 + C_2$ is an $[n, k, d]_q$ code with $k \leq k_1 + k_2$ and $d \leq \min\{d_1, d_2\}$.

Definition 1.26 (Tensor Product)**.** For two matrices $A \in R^{m \times n}$ and $B \in R^{m' \times n'}$ over the same ring R, the *tensor product* $A \otimes B$ is

$$A \otimes B := \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix} \in R^{mm' \times nn'}.$$

It is well known that the tensor product is a bilinear operator.

Example 1.27. Consider $\begin{pmatrix} 1 & 2 \\ 1 & 4 \end{pmatrix}, \begin{pmatrix} 3 & 1 \\ 2 & 2 \end{pmatrix} \in \mathbb{F}_5^{2\times 2}$. Then

$$\begin{pmatrix} 1 & 2 \\ 1 & 4 \end{pmatrix} \otimes \begin{pmatrix} 3 & 1 \\ 2 & 2 \end{pmatrix} = \begin{pmatrix} 1\begin{pmatrix} 3 & 1 \\ 2 & 2 \end{pmatrix} & 2\begin{pmatrix} 3 & 1 \\ 2 & 2 \end{pmatrix} \\ 1\begin{pmatrix} 3 & 1 \\ 2 & 2 \end{pmatrix} & 4\begin{pmatrix} 3 & 1 \\ 2 & 2 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 3 & 1 & 1 & 2 \\ 2 & 2 & 4 & 4 \\ 3 & 1 & 2 & 4 \\ 2 & 2 & 3 & 3 \end{pmatrix}.$$

Definition 1.28 (Tensor Product Code)**.** For a code $C \leq \mathbb{F}_q^n$ and a vector $\mathbf{v} \in \mathbb{F}_q^m$, the *tensor product code* is

$$\mathbf{v} \otimes C := \langle \mathbf{v} \otimes \mathbf{c} : \mathbf{c} \in C \rangle \leq \mathbb{F}_q^{mn}.$$

Remark 1.29. For a code C with generator matrix G, the code $\mathbf{v} \otimes C$ is generated by $\mathbf{v} \otimes G$.

Example 1.30 (Tensor Product Code)**.** If $C \leq \mathbb{F}_3^3$ is the code generated by

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

then the tensor product code $(1, 2) \otimes C \leq \mathbb{F}_3^6$ is generated by

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

Definition 1.31 (Schur Product)**.** Given two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{F}_q^n$, the *Schur product* of $\mathbf{a}$ and $\mathbf{b}$ is

$$\mathbf{a} * \mathbf{b} := (a_1 b_1, \ldots, a_n b_n) \in \mathbb{F}_q^n.$$

Less formally, the Schur product of **a** and **b** is the vector resulting from the ordered componentwise multiplication of their entries.

Example 1.32. The Schur product of $(1,0,3),(4,2,2) \in \mathbb{F}_5^3$ is

$$(1,0,3) * (4,2,2) = (4,0,1).$$

Definition 1.33 (Schur Product Code)**.** Given two codes $C_1, C_2 \leq \mathbb{F}_q^n$, the *Shur product code* of C_1 and C_2 is

$$C_1 * C_2 := \langle \mathbf{c_1} * \mathbf{c_2} : \mathbf{c_1} \in C_1, \mathbf{c_2} \in C_2 \rangle \leq \mathbb{F}_q^n.$$

Example 1.34. Let $C_1, C_2 \leq \mathbb{F}_3^5$ be the codes generated by

$$G_1 = \begin{pmatrix} 1 & 0 & 2 & 1 & 4 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix} \qquad \text{and} \qquad G_2 = \begin{pmatrix} 0 & 0 & 2 & 2 & 1 \\ 1 & 4 & 3 & 3 & 3 \end{pmatrix}.$$

We see that we can find the generator of the product code by taking the products of all pairs between the bases given in the rows of G_1 and G_2. Then the code $C_1 * C_2 \leq \mathbb{F}_5^6$ is the code which is generated by

$$\begin{pmatrix} 0 & 0 & 4 & 2 & 4 \\ 1 & 0 & 1 & 3 & 2 \\ 0 & 0 & 2 & 2 & 0 \end{pmatrix}.$$

Observe that C_1 and C_2 are both $[6,2]_5$ codes, but $C_1 * C_2$ is a $[6,3]_5$ code.

Definition 1.35 (Square Code)**.** For a code $C \leq \mathbb{F}_q^n$, the *square code* C^2 is $C^2 := C * C$.

Remark 1.36. If $(1, \ldots, 1) \in C$, then $C \leq C^2$. However, as shown below, this condition is necessary.

Example 1.37. Let $C \leq \mathbb{F}_7^5$ be generated by

$$\begin{pmatrix} 1 & 0 & 5 & 2 & 1 \\ 1 & 0 & 2 & 5 & 6 \end{pmatrix}.$$

Then the square code $C^2 \leq \mathbb{F}_7^5$ is generated by

$$\begin{pmatrix} 1 & 0 & 4 & 4 & 1 \\ 1 & 0 & 3 & 3 & 6 \end{pmatrix}.$$

Observe that in this case $C \neq C^2$, but the two codes have the same dimension.

Remark 1.38. See Propositions 5.1, 5.2, 5.3, 5.4, and 5.9 for results on how we can manipulate expressions involving the code operations described above.

Definition 1.39 (Trace Function)**.** For any $\alpha \in \mathbb{F}_{q^r}$, the *trace function* with respect to the extension $\mathbb{F}_{q^r}/\mathbb{F}_q$ is

$$\mathrm{tr}_{\mathbb{F}_{q^r}/\mathbb{F}_q}(\alpha) := \sum_{i=0}^{r-1} \alpha^{q^i}.$$

It is well-known this function satisfies $\mathbb{F}_{q^r} \to \mathbb{F}_q$. Viewing $\mathbb{F}_{q^r}$ as an $\mathbb{F}_q$-vector space, this function is $\mathbb{F}_q$-linear. For the remainder of this thesis, for an element $\alpha \in \mathbb{F}_{q^r}$, the notation $\mathrm{tr}(\alpha) := \mathrm{tr}_{\mathbb{F}_{q^r}/\mathbb{F}_q}(\alpha)$ will be used to denote the trace where the extension and base fields can be taken from context.

Example 1.40. Let $\alpha \in F_{2^3}$ be the root of the polynomial $x^3 + x + 1 \in \mathbb{F}_2[x]$. Then

$$\mathrm{tr}(\alpha) = \alpha^{2^2} + \alpha^{2^1} + \alpha^{2^0} = (\alpha + \alpha^2) + (\alpha^2) + (\alpha) = 0.$$

Definition 1.41 (Vector Trace)**.** For $\mathbf{c} \in \mathbb{F}_{q^r}^n$, *vector trace* of $\mathbf{c}$ is

$$\mathrm{tr}(\mathbf{c}) := (\mathrm{tr}(c_1), \ldots, \mathrm{tr}(c_n)) \in \mathbb{F}_q^n.$$

Example 1.42. Take α to be a generator of $\mathbb{F}_{5^2}^*$. Then the trace of $(\alpha, 1, \alpha^{12}, \alpha^{17}) \in \mathbb{F}_{5^2}^4$ is

$$\mathrm{tr}((\alpha, 1, \alpha^{12}, \alpha^{17})) = (\mathrm{tr}(\alpha), \mathrm{tr}(1), \mathrm{tr}(\alpha^{12}), \mathrm{tr}(\alpha^{17})) = (2, 1, 3\ 4) \in \mathbb{F}_5^4.$$

Definition 1.43 (Trace Code)**.** For a code $C \leq \mathbb{F}_{q^r}^n$, the *trace code* of C is

$$\mathrm{tr}(C) := \langle \mathrm{tr}(\mathbf{c}) : \mathbf{c} \in C \rangle \leq \mathbb{F}_q^n.$$

Example 1.44. Let α be a generator of $\mathbb{F}_{2^3}^*$. Let $C \leq \mathbb{F}_{2^3}^4$ be the code generated by

$$\begin{pmatrix} 1 & \alpha^2 & \alpha & 0 \\ 0 & \alpha & \alpha^5 & 0 \end{pmatrix}.$$

Then the trace code $\mathrm{tr}(C) \leq \mathbb{F}_2^4$ is generated by

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

Observe that $\dim \mathrm{tr}(C) > \dim C$ in this example.

1.2 Motivation (Quantum Error Correction)

Practicality has historically driven the development of coding theory. In a world built around computers and where signal transmission underpins the entire global economy, ensuring that errors do not occur or are corrected has only become more important with time. In the late 20th century, researchers realized that some of the stranger corollaries of quantum physics could be used to build a computer very different from others.

Quantum computers are machines which utilize quantum mechanics to perform computational tasks in a way which is fundamentally different from a classical computation. Whereas classical computers use *bits* with values 0 or 1, quantum computers use *qubits*, which quantum physics allows to hold all possible values simultaneously. The transition of quantum computers from theoretical objects to physical ones will likely have massive implications, especially in areas like public-key cryptography, which relies on the assumed difficulty of specific problems to prove security. Famously, the security of the RSA cryptosystem relies on the perceived difficulty of integer factorization, which Shor [9] proved can be performed in polynomial time on a quantum computer.

In the same way that classical error-correcting codes provide fault-tolerance for classical computers, quantum error-correcting codes provide fault-tolerance for quantum computers. The difference between classical and quantum error correction is that classical errors are simple relative to quantum ones. Whereas classical errors are bit operations (commonly parity flips) or erasures, quantum errors introduce an extra level of complexity to correct.

The *No Cloning Theorem* given in [11] states that *no* qubit can be exactly copied. Much of classical error-correction relies on copying bits for redundancy, so it was unclear whether quantum error-correction would be possible. Eventually, Shor [8] showed that quantum error-correction was possible, although his original construction was impractical, encoding a

single qubit into nine.

1.2.1 Quantum Computation and Stabilizer Codes

The qubit can be formally thought of as a vector $|\psi\rangle \in \mathbb{C}^2$. From a physical perspective, we write

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,,$$

where $\alpha^2 + \beta^2 = 1$, $\mathbb{P}[|\psi\rangle = |0\rangle] = |\alpha|^2$, and $\mathbb{P}[|\psi\rangle = |1\rangle] = |\beta|^2$ (where $\mathbb{P}[\text{condition}]$ is the probability that *condition* is true). Recall that a qubit is not restricted to holding its base values of $|0\rangle$ or $|1\rangle$. Because of the complicated nature of qubits, it is commonly assumed that an error E can be represented as

$$E = E_1 \otimes \cdots \otimes E_n,$$

where each E_i is a unitary (invertible with determinant ± 1) matrix in $\mathbb{C}^{2\times 2}$. If we expect an error to have this form, then it makes sense to try to find a quantum code which is unaffected by this matrix. Furthermore, if we expect that an error manifests as any set of such matrices, we call the code consisting of the vectors unchanged by these matrices the *stabilizer code*. Because this is the most common model of quantum errors, stabilizer codes are the most common model of quantum error-correction. In fact, some stabilizer codes can be constructed from classical binary codes.

1.2.2 CSS Codes

In 1995, Calderbank and Shor [2] and Steane [10] independently found a method for constructing quantum codes that allows for a wider array of parameters. Take an $[n, k_1]_2$ code

C_1 and an $[n, k_2]_2$ code C_2 satisfying $C_2 \leq C_1$. For $\mathbf{w} \in \mathbb{F}_2^n$, define a corresponding quantum codeword

$$|\mathbf{q_w}\rangle := 2^{-\frac{k_1}{2}} \sum_{\mathbf{c} \in C_1} (-1)^{\mathbf{c}\cdot\mathbf{w}} |\mathbf{c}\rangle .$$

Using this construction, Calderbank and Shor [2] and Steane [10] define the quantum CSS code generated by C_1 and C_2 as

$$Q(C_1, C_2) := \{|\mathbf{q_c}\rangle : \mathbf{c} \in C_2^{\perp}\}.$$

Observe that $Q(C_1, C_2)$ has the same length n and dimension $k_2 - k_1$ (see [2] for more details).

1.2.3 CSS-T Codes

Universal quantum computation is the quantum analogue of Turing completeness, i.e. universality allows one to simulate any quantum computer (subject to physical constraints). Rengaswamy et al. [7] provide a special type of CSS code which is of particular interest in the implementation of universality.

Definition 1.45 (CSS-T Pair [7])**.** For two codes $C_2 \leq C_1 \leq \mathbb{F}_2^n$, (C_1, C_2) is a *CSS-T pair* provided C_2 is even and, for all $\mathbf{c} \in C_2$, $C_1^{\perp}$ contains a self-dual subcode with dimension $\frac{|\mathbf{c}|}{2}$ (where $|\mathbf{c}|$ is the Hamming weight of $\mathbf{c}$) with support on $\mathbf{c}$.

Definition 1.46 (CSS-T Code [7])**.** A *CSS-T code* is a CSS code $Q(C_1, C_2)$ where the underlying binary codes form a CSS-T pair.

This construction is unwieldy for large C_2 (recall that $|C_2|$ is exponential with respect to the dimension of C_2). Camps-Moreno et al. [3] give a more algebraic interpretation of CSS-T pairs by proving the following result.

Theorem 1.47 ([3, Theorem 2.3]). *Given $C_1, C_2 \leq \mathbb{F}_2^n$, (C_1, C_2) is a CSS-T pair if and only if $C_2 \leq C_1 \cap (C_1^2)^\perp$.*

This construction of CSS-T codes will be the primary focus of the results presented here. In particular, a construction for the code of the form $C_1 \cap (C_1^2)^\perp$ for quasi-cyclic C_1 will be provided.

Chapter 2

Quasi-Cyclic Codes

2.1 Preliminaries

We now introduce a well-studied class of codes with many applications. Later in this section, we will use these codes to build an understanding of quasi-cyclic codes.

Definition 2.1 (Cyclic Code)**.** A *cyclic code* is a linear code $C \leq \mathbb{F}_q^n$ which is invariant under the shift function

$$\begin{aligned} \text{shift} : \mathbb{F}_q^n &\to \mathbb{F}_q^n \\ (c_1, c_2, \ldots, c_{n-1}, c_n) &\mapsto (c_n, c_1, c_2, \ldots, c_{n-1}), \end{aligned}$$

meaning $\text{shift}(C) = C$.

The natural extension of this concept generalizes the shift concept to a repeated shifting action and yields quasi-cyclic codes.

Definition 2.2 (Repeated Shift Function)**.** The *ℓ-folded shift function* $\text{shift}^\ell : \mathbb{F}_q^n \to \mathbb{F}_q^n$ is the shift function composed with itself ℓ times, i.e. the function which shifts individual entries by ℓ positions.

Definition 2.3 (Quasi-Cyclic Code)**.** An *ℓ-quasi-cyclic* (or ℓ-QC) code is a code which is

invariant under the shift$^\ell$ function, i.e. a code C such that shift$^\ell(C) = C$. A code is *quasi-cyclic* if and only if it is ℓ-quasi-cyclic for some integer $\ell \geq 1$.

Many of the basic traits of cyclic codes can be extended to quasi-cyclic codes. Now, we recall the traits which we will need later in this thesis.

Proposition 2.4. *A code $C \leq \mathbb{F}_q^{m\ell}$ is ℓ-quasi-cyclic if and only if for some basis $\mathcal{B}$ of C, for any $\mathbf{c} \in \mathcal{B}$,* shift$^\ell(\mathbf{c}) \in C$.

Proof. Suppose $\mathcal{B} = \{\mathbf{c}_1, \ldots, \mathbf{c}_k\}$ is a basis of C such that shift$^\ell(\mathbf{b}_i) \in C$ for every $\mathbf{b}_i \in \mathcal{B}$. Let $\mathbf{c} \in C$ be arbitrary. Then there exist $\beta_1, \ldots, \beta_k \in \mathbb{F}_q$ such that

$$\mathbf{c} = \sum_{i=1}^{k} \beta_i \mathbf{c}_i.$$

Therefore, we can write

$$\text{shift}^\ell(\mathbf{c}) = \text{shift}^\ell\left(\sum_{i=1}^{k} \beta_i \mathbf{c}_i\right) = \sum_{i=1}^{k} \beta_i \text{shift}^\ell(\mathbf{c}_i) \in C,$$

where we draw the last conclusion from the fact that C is closed under addition and each shift$^\ell(\mathbf{c}_i) \in C$. Therefore, C is ℓ-quasi-cyclic.

Now, suppose C is ℓ-quasi-cyclic. Then C contains the shifts of all its codewords, so for any basis $\mathcal{B}$ of C and any $\mathbf{c} \in \mathcal{B}$, shift$^\ell(\mathbf{c}) \in C$. □

Proposition 2.5. *If $C_1, C_2 \leq \mathbb{F}_q^{m\ell}$ are ℓ-quasi-cyclic codes, then $C_1 + C_2$ is an ℓ-quasi-cyclic code.*

Proof. Take an arbitrary codeword $\mathbf{c} \in C_1 + C_2$. Then $\mathbf{c} = \mathbf{c}_1 + \mathbf{c}_2$ with $\mathbf{c}_1 \in C_1$ and $\mathbf{c}_2 \in C_2$.

It follows that

$$\text{shift}^\ell(\mathbf{c}) = \text{shift}^\ell(\mathbf{c}_1 + \mathbf{c}_2) = \text{shift}^\ell(\mathbf{c}_1) + \text{shift}^\ell(\mathbf{c}_2) \in \text{shift}^\ell(C_1) + \text{shift}^\ell(C_2) = C_1 + C_2.$$

Therefore, $C_1 + C_2$ is ℓ-quasi-cyclic. □

Proposition 2.6. *If $C_1, C_2 \leq \mathbb{F}_q^{m\ell}$ are ℓ-quasi-cyclic codes, then $C_1 \cap C_2$ is an ℓ-quasi-cyclic code.*

Proof. Take any $\mathbf{c} \in C_1 \cap C_2$. Then

$$\text{shift}^\ell(\mathbf{c}) \in \text{shift}^\ell(C_1) = C_1 \qquad \text{and} \qquad \text{shift}^\ell(\mathbf{c}) \in \text{shift}^\ell(C_2) = C_2.$$

Thus, $\text{shift}^\ell(\mathbf{c}) \in C_1 \cap C_2$. Therefore, $C_1 \cap C_2$ is ℓ-quasi-cyclic. □

Proposition 2.7. *If $C_1, C_2 \leq \mathbb{F}_q^{m\ell}$ are ℓ-quasi-cyclic codes, then $C_1 * C_2$ is an ℓ-quasi-cyclic code.*

Proof. For any $\mathbf{c} \in C_1 * C_2$, we have $\mathbf{c}_1 \in C_1$ and $\mathbf{c}_2 \in C_2$ such that $\mathbf{c} = \mathbf{c}_1 * \mathbf{c}$. The definition of the Schur product code says that

$$S := \{\mathbf{c}_1 * \mathbf{c}_2 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2\}$$

spans and thus contains a basis for $C_1 * C_2$. Take any element of $\mathbf{c} \in S$. Then there exist $\mathbf{c}_1 \in C_1$ and $\mathbf{c}_2 \in C_2$ such that $\mathbf{c} = \mathbf{c}_1 * \mathbf{c}_2$. Therefore,

$$\text{shift}^\ell(\mathbf{c}) = \text{shift}^\ell(\mathbf{c}_1 * \mathbf{c}_2) = \text{shift}^\ell(\mathbf{c}_1) * \text{shift}^\ell(\mathbf{c}_2) \in \text{shift}^\ell(C_1) * \text{shift}^\ell(C_2) = C_1 * C_2.$$

By construction, S contains a basis for $C_1 * C_2$ and $\text{shift}^\ell(\mathbf{c}) \in C_1 * C_2$ for any $\mathbf{c} \in S$, so

Proposition 2.4 says that $C_1 * C_2$ is ℓ-quasi-cyclic. □

Proposition 2.8. *If $C \leq \mathbb{F}_q^{m\ell}$ is an ℓ-quasi-cyclic code, then $C^\perp \leq \mathbb{F}_q^{m\ell}$ is an ℓ-quasi-cyclic code.*

Proof. Take any $\mathbf{d} \in C^\perp$ and any $\mathbf{c} \in C$. Since C is ℓ-quasi-cyclic, $\text{shift}^{(m-1)\ell}(\mathbf{c}) \in C$. The dot product only uses the relative positions of elements in a vector, so shifting $\mathbf{d}$ ℓ positions one way is the same as shifting $\mathbf{c}$ the ℓ positions in the other way. More formally,

$$\text{shift}^{\ell}(\mathbf{d}) \cdot \mathbf{c} = \mathbf{d} \cdot \text{shift}^{(m-1)\ell}(\mathbf{c}) = 0.$$

Therefore, $\text{shift}^{\ell}(\mathbf{d}) \in C^\perp$, so $C^\perp$ is ℓ-quasi-cyclic. □

Corollary 2.9. *A code $C \leq \mathbb{F}_q^{m\ell}$ is ℓ-quasi-cyclic if and only if $C^\perp$ is ℓ-quasi-cyclic.*

Proof. Applying $C^\perp$ and $C = (C^\perp)^\perp$ to Proposition 2.8 yields the regular and converse implications, respectively. □

Theorem 2.10. *The set of ℓ-quasi-cyclic codes in a space $\mathbb{F}_q^{m\ell}$ is closed under taking sums, intersections, products, and duals.*

Proof. This follows from applying Propositions 2.5, 2.6, 2.7, and 2.8 to Theorem 5.12. □

2.2 Algebraic Properties

It is well known that a cyclic code C of length n can be viewed as an ideal of the quotient ring $\frac{\mathbb{F}_q[x]}{(x^n-1)}$. Under this framework, the structure of the ideal under multiplication by x and, by extension, any polynomial in $\frac{\mathbb{F}_q[x]}{(x^n-1)}$, imitates the structure of the cyclic code under the

shift operation. In particular, the homomorphism between C and R is given by

$$\begin{aligned} \phi : C &\to R \\ \mathbf{c} &\mapsto \sum_{i=0}^{n-1} c_i x^{n-1-i} + (x^n - 1). \end{aligned}$$

Under this homomorphism, we see that the shift function has a clean analogue in R. In particular,

$$\begin{aligned} \phi(\text{shift}(\mathbf{c})) &= c_1 x^{n-1} + \cdots + c_{n-1} x + c_0 + (x^n - 1) \\ &= c_0 x^n + c_1 x^{n-1} + \cdots + c_{n-1} x + (x^n - 1) \\ &= x(c_0 x^{n-1} + c_1 x^{n-2} + \cdots + c_{n-1}) + (x^n - 1) \\ &= x\phi(\mathbf{c}). \end{aligned}$$

From this, we also see that repeated shifting corresponds to repeated multiplication by x. We will now drop the $(x^n - 1)$ term but still take the elements to belong to the quotient ring. It is not hard to see that ϕ is $\mathbb{F}_q$-linear, so if we represent any codeword in C as a linear combination of shifts of $\mathbf{c}$,

$$\phi\left(\sum_{i=0}^{n-1} a_i \text{shift}^i(\mathbf{c})\right) = \sum_{i=0}^{n-1} \phi(a_i \text{shift}^i(\mathbf{c})) = \sum_{i=0}^{n-1} a_i \phi(\text{shift}^i(\mathbf{c})) = \sum_{i=0}^{n-1} a_i x^i \phi(\mathbf{c}) = \left(\sum_{i=0}^{n-1} a_i x^i\right) \phi(\mathbf{c}).$$

Thus, every linear combination of shifts of $\mathbf{c}$ has a corresponding polynomial in R. So the set of all linear combinations of shifts of $\mathbf{c}$, which is C, is isomorphic to the set of all polynomials times $\phi(\mathbf{c})$ in R. Therefore, C is isomorphic to the ideal generated by $\phi(\mathbf{c})$ in R.

Lally [6] extended this framework to quasi-cyclic codes. Now, we view an ℓ-quasi-cyclic code of length $m\ell$ as an $\frac{\mathbb{F}_q[x]}{(x^m-1)}$-submodule of the finitely-generated module $\left(\frac{\mathbb{F}_q[x]}{(x^m-1)}\right)^\ell$. The reason this works is that a shift by ℓ acts on m distinct parts of a codeword of length $m\ell$. Viewed

as a permutation on the entries of a codeword, we can write shift^ℓ as

$$\sigma_{\text{shift}^\ell} = (c_0\ c_m\ \cdots\ c_{m(\ell-1)})(c_1\ c_{m+1}\ \cdots\ c_{m(\ell-1)+1})\cdots(c_{m-1}\ \cdots\ c_{m(\ell-1)+(m-1)}).$$

We write a codeword $\mathbf{c}$ as a matrix

$$\begin{pmatrix} \mathbf{c}_0 \\ \vdots \\ \mathbf{c}_{m-1} \end{pmatrix},$$

where $\mathbf{c}_i$ is the vector made of all the ordered entries with indices congruent modulo m. Observe that

$$\text{shift}^\ell(\mathbf{c}) = \begin{pmatrix} \text{shift}(\mathbf{c}_0) \\ \vdots \\ \text{shift}(\mathbf{c}_{m-1}) \end{pmatrix}.$$

We can now define a homomorphism ψ which applies the ϕ used in the cyclic case to each codeword in the matrix representation of $\mathbf{c}$. Explicitly,

$$\psi(\mathbf{c}) = \begin{pmatrix} \phi(\mathbf{c}_0) \\ \vdots \\ \phi(\mathbf{c}_{m-1}) \end{pmatrix}.$$

This construction is useful because it preserves the shift structure from before, meaning

$$\psi(\text{shift}^\ell(\mathbf{c})) = \begin{pmatrix} \phi(\text{shift}(\mathbf{c}_0)) \\ \vdots \\ \phi(\text{shift}(\mathbf{c}_{m-1})) \end{pmatrix} = \begin{pmatrix} x\phi(\mathbf{c}_0) \\ \vdots \\ x\phi(\mathbf{c}_{m-1}) \end{pmatrix} = x\psi(\mathbf{c}).$$

As before, this also allows us to extend any polynomial in R to a linear combination of shifts of codewords in C.

While many analogues exist between cyclic and quasi-cyclic codes, some of the nicer properties of cyclic codes do not hold in the quasi-cyclic case. For example, a basic property of cyclic codes is that they can be described with a single generator polynomial[1].

Example 2.11. Let $C \leq \mathbb{F}_5^6$ be the code with generator matrix

$$\begin{pmatrix} 1 & 2 & 1 & 4 & 3 & 4 \\ 1 & 1 & 0 & 4 & 4 & 0 \\ 3 & 2 & 4 & 2 & 3 & 0 \end{pmatrix}.$$

It is clear that C is cyclic because another generator matrix for C is

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 4 & 4 \\ 1 & 1 & 0 & 4 & 4 & 0 \\ 1 & 0 & 4 & 4 & 0 & 1 \end{pmatrix}.$$

Therefore, $\phi(C)$ is generated by the polynomial corresponding to the codeword $(0, 1, 1, 0, 4, 4)$, which is $x^4 + x^3 - x - 1$.

Example 2.12. Let $C \leq \mathbb{F}_5^6$ be the 3-quasi-cyclic code with generator matrix

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 4 & 4 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

We see that as a $\frac{\mathbb{F}_q[x]}{\langle x^2-1 \rangle}$-submodule, C is generated by $(0, x-1, x-1)$ and $(0, 0, x+1)$. We see that there is no way to get the second tuple from the first nor the first tuple from the second. Thus, there is no single generator element in the $\frac{\mathbb{F}_5[x]}{\langle x^2-1 \rangle}$-submodule corresponding to C.

[1] Without loss of generality, the generator can be written as a factor of $x^m - 1$.

Because $\left(\frac{\mathbb{F}_q[x]}{\langle x^m-1\rangle}\right)^\ell$ is a finitely-generated $\frac{\mathbb{F}_q[x]}{\langle x^m-1\rangle}$-module, any ℓ-quasi-cyclic code must have a *set* of generators. This set can contain more than one element adding a great deal of complexity to the analysis of ℓ-quasi-cyclic codes for $\ell > 1$. It is also worth noting that single-generator quasi-cyclic codes do exist and are worth studying. Abdukhalikov et al. [1] provide an analysis of single-generator quasi-cyclic codes and their dual codes.

2.3 Quasi-Cyclic Codes Over Extension Fields

The construction which we will study in this thesis is the method described in [4] for constructing ℓ-quasi-cyclic extension codes from ℓ-quasi-cyclic codes. We now present this construction. Define $\psi(C)$ as the $\frac{\mathbb{F}_q[x]}{\langle x^m-1\rangle}$-submodule of $\left(\frac{\mathbb{F}_q[x]}{\langle x^m-1\rangle}\right)^\ell$ which is isomorphic to C. Let $\tilde{C}$ be the pre-image of $\psi(C)$ under the natural homomorphism $\pi : (\mathbb{F}_q[x])^\ell \to \left(\frac{\mathbb{F}_q[x]}{\langle x^m-1\rangle}\right)^\ell$. Since $\tilde{C}$ is an $\mathbb{F}_q[x]$-submodule of the free module $(\mathbb{F}_q[x])^\ell$, $\tilde{C}$ must be free. Let $\mathbf{u}_1(x), \ldots, \mathbf{u}_n(x) \in (\mathbb{F}_q[x])^\ell$ form a basis of $\tilde{C}$. We use these to construct

$$\tilde{U}(x) = \begin{pmatrix} u_{1,1}(x) & \cdots & u_{1,\ell}(x) \\ \vdots & \ddots & \vdots \\ u_{n,1}(x) & \cdots & u_{n,\ell}(x) \\ x^m - 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & x^m - 1 \end{pmatrix}.$$

We can reduce $\tilde{U}(x)$ to its unique Hermite normal form

$$\tilde{G}(x) = \begin{pmatrix} g_{1,1}(x) & g_{1,2}(x) & \cdots & g_{1,\ell}(x) \\ 0 & g_{2,2}(x) & \cdots & g_{2,\ell}(x) \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g_{\ell,\ell}(x) \end{pmatrix}.$$

Let $\mathbb{F}_{q^r}$ be a splitting field of $x^m - 1$ over $\mathbb{F}_q$. For a given $\lambda \in \mathbb{F}_{q^r}$ where $\det \tilde{G}(\lambda) = 0$, we define the right eigenspace

$$W_\lambda := \{\mathbf{v}\tilde{G}(\lambda) : \mathbf{v} \in \mathbb{F}_{q^r}^\ell\}.$$

If $\lambda_1, \ldots, \lambda_k$ are all the roots of $\det(\tilde{G}(x))$, then we define H_i as the parity check matrix of W_{λ_i}. We use these matrices to build

$$\hat{H} = \begin{pmatrix} p(\lambda_1) \otimes H_1 \\ p(\lambda_2) \otimes H_2 \\ \vdots \\ p(\lambda_k) \otimes H_k \end{pmatrix},$$

where $p(\lambda) := (1, \lambda, \lambda^2, \cdots, \lambda^{m-1})$ for $\lambda \in \mathbb{F}_{q^r}$. In Chimal-Dzul et al. [4], it is shown that $\hat{H}$ is a matrix with elements in $\mathbb{F}_{q^r}$ which serves as a parity check matrix the original code $C \leq \mathbb{F}_q^{m\ell}$.

Lastly, we define the code $\hat{C}$ as the code in $\mathbb{F}_{q^r}^{m\ell}$ with parity check matrix $\hat{H}$ and observe that $\hat{C}$ is ℓ-quasi-cyclic. We present those results on $\hat{C}$ which are most relevant to this work.

Proposition 2.13 ([4, Corollary 6]). *If G generates C over $\mathbb{F}_q^k$, then G generates $\hat{C}$ over $\mathbb{F}_{q^r}^k$.*

Proposition 2.14 ([4, Corollary 6]). *The restriction of $\hat{C}$ to $\mathbb{F}_q$ is C. That is, $C = \hat{C} \cap \mathbb{F}_q^{m\ell}$.*

Theorem 2.15 ([4, Lemma 7]). *For any ℓ-quasi-cyclic code $C \leq \mathbb{F}_q^{m\ell}$, $\mathrm{tr}(\hat{C}) = C$.*

Proof. This is a special case of Theorem 3.30, which will be proven later. □

Theorem 2.16 ([4, Theorem 11])**.** *If* $(x^{q^r-1} - 1) \mid \det(\tilde{G}(x))$ *and* $\alpha \in \mathbb{F}_{q^r}$ *is a primitive* m*-th root of unity, then* $\hat{C}$ *has a generator matrix of the form*

$$\begin{pmatrix} p(1) \otimes G_0 \\ p(\alpha) \otimes G_1 \\ \vdots \\ p(\alpha^{m-1}) \otimes G_{m-1} \end{pmatrix},$$

where each G_i *generates the row space of* $\tilde{G}(\alpha^{-i})$*.*

Remark 2.17. The existence of a primitive m-th root of unity is assumed when $\mathbb{F}_{q^r}$ is taken to be a splitting field of $x^m - 1$ over $\mathbb{F}_q$. It is also equivalent to the somewhat strict condition $m \mid q^r - 1$ (see Lemma 5.11 for a formal explanation). For any m and q, such an r exists exactly when $\gcd(m, q) = 1$. Furthermore, $r \mid \phi(q)$, where ϕ is Euler's totient function[2]. This is not always so bad, if r need not be fixed. For example, for $q = 2$ and odd m, choosing r as the order of $2 \in \mathbb{Z}_q$ is sufficient.

[2] $\phi(n) = |\{1 \leq m < n : \gcd(m, n) = 1\}|$.

Chapter 3

Decomposable Extensions of Quasi-Cyclic Codes

3.1 Decomposable Codes

Motivated by the structure of the generator matrix provided for the code $\hat{C}$ in [4], we wish to understand the broader class of codes of this form. Instead of defining such codes with the generator matrix, we will define such codes in $\mathbb{F}_{q^r}^{m\ell}$ as sums of specific tensor product codes of a vector involving α and codes in $\mathbb{F}_{q^r}^{m\ell}$. We want to understand what these codes look like under sums, intersections, products, and duals. Why this construction is worth studying and its relation to quasi-cyclic codes will be demonstrated.

Given an integer s, we use the notation $\overline{s}$ to denote the reduction of s modulo m, especially in this section.

Definition 3.1 (Decomposable Code)**.** An $[m\ell, k_\ell]$ code $C \leq \mathbb{F}_{q^r}^{m\ell}$ with $m \mid q^r - 1$ is said to be (α, ℓ)-decomposable if there exists primitive m-th root of unity $\alpha \in \mathbb{F}_{q^r}$ and codes $C_0, C_1, \ldots, C_{m-1} \leq \mathbb{F}_{q^r}^{\ell}$ such that

$$C = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i,$$

where $p : \mathbb{F}_{q^r} \to \mathbb{F}_{q^r}^m$ is defined by

$$p(\beta) := (1, \beta, \beta^2, \ldots, \beta^{m-1}) \in \mathbb{F}_{q^r}^m.$$

For the rest of this section, assume $\alpha \in \mathbb{F}_{q^r}$ is a primitive m-th root of unity and that the length of all codes mentioned is $m\ell$. We will say *decomposable* in place of (α, ℓ)-decomposable.

The motivation for decomposable codes comes from the structure of the generator matrix given in Theorem 2.15. We now show that decomposability is equivalent to having a generator matrix of the same form.

Proposition 3.2. *A code $C \leq \mathbb{F}_{q^r}^{m\ell}$ is decomposable if and only if C has a generator matrix of the form*

$$G = \begin{pmatrix} p(1) \otimes G_0 \\ p(\alpha) \otimes G_1 \\ \vdots \\ p(\alpha^{m-1}) \otimes G_{m-1} \end{pmatrix}.$$

Proof. Assume C is decomposable. Then we have $C_i \leq \mathbb{F}_{q^r}^{\ell}$ such that

$$C = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i.$$

We can create a generator matrix for C out of block matrices of the form $p(\alpha^i) \otimes C_i$, because we can find a basis for C by taking a basis for each C_i and taking the tensor product of $p(\alpha^i)$

with each member. Now, suppose C has a generator matrix

$$G = \begin{pmatrix} p(1) \otimes G_0 \\ \vdots \\ p(\alpha^{m-1}) \otimes G_{m-1} \end{pmatrix}$$

for some matrices $G_0, \cdots, G_{m-1}$. Let C_i be the code generated by G_i. Then

$$C = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i.$$

By definition, C is decomposable. □

A basic corollary of Proposition 3.2 is that we can *always* write the basis of a decomposable code in a decomposable form.

Corollary 3.3. *A decomposable code $C \leq \mathbb{F}_{q^r}^{m\ell}$ has a basis where every codeword is of the form $p(\alpha^i) \otimes \mathbf{c}$ for some $0 \leq i \leq m-1$.*

Proof. Take G to be the generator matrix for C in the form given in Proposition 3.2. Every row of G is of the form $p(\alpha^i) \otimes \mathbf{c}$ for some $\mathbf{c}$ where $\mathbf{c}$ is a row of G_i. □

An important property of decomposable codes is that their decompositions are not redundant in the sense that no code of the form $p(\alpha^i) \otimes C_i$ can be created by linear combinations of codewords of the other codes in the decomposition. This property will help us prove the next lemma.

Lemma 3.4. *For any $\mathbf{c}_0, \ldots, \mathbf{c}_{m-1} \in \mathbb{F}_{q^r}^{m\ell}$ and any $0 \leq j < m$ such that $\mathbf{c}_j \neq 0$,*

$$p(\alpha^j) \otimes \mathbf{c}_j \notin \operatorname{span}\{p(\alpha^i) \otimes \mathbf{c}_i\}_{i \neq j}.$$

Proof. Suppose, towards contradiction, that $p(\alpha^j) \otimes \mathbf{c}_j \in \text{span}\{p(\alpha^i) \otimes \mathbf{c}_i\}_{i \neq j}$. Then there exist $\beta_0, \ldots, \beta_{m-1} \in \mathbb{F}_{q^r}$ such that

$$p(\alpha^j) \otimes \mathbf{c}_j = \sum_{i \neq j} \beta_i (p(\alpha^i) \otimes \mathbf{c}_i) = \sum_{i \neq j} p(\alpha^i) \otimes (\beta_i \mathbf{c}_i).$$

Breaking up each tensor product vector into its original components,

$$\begin{aligned} \mathbf{c}_j &= \textstyle\sum_{i \neq j} \beta_i \mathbf{c}_i \\ \alpha^j \mathbf{c}_j &= \textstyle\sum_{i \neq j} \alpha^i \beta_i \mathbf{c}_i \\ &\vdots \\ \alpha^{(m-1)j} \mathbf{c}_j &= \textstyle\sum_{i \neq j} \alpha^{(m-1)i} \beta_i \mathbf{c}_j. \end{aligned}$$

In fact, for any $0 \leq s < m$,

$$\alpha^{sj} \mathbf{c}_j = \sum_{i \neq j} \alpha^{si} \beta_i \mathbf{c}_j.$$

Breaking each vector into its individual entries gives us, for any $0 \leq t < \ell$,

$$\alpha^{sj} c_{j,t} = \sum_{i \neq j} \alpha^{si} \beta_i c_{i,t}.$$

For all $0 \leq t < \ell$, we define

$$f_t(x) := -c_{j,t} x^j + \sum_{i \neq j} \beta_i c_{i,t} x^i.$$

Observe that we know $f_t(\alpha^s) = 0$ for any $0 \leq s < m$, so

$$\prod_{s=0}^{m-1} (x - \alpha^s) = x^m - 1$$

is a divisor of $f_t(x)$. However, it must also be true that $\deg f_t(x) \leq m - 1$ from our con-

struction, so this is a contradiction. □

A useful fact about decomposable codes is that we can find the dimension of a decomposable code using the dimensions of the codes that construct it. We will need this property to characterize the dual of a decomposable code in the next proposition.

Lemma 3.5. *For any codes* $C_0, \ldots, C_{m-1} \leq \mathbb{F}_{q^r}^{\ell}$,

$$\dim\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i\right) = \sum_{i=0}^{m-1} \dim(C_i).$$

Proof. Define, for $0 \leq i < m$,

$$\tilde{C}_i := (p(\alpha^i) \otimes C_i) + \tilde{C}_{i-1}$$

with $\tilde{C}_0 = p(1) \otimes C_0$. Assume $\dim(\tilde{C}_n) = \sum_{i=0}^{n} \dim(C_i)$ for some $0 \leq n < m$. Lemma 3.4 tells us

$$(p(\alpha^{n+1}) \otimes C_{n+1}) \cap \tilde{C}_n \leq (p(\alpha^{n+1}) \otimes C_{n+1}) \cap \operatorname{span}\{p(\alpha^i) \otimes C_i\}_{i \neq j} = \{\mathbf{0}\}$$

if $C_{n+1} \neq \{\mathbf{0}\}$. In this case,

$$\dim(\tilde{C}_{n+1}) = \dim(p(\alpha^{n+1}) \otimes C_{n+1}) + \dim(\tilde{C}_n) = \sum_{i=0}^{n+1} \dim(C_i).$$

If $C_{n+1} = \{\mathbf{0}\}$, then

$$\dim(\tilde{C}_{n+1}) = \dim(\tilde{C}_n) = \sum_{i=0}^{n} \dim(C_i) = \sum_{i=0}^{n+1} \dim(C_i).$$

In either case,

$$\dim(\tilde{C}_{n+1}) = \sum_{i=0}^{n+1} \dim(C_i).$$

We also have $\dim(\tilde{C}_0) = \dim C_0$ because the tensor product is bilinear and $p(1)$ is non-zero. Therefore, by induction,

$$\dim\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i\right) = \dim(\tilde{C}_{m-1}) = \sum_{i=0}^{m-1} \dim(C_i). \qquad \square$$

The property of decomposable codes which makes them particularly useful is the structure of their dual codes. Indeed, dreams of this structure motivated the authors to form decomposable codes in the first place. We now show and prove the structure of the dual of a decomposable code.

Proposition 3.6. *If a code $C \leq \mathbb{F}_{q^r}^{m\ell}$ can be decomposed as*

$$C = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i,$$

then $C^\perp$ can be decomposed as

$$C^\perp = \sum_{j=0}^{m-1} p(\alpha^j) \otimes C^\perp_{\overline{m-j}}.$$

Proof. Suppose

$$C = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i.$$

By Proposition 3.2, there exists a generator matrix of the form

$$G = \begin{pmatrix} p(1) \otimes G_0 \\ p(\alpha) \otimes G_1 \\ \vdots \\ p(\alpha^{m-1}) \otimes G_{m-1} \end{pmatrix},$$

where G_i generates C_i. We now construct a matrix

$$H = \begin{pmatrix} p(1) \otimes G_0^{\perp} \\ p(\alpha) \otimes G_{m-1}^{\perp} \\ p(\alpha^2) \otimes G_{m-2}^{\perp} \\ \vdots \\ p(\alpha^{m-1}) \otimes G_1^{\perp} \end{pmatrix}.$$

We compute the block matrix

$$GH^T = \begin{pmatrix} M_{0,0} & \cdots & M_{0,n-k+1} \\ \vdots & \ddots & \vdots \\ M_{k-1,0} & \cdots & M_{k-1,n-k+1} \end{pmatrix},$$

where each block $M_{i,j}$ (with $0 \leq i \leq k-1$ and $0 \leq j \leq n-k+1$) is of the form

$$M_{i,j} = \sum_{s=0}^{m-1} (\alpha^{ri} G_i)(\alpha^{rj} G_{\overline{m-j}}^{\perp})^T = G_i (G_{\overline{m-j}}^{\perp})^T \sum_{r=0}^{m-1} (\alpha^{i+j})^r.$$

If $i + j \neq 0 \bmod m$, then $\alpha^m = 1$ and $\alpha^{m(i+j)} = 1$, which yields

$$M_{i,j} = G_i (G_{m-j}^{\perp})^T \sum_{r=0}^{m-1} (\alpha^{i+j})^r = G_i (G_{m-j}^{\perp})^T \cdot \frac{\alpha^{m(i+j)} - 1}{\alpha^{i+j} - 1} = \mathbf{0}.$$

If $i + j = 0 \bmod m$, then $j = \overline{m - i}$, so

$$M_{i,j} = G_i(G^{\perp}_{\overline{m-(m-i)}})^T \sum_{s=0}^{m-1} 1 = mG_i(G_i^{\perp})^T = \mathbf{0}.$$

Therefore, $GH^T = \mathbf{0}$. Turning our attention to finding the rank of H, by definition H generates the code

$$D = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C^{\perp}_{\overline{m-i}}.$$

Using Lemma 3.5,

$$\begin{aligned}
\operatorname{rank}(H) &= \dim(D) \\
&= \dim\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes C^{\perp}_{\overline{m-i}}\right) \\
&= \sum_{i=0}^{m-1} \dim(C^{\perp}_{\overline{m-i}}) \\
&= \sum_{i=0}^{m-1} \ell - \dim(C_{\overline{m-i}}) \\
&= m\ell - \sum_{i=0}^{m-1} \dim(C_i).
\end{aligned}$$

Applying Lemma 3.5 again,

$$\begin{aligned}
\operatorname{rank}(H) &= m\ell - \sum_{i=0}^{m-1} \dim(C_i) \\
&= m\ell - \dim\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i\right) \\
&= m\ell - \dim(C) \\
&= \dim(C^{\perp}).
\end{aligned}$$

Therefore, H generates $C^\perp$, so $C^\perp$ is decomposable by Proposition 3.2. Because H generates $C^\perp$, it follows that

$$C^\perp = \sum_{j=0}^{m-1} p(\alpha^j) \otimes C^\perp_{\overline{m-j}}.$$ □

Theorem 3.7. *A code $C \leq \mathbb{F}_{q^r}^{m\ell}$ is decomposable if and only if $C^\perp$ is decomposable.*

Proof. Proposition 3.6 gives us the reverse implication of the statement. The forward implication follows from the fact that if $C^\perp$ is decomposable, then Proposition 3.6 implies that $C = (C^\perp)^\perp$ is decomposable. □

Besides closure under duals, we also see that decomposable codes are closed under a few other basic operations.

Proposition 3.8. *If $A, B \leq \mathbb{F}_{q^r}^{m\ell}$ are decomposable codes, then $A+B$ is a decomposable code.*

Proof. We can use the decompositions of A and B as

$$A+B = \sum_{i=0}^{m-1} p(\alpha^i) \otimes A_i + \sum_{i=0}^{m-1} p(\alpha^i) \otimes B_i = \sum_{i=0}^{m-1} p(\alpha^i) \otimes A_i + p(\alpha^i) \otimes B_i = \sum_{i=0}^{m-1} p(\alpha^i) \otimes (A_i + B_i)$$

so $A + B$ is decomposable. □

Code intersections are not always trivial, but intersections of decomposable codes are structured enough that things become somewhat simpler. In particular, the intersection of two decomposable codes can be found by finding the intersections of the corresponding parts of their decompositions.

Proposition 3.9. *If $A, B \leq \mathbb{F}_{q^r}^{m\ell}$ are decomposable codes, then $A \cap B$ is a decomposable code.*

Specifically,

$$\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes A_i\right) \cap \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes B_i\right) = \sum_{i=0}^{m-1} p(\alpha^i) \otimes (A_i \cap B_i).$$

Proof. We know by Theorem 3.7 that $A^\perp$ and $B^\perp$ are decomposable. Therefore, by Proposition 3.8, $A^\perp + B^\perp = (A \cap B)^\perp$ is decomposable, and by Theorem 3.7, $A \cap B$ is decomposable. Writing this out formally for two decomposable codes,

$$\begin{aligned}
\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes A_i\right) \cap \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes B_i\right) &= \left(\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes A_i\right)^\perp + \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes B_i\right)^\perp\right)^\perp \\
&= \left(\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes A^\perp_{\overline{m-i}}\right) + \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes B^\perp_{\overline{m-i}}\right)\right)^\perp \\
&= \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes A^\perp_{\overline{m-i}} + p(\alpha^i) \otimes B^\perp_{\overline{m-i}}\right)^\perp \\
&= \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes (A^\perp_{\overline{m-i}} + B^\perp_{\overline{m-i}})\right)^\perp \\
&= \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes (A_{\overline{m-i}} \cap B_{\overline{m-i}})^\perp\right)^\perp \\
&= \sum_{i=0}^{m-1} p(\alpha^i) \otimes (A_i \cap B_i),
\end{aligned}$$

where the expansion of the dual code is provided by Proposition 3.6. □

It turns out that decomposable codes have a very nice structure under Schur products. We will characterize this structure and then use it to observe the closure of decomposable codes under Schur products. While it is not as clean as in the sum or intersection case, it is certainly preferable to the complete lack of structure informed by the naïve approach.

Lemma 3.10. *For any codes $A_0, \ldots, A_{m-1}, B_1, \ldots, B_{m-1} \leq \mathbb{F}_{q^r}^{m\ell}$,*

$$\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes A_i\right) * \left(\sum_{j=0}^{m-1} p(\alpha^j) \otimes B_j\right) = \sum_{k=0}^{m-1} p(\alpha^k) \otimes \left[\sum_{i+j=0 \bmod m} A_i * B_j\right].$$

Proof. Observe that

$$\begin{aligned}
\left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes A_i\right) * \left(\sum_{j=0}^{m-1} p(\alpha^j) \otimes B_j\right) &= \sum_{i=0}^{m-1}\sum_{j=0}^{m-1} (p(\alpha^i) \otimes A_i) * (p(\alpha^j) \otimes B_j) \\
&= \sum_{k=0}^{m-1} \sum_{i+j=k \bmod m} (p(\alpha^i) \otimes A_i) * (p(\alpha^j) \otimes B_j) \\
&= \sum_{k=0}^{m-1} \sum_{i+j=k \bmod m} p(\alpha^k) \otimes (A_i * B_j) \\
&= \sum_{k=0}^{m-1} p(\alpha^k) \otimes \left[\sum_{i+j=k \bmod m} A_i * B_j\right].
\end{aligned}$$

□

Proposition 3.11. *If $A, B \leq \mathbb{F}_{q^r}^{m\ell}$ are decomposable codes, then $A*B$ is a decomposable code.*

Proof. Because A and B are decomposable, there exist codes $A_0, \ldots, A_{m-1} \leq \mathbb{F}_{q^r}^{\ell}$ and $B_0, \ldots, B_{m-1} \leq \mathbb{F}_{q^r}^{\ell}$ where

$$A = \sum_{i=0}^{m-1} p(\alpha^i) \otimes A_i \qquad \text{and} \qquad B = \sum_{j=0}^{m-1} p(\alpha^j) \otimes B_j.$$

Then, from Lemma 3.10,

$$A * B = \sum_{k=0}^{m-1} p(\alpha^k) \otimes \left[\sum_{i+j=k \bmod m} A_i * B_j\right],$$

so $A * B$ is decomposable. □

We spend the next lemma and proposition on the uniqueness of decomposable codes. We

will soon see that this gives us a clean characterization of self-dual decomposable codes.

Lemma 3.12. *For any codes $C_0, \ldots, C_{m-1} \leq \mathbb{F}_{q^r}^{\ell}$, for any $0 \leq j \leq m-1$,*

$$\left[\sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i\right] \cap [p(\alpha^j) \otimes \mathbb{F}_{q^r}^{\ell}] = p(\alpha^j) \otimes C_j.$$

Proof. Proposition 3.9 tells us that we can use component-wise intersections to find the intersection of two decomposable codes, so

$$\left[\sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i\right] \cap [p(\alpha^j) \otimes \mathbb{F}_{q^r}^{\ell}] = p(\alpha^j) \otimes C_j. \qquad \square$$

Proposition 3.13. *The decomposition of a decomposable code $C \leq \mathbb{F}_{q^r}^{m\ell}$ into the form*

$$C = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i$$

is unique.

Proof. Suppose, towards contradiction, that C is a decomposable code. Then it can be written as

$$C = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i'$$

for some codes where C_i and C_i' are potentially different codes. Then, for a fixed $0 \leq j \leq m-1$, we use Lemma 3.12 to see that

$$\begin{aligned} p(\alpha^j) \otimes C_j &= \left[\sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i\right] \cap [p(\alpha^j) \otimes \mathbb{F}_{q^r}^{\ell}] \\ &= \left[\sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i'\right] \cap [p(\alpha^j) \otimes \mathbb{F}_{q^r}^{\ell}] \\ &= p(\alpha^j) \otimes C_j'. \end{aligned}$$

Therefore, for any codeword $\mathbf{c} \in C_j$, there exists $\mathbf{c}'_j \in C'_j$ such that $p(\alpha^j) \otimes \mathbf{c}_j = p(\alpha^j) \otimes \mathbf{c}'_j$. Because the first entry of $p(\alpha^j)$ must be 1, we conclude that $\mathbf{c}_j = \mathbf{c}'_j$. □

Theorem 3.14. *Decomposable codes in $\mathbb{F}_q^{m\ell}$ are closed under finite expressions involving sums, intersections, products, or duals.*

Proof. This follows from Theorem 5.12. □

One nice result of the decomposable code framework and the uniqueness of decomposition is that the description of self-dual decomposable codes is precise. We can find an exact condition on the types of component codes which form a self-dual decomposable code.

Proposition 3.15. *A decomposable code C with decomposition*

$$C = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i$$

is self-dual if and only if $C_{\overline{m-i}} = C_i^\perp$ for all $0 \leq i \leq m-1$.

Proof. If $C_{m-i} = C_i^\perp$ for all $0 \leq i \leq m-1$, then

$$C^\perp = \sum_{i=0}^{m=1} p(\alpha^i) \otimes C_{\overline{m-i}}^\perp = \sum_{i=0}^{m=1} p(\alpha^i) \otimes (C_i^\perp)^\perp = \sum_{i=0}^{m-1} p(\alpha) \otimes C_i = C,$$

so C is self-dual. If there is some $0 \leq j \leq m-1$ where $C_{\overline{m-j}} \neq C_j^\perp$, then the uniqueness condition from Proposition 3.13 gives us

$$C^\perp = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_{\overline{m-i}}^\perp \neq \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i = C.$$

Therefore, C is not self-dual. □

Despite this section, this thesis is not *about* decomposable codes. The overall desired focus of this thesis is quasi-cyclic codes. We now demonstrate that, as this segment forms part of a larger piece on quasi-cyclic codes, decomposable codes are but a special class of quasi-cyclic codes.

Theorem 3.16. *If $C \leq \mathbb{F}_{q^r}^{m\ell}$ is decomposable, then C is ℓ-quasi-cyclic.*

Proof. Suppose C is decomposable with

$$C = \sum_{i=0}^{m-1} p(\alpha^i) \otimes C_i$$

for $C_0, \ldots, C_{m-1} \leq \mathbb{F}_{q^r}^{\ell}$. Take an arbitrary codeword $p(\alpha^i) \otimes \mathbf{c}$ from the basis given in Corollary 3.3. Then

$$\begin{aligned} \text{shift}^{\ell}(p(\alpha^i) \otimes \mathbf{c}) &= \text{shift}^{\ell}((\mathbf{c}\ \alpha^i\mathbf{c}\ \cdots\ \alpha^{(m-1)i}\mathbf{c})) \\ &= (\alpha^i\mathbf{c}\ \cdots\ \alpha^{(m-1)i}\mathbf{c}\ \mathbf{c}) \\ &= \alpha^i(\mathbf{c}\ \alpha^i\mathbf{c}\ \cdots\ \alpha^{(m-1)i}\mathbf{c}) \\ &= \alpha^i(p(\alpha^i) \otimes \mathbf{c}). \end{aligned}$$

Therefore, $\text{shift}^{\ell}(p(\alpha^i) \otimes \mathbf{c}) \in C$. Because $p(\alpha^i) \otimes \mathbf{c}$ was taken as an arbitrary basis element, it follows that C is ℓ-quasi-cyclic. □

While the statement of Theorem 3.16 is clean, we break it into two cases. To describe self-dual decomposable codes, we depend on the parity of m. In this light, the next corollary paints a fuller mural of self-dual decomposable codes.

Corollary 3.17. *We can construct a self-dual decomposable code in* $\mathbb{F}_{q^r}^{m\ell}$ *as*

$$p(1) \otimes C_0 + \sum_{i=1}^{\lfloor m/2 \rfloor} p(\alpha^i) \otimes C_i + \sum_{i=\lfloor m/2 \rfloor + 1}^{m-1} p(\alpha^i) \otimes C_{m-i}^{\perp}$$

with any codes $C_0, \ldots, C_{\lfloor m/2 \rfloor} \leq \mathbb{F}_{q^r}^{\ell}$*, where* C_0 *is self-dual and where* $C_{\lfloor m/2 \rfloor}$ *is self-dual if* m *is even. Furthermore,* all *self-dual decomposable codes follow this construction.*

Proof. Any such code conforms to the conditions of Proposition 3.15, so the code is self-dual. That all self-dual codes follow this construction also tracks from Proposition 3.15. □

Remark 3.18. 3.16 tells us that Corollary 3.17 also provide recipes for self-dual ℓ-quasi-cyclic codes, although only the *decomposable* self-dual ℓ-quasi-cyclic codes in $\mathbb{F}_{q^r}^{m\ell}$ follow this construction.

3.2 Traces of Extension Codes

In this section, we take the construction of $\hat{C} \leq \mathbb{F}_{q^r}^{m\ell}$ from an ℓ-quasi-cyclic code $C \leq \mathbb{F}_q^{m\ell}$ and study the more general concept of codes over extension fields with the same basis as a code over a base field. The results in this section gradually build towards the proof of Theorem 3.28, which provides a strong result on the traces of expressions of code operations on extension codes.

Definition 3.19. For a code $C \leq \mathbb{F}_q^n$ and an extension field $\mathbb{F}_{q^r}/\mathbb{F}_q$, the *extension code* of C in $\mathbb{F}_{q^r}$ is

$$\hat{C} := \langle \alpha \mathbf{c} : \alpha \in \mathbb{F}_{q^r}, \mathbf{c} \in C \rangle \leq \mathbb{F}_{q^r}^n.$$

Definition 3.20. The *extension operator* $\widehat{\ }$ is the operator which maps a code $C \leq \mathbb{F}_q^n$ to its extension $\hat{C} \leq \mathbb{F}_{q^r}^n$.

We will use the notations $C^{\frown}$ and $\hat{C}$ interchangeably, i.e. $C^{\frown} := \hat{C}$.

In the next few results, we show that the extension operator is very well-behaved under sums, products, intersections, and duals. This will be useful for understanding what the traces of these operations taken on extension fields look like at the end of this section.

Proposition 3.21. *For any codes $C_1, C_2 \leq \mathbb{F}_q^{m\ell}$, $\widehat{C_1 + C_2} = \hat{C}_1 + \hat{C}_2$.*

Proof. If we have generators

$$G_1 = \begin{pmatrix} \mathbf{c}_{1,1} \\ \vdots \\ \mathbf{c}_{1,k_1} \end{pmatrix} \qquad \text{and} \qquad G_2 = \begin{pmatrix} \mathbf{c}_{2,1} \\ \vdots \\ \mathbf{c}_{2,k_2} \end{pmatrix}$$

of C_1 and C_2, respectively, then $G := \begin{pmatrix} G_1 \\ G_2 \end{pmatrix}$ must be a generator for $C_1 + C_2$ because every codeword in $C_1 + C_2$ must be a sum of codewords in either code, each of which is a sum of codewords in the respective basis. Because $\widehat{C_1 + C_2}$ is just the code generated by G extended to $\mathbb{F}_{q^r}$, G generates $\widehat{C_1 + C_2}$ by definition. Then for any element of $\tilde{\mathbf{a}} + \tilde{\mathbf{b}} \in \hat{C}_1 + \hat{C}_2$, there exist some $\mathbf{a} \in \mathbb{F}_{q^r}^{k_1}$ and $\mathbf{b} \in \mathbb{F}_{q^r}^{k_2}$ where $\tilde{\mathbf{a}} = \mathbf{a}G_1$ and $\tilde{\mathbf{b}} = \mathbf{b}G_2$. Therefore,

$$\tilde{\mathbf{a}} + \tilde{\mathbf{b}} = \sum_{i=0}^{k_1} a_i \mathbf{c}_{1,i} + \sum_{j=0}^{k_2} b_j \mathbf{c}_{2,j},$$

so G generates $\hat{C}_1 + \hat{C}_2$. Thus, $\widehat{C_1 + C_2} = \hat{C}_1 + \hat{C}_2$. □

Proposition 3.22. *For any codes $C_1, \ldots, C_n \leq \mathbb{F}_q^{m\ell}$,*

$$(C_1 + \cdots + C_n)^{\frown} = \hat{C}_1 + \cdots + \hat{C}_n.$$

Proof. We proceed by induction. We will assume

$$(C_1 + \cdots + C_{n-1})\hat{} = \hat{C}_1 + \cdots + \hat{C}_{n-1}.$$

We know that $C_1 + \cdots + C_{n-1}$ is a code in $\mathbb{F}_q^n$, so we can apply Proposition 3.21 as

$$(C_1+\cdots+C_n)\hat{} = ((C_1+\cdots+C_{n-1})+C_n)\hat{} = (C_1+\cdots+C_{n-1})\hat{}+\hat{C}_n = \hat{C}_1+\cdots+\hat{C}_{n-1}+\hat{C}_n.$$

The case where $n = 1$ must be true because we have $(C)\hat{} = \hat{C}$. Therefore, by induction, the statement holds for all $n > 0$. □

Proposition 3.23. *For any codes $C_1, C_2 \leq \mathbb{F}_q^n$, $\widehat{C_1 * C_2} = \hat{C}_1 * \hat{C}_2$.*

Proof. Suppose G_1 and G_2 generate C_1 and C_2, respectively, with

$$G_1 = \begin{pmatrix} \mathbf{c}_{1,1} \\ \vdots \\ \mathbf{c}_{1,k_1} \end{pmatrix} \qquad \text{and} \qquad G_2 = \begin{pmatrix} \mathbf{c}_{2,1} \\ \vdots \\ \mathbf{c}_{2,k_2} \end{pmatrix}.$$

Then, observe that the matrix

$$G_* = \begin{pmatrix} \mathbf{c}_{1,1} * \mathbf{c}_{2,1} \\ \vdots \\ \mathbf{c}_{1,1} * \mathbf{c}_{2,k_2} \\ \mathbf{c}_{1,k_1} * \mathbf{c}_{2,1} \\ \vdots \\ \mathbf{c}_{1,k_1} * \mathbf{c}_{2,k_2} \end{pmatrix}$$

generates $C_1 * C_2$. From Proposition 2.13, we have that $\widehat{C_1 * C_2}$ is generated by G_* over $\mathbb{F}_{q^r}^n$. Now, take an arbitrary element in $\hat{C}_1 * \hat{C}_2$, that is an arbitrary $\tilde{\mathbf{a}} \in C_1$ and $\tilde{\mathbf{b}} \in C_1$. Then

we have $\mathbf{a} \in \mathbb{F}_{q^r}^{k_1}$ and $\mathbf{b} \in \mathbb{F}_{q^r}^{k_2}$ such that $\tilde{\mathbf{a}} = \mathbf{a}G_1$ and $\tilde{\mathbf{b}} = \mathbf{b}G_2$. Then we have

$$\tilde{\mathbf{a}} * \tilde{\mathbf{b}} = \left(\sum_{i=1}^{k_1} a_i \mathbf{c}_{1,i}\right) * \left(\sum_{j=1}^{k_2} b_i \mathbf{c}_{2,j}\right) \sum_{i=1}^{k_1}\sum_{j=1}^{k_2} (a_i \mathbf{c}_{1,i}) * (b_j \mathbf{c}_{2,j}) \sum_{i=1}^{k_1}\sum_{j=1}^{k_2} (a_i b_j)(\mathbf{c}_{1,i} * \mathbf{c}_{2,j}),$$

so G_* generates $\hat{C}_1 * \hat{C}_2$ over $\mathbb{F}_{q^r}^n$. Thus, $\widehat{C_1 * C_2} = \hat{C}_1 * \hat{C}_2$. □

Proposition 3.24. *For any codes $C_1, \ldots, C_n \leq \mathbb{F}_q^n$,*

$$(C_1 * \cdots * C_s)\hat{} = \hat{C}_1 * \cdots * \hat{C}_s.$$

Proof. We proceed by induction. Suppose

$$(C_1 * \cdots * C_{n-1})\hat{} = \hat{C}_1 * \cdots * \hat{C}_{n-1}.$$

We know that $C_1 * \cdots * C_{n-1}$ is a code, so we can apply Proposition 3.23 to obtain

$$(C_1 * \cdots * C_n)\hat{} = ((C_1 * \cdots * C_{n-1}) * C_n)\hat{} = (C_1 * \cdots * C_{n-1})\hat{} * \hat{C}_n = \hat{C}_1 * \cdots * \hat{C}_{n-1} * \hat{C}_n.$$

We observe that the base case $n = 1$ is trivially true because $(C)\hat{} = \hat{C}$, so the statement holds for all integers $n > 0$ by induction. □

Proposition 3.25. *For any code $C \leq \mathbb{F}_q^n$, $[C^\perp]\hat{} = [\hat{C}]^\perp$.*

Proof. Let H be a generator matrix for $C^\perp$, i.e. a parity-check matrix for C. Then $GH^T = 0$. Because G generates $\hat{C}$ over $\mathbb{F}_{q^r}$, H must generate a subcode of $\hat{C}^\perp$. Furthermore, because G generates C and $\hat{C}$, $\dim(C) = \dim(\hat{C})$. This implies

$$\operatorname{rank}(H) = \dim(C^\perp) = n - \dim(C) = n - \dim(\hat{C}) = \dim(C^\perp).$$

Thus, H generates $\hat{C}^{\perp}$. By definition, H generates $[C^{\perp}]\hat{} \leq FF_{q^r}^n$, so $[C^{\perp}]\hat{} = \hat{C}^{\perp}$. □

Lemma 3.26. *For any codes $C_1, C_2 \leq \mathbb{F}_q^n$, $\widehat{C_1 \cap C_2} = \hat{C}_1 \cap \hat{C}_2$.*

Proof. We use Propositions 3.24 and 3.25 to get

$$[C_1 \cap C_2]\hat{} = [(C_1^{\perp} + C_2^{\perp})^{\perp}]\hat{} = ([C_1^{\perp} + C_2^{\perp}]\hat{})^{\perp} = (\hat{C}_1^{\perp} + \hat{C}_2^{\perp})^{\perp} = \hat{C}_1 \cap \hat{C}_2.$$ □

Proposition 3.27. *For any codes $C_1, \ldots, C_s \leq \mathbb{F}_q^n$,*

$$(C_1 \cap \cdots \cap C_s)\hat{} = \hat{C}_1 \cap \cdots \cap \hat{C}_s.$$

Proof. We proceed by induction. By definition, $(C)\hat{} = \hat{C}$ holds for any code $C \leq \mathbb{F}_q^n$. Suppose

$$(C_1 \cap \cdots \cap C_{s-1})\hat{} = \hat{C}_1 \cap \cdots \cap \hat{C}_{s-1}$$

for any codes $C_1, \cdots, C_{s-1} \leq \mathbb{F}_q^n$. Then, for any codes $C_1, \ldots, C_n \leq \mathbb{F}_q^n$, Lemma 3.26 implies

$$(C_1 \cap \cdots \cap C_n)\hat{} = ([C_1 \cap \cdots \cap C_{s-1}] \cap C_s)\hat{} = (C_1 \cap \cdots \cap C_{s-1})\hat{} \cap \hat{C}_s = \hat{C}_1 \cap \cdots \cap \hat{C}_{s-1} \cap \hat{C}_s.$$

Because we have completed the base case $s = 1$, the statement holds for all $n \geq 1$ by induction. □

We will combine these last few results to say something much more general about how these operations behave under the extension operator. In fact, the extension of any expression on these operations results in the same code as taking the expression with the extensions as inputs.

Theorem 3.28. *For any function f which can be expressed in a closed form in terms of*

sums, products, intersections, or dual operators on codes $C_1, \ldots, C_s \leq \mathbb{F}_q^n$,

$$[f(C_1, \ldots, C_s)]\hat{} = f(\hat{C}_1, \ldots, \hat{C}_s).$$

Proof. Take a function f with a closed form expression $\mathcal{T}$ involving sums, products, intersections, and duals of codes $C_1, \ldots, C_s \leq \mathbb{F}_q^n$. Without loss of generality, we can assume that each term in $\mathcal{T}$ is unique (i.e. no specific code C_i appears twice in the expression). We proceed using induction. Suppose that for every function g with a closed form in less than s terms, where s is the number of terms in $\mathcal{T}$. Then

$$[g(C_1, \ldots, C_t)]\hat{} = g(\hat{C}_1, \ldots, \hat{C}_t).$$

For notational convenience, let us describe the inputs $C_1, \ldots, C_s$ as the ordered set $\mathcal{C}$. We know that every closed form can be computed in finite time, so, once the order of operations is applied, we have a top layer which can take one of two forms. In the first, we can write

$$f(\mathcal{C}) = h_1(\mathcal{C}_1) \circ \cdots \circ h_k(\mathcal{C}_k)$$

for some $k \geq 2$ where $\circ$ is an operation which denotes either $+$, $*$, or $\cap$ and where each $\mathcal{C}_i$ is non-empty and represents an ordered set of inputs with

$$\mathcal{C}_1 \sqcup \cdots \sqcup \mathcal{C}_k = \{C_1, \ldots, C_s\}.$$

Note that the disjoint partition is not problematic, because each code in the input can only appear once. Because the closed form of each $h_i(\mathcal{C}_i)$ can contain at most $n - k + 1$ variables (allowing for at least one other code in the input of each other function), we can apply our

assumption and get

$$[f(C_1, \ldots, C_s)]^\frown = [h_1(\mathcal{C}_1) \circ \cdots \circ h_k(\mathcal{C}_k)]^\frown.$$

Because $h_, \ldots, h_k$ are functions which can be expressed as a non-empty combination of finite sums, products, and intersections of distinct codes, $[h_i(\mathcal{C}_i)]^\frown = h_i(\hat{\mathcal{C}}_i)$[1]. Because $\circ$ is either $+$, $*$, or $\cap$, we can use prior results to obtain

$$\begin{aligned}
[f(\mathcal{C})]^\frown &= [h_1(\mathcal{C}_1) \circ \cdots \circ h_k(\mathcal{C}_k)]^\frown \\
&= [h_1(\mathcal{C}_1)]^\frown \circ \cdots \circ [h_k(\mathcal{C}_k)]^\frown \\
&= h_1(\hat{\mathcal{C}}_1) \circ \cdots \circ h_k(\hat{\mathcal{C}}_k) \\
&= f(\hat{\mathcal{C}}).
\end{aligned}$$

The other case is when $f(\mathcal{C}) = [g(\mathcal{C})]^\perp$ for some function g with a closed expression in at most n terms. Without loss of generality, we can assume g can be decomposed in the same way as in the first case of f, so

$$g(\mathcal{C}) = h_1(\mathcal{C}_1) \circ \cdots \circ h_k(\mathcal{C}_k).$$

In the same way as before, $[g(\mathcal{C})]^\frown = g(\hat{\mathcal{C}})$. Also as before, we know that g describes some code in $\mathbb{F}_q^n$, so

$$\begin{aligned}
[f(\mathcal{C})]^\frown &= [[g(\mathcal{C})]^\perp]^\frown \\
&= [[g(\mathcal{C})]^\frown]^\perp \\
&= [g(\hat{\mathcal{C}})]^\perp \\
&= f(\hat{\mathcal{C}}).
\end{aligned}$$

[1] We use the notation $\hat{\mathcal{C}}$ to represent the $\frown$ operation applied to each individual code in $\hat{\mathcal{C}}$.

Thus, in either case, we get $[f(\mathcal{C})]\hat{} = f(\hat{\mathcal{C}})$. We now prove the base case $m = 1$. There are only two closed form expressions with a single term, $f(C) := C$ and $g(C) := C^{\perp}$. In either case,

$$[f(C)]\hat{} = C\hat{} = \hat{C} = f(\hat{C}) \qquad \text{or} \qquad [g(C)]\hat{} = (C^{\perp})\hat{} = (\hat{C})^{\perp} = g(\hat{C}).$$

Therefore, by induction, the statement holds for all closed expressions involving sums, products, intersections, and duals that have at least one term. □

We now wish to understand what the trace of an extension looks like. We will find that the trace operator is a way to get back to the original code using the extension code. Before we can prove this, we need one small lemma.

Lemma 3.29. *For any matrix $G \in \mathbb{F}_q^{n\times k}$ and any $\mathbf{v} \in \mathbb{F}_{q^r}^k$, $\mathrm{tr}(\mathbf{v}G) = \mathrm{tr}(\mathbf{v})G$.*

Proof. Let $\mathbf{c}_1, \ldots, \mathbf{c}_k \in \mathbb{F}_q^n$ be the rows of G. Then

$$\mathbf{v}G = \sum_{i=1}^{k} v_i \mathbf{c}_i.$$

Therefore,

$$\begin{aligned}
\operatorname{tr}(\mathbf{v}G) &= \operatorname{tr}\left(\sum_{i=1}^{k} v_i \mathbf{c}_i\right) \\
&= \sum_{i=1}^{k} \operatorname{tr}(v_i \mathbf{c}_i) \\
&= \sum_{i=1}^{k} \operatorname{tr}((v_i c_{i,1}, \ldots, v_i c_{i,n})) \\
&= \sum_{i=1}^{k} (\operatorname{tr}(v_i c_{i,1}), \ldots, \operatorname{tr}(v_i c_{i,n})) \\
&= \sum_{i=1}^{k} (\operatorname{tr}(v_i) c_{i,1}), \ldots, \operatorname{tr}(v_i) c_{i,n}) \\
&= \sum_{i=1}^{k} \operatorname{tr}(v_i) \mathbf{c}_i \\
&= \operatorname{tr}(\mathbf{v})G. \qquad \square
\end{aligned}$$

Theorem 3.30 (Folklore)**.** *For any code* $C \leq \mathbb{F}_q^n$, $\operatorname{tr}(\hat{C}) = C$.

Proof. Let G be a generator for C. By construction, G generates $\hat{C}$ as well, so if we take an arbitrary codeword $\hat{\mathbf{c}} \in \hat{C}$, then $\mathbf{v} \in \mathbb{F}_{q^r}^k$ where $\hat{\mathbf{c}} = \mathbf{v}G$. By Lemma 3.29, $\operatorname{tr}(\hat{\mathbf{c}}) = \operatorname{tr}(\mathbf{v}G) = \operatorname{tr}(\mathbf{v})G$. We know $\operatorname{tr}(\mathbf{v}) \in \mathbb{F}_q^n$ and G generates C, so $\operatorname{tr}(\hat{\mathbf{c}}) \in C$ and $\operatorname{tr}(\hat{C}) \leq C$. Now, take any $\mathbf{c} \in C$. Then $\mathbf{w} \in \mathbb{F}_q^k$ where $\mathbf{c} = \mathbf{w}G$. It is well-known that the trace function is onto, so there must exist $\mathbf{v} \in \mathbb{F}_{q^r}^k$ where $\operatorname{tr}(\mathbf{v}) = \mathbf{w}$. Therefore,

$$\operatorname{tr}(\mathbf{v}G) = \operatorname{tr}(\mathbf{v})G = \mathbf{w}G = \mathbf{c}.$$

Because G generates $\hat{C}$, $\mathbf{v}G \in \hat{C}$, so $C \leq \operatorname{tr}(\hat{C})$. $\square$

Theorem 3.28 tells us that doing $\widehat{\ }$ operator on some a code which is some composition of

sums, intersections, products, and duals of input codes is the same as doing it to each input code first. Theorem 3.30 says that the trace of the extension of a code is that original code. We combine these to get a more generalized version of Theorem 2.15.

Corollary 3.31. *For any function f which can be represented as a closed form expression of sums, products, intersections, and duals of codes and for any codes $C_1, \ldots, C_s \leq \mathbb{F}_q^n$,*

$$\operatorname{tr}(f(\hat{C}_1, \ldots, \hat{C}_s)) = f(C_1, \ldots, C_s).$$

Proof. We know that $f(C_1, \ldots, C_s)$ will be a code. Using Theorems 3.30 and 3.28,

$$f(C_1, \ldots, C_s) = \operatorname{tr}([f(C_1, \ldots, C_s)]\hat{}) = \operatorname{tr}(f(\hat{C}_1, \ldots, \hat{C}_s)). \qquad \square$$

3.3 Decompositions of ℓ-Quasi-Cyclic Codes

We will now combine the previous two sections with the previous results we have on quasi-cyclic codes. Take a code $C \leq \mathbb{F}_q^{m\ell}$ and let $\mathbb{F}_{q^r}$ be the splitting field of $\det \tilde{G}(x)$ (see Section 2.3 for a refresher on what this means). If we further assume that the zeros of $\det \tilde{G}(x)$ are the set $\mathbb{F}_{q^r}^*$ and that a primitive m-th root of unity $\alpha \in \mathbb{F}_{q^r}$ exists, then we have that $\hat{C}$ is an (α, ℓ)-decomposable q^r-extension of C, so we can apply the results we have found for those types of codes. In particular, we want to see if we can construct CSS-T pairs of codes using C using the codes in the extension space containing $\hat{C}$. We wish to find the structure of $\hat{C} \cap (\hat{C}^2)^\perp$ and then to see what this tells us about $C \cap (C^2)^\perp$, building towards Theorem 3.34, which provides a construction of a CSS-T partner for C using a code in the extension space $\mathbb{F}_{q^r}^{m\ell}$.

For a summary representation of the square code $\hat{C}^2$, we define the notation

$$D_{*k} := \sum_{i+j=k \bmod m} D_i * D_j$$

for each $0 \leq i \leq m-1$, where $D := \hat{C}$ and

$$D = \sum_{i=0}^{m-1} p(\alpha^i) \otimes D_i.$$

The next two results will apply results from Section 3.1 towards trying to find what $\hat{C} \cap (\hat{C}^2)^\perp$ looks like.

Proposition 3.32. *If $C \leq \mathbb{F}_q^{m\ell}$ is an ℓ-quasi-cyclic code with (a, ℓ)-decomposable q^r-extension $D := \hat{C}$, where*

$$D = \sum_{i=0}^{m-1} p(\alpha) \otimes D_i$$

for some $D_i \leq \mathbb{F}_{q^r}^\ell$, then

$$D^2 = \sum_{i=0}^{m-1} p(\alpha^i) \otimes D_{*i}.$$

Proof. Since D is decomposable, $D^2 = D * D$ is decomposable as well (by Proposition 3.11). The exact structure of D^2 follows directly from Lemma 3.10. □

Theorem 3.33. *If $C \leq \mathbb{F}_q^{m\ell}$ is an ℓ-quasi-cyclic code with (a, ℓ)-decomposable q^r-extension $D := \hat{C}$, where*

$$D = \sum_{i=0}^{m-1} p(\alpha) \otimes D_i$$

for some $D_i \leq \mathbb{F}_{q^r}^\ell$, then

$$(D^2)^\perp \cap D = \sum_{i=0}^{m-1} p(\alpha^i) \otimes (D^\perp_{*\overline{(m-i)}} \cap D_i).$$

Proof. By Theorem 3.14, we know that $(D^2)^\perp$ and $D \cap (D^2)^\perp$ are decomposable. Proposition 3.6 tells us

$$(D^2)^\perp = \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes D_{*i} \right)^\perp = \sum_{i=0}^{m-1} p(\alpha^i) \otimes D^\perp_{*(\overline{m-i})}.$$

Therefore, Proposition 3.9 implies

$$D \cap (D^2)^\perp = \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes D_i \right) \cap \left(\sum_{i=0}^{m-1} p(\alpha^i) \otimes D^\perp_{*(\overline{m-i})} \right) = \sum_{i=0}^{m-1} p(\alpha^i) \otimes (D_i \cap D^\perp_{*(\overline{m-i})}). \qquad \square$$

Now, we will use our understanding of the relationships between traces and extensions to give a strong result on the construction of a CSS-T pair involving C using a specific (α, ℓ)-decomposable subcode of $\hat{C}$.

Theorem 3.34. *Let $C \leq \mathbb{F}_2^{m\ell}$ be an ℓ-quasi-cyclic code with (α, ℓ)-decomposable q^r-extension $D := \hat{C}$, where*

$$D = \sum_{i=0}^{m-1} p(\alpha^i) \otimes D_i.$$

If $\tilde{C} \leq \mathbb{F}_{2^r}^{m\ell}$ is any decomposable code of the form

$$\tilde{C} = \sum_{i=0}^{m-1} p(\alpha^i) \otimes \tilde{C}_i,$$

where $\tilde{C}_0, \ldots, \tilde{C}_{m-1} \leq \mathbb{F}_{2^r}^{\ell}$ are any codes such that $\tilde{C}_i \leq D^\perp_{*(\overline{m-i})} \cap D_i$ holds for $0 \leq i \leq m-1$, then $(C, \mathrm{tr}(\tilde{C}))$ is a CSS-T pair.

Proof. As we have constructed it, $\tilde{C} \leq D \cap (D^2)^\perp$. Combining this with Corollary 3.31 yields

$$\mathrm{tr}(\tilde{C}) \leq \mathrm{tr}(D \cap (D^2)^\perp) = \mathrm{tr}(\hat{C} \cap (\hat{C}^2)^\perp) = C \cap (C^2)^\perp.$$

By Theorem 1.47, $(C, \mathrm{tr}(\tilde{C}))$ is a CSS-T pair. $\square$

Now, we demonstrate this construction with a simple example.

Example 3.35. Let C_1 be the $[12, 3]_2$ 4-quasi-cyclic code generated by

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

Using the method explained in Section 2.3, we find the matrix

$$\tilde{U}(x) = \begin{pmatrix} 1 & 1 & x^2+1 & x+1 \\ x^3+1 & 0 & 0 & 0 \\ 0 & x^3+1 & 0 & 0 \\ 0 & 0 & x^3+1 & 0 \\ 0 & 0 & 0 & x^3+1 \end{pmatrix}.$$

Taking the Hermite normal form, we get

$$\tilde{G}(x) = \begin{pmatrix} 1 & 1 & x^2+1 & x+1 \\ 0 & x^3+1 & 0 & 0 \\ 0 & 0 & x^3+1 & 0 \\ 0 & 0 & 0 & x^3+1 \end{pmatrix}.$$

We see that $\mathbb{F}_{2^2}$ is the splitting field of $\det \tilde{G}(x)$. Let α be a multiplicative generator of $\mathbb{F}_{2^2}$. Defining G_i as a matrix which generates the row space of $\tilde{G}(\alpha^{-i})$, we compute

$$G_0 = \begin{pmatrix} 1 & 1 & 0 & 0 \end{pmatrix}$$

$$G_1 = \begin{pmatrix} 1 & 1 & \alpha^2 & \alpha \end{pmatrix}$$

$$G_2 = \begin{pmatrix} 1 & 1 & \alpha & \alpha^2 \end{pmatrix}.$$

These give us the generator matrix for $D := \hat{C}$,

$$G = \begin{pmatrix} \begin{pmatrix} 1 & 1 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 1 & 0 & 0 \end{pmatrix} \\ \begin{pmatrix} 1 & \alpha & \alpha^2 \end{pmatrix} \otimes \begin{pmatrix} 1 & 1 & \alpha^2 & \alpha \end{pmatrix} \\ \begin{pmatrix} 1 & \alpha^2 & \alpha \end{pmatrix} \otimes \begin{pmatrix} 1 & 1 & \alpha & \alpha^2 \end{pmatrix} \end{pmatrix}$$
$$= \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & \alpha^2 & \alpha & \alpha & \alpha & 1 & \alpha^2 & \alpha^2 & \alpha^2 & \alpha & 1 \\ 1 & 1 & \alpha & \alpha^2 & \alpha^2 & \alpha^2 & 1 & \alpha & \alpha & \alpha & \alpha^2 & 1 \end{pmatrix},$$

and the codes $D_0, D_1, D_2 \leq \mathbb{F}_{2^2}^4$. We must now find the generators G_{*i} and the parity-check matrices H_{*i} for the codes D_{*i}.

$$G_{*0} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \qquad H_{*0} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

$$G_{*1} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & \alpha^2 \end{pmatrix} \qquad H_{*1} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & \alpha \end{pmatrix}$$

$$G_{*2} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & \alpha \end{pmatrix} \qquad H_{*2} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & \alpha^2 \end{pmatrix}.$$

Notice that this matrix generates a decomposable code which satisfies the conditions given in Theorem 3.16, so $\hat{C}^2$ is self-dual. Furthermore, $(1\ 0)\, G_{*0} = G_0$, $(1\ \alpha^2)\, H_{*2} = G_1$, and $(1\ \alpha)\, H_{*1} = G_2$. This tells us $D_0 \leq D_0 \cap D_{*0}^{\perp}$, $D_1 \leq D_1 \cap D_{*2}^{\perp}$, and $D_2 \leq D_2 \cap D_{*1}^{\perp}$. Therefore,

choosing $\tilde{C}_1, \tilde{C}_2 := \{\mathbf{0}\}$ and $\tilde{C}_0 := D_0$, we can write a generator matrix of $\tilde{C} \leq D$ as

$$\left(\begin{pmatrix}1 & 1 & 1\end{pmatrix} \otimes \begin{pmatrix}1 & 1 & 0 & 0\end{pmatrix}\right) = \begin{pmatrix}1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0\end{pmatrix}.$$

Let $C_2 := \mathrm{tr}(\tilde{C})$. It is clear that C_2 is generated by the same matrix. By Theorem 3.34, (C_1, C_2) is a CSS-T pair.

Chapter 4

Conclusion

As we have seen, extensions of certain quasi-cyclic codes have a rich structure with many useful properties. We saw in Section 3.1 that decomposable codes have a rich structure that can be used to simplify code operations like sums, products, intersections, and duals. In addition, decomposable codes have an interesting interpretation of self-duality (see Corollary 3.17).

Theorem 3.30 provides a way to understand a complicated code operation as the trace of the same operation done over the extensions of the original codes. Combining these concepts, we may take quasi-cyclic codes that meet the conditions of Theorem 2.16 and use the properties of decomposable and extension codes to simplify operations between them.

Working with quasi-cyclic codes over $\mathbb{F}_2$, we may be able take the extension code in some extension field to get a decomposable code of the same length and dimension over $\mathbb{F}_{q^r}$. In particular, we can take the $C \cap (C^2)^\perp$ expression from [3] and work with the same expression over decomposable codes in an extension field, and we need only find the trace code to get a code over $\mathbb{F}_2$ with the desired property.

Chapter 5

Appendix

Here, we gather rigorous proofs of some of the non-trivial but intuitive results in a way that avoids derailing the content of other sections.

Proposition 5.1. *For any codes* $C_1, C_2, C_3 \leq \mathbb{F}_q^n$,

$$C_1 + (C_2 + C_3) = (C_1 + C_2) + C_3.$$

Proof. We use the commutativity of addition in $\mathbb{F}_q^n$ to find

$$\begin{aligned}
C_1 + (C_2 + C_3) &= C_1 + \langle \mathbf{c}_2 + \mathbf{c}_3 : \mathbf{c}_2 \in C_2, \mathbf{c}_3 \in C_3 \rangle \\
&= \langle \mathbf{c}_1 + \mathbf{c} : \mathbf{c}_1 \in C_1, \mathbf{c} \in C_2 + C_3 \rangle \\
&= \langle \mathbf{c}_1 + (\mathbf{c}_2 + \mathbf{c}_3) : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2, \mathbf{c}_3 \in C_3 \rangle \\
&= \langle (\mathbf{c}_1 + \mathbf{c}_2) + \mathbf{c}_3 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2, \mathbf{c}_3 \in C_3 \rangle \\
&= \langle \mathbf{c} + \mathbf{c}_3 : \mathbf{c} \in C_1 + c_2 \in C_2, \mathbf{c}_3 \in C_3 \rangle \\
&= \langle \mathbf{c}_1 + \mathbf{c}_2 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle + C_3 \\
&= (C_1 + C_2) + C_3.
\end{aligned}$$

The same proof holds for Schur products, which are also associative (due to the associativity of multiplication in $\mathbb{F}_q$). □

Proposition 5.2. *For any codes* $C_1, C_2 \leq \mathbb{F}_q^n$,

$$C_1 + C_2 = C_2 + C_1 \qquad \textit{and} \qquad C_1 * C_2 = C_2 * C_1.$$

Proof. The first part follows from the commutativity of addition in $\mathbb{F}_q^n$. We see

$$\begin{aligned} C_1 + C_2 &= \langle \mathbf{c}_1 + \mathbf{c}_2 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle \\ &= \langle \mathbf{c}_2 + \mathbf{c}_1 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle \\ &= C_2 + C_1. \end{aligned}$$

The second comes from the fact that the Schur product is commutative (because multiplication in $\mathbb{F}_q$ is commutative), so

$$\begin{aligned} C_1 * C_2 &= \langle \mathbf{c}_1 * \mathbf{c}_2 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle \\ &= \langle \mathbf{c}_2 * \mathbf{c}_1 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle \\ &= C_2 * C_1. \end{aligned}$$

□

Proposition 5.3. *For any vector* $\mathbf{v} \in \mathbb{F}_q^m$ *and any two codes* $C_1, C_2 \leq \mathbb{F}_q^n$,

$$\mathbf{v} \otimes (C_1 + C_2) = \mathbf{v} \otimes C_1 + \mathbf{v} \otimes C_2.$$

Proof. We use that the tensor product is bilinear to see

$$\begin{aligned}\mathbf{v} \otimes (C_1 + C_2) &= \langle \mathbf{v} \otimes (\mathbf{c}_1 + \mathbf{c}_2) : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle \\ &= \langle \mathbf{v} \otimes \mathbf{c}_1 + \mathbf{v} \otimes \mathbf{c}_2 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle \\ &= \langle \mathbf{v} \otimes \mathbf{c}_1 : \mathbf{c}_1 \in C_1 \rangle + \{ \mathbf{v} \otimes \mathbf{c}_2 : \mathbf{c}_2 \in C_2 \rangle \\ &= \mathbf{v} \otimes C_1 + \mathbf{v} \otimes C_2. \end{aligned}$$

□

Proposition 5.4. *For any codes* $C_1, C_2, C_3 \leq \mathbb{F}_q^n$,

$$C_1 * (C_2 + C_3) = C_1 * C_2 + C_1 * C_3.$$

Proof. We need only observe

$$\begin{aligned}C_1 * (C_2 + C_3) &= \langle \mathbf{c}_1 * \mathbf{c} : \mathbf{c}_1 \in C, \mathbf{c} \in C_2 + C_3 \rangle \\ &= \langle (\mathbf{c}_1 * (\mathbf{c}_2 + \mathbf{c}_3) : \mathbf{c}_2 \in C_2, \mathbf{c}_3 \in C_3 \rangle \\ &= \langle \mathbf{c}_1 * \mathbf{c}_2 + \mathbf{c}_1 * \mathbf{c}_3 : \mathbf{c}_2 \in C_2, \mathbf{c}_3 \in C_3 \rangle \\ &= \langle \mathbf{c}_1 * \mathbf{c}_2 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle + \langle \mathbf{c}_1 * \mathbf{c}_3 : \mathbf{c}_1 \in C_1, \mathbf{c}_3 \in C_3 \rangle \\ &= C_1 * C_2 + C_1 * C_3. \end{aligned}$$

□

Proposition 5.5. *Let* $A, B \leq \mathbb{F}_q^n$ *be any two codes. If* $\{\mathbf{a}_1, \ldots, \mathbf{a}_{k_1}\}$ *is a basis for* A *and* $\{\mathbf{b}_1, \ldots, \mathbf{b}_{k_2}\}$ *is a basis for* B *then the set*

$$S := \{\mathbf{a}_1 * \mathbf{b}_1, \ldots, \mathbf{a}_1 * \mathbf{b}_{k_2}, \ldots, \mathbf{a}_{k_1} * \mathbf{b}_1, \ldots, \mathbf{a}_{k_1} * \mathbf{b}_{k_2}\}$$

spans $A * B$.

Proof. From the definition of a Schur product code, it is clear that the set $T := \{\mathbf{a} *$

$\mathbf{b} : \mathbf{a} \in A, \mathbf{b} \in B\}$ spans $A * B$. Take any member of this set $\mathbf{a} * \mathbf{b}$. Then there exist $a_1, \ldots, a_{k_1}, b_1, \ldots, b_{k_2} \in \mathbb{F}_q$ such that

$$\mathbf{a} = \sum_{i=1}^{k_1} a_i \mathbf{a}_i \qquad \text{and} \qquad \mathbf{b} = \sum_{j=1}^{k_2} b_j \mathbf{b}_j.$$

Using the distributive properties of the Schur product,

$$\mathbf{a} * \mathbf{b} = \left(\sum_{i=1}^{k_1} a_i \mathbf{a}_i \right) * \left(\sum_{j=1}^{k_2} b_j \mathbf{b}_j \right) = \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} (a_i b_j)(\mathbf{a}_i * \mathbf{b}_j),$$

so S spans T and therefore $A * B$. □

Proposition 5.6. *For any code $C \leq \mathbb{F}_q^n$ and any $\mathbf{v} \in \mathbb{F}_q^m$, if $\mathcal{B}$ is a basis for C then $\mathbf{v} \otimes \mathcal{B}$ is a basis of $\mathbf{v} \otimes C$.*

Proof. The tensor product is a bilinear operator, so the function

$$\begin{aligned} f : \quad C &\rightarrow \mathbf{v} \otimes C \\ \mathbf{c} &\mapsto \mathbf{v} \otimes \mathbf{c} \end{aligned}$$

is linear and surjective. Therefore, any basis $\mathcal{B}$ for C spans $\mathbf{v} \otimes C$. Furthermore, if two vectors are not equal, then for any non-zero entry v_j of $\mathbf{v}$, the corresponding blocks will not be equal either, so f is injective. If $\mathbf{v} = 0$, then $\mathbf{v} \otimes \mathcal{B} = \{\mathbf{0}\}$, so the statement still holds. It follows that $\mathbf{v} \otimes \mathcal{B}$ is a basis for $\mathbf{v} \otimes C$. □

Proposition 5.7. *For any codes $C_1, C_2 \leq \mathbb{F}_q^n$ and any $\mathbf{v} \in \mathbb{F}_q^m$,*

$$\mathbf{v} \otimes (C_1 + C_2) = \mathbf{v} \otimes C_1 + \mathbf{v} \otimes C_2.$$

Proof. This follows from the properties of the tensor product. In particular,

$$\begin{aligned}
\mathbf{v} \otimes (C_1 + C_2) &= \langle \mathbf{v} \otimes \mathbf{c} : \mathbf{c} \in C_1 + C_2 \rangle \\
&= \langle \mathbf{v} \otimes (\mathbf{c}_1 + \mathbf{c}_2) : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle \\
&= \langle \mathbf{v} \otimes \mathbf{c}_1 + \mathbf{v} \otimes \mathbf{c}_2 : \mathbf{c}_1 \in C_1, \mathbf{c}_2 \in C_2 \rangle \\
&= \langle \mathbf{v} \otimes \mathbf{c}_1 : \mathbf{c}_1 \in C_1 \rangle + \langle \mathbf{v} \otimes \mathbf{c}_2 : \mathbf{c}_2 \in C_2 \rangle \\
&= \mathbf{v} \otimes C_1 + \mathbf{v} \otimes C_2.
\end{aligned}$$

□

Proposition 5.8. *For any codes $C_1, C_2 \leq \mathbb{F}_q^n$ and any $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{F}_q^m$,*

$$(\mathbf{v}_1 * \mathbf{v}_2) \otimes (C_1 * C_2) = (\mathbf{v}_1 \otimes C_1) * (\mathbf{v}_2 \otimes C_2).$$

Proof. Suppose $\mathcal{A} := \{\mathbf{a}_1, \ldots, \mathbf{a}_{k_1}\}$ is a basis for C_1 and $B := \{\mathbf{b}_1, \ldots, \mathbf{b}_{k_2}\}$ is a basis for C_2. Then we can apply Propositions 5.6 and 5.7 to find

$$\begin{aligned}
(\mathbf{v}_1 * \mathbf{v}_2) \otimes (C_1 * C_2) &= (\mathbf{v}_1 * \mathbf{v}_2) \otimes \langle \mathbf{a} * \mathbf{b} : \mathbf{a} \in A, \mathbf{b} \in B \rangle \\
&= \langle (\mathbf{v}_1 * \mathbf{v}_2) \otimes (\mathbf{a} * \mathbf{b}) : \mathbf{a} \in \mathcal{A}, \mathbf{b} \in \mathcal{B} \rangle \\
&= \langle (\mathbf{v}_1 \otimes \mathbf{a}) * (\mathbf{v}_2 \otimes \mathbf{b}) : \mathbf{a} \in \mathcal{A}, \mathbf{b} \in \mathcal{B} \rangle \\
&= \langle \mathbf{a}' * \mathbf{b}' : \mathbf{a}' \in \mathbf{v}_1 \otimes \mathcal{A}, \mathbf{b}' \in \mathbf{v}_2 \otimes \mathcal{B} \rangle \\
&= (\mathbf{v}_1 \otimes C_1) * (\mathbf{v}_2 \otimes C_2).
\end{aligned}$$

□

Proposition 5.9. *For any two codes $C_1, C_2 \leq \mathbb{F}_q^n$,*

$$(C_1 + C_2)^\perp = C_1^\perp \cap C_2^\perp.$$

Proof. Take any $\mathbf{c} \in (C_1 + C_2)^\perp$. Because $C_1, C_2 \leq C_1 + C_2$, for any $\mathbf{c}_1 \in C_1$ and any

$\mathbf{c}_2 \in C_2$, $\mathbf{c} \cdot \mathbf{c}_1 = 0$ and $\mathbf{c} \cdot \mathbf{c}_2 = 0$. Therefore, $\mathbf{c} \in C_1^\perp$ and $\mathbf{c} \in C_2^\perp$, so $\mathbf{c} \in C_1^\perp \cap C_2^\perp$ and $(C_1 + C_2)^\perp \leq C_1^\perp \cap C_2^\perp$. Now, take any $\mathbf{c} \in C_1^\perp \cap C_2^\perp$. Choose arbitrary $\mathbf{c}_1 + \mathbf{c}_2 \in C_1 + C_2$. Then

$$\mathbf{c} \cdot (\mathbf{c}_1 + \mathbf{c}_2) = \mathbf{c} \cdot \mathbf{c}_1 + \mathbf{c} \cdot \mathbf{c}_2 = 0 + 0 = 0.$$

Therefore, $\mathbf{c} \in (C_1 + C_2)^\perp$ and $C_1^\perp \cap C_2^\perp \leq (C_1 + C_2)^\perp$. □

Corollary 5.10. *For any two codes $C_1, C_2 \leq \mathbb{F}_q^n$,*

$$(C_1 \cap C_2)^\perp = C_1^\perp + C_2^\perp.$$

Proof. We use Proposition 5.9 to find

$$(C_1^\perp + C_2^\perp)^\perp = (C_1^\perp)^\perp \cap (C_2^\perp)^\perp = C_1 \cap C_2.$$

Because the duals are equal, we conclude that the original codes are equal as well. □

Lemma 5.11. *$\mathbb{F}_q$ contains a primitive m-th root of unity if and only if $m \mid q - 1$.*

Proof. If $\alpha \in \mathbb{F}_q$ is a primitive m-th root of unity, then $\langle \alpha \rangle$ is a subgroup of $\mathbb{F}_q^*$, so Lagrange's Theorem requires $m \mid q - 1$. If $m \mid q - 1$, then we can write $md = q - 1$ for some positive integer d. Let γ be the generator of $\mathbb{F}_q^*$. Then γ^d has order m in $\mathbb{F}_q^*$, so γ^d is a primitive m-th root of unity. □

Theorem 5.12. *If some family of codes $\mathcal{F} \leq \mathcal{P}(\mathbb{F}_q^n)$ is closed under binary sums, intersections, and products of codes and contains the dual codes of all its codes, then any code which can be expressed in a closed form involving sums, intersections, products, or duals of codes in $\mathcal{F}$ is itself in $\mathcal{F}$.*

Proof. Let the $\mathcal{F}$ denote some family of codes in $\mathbb{F}_q^n$ which is closed under sums, intersections,

and products of two codes and under dual operators. Suppose some function $f : \mathcal{F}^m \to \mathcal{P}(\mathbb{F}_q^n)$ can be expressed in a closed form $\mathcal{T}$ involving sums, intersections, products, and duals of codes. Without loss of generality, suppose that each input code appears exactly once in $\mathcal{T}$. We will proceed by induction on m. Suppose that for any $1 \leq k < n$, any such closed form on k codes in $\mathcal{F}$ results in a code in $\mathcal{F}$. Let $\mathcal{C}$ be the ordered set of input terms of f. Then f must have one of two forms. The first possible form is

$$f(\mathcal{C}) = h_1(\mathcal{C}_1) \circ \cdots h_s(\mathcal{C}_s),$$

where $s > 1$, $\mathcal{C} = \mathcal{C}_1 \sqcup \cdots \sqcup \mathcal{C}_s$ with $\mathcal{C}_i \neq$ for all $1 \leq i \leq s$, where $h_1, \ldots, h_s$ are functions, each of which can be expressed in a closed form involving sums, intersections, products, and duals of codes, and where $\circ$ is either $+$, $\cap$, or $*$. Because no $\mathcal{C}_i$ can be empty, we have that any given $\mathcal{C}_i$ can contain at most $m - s$ codes, with $m - s \leq m - 2$. Therefore, each h_i function takes at most $m - 2$ input codes, so, by assumption, if $\mathcal{C}$ is contains only codes in $\mathcal{F}$, then each $h_i(\mathcal{C}_i)$ will result in a code in $\mathcal{F}$. Therefore, if we write

$$C'_i := C'_{i-1} \circ h_i(\mathcal{C}_i)$$

and define $C'_1 := h_1(\mathcal{C}_1)$, $f(\mathcal{C}) = C'_s$. Lastly, we observe that if $C'_i \in \mathcal{F}$, then C'_{i+1} is either a sum, intersection, or product of C'_{i-1} and $h_i(\mathcal{C}_i)$, each of which is in $\mathcal{F}$. Therefore, $f(\mathcal{C}) \in \mathcal{F}$. The other form f could take is

$$f(\mathcal{C}) = [g(\mathcal{C})]^\perp,$$

where $g(\mathcal{C})$ is a closed form involving sums, intersections, products, and duals of codes. Importantly, we can say that g is not of the form $[h(\mathcal{C})]^\perp$ for some h without loss of generality, because consecutive $\perp$ operations cancel. Therefore, g is of the same form as the first form of f discussed. In the same way, $g(\mathcal{C}) \in \mathcal{F}$ for $\mathcal{C} \leq \mathcal{F}$. Therefore, $f(\mathcal{C})$ is the dual code

of a code in $\mathcal{F}$, so $f(\mathcal{C}) \in \mathcal{F}$. Finally, we establish the base case, which is a closed form expression on a single element. Such an expression must take the form

$$f(C) = C \qquad \text{or} \qquad f(C) = C^{\perp}.$$

In this case, it is clear that if $C \in \mathcal{F}$, then $f(C) \in \mathcal{F}$. Therefore, for any closed form of sums, intersections, products, and duals involving codes in $\mathcal{F}$ and which operates on at least one code, the resulting code is contained in $\mathcal{F}$. $\square$

Bibliography

[1] Kanat Abdukhalikov, Tushar Bag, and Daniel Panario. One-Generator Quasi-Cyclic Codes and their Dual Codes. *Discrete Mathematics*, 346, 2023.

[2] A. R. Calderbank and Peter W. Shor. Good Quantum Error-Correcting Codes Exist. *Physical Review A*, 54(2):1098–1106, 1996.

[3] Eduardo Camps-Moreno, Hiram H. López, Gretchen L. Matthews, Diego Ruano, Rodrigo San-José, and Ivan Soprunov. An Algebraic Characterization of Binary CSS-T Codes and Cyclic CSS-T Codes for Quantum Fault Tolerance. *Quantum Information Processing*, 23, 2024.

[4] Henry Chimal-Dzul, Julia Lieb, and Joachim Rosenthal. Generator Matrices of Quasi-Cyclic Codes over Extension Fields Obtained from Gröbner Basis. *IFAC-PapersOnLine*, 55:61–66, 2022.

[5] W. Cary Huffman and Vera Pless. *Fundamentals of Error-Correcting Codes*. Cambridge University Press, 2003.

[6] Kristine Lally. Quasicyclic Codes of Index ℓ over $\mathbb{F}_q$ viewed as $\mathbb{F}_q[x]$-Submodules of $\mathbb{F}_q^\ell[x]/\langle x^m - 1\rangle$. In *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes*, volume 15, Toulouse, France, May 2003.

[7] Narayanan Rengaswamy, Robert Calderbank, Michael Newman, and Henry D. Pfister. Classical Coding Problem from Transversal T Gates. pages 1891–1896, 2020.

[8] Peter W. Shor. Scheme for Reducing Decoherence in Quantum Computer Memory. *Physical Review A*, 52(4), 1995.

[9] Peter W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Scientific & Statistical Computing*, 26, 1997.

[10] Andrew Steane. Multiple Particle Interference and Quantum Error Correction. *Proceedings of the Royal Society of London*, (452), 1996.

[11] W. K. Wootters and W. H. Zurek. A Single Quantum Cannot be Cloned. *Nature*, 299: 802–803, 1928.