CODE CONSTRUCTIONS AND CODE FAMILIES FOR NONBINARY QUANTUM STABILIZER CODES

A Thesis

by

AVANTI ULHAS KETKAR

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

August 2004

Major Subject: Computer Science

CODE CONSTRUCTIONS AND CODE FAMILIES FOR NONBINARY QUANTUM STABILIZER CODES

A Thesis

by

AVANTI ULHAS KETKAR

Submitted to Texas A&M University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Andreas Klappenecker (Chair of Committee)	Rabi Mahapatra (Member)
Sue Geller (Member)	Valerie Taylor (Head of Department)

August 2004

Major Subject: Computer Science

ABSTRACT

Code Constructions and Code Families for Nonbinary Quantum Stabilizer Codes.

(August 2004)

Avanti Ulhas Ketkar, B.E., Pune University

Chair of Advisory Committee: Dr. Andreas Klappenecker

Stabilizer codes form a special class of quantum error correcting codes. Nonbinary quantum stabilizer codes are studied in this thesis. A lot of work on binary quantum stabilizer codes has been done. Nonbinary stabilizer codes have received much less attention. Various results on binary stabilizer codes such as various code families and general code constructions are generalized to the nonbinary case in this thesis. The lower bound on the minimum distance of a code is nothing but the minimum distance of the currently best known code. The focus of this research is to improve the lower bounds on this minimum distance. To achieve this goal, various existing quantum codes are studied that have good minimum distance. Some new families of nonbinary stabilizer codes such as quantum BCH codes are constructed. Different ways of constructing new codes from the existing ones are also found. All these constructions together help improve the lower bounds.

To my parents

ACKNOWLEDGMENTS

I would like to thank Dr. Andreas Klappenecker for solving all my difficulties and keeping me motivated throughout. Without his contribution and encouragement, this work was not possible. I would like to thank my committee members Dr. Rabi Mahapatra and Dr. Sue Geller for their continued support. I would also like to thank Neelima Chinthamani, Santosh Kumar and Pradeep Kiran Sarvepalli for their continuous feedback on my work. Many people have assisted in editing my thesis. I am thankful to all of them. This research has been supported by the NSF grant CCR 0218582 and a Texas A&M TITF initiative.

TABLE OF CONTENTS

CHAPTER Page

LIST OF TABLES

CHAPTER I

INTRODUCTION

A. Background

Quantum information and computation is the study of information processing that uses quantum mechanical systems. The motivation behind studying quantum computers is their enormous information storing and processing capacity. The discovery of Shor's factorization algorithm [1] and Grover's Search algorithm [1] made this power practically useful. This discovery motivates us to compare the capabilities of classical and quantum computers. Small quantum computers doing several operations represent the state of the art quantum computation. The study of quantum cryptography, quantum error correcting codes and fault tolerant quantum computing makes the implementation of a quantum computer practically realizable. This chapter covers the basics of this quantum computation and error correcting codes.

The unit of information in quantum computing is called a quantum bit or *qubit*. A qubit can be in a state $|0\rangle$ or $|1\rangle$, or a linear superposition

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

where $\alpha, \beta \in \mathbf{C}$. The basis states $|0\rangle$ or $|1\rangle$ play the same role as the bits 0 and 1, in classical information theory. The coefficients α, β are assumed to satisfy the condition $|\alpha|^2 + |\beta|^2 = 1$. If the qubit is measured in the computational basis, $|0\rangle$ will be observed with the probability $|\alpha|^2$ and $|1\rangle$ with the probability $|\beta|^2$. The *Dirac ket* notation $\alpha|0\rangle + \beta|1\rangle$ is customary in quantum computing, but it simply represents a vector (a, b) in $\mathbf{C}^2$. Similarly, multiple qubits can be in a superposition of states with

---

The journal model is *IEEE Transactions on Automatic Control.*

different amplitudes of the basis states. For instance, a *two-qubit* state can be given as

$$\psi = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle,$$

where $\alpha_{00}, \alpha_{01}, \alpha_{10}, \alpha_{11} \in \mathbf{C}$ and $|\alpha_{00}|^2 + |\alpha_{01}|^2 + |\alpha_{10}|^2 + |\alpha_{11}|^2 = 1$. Analogous to classical computers, quantum computers have logical gates, wires and quantum circuits. The quantum gates are nothing but unitary matrices acting on quantum states. *Pauli* matrices denote the most important single qubit gates. These gates include the I gate, which is an Identity matrix or no operation, the X gate, which is analogous to the classical NOT gate, the Z gate, which is a phase gate and the Y gate, which is a combination of X and Z gates. Another important single qubit gate, sometimes called as *square root of NOT gate*, is *Hadamard* gate, which is given by

$$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

The controlled-NOT gate and the Toffoli gate are the multiple qubit gates. The controlled-NOT gate flips the target bit if the control bit is 1. The Toffoli gate flips the target bit if both the control-bits are 1. Quantum computations are performed using these basic gates and other arbitrary unitary operations. Quantum information is very sensitive to the environment. When the computer interacts with its environment, the quantum state of the computer becomes entangled with the state of the environment; hence, the quantum state of the computer decays into an incoherent mixed state. This phenomenon is known as *decoherence*. Decoherence drastically compromises the performance of the machine. Hence, one needs a means to protect the quantum information.

B. Mathematical Basics

We now study some mathematics fundamentals that need to be understood for the theory of error correction.

1. Galois fields

A field is a set $\mathbf{F}$ which contains at least two elements and has two operations $+$ and $\times$. A field is closed under addition and multiplication and satisfies the associative, commutative and distributive laws. It has additive and multiplicative identities and inverses for each element except that 0 does not have a multiplicative inverse. A field with a finite number of elements is called a *finite field* or a *Galois field*. A finite field has p^m elements, where p is a prime and m is an integer. The set of nonzero elements of the field is a cyclic group under multiplication. A generator of this cyclic group is called a *primitive element* of the field. A finite field can be considered as the set of polynomials with coefficients in $\mathbf{F}_p$, when addition and multiplication is taken modulo an irreducible polynomial of degree n. Quantum information theory has a special interest in the Galois field of 4 elements. It is generated by polynomials over $\mathbf{F}_2$ modulo the irreducible polynomial $x^2 + x + 1$. Its elements are denoted as $\{0, 1, \omega, \omega^2\}$.

2. Notions of inner product

Two vectors are orthogonal to each other with respect to some notion of inner product if the product is zero. There are several notions of inner product defined, namely, the standard inner product, the trace-symplectic form, the Hermitian inner product, the alternating Hermitian product, the symplectic inner product or the twisted inner product, the trace-alternating form and so on. Each of these notions has different

definitions and different degrees of generalization. The specific notion of inner product will be explained whenever it is used for the first time in the following chapters.

C. Error Correcting Codes

1. Classical error correcting codes

The basic idea behind the error correcting codes is to add some redundancy to the message word of length k to give a code word of length n. In a q-ary error correcting code, every bit can take q values from $\mathbf{F}_q$, where q is a prime power. If $q = 2$, the code is called a *binary code*. If $q \geq 2$, the code is called a *nonbinary code*. Hence, the code space contains q^k different code words that are taken from the vector space $\mathbf{F}_q^n$. The Hamming distance between two code vectors is the number of coordinates in which they differ. The *minimum distance* of a code C is the minimum Hamming distance between all distinct pairs of code words in C. A code with minimum distance d can detect all the errors of weight less than or equal to $d-1$ and can correct all the errors of weight less than or equal to $\lfloor (d-1)/2 \rfloor$, where weight of a vector is defined as the number of nonzero components in the vector. The minimum distance of a code thus determines the error correcting capability of the code.

If k information bits are encoded into n information bits with minimum distance d, then the classical error correcting code is said to be an $[n, k, d]$ code. A binary $[n, k]$ code C is a k-dimensional subspace of $\mathbf{F}_2^n$. It is customary in coding theory to write the code words as row vectors. The matrix whose rows form a basis of C is called a *generator matrix*. Consider, for example, a $[5, 3]$ binary code C whose generator matrix G can be given as

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

All the code words in C can be obtained by the linear combination of theses three vectors. Another way of describing the code is via its *parity check matrix* H; in this case, the code is given by the kernel of the map $x \to xH^T$. A parity check matrix of the $[5, 3]$ code C is therefore given as

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

For any error correcting code, $GH^T = 0$. Also, if an $[n, k]$ code C has generator matrix $G = (I, A)$ in the standard form, then the parity check matrix of C is $H = (-A^T, I)$, where A^T is the transpose of A and is an $(n-k) \times k$ matrix and I is an $(n-k) \times (n-k)$ identity matrix.

The error correction is now explained. Let e denote the error acting on the code word x. Consider a received vector y, which is given by the sum of the code word x and error vector e, i.e., $y = x + e$. Then y is in some coset of C, and this coset contains all expressions $e = y - x$, where $x \in C$. Let H be the parity check matrix of code C with rows $h_1, h_2 \ldots h_{n-k}$. If $y \in V$, then the *syndrome* of y is defined to be a column vector

$$syn(y) = \begin{pmatrix} y.h_1 \\ y.h_2 \\ . \\ . \\ . \\ y.h_{n-k} \end{pmatrix}.$$

An appropriate error correcting scheme is applied depending upon the syndrome measured for the received word.

2. Quantum error correcting codes

The basic idea of error correction, i.e., adding redundancy to information bits to allow for error correction in classical error correction, remains the same in quantum error correction. However, some new ideas need to be introduced in quantum error correction to deal with the following difficulties [1]:

- *No cloning*: The no cloning theorem states that quantum information cannot be copied.
- *Errors are continuous*: A continuum of errors might occur on a single qubit.
- *Measurement destroys quantum information.*

However, these difficulties can be overcome. This can be explained using a 3-bit repetition code. Suppose a single state $a|0\rangle + b|1\rangle$ is encoded as $a|000\rangle + b|111\rangle$. The error correction is done in two stages.

1. *Error detection and syndrome measurement*: The measurement result of the received state is called *error syndrome*. If the error flips the bit value(s) of the state, then it is called a *bit flip* error. If the error changes the phase of the state, then it is called a *phase flip* error. For the bit flip channel, this code has four error syndromes:

$$P_0 = |000\rangle\langle 000| + |111\rangle\langle 111| \tag{1.1}$$

$$P_1 = |100\rangle\langle 100| + |011\rangle\langle 011| \tag{1.2}$$

$$P_2 = |010\rangle\langle 010| + |101\rangle\langle 101| \tag{1.3}$$

$$P_3 = |001\rangle\langle 001| + |110\rangle\langle 110| \tag{1.4}$$

These syndromes represent no error, bit flip on qubit one, bit flip on qubit two and bit flip on qubit three, respectively. Suppose the first bit is flipped to give $\psi = a|100\rangle + b|011\rangle$. Then, $\langle\psi|P_1|\psi\rangle = 1$. Thus, the output tells that the error is on the first bit without measuring or destroying the state.

2. *Recovery*: Depending on the syndrome, an appropriate correction is applied. So, if the first bit is flipped by the error, it is flipped back by the correction circuit.

The code $Q = \{a|000\rangle + b|111\rangle | a, b \in \mathbf{C}\}$ is a 2-dimensional subspace of $\mathbf{C}^q$. It is an example of a so called stabilizer code. Let

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

denote the phase gate. Consider the group with 4 elements $S = \{I \otimes I \otimes I, Z \otimes Z \otimes I, I \otimes Z \otimes Z, Z \otimes I \otimes Z\}$. The projection onto the code Q is defined by:

$$P_Q = \frac{1}{|S|} \sum_{M \in S} M.$$

This quantum code Q detects an error if and only if $P_Q E P_Q = c_E P_Q$ holds, where $c_E \in \mathbf{C}$. It follows that a phase error $Z \otimes I \otimes I$ is not detectable. Indeed, one notes that

$$P_Q(Z \otimes I \otimes I)P_Q = |000\rangle\langle 000| - |111\rangle\langle 111| \neq cP_Q$$

for every $c \in \mathbf{C}$, so $Z \otimes I \otimes I$ cannot be detected by this code. This discussion provides strong background to study the special class of codes called stabilizer codes. Continuing from here, chapter II explains the connection between classical self-orthogonal codes and quantum stabilizer codes that helps in constructing the quantum codes. Chapter III then explains the various existing quantum code families and shows how the new code families such as quantum BCH codes are constructed. Chapter IV

proves general code constructions where new quantum codes are constructed from the existing ones. Using all these results, how the actual improvement in the lower bounds is achieved is explained in chapter V. The conclusions and future work then follow.

CHAPTER II

CONNECTION BETWEEN CLASSICAL SELF-ORTHOGONAL CODES AND QUANTUM STABILIZER CODES

Error correcting codes form a very useful and significant part of classical information theory. This area is very well studied. The results presented in this chapter make use of these classical error correcting codes to build nonbinary quantum stabilizer codes. A brief overview of the generalization of the binary stabilizer codes to the nonbinary case is given. The material is based on [2, 3, 4, 5, 6, 7]. For the sake of completeness some of the proofs are restated and expanded.

To relate the quantum stailizer codes to classical self-orthogonal codes, we make use of an error basis, an error group and the stabilizer group. An error basis is nothing but a set of matrices that is basis for all the error operators in the q-ary system. Section A will explain the error basis of our interest in more detail. The set of all the error operators generated by the error basis is called error group G_n. An abelian subgroup S, of this error group is the stabilizer group which is of our interest. If we operate in a q-ary system, then the problem of constructing the quantum error correcting codes reduces to two steps. The first step is to find the appropriate error basis and the group generated by this error basis. In the next step, an abelian subgroup of this error group is chosen as a stabilizer S. The commutativity plays an important role in the construction of the stabilizer codes. While constructing the quantum codes from classical codes, some notion of inner product is used to arrive at the commutativity of operators in the stabilizer. We explain this construction, step by step, starting out with the appropriate selection of error basis.

A. Error Basis

In [2] nonbinary error bases are discussed in detail and their application to quantum codes is discussed in [5]. The relevant results now follow. Even though many bases are possible, the bases where the operators are invertible and unitary are useful.

Lemma 1. *Let $\mathcal{E}$ denote a set of q^2 unitary error operators over $\mathbf{F}_q$. This set $\mathcal{E}$ is called a nice error basis if and only if*

- *it contains the identity matrix*
- *the product of two elements in $\mathcal{E}$ is a scalar multiple of another element in $\mathcal{E}$.*
- *the trace $\mathrm{Tr}(A^\dagger B) = 0$ for distinct elements A, B of $\mathcal{E}$.*

More about these nice error basis can be found in [8]. The nice error basis that we choose for our connection between classical and quantum codes, is the following:

$$\{T_i R_j : i, j \in \mathbf{F}_p\} \tag{2.1}$$

where $T_{ij} = \delta_{i,j-1 \mod p}$, $R_{ij} = \epsilon^i \delta_{i,j}$, $\epsilon = e^{-\iota 2\pi/p}$ and $\iota = \sqrt{-1}$. A direct consequence of the definitions can be stated as

$$T_i T_j = T_{(i+j) \mod p}$$
$$R_i R_j = R_{(i+j) \mod p}$$

where $i, j \in \mathbf{F}_p$.

Lemma 2. *The set of matrices given by $T_i R_j$ form an orthogonal basis for the set of complex matrices of size p.*

Proof. The proof can be found in [5]. □

An interesting observation is that this basis separates the errors into amplitude errors and phase errors and any error can be considered as a combination of these two basic errors.

The extension of this basis to the case of $q = p^m$ is now given by recognizing that $\mathbf{F}_{p^m} \cong \mathbf{F}_p^m$ as vector spaces over $\mathbf{F}_p$. The error basis can therefore be generated by tensoring the basis over $\mathbf{F}_p$, m number of times. That the basis so generated is invertible and orthogonal is proven in [5]. This leads to the following lemma. Proof is omitted.

Lemma 3. *The m-fold tensor products* $\{(T_aR_b)_1 \otimes (T_aR_b)_2 \otimes \ldots (T_aR_b)_m\}$ *of* T_aR_b, *where* $a, b \in \mathbf{F}_p$, *form a nice error basis over* $\mathbf{F}_q$.

The operators in the stabilizer commute. Therefore, the crucial information that we need to extract from the error basis are the relations with respect to the commutativity of two error operators. We relate the classical self-orthogonal codes to the stabilizer codes in such a way that some notion of inner product between the error basis in the classical code get converted to the commutativity of the operators in the stabilizer. For the p-ary case we have

$$T_iR_j = \epsilon^{-ij} R_jT_i \tag{2.2}$$

Let $a, b \in \mathbf{F}_q^n$, then $\langle a, b\rangle = \sum_{i=1}^n a_ib_i$ denotes the standard inner product of a, b. Extending this to the p^m-ary case, we get:

$$T_aR_b = \epsilon^{-\langle a,b\rangle} R_bT_a \tag{2.3}$$

where the standard inner product is over $\mathbf{F}_p$. When this is extended to an operator

for n qubits we get the following relation for an error operator of n qubits:

$$\begin{aligned} E_{\mathbf{a},\mathbf{b}} &= T_{a^{(1)}} R_{b^{(1)}} \otimes T_{a^{(2)}} R_{b^{(2)}} \cdots \otimes T_{a^{(n)}} R_{b^{(n)}} \\ &= \epsilon^{-\sum_{i=1}^{n} \langle a^{(i)}, b^{(i)} \rangle} R_{b^{(1)}} T_{a^{(1)}} \otimes R_{b^{(2)}} T_{a^{(2)}} \cdots \otimes R_{b^{(n)}} T_{a^{(n)}} \\ &= \epsilon^{-\langle \mathbf{a},\mathbf{b} \rangle_{\mathbf{v}}} R_{b^{(1)}} T_{a^{(1)}} \otimes R_{b^{(2)}} T_{a^{(2)}} \cdots \otimes R_{b^{(n)}} T_{a^{(n)}} \end{aligned}$$

where the vector inner product between a,b is given as:

$$\langle \mathbf{a}, \mathbf{b} \rangle_{\mathbf{v}} = \sum_{i=1}^{n} \langle a^{(i)}, b^{(i)} \rangle = \sum_{i=1}^{n} \sum_{j=1}^{m} a_j^{(i)} b_j^{(i)}.$$

From these relations we can deduce that

$$E_{\mathbf{a},\mathbf{b}} = \epsilon^{-\langle \mathbf{b},\mathbf{c} \rangle_{\mathbf{v}}} E_{\mathbf{a}+\mathbf{d},\mathbf{b}+\mathbf{c}}$$

$$E_{\mathbf{c},\mathbf{d}} = \epsilon^{-\langle \mathbf{a},\mathbf{d} \rangle_{\mathbf{v}}} E_{\mathbf{a}+\mathbf{d},\mathbf{b}+\mathbf{c}}$$

Lemma 4. *The error operators $E_{\mathbf{a},\mathbf{b}}$ and $E_{\mathbf{c},\mathbf{d}}$ commute if and only if*

$$\langle \mathbf{a}, \mathbf{d} \rangle_{\mathbf{v}} - \langle \mathbf{b}, \mathbf{c} \rangle_{\mathbf{v}} = \sum_{i=1}^{n} \langle a^{(i)}, d^{(i)} \rangle - \langle b^{(i)}, c^{(i)} \rangle = 0 \tag{2.4}$$

holds.

Thus, we translated the commutativity relation in quantum stabilizer codes to the inner product notion in classical codes.

B. From Quantum Stabilizer Codes to Classical Codes

A stabilizer code Q is the joint $+1$ eigenspace of an abelian subgroup S of the error group G_n. If $Q \leq \mathbf{C}^{q^n}$ is a quantum code with stabilizer S, then the centralizer $C(S)$ of the stabilizer is defined as a subgroup of G_n such that elements in $C(S)$ commute with all the elements of S. All the errors outside $C(S) - \langle S \rangle$ are detectable. The code Q can correct errors of weight less than or equal to t, if $C(S) - \langle S \rangle$ does

not contain errors of weight less than or equal to $2t$. Note that, when a quantum stabilizer code Q is derived from the classical self-orthogonal code C, the classical code C corresponds to the stabilizer of the quantum code and the dual code of C corresponds to the centralizer of the stabilizer. Hence, $S \subset C(S)$ gives us a condition that, classical codes should be contained in their dual (self-orthogonal code) with an appropriate definition of inner product. We also have a condition that there cannot be any vectors of weight less than d in $C^{\perp} \setminus C$ if Q has distance d.

First we connect the stabilizer over $\mathbf{F}_q$ to $\mathbf{F}_q^{2n}$. As mentioned in the last section every error operator is defined by two vectors $a, b \in \mathbf{F}_q^n$. We can associate a vector $(a|b)$ of length $2n$ for every operator and thus form an isomorphism over $\mathbf{F}_q^{2n}$.

Let $v, v' \in \mathbf{F}_q^{2n}$ be two operatots such that $v = (a|b)$ and $v' = (a'|b')$. These operators commute if the following relation holds

$$\langle \mathbf{a}, \mathbf{b}' \rangle_{\mathbf{v}} - \langle \mathbf{a}', \mathbf{b} \rangle_{\mathbf{v}} = 0$$
$$\sum_{i=1}^{n} \langle a^i, b'^i \rangle - \langle a'^i, b^i \rangle = 0.$$

This is the vector symplectic product between v, v' denoted as $\langle v, v' \rangle_{vs}$. As similar notion of inner product, called the symplectic inner product, denoted by $\langle .|. \rangle_s$, is used later where the summation for the vectors is not present. The next lemma therefore follows.

Lemma 5. *The stabilizer of a $q-$ary quantum code is isomorphic to $C \subset \mathbf{F}_q^{2n}$ and C(S) is isomorphic to the symplectic dual of C over $\mathbf{F}_p$.*

However, this symplectic product is actually over $\mathbf{F}_p$. Though we have a classical code at this point the notion of the inner product is defined in a rather inconvenient form and does not have any apparent relation to one of the standard inner products over $\mathbf{F}_q$. To take advantage of classical codes we now define an automorphism over

$\mathbf{F}_q$ so that the current symplectic inner product can be more conveniently related to the inner product over $\mathbf{F}_q$. This motivates the middle connection between classical and quantum codes.

1. Connecting to classical codes over $\mathbf{F}_q^{2n}$

At this point, we have the p-ary error basis. To operate in a q-ary system, we have to tensor these basis m times. To simplify this and get the q-ary basis, we define an automorphism φ. Let φ^* be an automorphism of $\mathbf{F}_p^m$. For $\mathbf{a} = (a^{(1)}, a^{(2)}, \ldots, a^{(n)}) \in \mathbf{F}_{p^m}^n$,

$$\varphi_*^*(\mathbf{a}) = (\varphi^*(a^{(1)}), \varphi^*(a^{(2)}), \ldots, \varphi^*(a^{(n)})).$$

Let us further define the automorphism of $\mathbf{F}_q^{2n}$. For all vectors $v = (a|b) \in \mathbf{F}_q^{2n}$, define

$$\varphi(v) = (a|\varphi_*^*(b))$$

where $a, b \in \mathbf{F}_q^n$. Let $C \leq \mathbf{F}_q^{2n}$ be an additive self-orthogonal code where the orthogonality is with respect to the symplectic product over $\mathbf{F}_p$. Let the code generated by the automorphism be $\varphi(C)$. Thus if we have $(a|b) \in \varphi(C)$ then $(a|\varphi^{-1}(b)) \in C$. Let the matrix M defining φ^{*-1} be given by

$$M_{i,j} = \mathrm{tr}_{p^m/p}(\alpha_i \alpha_j),$$

where α is a basis of $\mathbf{F}_{p^m}$ over $\mathbf{F}_p$. The code C is self-orthogonal with respect to the symplectic product defined over $\mathbf{F}_p$. We have:

$$\langle (\mathbf{a}|\varphi^{*-1}(\mathbf{b})), (\mathbf{a}'|\varphi^{*-1}(\mathbf{b}'))\rangle_{vs} = \langle \mathbf{a}, \varphi^{*-1}(\mathbf{b}')\rangle_v - \langle \mathbf{a}', \varphi^{*-1}(\mathbf{b})\rangle_v. \tag{2.5}$$

For $a, b \in \mathbf{F}_q$ we have $a^T = (a_1, a_2, \ldots, a_m)$ and $b^T = (b_1, b_2, \ldots, b_m) \in \mathbf{F}_{p^m}$ where a,b are in matrix form. This leads to

$$\begin{aligned} a^T M b &= \sum_{i=1}^{m}\sum_{j=1}^{m} a_i b_j \operatorname{tr}(\alpha_i \alpha_j) = \sum_{i=1}^{m}\sum_{j=1}^{m} \operatorname{tr}(a_i b_j \alpha_i \alpha_j) \\ &= \operatorname{tr}((\sum_{i=1}^{m} a_i \alpha_i)(\sum_{j=1}^{m} b_j \alpha_j)) = \operatorname{tr}(ab) \end{aligned}$$

This automorphism conveniently transforms the symplectic product over $\mathbf{F}_p$ to trace of the symplectic product over $\mathbf{F}_q$ as follows

$$\langle (\mathbf{a}|\varphi_*^{-1}\mathbf{b}), (\mathbf{a}'|\varphi_*^{-1}\mathbf{b}')\rangle_{vs} = \operatorname{tr}(\langle a, b'\rangle - \langle a', b\rangle) \quad (2.6)$$

$$= \operatorname{tr}_{q/p}(\langle (a|b), (a'|b')\rangle_s). \quad (2.7)$$

So we need to find codes that are self-orthogonal with respect to the inner product defined by $\operatorname{tr}_{q/p}(\langle \mathbf{a}, \mathbf{b}'\rangle - \langle \mathbf{a}', \mathbf{b}\rangle)$. This is nothing but trace of the symplectic inner product defined earlier and is called trace-symplectic form. Note that it is possible to choose the basis $\{\alpha_i; i = 1, 2 \ldots m\}$ such that φ_* is an identity.

2. Connecting to classical codes over $\mathbf{F}_{q^2}^n$

In this section a connection is established between classical self-orthogonal codes of length n over $\mathbf{F}_{q^2}$ to self-orthogonal codes over length $2n$ over $\mathbf{F}_q$.

Suppose $(a|b), (a'|b') \in \mathbf{F}_q^{2n}$, then we define the trace-symplectic form between $(a|b)$, $(a'|b')$ as

$$\operatorname{tr}_{q/p}(\langle (a|b), (a'|b')\rangle_s) = \operatorname{tr}_{q/p}(\langle a, b'\rangle - \langle a', b\rangle) \quad (2.8)$$

It is obvious that vectors orthogonal with respect to the symplectic inner product will be orthogonal with respect to the trace-symplectic form. Let $[\omega, \omega^q]$ be a normal

basis of $\mathbf{F}_{q^2}$ over $\mathbf{F}_q$. Then, for each vector $v = (a|b) \in \mathbf{F}_q^{2n}$, we define a mapping $\phi : \mathbf{F}_q^{2n}/\mathbf{F}_{q^2}^n$ given by

$$\phi(v) = \omega a + \omega^q b. \tag{2.9}$$

It is clear that the weight of v is equal to the Hamming weight of $\phi(v)$, and the distance between vectors $v = (a|b), v' = (a'|b') \in \mathbf{F}_q^{2n}$ is equal to $\text{dist}(\phi(v), \phi(v'))$.

Suppose $v, v' \in \mathbf{F}_q^{2n}$, then we define the trace-alternating form as

$$\langle \phi(v), \phi(v') \rangle_a = \text{Tr}_{q/p}(\frac{\langle \phi(v), \phi(v')^q \rangle - \langle \phi(v)^q, \phi(v') \rangle}{\omega^2 - \omega^{q2}}). \tag{2.10}$$

Lemma 6. *The trace-symplectic form of v and v' is equal to the trace-alternating form.*

Proof. Suppose that $\phi(v) = \omega a + \omega^q b$ and $\phi(v') = \omega a' + \omega^q b'$. Then the trace-alternating form of $T = \langle \phi(v)\phi(v') \rangle_a$ is given by

$$T = \text{tr}_{q/p}\left(\frac{\langle (\omega a + \omega^q b), (\omega^q a' + \omega^{q^2} b') \rangle - \langle (\omega^q a + \omega^{q^2} b), (\omega a' + \omega^q b') \rangle}{\omega\omega^{q^2} - \omega^q \omega^q} \right).$$

After multiplying and taking out the common terms, we get the standard inner products as:

$$T = \text{tr}_{q/p}\left(\frac{\langle a, b' \rangle((\omega\omega^{q^2}) - (\omega^q\omega^q)) + \langle a', b \rangle((\omega^q\omega^q) - (\omega^{q^2}\omega))}{\omega^2 - \omega^{q^2}} \right).$$

Note that the terms $\omega\bar{\omega}aa'$ and $\omega^q\bar{\omega}^q bb'$ get canceled. Further simplifying we get,

$$\begin{aligned} T &= \text{tr}_{q/p}\left(\frac{(\omega^2 - \omega^{q^2})(\langle a, b' \rangle - \langle a', b \rangle)}{\omega^2 - \omega^{q^2}} \right) \\ &= \text{tr}_{q/p}(\langle a, b' \rangle - \langle a', b \rangle). \end{aligned}$$

Hence proved. □

Theorem 7. *There exists an additive classical code over $\mathbf{F}_q^{2n}$, self-orthogonal with*

respect to the trace-symplectic form if and only if there exists an additive classical code over $\mathbf{F}_{q^2}^n$ *self-orthogonal with respect to the trace-alternating form.*

Proof. This follows from Lemma 6 and the fact that mapping ϕ is an isometry. □

C. Existence of Quantum Codes

We can now formally relate the existence of quantum codes by examining the corresponding classical codes based on the previous results.

Theorem 8. *Let* C *be a* $[n, k, d]_q$ *classical code such that* $C^\perp \subseteq C$ *where the inner product is with respect to the standard inner product. Then there exists* $[[n, 2k-n, d]]_q$ *quantum code.*

Proof. This construction is exact generalization of the CSS codes. Let C be defined by the generator matrix G and parity check matrix H. Since $C^\perp \subseteq C$ it is self-orthogonal. If we define C' as the direct sum $C^\perp \oplus C^\perp$, then we have a $\mathbf{F}_q^{2n}$ code whose generator matrix is given as

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

Any vector $v \in C'$ is of the form $(a|0)$ or $(0|b)$ where $a, b \in C^\perp$. When we take the symplectic product we get

$$\langle (a|0), (b|0) \rangle_s = \langle a, 0 \rangle - \langle 0, b \rangle = 0 \tag{2.11}$$

$$\langle (a|0), (0|b) \rangle_s = \langle a, b \rangle - \langle 0, 0 \rangle = 0 \tag{2.12}$$

where the second symplectic product vanishes because $C^\perp$ is self-orthogonal. This has a dimension of $2n - 2k$. Therefore the stabilizer has a dimension $2n - 2k$. The quantum code has a dimension $n - (2n - 2k) = 2k - n$. Since C has a minimum

distance of d, $C^{\perp_s}\backslash C$ has a distance greater or equal to d. The corresponding quantum code therefore has a distance greater than or equal to d. □

Corollary 9. *If there exists a classical code* $C, [n, (n-k)/2]_{q^2}$ *self-orthogonal with respect to the trace-alternating form, then there exists a quantum code with the parameters* $[[n, k, d]]$ *where* $d = min\ \{w(x) : x \in C^{\perp_a}\backslash C\}$.

Corollary 10. *If there exists a classical* $[n, (n-k)]_q$ *code* C*, self-orthogonal with respect to the trace-symplectic form, then there exists a quantum code with the parameters* $[[n, k, d]]$ *where* $d = min\ \{w(x) : x \in C^{\perp_{ts}}\backslash C\}$.

It should be noted that at no point have we considered the code C to be linear over $\mathbf{F}_{q^2}$, hence the codes so constructed will be purely additive codes. If however the code is linear then it is sufficient for the existence of self-orthogonal code C, $[n, (n-k)/2]_{q^2}$ for the existence of an $[[n, k, d]]_q$ quantum code. Because the code is linear, we need to only list $(n-k)/2$ generators, as the other generators can be obtained by multiplication of one of the basis elements used for expanding $\mathbf{F}_q^2$ over $\mathbf{F}_q$ in the mapping ϕ.

D. Conclusion

To summarize, we have looked in detail the various links that connect the quantum codes to classical codes. And we have shown how the problem of nonbinary quantum stabilizer code maybe translated to design of self-orthogonal codes over $\mathbf{F}_{q^2}^n$. We have clarified the various notions of inner product that come into picture at various stages in this process. We have also looked at two constructions for quantum codes from classical codes.

CHAPTER III

QUANTUM CODE FAMILIES

As stated earlier, the goal of this research is to improve the lower bounds on the nonbinary quantum stabilizer codes. The following chapters explain how this goal is achieved.

A. Existing Quantum Codes

With the aim of meeting the proposed upper bounds, the existing literature is searched for good nonbinary quantum stabilizer codes. The following results prove to be useful.

Theorem 11. *For any prime power q and an integer in the range $0 \leq \mu < (q-1)$, there exist quantum stabilizer codes with parameters $[[q^2, q^2-2\mu-2, \mu+2]]_q$ and $[[q^2-1, q^2-2\mu-1, \mu+1]]_q$.*

Proof. Please see [9] for proof. □

For instance, the MDS code $[[9,5,3]]_3$ can be constructed using this theorem. MDS code is the *Maximally Distance Seperable* code and therefore has excellent error correcting capability.

Theorem 12. *By shortening of MDS code, for an integer s, and an integer d, $2 \leq d \leq q$, there exist quantum stabilizer codes $[[q^2-s, q^2-2d+2-s, d]]_q$.*

Proof. Please see [9] for proof. □

For example, $[[7,3,3]]_3$ code can be constructed from $[[9,5,3]]_3$ code. These results are taken from [9]. Both these results give us some good codes to start with.

Theorem 13. *Let q be a prime power. For an even integer r, there exist quantum stabilizer codes with parameters*

$$[[\frac{q^{r+2}-1}{q^2-1}, \frac{q^{r+2}-1}{q^2-1} - (r+2), 3]]_q.$$

For an odd integer r there exist quantum stabilizer codes with parameters

$$[[\frac{q^3(q^{r+2}-1)}{(q^2-1)}, \frac{q^3(q^{r+2}-1)}{(q^2-1)} - (r+2), 3]]_q.$$

The details of these two results can be found in [6]. Another simple result can be stated as:

Lemma 14. *An $[[n, n, 1]]_q$ quantum stabilizer code always exists.*

In this section, we use the Hermitian duality between vectors. For any element $\alpha \in \mathbf{F}_{q^2}$, we define the conjugate of that element as α^q and it is denoted by $\bar{\alpha}$. For a vector $a \in \mathbf{F}_{q^2}^n$, we define $\bar{a}$ as the conjugate of a and the conjugate is taken over each component. The Hermitian inner product of two vectors in $\mathbf{F}_{q^2}^n$ is defined as

$$\langle a, b\rangle_h = \langle a, \bar{b}\rangle = \sum_{i=1}^{n} a_i \bar{b}_i = \sum_{i=1}^{n} a_i b_i^q, \text{ where } a, b \in \mathbf{F}_{q^2}^n.$$

Lemma 15. *Any classical code self-orthogonal with respect to the Hermitian product will be self-orthogonal with respect to the trace-alternating form.*

Proof. The Hermitian product between $a, b \in F_{q^2}^n$ is given by

$$\sum_{i=1}^{n} a_i b_i^q = 0 \Rightarrow \sum_{i=1}^{n} (a_i b_i^q)^q = 0 \Rightarrow \sum_{i=1}^{n} a_i^q b_i = 0$$

Hence,

$$\sum_{i=1}^{n} a_i b_i^q - \sum_{i=1}^{n} a_i^q b_i = 0$$

For $a, b \in F_{q^2}^n$, the trace-alternating form between them is given by

$$\frac{\sum_{i=1}^{n} a_i b_i^q - \sum_{i=1}^{n} a_i^q b_i}{\omega^2 - \omega^{2q}},$$

where (ω, ω^q) is a normal basis of $\mathbf{F}_{q^2}$ over $\mathbf{F}_q$. Thus, the trace-alternating form vanishes when the Hermitian product vanishes. □

The connection between the classical and quantum codes from Chapter II can then be used to construct corresponding quantum codes.

B. Searching for Codes

The very first approach taken for finding quantum codes is to search for the classical self-orthogonal codes exhaustively (brute force approach). Choosing all combinations of $(q^2)^k$ vectors from the $(q^2)^n$ vectors is, however, very expensive, and hence, some optimizations are necessary. The following approach is then taken

- For a classical $[n, k, d]$ code, the generator matrix G is a $k \times n$ matrix that can always be reduced to a *standard form*. This standard form is given as

  $$G = \left[\; I_{k\times k} \;\middle|\; A_{k\times n-k} \; \right],$$

  where I is an identity matrix and A is an arbitrary matrix.

- Instead of choosing $(q^2)^k$ vectors from $(q^2)^n$ vectors, k vectors are generated each time that satisfy this standard form. This approach of choosing the generators instead of choosing all the code vectors reduces the search space by a very large factor. This *choosing* operation is saved and the reduction of factor $(q^2)^k$ is achieved due to the standard form.

- The all zero vector is eliminated and the self-orthogonality of the generators is

checked.

- An $[n, k, d]$ code is pure if the dual of the code does not contain vectors of weight less than d; otherwise, the code is impure. Since the search was done for pure codes, the vectors with weight less than d are eliminated.

- The most expensive operation of checking linear independence is then done.

Even though the search algorithm is optimized over the exhaustive search, it did not give significant results. This is because of the exponential complexity of operations such as checking the linear independence of code vectors. It might be improved by making use of the exact weight enumerators and standard form of the dual code's generator matrix. Machines with higher power might also help.

C. Quantum BCH Codes

It has been shown previously that binary quantum codes can be constructed from classical cyclic codes or, in particular, Bose-Chaudhuri-Hocquenghem codes (BCH codes) [4, 9, 11]. BCH codes form an extremely important class of error correcting codes.

Lemma 16. *Let ω be a primitive n^{th} root of unity over $\mathbf{F}_q$ and let $g(X)$ be a monic polynomial over $\mathbf{F}_q$ of smallest degree that has the $\delta - 1$ numbers $\omega^b, \omega^{b+1}, \ldots, \omega^{b+\delta-2}$ among its zeros, where $b \geq 0$ and $\delta \geq 1$. Thus,*

$$g(X) = lcm\{m_b(X), m_{b+1}(X), \ldots, m_{b+\delta-2}(X)\},$$

where $m_i(X)$ is the minimal polynomial of ω^i. The q-ary cyclic code with this $g(X)$ as a generating polynomial is called the BCH code with designed distance δ.

We can construct nonbinary quantum codes if we start with classical self-orthogonal BCH codes using the connection given in Chapter II. Such quantum codes derived from classical self-orthogonal BCH codes are called quantum BCH codes. The first construction gives us smaller codes and hence is more useful in practice. However, the last two constructions help in constructing higher dimensional codes. In general, three families of quantum BCH codes are constructed. Furthermore, some of these codes meet the proposed upper bounds on minimum distance. Hence, they provide us with the starting point for improving the lower bounds. Let C be an $[n, k, d]$ classical BCH code which is used to construct the quantum BCH code. Let Z_C denote the zero set of C over a field $\mathbf{F}_{q^2}$. The generator polynomial of C is then given by

$$g(X) = \prod_{z \in Z_c} (X - \alpha^z).$$

Let $h(X)$ be of minimal degree so that $g(X)h(X) = 0 \mod (X^n - 1)$. Then the generator polynomial of the Hermitian dual code is defined as [9]

$$h^*(X) = \prod_{z \in Z_n \backslash Z_c} (X - \alpha^{-qz}),$$

where Z_n is the set of all n roots of $X^n - 1 = 0$. Hence, the zero set of the orthogonal set is given as

$$Z_{C^\perp} = \{-qz \mod n : z \in \{0, 1, 2 \ldots n-1\} \backslash Z_C\}.$$

The roots are $-qz$ for this dual code because the duality is with respect to the Hermitian product. Therefore, each element is raised to power q.

Lemma 17. *A classical BCH code is self-orthogonal if the set of the roots of the dual code is a subset of the set of the roots of the BCH code.*

Proof. Let $g(X)$ be a generating polynomial of a BCH code C over $\mathbf{F}_q$ and let $h^*(X)$ be the generating polynomial of the Hermitian dual code of C. The result, however,

holds for any notion of inner product. Any codeword in C can be given as

$$c(X) = g(X)m_1(X)$$

where $m_1 \in \mathbf{F}[X]$. Any code word in the dual code of C can be given as

$$c^*(X) = h^*(X)m_2(X)$$

where $m_2 \in \mathbf{F}[X]$. But if all the roots of $h^*(X)$ are contained in the roots of $g(X)$, $c(X)$ can be represented as

$$c(X) = g(X)m_1(X) = h^*(X)m_3(X)m_1(X) = h^*(X)m'(X),$$

where $g(X) = m_3(X)h^*(X)$ and $m'(X) \in \mathbf{F}[X]$. Thus, every code word generated by $g(X)$ can be generated by $h^*(X)$. Thus, the code C is contained in its dual code and hence is self-orthogonal. □

Since the roots of the dual code lie in the set of the roots of the BCH code C, all the vectors generated by the generator matrix of C lie in the set of all the vectors of the dual code $C^{\perp}$. Hence, the quantum BCH code can be derived from a classical BCH code if $Z_{C^{\perp}} \subseteq Z_C$. The following algorithm gives us quantum BCH codes over $\mathbf{F}_q$ derived from classical linear self-orthogonal codes over $\mathbf{F}_{q^2}$. The self-orthogonality is with respect to the Hermitian product.

1. Take n, d, q as integer input parameters where q is a perfect square.

2. Calculate the exponent m such that n divides q^m.

3. Choose n elements from $\mathbf{F}_{q^m}$ such that they are n-th roots of unity and order them. Call this set of ordered roots as Z_n.

4. The minimum distance of the code should be at least d. So the dual code should

have at least $d-1$ consecutive elements from the ordered roots as the zeros of its generator polynomial. Hence, choose consecutive $d-1$ roots from the ordered roots as the zeros of the dual code.

5. To calculate all the conjugates, raise the elements to power q, $m-1$ times. Note that some elements might be repeated. All these roots will give us the set of zeros of the dual code. Call this set Z_H.

6. To obtain roots of $h^*(X)$, raise all these roots to the power $-\sqrt{q}$. This is because the dual code is with respect to Hermitian inner product defined earlier. The square root is taken because the field entered as input is q^2. Call this set Z_D.

7. The roots of $g(X)$ can be found out as $Z_n \backslash Z_D$. Call this set as Z_C.

8. If $Z_H \subseteq Z_C$, then the BCH code is self-orthogonal. If the number of roots of $g(X)$ is k', $k = n - k'$. Hence, we obtain $[n, k, d]$ classical BCH code.

9. From the connection between classical and quantum codes, it can be seen as the corresponding quantum code $[[n, n-2k, d]]$ where $k = n - k'$. The actual distance of the quantum code can, however, be higher than this distance. This quantum BCH code is over $\mathbf{F}_{\sqrt{q}}$.

10. The roots of the classical BCH code and its dual are then given as powers of the primitive element in $\mathbf{F}_{q^m}$.

11. Repeat steps 4 to 10 for all the n combinations of $d-1$ consecutive elements. There are n combinations because the consecutiveness is cyclic.

We used this algorithm to construct quantum BCH codes. We used *The Mathematica Book* [10] to search for BCH codes over $\mathbf{F}_9$ and $\mathbf{F}_{16}$ to give us codes over $\mathbf{F}_3$ and $\mathbf{F}_4$.

We can also calculate the generator polynomials for these codes and their dual codes using the roots given by the program. Using these codes, we provide the following parameters of nonbinary quantum BCH codes over $\mathbf{F}_3$ and $\mathbf{F}_4$. After finding out some quantum BCH codes along with the zeros of the code and its dual, the following observations can be made

- The codes currently found do not cover the continuous values of n. This is because $n|(q^m - 1)$. As m increases, the program takes longer time to run. For values of q as 9 and 16, m can, at most, be 3 for the program to run in a decent amount of time. This puts restrictions on the values of n.

- The minimum distances of these quantum BCH codes are not very good. However, some of them are MDS codes. Thus, they help in improving the lower bounds to a large extent in combination with the general code constructions which are explained in the next chapter.

Table I gives the quantum BCH codes over $\mathbf{F}_3$. In the following tables, the codes shown in **bold** font are the codes that meet the proposed upper bound on the minimum distance. The roots of the generator polynomial of the code $g(x)$ and the generator polynomial of the dual code $h^*(x)$ are given. The roots are given as the powers of the primitive element in the field $\mathbf{F}_{q^m}$. The codes over $\mathbf{F}_4$ are given in Table II.

Table I.: Parameters of quantum BCH codes over $\mathbf{F}_3$

$[[n,k,d]]$ codes	Roots of $g(x)$	Roots of $h^*(x)$
$[[\mathbf{8},\mathbf{6},\mathbf{2}]]$ (m=1)	$1,0,7,2,4,6,3$	1
$[[10,2,3]]$ (m=2)	$0,24,56,40,32,48$	$24,48,32,56$
$[[10,6,2]]$ (m=2)	$0,24,56,40,64,32,16,48$	$24,56$
$[[13,1,3]]$ (m=3)	$504,336,0,280,56,168,112$	$56,112,168,280,$ $336,504$
$[[\mathbf{13},\mathbf{7},\mathbf{3}]]$ (m=3)	$560,224,504,336,0,280,$ $56,168,672,112$	$112,280,336$
$[[16,6,4]]$ (m=2)	$55,15,5,20,45,60,0,30,$ $10,40,70$	$45,10,55,5,15$
$[[16,10,3]]$ (m=2)	$55,15,75,5,20,45,60,35,$ $0,30,10,40,70$	$45,10,5$
$[[\mathbf{16},\mathbf{12},\mathbf{2}]]$ (m=2)	$55,15,75,5,20,45,60,35,$ $0,30,10,40,50,70$	$45,5$
$[[\mathbf{16},\mathbf{14},\mathbf{2}]]$ (m=2)	$55,15,25,75,5,20,65,45,$ $60,35,0,30,10,40,70$	10
$[[20,4,5]]$ (m=2)	$20,60,0,72,8,40,52,64,$ $44,16,76,68$	$44,8,52,16,68,$ $76,64,72$
$[[20,8,4]]$ (m=2)	$20,60,0,72,8,40,52,64,$ $32,44,16,76,68,48$	$44,8,52,68,$ $76,72$
$[[20,12,3]]$ (m=2)	$20,60,0,72,8,4,36,40,$ $52,64,32,44,16,76,68,48$	$44,8,52,68,$ $76,72$

Continued on next page

Table I – Continued from previous page

$[[n, k, d]]$ codes	Roots of $g(x)$	Roots of $h^*(x)$
$[[\mathbf{20}, \mathbf{16}, \mathbf{2}]]$ (m=2)	$20, 60, 0, 24, 72, 8, 4, 56, 36, 40,$ $52, 64, 32, 44, 16, 76, 68, 48$	$44, 76$
$[[26, 14, 3]]$ (m=3)	$196, 560, 224, 700, 504, 588, 0, 84,$ $476, 308, 252, 392, 364, 56, 448,$ $616, 644, 28, 168, 672$	$196, 392, 588,$ $448, 308, 616$
$[[26, 20, 2]]$(m=3)	$196, 560, 224, 700, 504, 588, 336, 0,$ $84,476,$ $308, 252, 392, 280, 364, 56, 448, 616,$ $644, 28, 168, 672, 112$	$196, 588, 308$
$[[40, 24, 5]]$ (m=2)	$2, 6, 42, 46, 58, 54, 20, 18,$ $60, 14, 0, 12, 72, 8, 30, 4$ $1, 0, 28, 36, 40, 22,$ $64, 32, 38, 44, 16, 76, 66, 50,$ $34, 70, 48$	$2, 4, 6, 8, 18, 36,$ $54, 72$
$[[40, 26, 5]]$ (m=2)	$2, 6, 42, 46, 58, 54, 20, 18,$ $60, 14, 0, 12, 72, 8, 74, 30,$ $4, 26, 10, 28, 36,$ $40, 22, 64, 32, 38, 44, 16, 76,$ $66, 34, 70, 48$	$4, 6, 8, 10, 36,$ $54, 72$
$[[40, 28, 4]]$ (m=2)	$2, 6, 42, 46, 58, 54, 20, 18,$ $60, 14, 0,$ $24, 72, 8, 30, 4, 56, 10, 28,$	$2, 4, 6, 18, 36,$ $54, 12,$

Continued on next page

Table I – Continued from previous page

$[[n, k, d]]$ codes	Roots of $g(x)$	Roots of $h^*(x)$
$[[40, 30, 4]]$ (m=2)	$36, 40, 22, 64, 32,$ $38, 44, 16, 76, 66, 50, 34, 70,$ 48 $2, 6, 42, 46, 58, 54, 20, 18,$ $60, 14, 0, 12, 72,$ $8, 74, 30, 4, 26, 10, 28, 36,$ $40, 52, 22, 64,$ $32, 38, 44, 16, 76, 66, 68, 34,$ $70, 48$	$6, 8, 10, 54, 72$
$[[40, 32, 3]]$ (m=2)	$2, 6, 42, 46, 58, 54, 20, 18,$ $60, 14, 0, 12, 24, 78, 72, 62,$ $8, 30, 4, 56, 10, 28, 36, 40,$ $22, 64, 32, 38, 44, 16, 76, 66,$ $50, 34, 70, 48$	$2, 4, 28, 36$
$[[40, 34, 3]]$ (m=2)	$2, 6, 42, 46, 58, 54, 20, 18,$ $60, 14, 0, 12, 78, 72, 62, 8,$ $30, 4, 26, 10, 28, 36, 40, 52,$ $22, 64, 32, 38, 44, 16, 76, 66,$ $74, 34, 70, 48, 68,$	$8, 10, 72$
$[[40, 36, 2]]$ (m=2)	$2, 6, 42, 46, 58, 54, 20, 18, 60, 14, 0, 12, 24,$ $78, 72, 62, 8, 30, 4, 56, 10, 28, 36, 40, 52, 22,$ $64, 32, 38, 44, 16, 76, 66, 68, 50, 34, 70, 48$	$2, 18$

Continued on next page

Table I – Continued from previous page

$[[n,k,d]]$ codes	Roots of $g(x)$	Roots of $h^*(x)$
$[[40,38,2]]$ (m=2)	$2,6,42,46,58,54,20,18,60,14,0,12$ $24,78,72,62,8,74,30,4,56,26,10,28,36,$ $40,52,22,64,32,38,44,16,76,66,50,34,$ $70,48$	$4,36$

Table II.: Parameters of quantum BCH codes over $\mathbf{F}_4$

$[[n,k,d]]$ codes over $\mathbf{F}_4$	$[[n,k,d]]$ codes
$[[7,1,3]]$ (m=3)	$[[21,13,3]]$ (m=3)
$[[9,1,3]]$ (m=3)	$[[21,15,2]]$ (m=3)
$[[9,3,2]]$ (m=3)	$[[21,19,2]]$ (m=3)
$[[\mathbf{9},\mathbf{7},\mathbf{2}]]$ (m=3)	$[[35,5,6]]$ (m=3)
$[[17,9,3]]$ (m=2)	$[[35,17,4]]$ (m=3)
$[[\mathbf{17},\mathbf{13},\mathbf{3}]]$ (m=2)	$[[35,23,3]]$ (m=3)
$[[\mathbf{15},\mathbf{11},\mathbf{3}]]$ (m=1)	$[[35,29,2]]$ (m=3)
$[[\mathbf{15},\mathbf{13},\mathbf{2}]]$ (m=1)	$[[39,27,3]]$ (m=3)
$[[21,1,5]]$ (m=3)	$[[39,31,3]]$ (m=3)
$[[21,3,5]]$ (m=3)	$[[39,33,2]]$ (m=3)
$[[21,7,4]]$ (m=3)	$[[39,37,2]]$ (m=3)
$[[21,9,3]]$ (m=3)	

Example 18. Consider the example of $[[8, 6, 2]]_3$. The generator polynomial of this code is given by

$$g(x) = x^7 + \omega^5 x^6 + \omega^2 x^5 + \omega^7 x^4 + \omega^4 x^3 + \omega x^2 + \omega^6 x + \omega^3.$$

The generator polynomial of its dual code is

$$h^*(x) = x - \omega.$$

Note that degree of $g(x)$ is 7 and that of $h^*(x)$ is 1. Hence, $k = 7 - 1 = 6$ and the quantum BCH code obtained is a $[[8, 6, 2]]_3$ code. It can also be verified that these codes are orthogonal with respect to the Hermitian product by constructing their generator matrices. Table II gives the quantum BCH codes over $\mathbf{F}_4$. Roots of these codes can also be provided.

The third construction in [9] can be generalized. This construction is mainly useful for finding higher dimensional codes.

Lemma 19. *Let $C = [n, k, d]$ be a self-orthogonal linear code over $\mathbf{F}_{q^l}$ that has the dual code $C^{\perp_s} = [n, n-k, d^{\perp_s}]$. Let B be a self dual basis of $\mathbf{F}_{q^l}$ over $\mathbf{F}_q$. Expanding each element of $\mathbf{F}_{q^l}$ with respect to the basis B gives a linear self-orthogonal code $C_2 = [ln, lk, d_2 \geq d]$ over $\mathbf{F}_q$. The dual $C^{\perp_s} = [ln, l(n-k), {d_2}^{\perp_s} \geq d^{\perp_s}]$ is obtained in the same manner. The results in chapter II can then be used to construct the quantum error correcting code.*

Proof. The theorem is proved by expanding the vectors in self dual basis [15]. Let C and $C^{\perp}$ denote the BCH code and its dual with respect to the standard inner product over $\mathbf{F}_{q^l}$. Let u, v be length n vectors such that $u \in C$ and $v \in C^{\perp}$. Let $\alpha = \alpha_1, \alpha_2 \ldots \alpha_l$ denote the self dual basis over $\mathbf{F}_q$ from $\mathbf{F}_{q^l}$, then

$$\langle u, v\rangle = \sum_{i=1}^{n} u_i v_i = 0.$$

Taking the trace operation from $\mathbf{F}_{q^l}$ to $\mathbf{F}_q$ and expanding over the l-ary self dual basis,

$$\begin{aligned} 0 &= \operatorname{tr}(\sum_{i=1}^{n} u_i v_i) \\ &= \operatorname{tr}(\sum_{i=1}^{n}\sum_{j=1}^{l} u_{ij}\alpha_j \sum_{k=1}^{l} v_{ik}\alpha_k) \\ &= \operatorname{tr}(\sum_{i=1}^{n}\sum_{j=1}^{l} u_{ij}v_{ij}\alpha_j\alpha_j) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{l} u_{ij}v_{ij}\operatorname{tr}(\alpha_j\alpha_j) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{l} u_{ij}v_{ij}. \end{aligned}$$

Thus, expanding in the specific bases retains the self-orthogonality over the new code and the new code remains self-orthogonal. The results in Chapter II can then be used to construct the corresponding quantum code. □

We use this result to construct quantum BCH codes. Tables III and IV show codes constructed from the classical BCH codes over higher fields.

Similar construction is now explained which again gives us quantum codes with large values of n and k.

Theorem 20. *If a BCH code exists over $\mathbf{F}_{q^{2l}}$ self-orthogonal with respect to the product, $\langle u, v\rangle^h = \sum_{i=1}^{n} u_i v_i^q = 0$, all the q^2-ary images of this code will be again Hermitian self-orthogonal where the l-ary expansion is done over specific basis.*

Proof. Let C and $C^\perp$ denote the BCH code and its dual with respect to the defined product over $\mathbf{F}_{q^{2l}}$. Let u, v be length n vectors such that $u \in C$ and $v \in C^\perp$. Let

Table III. Parameters of quantum BCH codes over $\mathbf{F}_3$ from $\mathbf{F}_{q^l}$

Extension field $\mathbf{F}_{q^l}$ for classical code	Quantum BCH code over $\mathbf{F}_q$
27	$[[24, 12, 2]]$, $[[32, 8, 4]]$, $[[32, 16, 3]]$, $[[32, 24, 2]]$, $[[39, 27, 3]]$, $[[39, 21, 4]]$, $[[39, 15, 5]]$, $[[39, 9, 6]]$, $[[39, 3, 7]]$, $[[78, 6, 13]]$, $[[78, 12, 12]]$, $[[78, 18, 11]]$ $[[78, 24, 10]]$, $[[78, 30, 9]]$, $[[78, 36, 8]]$, $[[78, 42, 7]]$ $[[78, 48, 6]]$, $[[78, 54, 5]]$, $[[78, 60, 4]]$, $[[78, 66, 3]]$ $[[78, 72, 2]]$
81	$[[40, 8, 5]]$, $[[40, 16, 4]]$, $[[40, 24, 3]]$, $[[40, 32, 2]]$, $[[64, 8, 8]]$, $[[64, 16, 7]]$, $[[64, 24, 6]]$, $[[64, 32, 5]]$, $[[64, 40, 4]]$, $[[64, 48, 3]]$, $[[64, 56, 2]]$, $[[80, 8, 10]]$ $[[80, 16, 9]]$, $[[80, 24, 8]]$, $[[80, 32, 7]]$, $[[80, 40, 6]]$ $[[80, 48, 5]]$, $[[80, 56, 4]]$, $[[80, 64, 3]]$, $[[80, 72, 2]]$

Table IV. Parameters of quantum BCH codes over $\mathbf{F}_4$ from $\mathbf{F}_{q^l}$

Extension field $\mathbf{F}_{q^l}$ for classical code	Quantum BCH code over $\mathbf{F}_q$
64	$[[21,3,4]]$, $[[21,9,3]]$, $[[21,15,2]]$, $[[27,3,5]]$, $[[27,9,4]]$, $[[27,15,3]]$, $[[27,21,2]]$, $[[45,21,3]]$, $[[45,27,3]]$, $[[45,33,2]]$, $[[45,39,2]$, $[[63,3,11]]$ $[[63,3,11]]$, $[[63,3,11]]$, $[[63,3,11]]$, $[[63,3,11]]$ $[[63,3,11]]$, $[[63,9,10]]$, $[[63,15,9]]$, $[[63,21,8]]$ $[[63,27,7]]$, $[[63,33,6]]$, $[[63,39,5]]$, $[[63,45,4]]$, $[[63,51,3]]$, $[[63,57,2]]$, $[[105,93,2]]$, $[[105,99,2]]$, $[[105,81,3]]$, $[[105,87,3]]$, $[[105,69,4]]$, $[[105,75,4]]$, $[[105,57,5]]$, $[[105,63,5]]$, $[[105,51,6]]$,
256	$[[20,4,3]]$, $[[20,12,2]]$, $[[60,4,8]]$, $[[60,12,7]]$, $[[60,20,6]]$, $[[60,28,5]]$, $[[60,36,4]]$, $[[60,44,3]]$, $[[60,52,2]]$, $[[68,4,9]]$, $[[68,12,8]]$, $[[68,20,7]]$, $[[68,28,6]]$, $[[68,36,5]]$, $[[68,44,4]]$, $[[68,52,3]]$, $[[68,60,2]]$

$\alpha = \alpha_1, \alpha_2 \ldots \alpha_l$ denote the self dual basis over $\mathbf{F}_{q^2}$ from $\mathbf{F}_{q^{2l}}$. Let $\beta = \beta_1, \beta_2, \ldots, \beta_l$ be another basis where $\beta = \alpha^{q^{2l-1}}$.

$$\langle u, v \rangle_h = \sum_{i=1}^{n} u_i^q v_i = 0.$$

Taking the trace operation from $\mathbf{F}_{q^{2l}}$ to $\mathbf{F}_{q^2}$ and expanding over the l-ary self dual basis,

$$\begin{aligned}
0 &= \operatorname{tr}(\sum_{i=1}^{n} u_i^q v_i) \\
&= \operatorname{tr}(\sum_{i=1}^{n} \sum_{j=1}^{l} u_{ij}^q \beta_j^q \sum_{k=1}^{l} v_{ik} \alpha_k) \\
&= \operatorname{tr}(\sum_{i=1}^{n} \sum_{j=1}^{l} u_{ij}^q v_{ij} \beta_j^q \alpha_j) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{l} u_{ij}^q v_{ij} \operatorname{tr}(\alpha_j^{q^l} \alpha_j) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{l} u_{ij}^q v_{ij} \operatorname{tr}(\alpha_j \alpha_j) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{l} u_{ij}^q v_{ij}.
\end{aligned}$$

Thus, expanding in specific bases retains the self-orthogonality over the new code and the new code remains self-orthogonal. The connection given in Chapter II can then be used to construct the corresponding quantum code. □

We use this result to construct the following codes. Tables V and VI show codes constructed from the classical BCH codes over higher fields.

The list of codes provided here is not complete and can be extended by using more powerful machines. The codes provided here, however, illustrate how the quantum BCH code families are constructed in general. We start with these codes in the tables

Table V. Parameters of quantum BCH codes over $\mathbf{F}_3$ from $\mathbf{F}_{q^{2l}}$

Extension field $\mathbf{F}_{q^{2l}}$ for classical code	Quantum BCH code over $\mathbf{F}_q$
27	$[[24, 12, 2]]$, $[[39, 33, 2]]$, $[[39, 27, 3]]$, $[[39, 21, 4]]$, $[[39, 15, 5]]$, $[[39, 9, 6]]$, $[[39, 3, 7]]$
81	$[[20, 12, 2]]$, $[[20, 4, 3]]$, $[[32, 16, 3]]$, $[[32, 8, 4]]$, $[[32, 24, 2]]$, $[[64, 8, 8]]$, $[[64, 16, 7]]$, $[[64, 24, 6]]$, $[[64, 32, 5]]$, $[[40, 32, 2]]$, $[[40, 24, 3]]$, $[[40, 16, 4]]$, $[[64, 40, 4]]$, $[[64, 48, 3]]$, $[[64, 56, 2]]$, $[[80, 8, 10]]$, $[[80, 16, 9]]$, $[[80, 24, 8]]$, $[[80, 32, 7]]$, $[[80, 40, 6]]$, $[[80, 48, 5]]$, $[[80, 56, 4]]$, $[[80, 64, 3]]$, $[[80, 72, 2]]$, $[[40, 8, 5]]$

Table VI. Parameters of quantum BCH codes over $\mathbf{F}_4$ from $\mathbf{F}_{q^{2l}}$

Extension field $\mathbf{F}_{q^{2l}}$ for classical code	Quantum BCH code over $\mathbf{F}_q$
64	$[[21, 3, 4]]$, $[[21, 9, 3]]$, $[[21, 15, 2]]$, $[[63, 39, 5]]$, $[[63, 45, 4]]$, $[[63, 51, 3]]$, $[[63, 57, 2]]$
256	$[[20, 4, 3]]$, $[[20, 12, 2]]$, $[[60, 4, 8]]$, $[[60, 12, 7]]$, $[[60, 20, 6]]$, $[[60, 28, 5]]$, $[[60, 36, 4]]$, $[[60, 44, 3]]$ $[[60, 52, 2]]$

and go on improving the lower bounds throughout the tables using different code constructions. These code constructions are explained in Chapter IV.

CHAPTER IV

GENERAL CODE CONSTRUCTIONS

The code constructions given in this chapter construct nonbinary quantum stabilizer codes from the existing ones. We use two notions of duality, the trace-symplectic form and the trace-alternating form. The main sources of the construction of stabilizer codes are in the following theorems that are discussed in Chapter II

Theorem 21. *If there exists a classical* $[2n, q^{n-k}]_q$ *code* C*, self-orthogonal with respect to the trace-symplectic form, then there exists a quantum code with the parameters* $[[n, k, d]]_q$*, where* $d = \min\{w(x) : x \in C^{\perp_{ts}} \backslash C\}$.

Theorem 22. *If there exists a classical* $[n, q^{n-k}]_{q^2}$ *code* C*, self-orthogonal with respect to the trace-alternating form, then there exists a quantum code with the parameters* $[[n, k, d]]$*, where* $d = \min\{w(x) : x \in C^{\perp_a} \backslash C\}$.

A. Constructions with Single Starting Code

Lemma 23. *If an* $[[n, k, d]]_q$ *stabilizer code exists for* $k > 0$*, then there exists an impure* $[[n + 1, k, d]]_q$ *stabilizer code.*

Proof. If an $[[n, k, d]]_q$ stabilizer code exists, then there exists an additive subcode $C \leq \mathbf{F}_q^{2n}$ such that $|C| = q^{n-k}$, $C \leq C^{\perp_{ts}}$, and $\mathrm{swt}(C^{\perp_{ts}} \setminus C) = d$. Define the additive code

$$C' = \{(a\alpha|b0) \,|\, \alpha \in \mathbf{F}_q, (a|b) \in \mathbf{F}_q^{2n}\}.$$

We have $|C'| = q^{n-k+1}$. The definition ensures that C' is self-orthogonal with respect to the trace-symplectic form. Indeed, two arbitrary elements $(a\alpha|b0)$ and $(a'\alpha'|b'0)$ of

C' satisfy the duality condition

$$\langle (a\alpha|b0)|(a'\alpha'|b'0)\rangle_{ts} = \langle (a|b)|(a'|b')\rangle_{ts} + \mathrm{tr}(\alpha\cdot 0 - \alpha'\cdot 0) = 0.$$

A vector in the trace-symplectic dual of C' has to be of the form $(a\alpha|b0)$ with $(a|b) \in C^{\perp_{ts}}$ and $\alpha \in \mathbf{F}_q$. Furthermore, the symplectic weight given by

$$\mathrm{swt}(C'^{\perp_{ts}} \setminus C') = \min\{\mathrm{swt}(a\alpha|b0) \,|\, \alpha \in \mathbf{F}_q, a, b \in C^{\perp_{ts}} \setminus C\},$$

coincides with $\mathrm{swt}(C^{\perp_{ts}} \setminus C)$. Therefore, an $[[n+1, k, d]]_q$ stabilizer code exists. If $d > 1$, then the code is impure, because $C'^{\perp_{ts}}$ contains the vector $(0\alpha|00)$ of symplectic weight 1. □

Lemma 24. *If a pure $[[n, k, d]]_q$ stabilizer code exists with $n \geq 2$ and $d \geq 2$, then there exists a pure $[[n-1, k+1, d-1]]_q$ stabilizer code.*

Proof. If a pure $[[n, k, d]]_q$ stabilizer code exists, then there exists an additive code $D \leq \mathbf{F}_{q^2}^n$ that is self-orthogonal with respect to the trace-alternating form, so that $|D| = q^{n-k}$ and $\mathrm{wt}(D^{\perp_a}) = d$. Let $D_0^{\perp_a}$ denote the code obtained by puncturing the first coordinate of $D^{\perp_{ta}}$. Since the minimum distance of $D^{\perp_a}$ is at least 2, we know that $|D_0^{\perp_a}| = |D^{\perp_a}| = q^{n+k}$, and we note that the minimum distance of $D_0^{\perp_a}$ is $d-1$. The dual of $D_0^{\perp_a}$ consists of all vectors u in $\mathbf{F}_{q^2}^{n-1}$ such that $0u$ is contained in D. Furthermore, if u is an element of D_0, then $0u$ is contained in D; hence, D_0 is a self-orthogonal additive code. The code D_0 is of size $q^{(n-1)-(k+1)}$, because

$$\dim D_0 + \dim D_0^{\perp_a} = \dim \mathbf{F}_{q^2}^{n-1}$$

when we view D_0 and its dual as $\mathbf{F}_p$–vector spaces. It follows that there exists a pure $[[n-1, k+1, d-1]]_q$ stabilizer code. □

Lemma 25. *If a (pure)* $[[n,k,d]]_q$ *stabilizer code exists with* $n \geq 2$ *(*$k \geq 1$*), then there exists a (pure)* $[[n,k-1,d^*]]_q$ *stabilizer code such that* $d^* \geq d$*.*

Proof. If an $[[n,k,d]]_q$ stabilizer code exists, then there exists an additive code $D \leq \mathbf{F}_{q^2}^n$ such that $D \leq D^{\perp_a}$ with $\mathrm{wt}(D^{\perp_a} \setminus D) = d$ and $|D| = q^{n-k}$. Choose an additive code D_b of size $|D_b| = q^{n-k+1}$ such that $D \leq D_b \leq D^{\perp_a}$. Since $D \leq D_b$, we have $D_b^{\perp_a} \leq D^{\perp_a}$. The set $\Sigma_b = D_b^{\perp_a} \setminus D_b$ is a subset of $D^{\perp_a} \setminus D$, hence, the minimum weight d^* of Σ_b is at least d. This proves the existence of an $[[n,k-1,d^*]]$ code.

If the code is pure, then $\mathrm{wt}(D^{\perp_a}) = d$. It follows from $D_b^{\perp_a} \leq D^{\perp_a}$ that $\mathrm{wt}(D_b^{\perp_a}) \geq d$, hence the smaller code is pure as well. □

Corollary 26. *If a pure* $[[n,k,d]]_q$ *stabilizer code with* $n \geq 2$ *and* $d \geq 2$ *exists, then there exists a pure* $[[n-1,k,d-1]]_q$ *stabilizer code.*

Proof. Combine Lemmas 24 and 25. □

Lemma 27. *Suppose that* Q *is a pure* $[[n,k,d]]_q$ *stabilizer code with* $n \geq 2$ *and* $d \geq 2$ *such that the stabilizer group contains an element of weight 1. Then there exists an* $[[n-1,k,d]]_q$ *stabilizer code.*

Proof. It follows from the hypothesis that there exists an additive code $D \leq \mathbf{F}_{q^2}^n$ of size $|D| = q^{n-k}$ that contains a vector v of weight 1, and satisfies $\mathrm{wt}(D_a^{\perp} \setminus D) = d$. Let D_0 denote the code obtained by puncturing D at the nonzero position of v. □

B. Constructions with Two Starting Codes

Lemma 28. *Consider two codes* $[[n_1,k_1,d_1]]_q$ *and* $[[n_2,k_2,d_2]]_q$ *defined by* Q_1 *and* Q_2*, respectively. The direct sum of these two codes, defined as* $Q_1 \oplus Q_2 = \{uv : u \in Q_1, v \in Q_2\}$*, will give a code* $[[n_1+n_2,k_1+k_2,d]]_q$*, where* $d = min\{d_1,d_2\}$ *and* uv *is the concatenation of vectors.*

Proof. The proof of the direct sum method from [4] generalizes. □

Lemma 29. *Suppose that an* $[[n, K, d]]_q$ *and an* $[[n', K', d']]_q$ *stabilizer code exists. K and K' are the dimensions of these codes. Then there exists an* $[[n+n', KK', \min(d, d')]]_q$ *stabilizer code.*

Proof. Suppose that P and P' are the orthogonal projectors onto the stabilizer codes for the $[[n, K, d]]_q$ and $[[n', K', d']]_q$ stabilizer codes, respectively. Then $P \otimes P'$ is an orthogonal projector onto a KK'-dimensional subspace Q^* of $\mathbf{C}^d$, where $d = q^{n+n'}$. Let S and S', respectively, denote the stabilizer groups of the images of P and P'. Then $S^* = \{E \otimes E' \mid E \in S, E' \in S'\}$ is the stabilizer group of Q^*.

If an element $F \otimes F^*$ of $G_n \otimes G_{n'} = G_{n+n'}$ is not detectable, then F has to commute with all elements in S, and F' has to commute with all elements in S'. It is not possible that both $F \in Z(G_n)S$ and $F' \in Z(G_{n'})S'$ hold, because this would imply that $F \otimes F'$ is detectable. $Z(G_n)S$ is nothing but the center of the error group G_n. Therefore, either F or F' is not detectable, which shows that the weight of $F \otimes F'$ is at least $\min(d, d')$. □

Another direct sum construction for codes over $\mathbf{F}_4$ was discussed in [4]. We now explain a similar construction for nonbinary codes over $\mathbf{F}_{q^2}$.

Lemma 30. *Let* C_1 *be an* $[n, q^{n-k_1}]$ *self-orthogonal code corresponding to an* $[[n, k_1, d_1]]$ *code and* C_2 *be an* $[n, q^{n-k_2}]$ *self-orthogonal code corresponding to an* $[[n, k_2, d_2]]$ *code, such that* $C_1 \subseteq C_2$. *The self-orthogonality is with respect to the trace-alternating form. Then there exists a* $[[2n, k_1 + k_2, d]]$ *code, where* $d = min\{2d_2, d_1\}$.

Proof. Take C to be the $(2n, q^{2n-(k_1+k_2)})$ additive code consisting of vectors $u|u+v$, such that $u \in C_1$ and $v \in C_2$, where bar denotes concatenation. Then $C^{\perp_a} = \{u+v|v : u \in C_1^{\perp_a}, v \in C_2^{\perp_a}\}$ that has minimum distance $\min\{2d_2, d_1\}$. Let $v_1 = (u|u+v) \in C$

and $v_2 = (u' + v'|v') \in C^{\perp_a}$. These vectors remain self-orthogonal because

$$\begin{aligned} \langle v_1, v_2 \rangle_a &= \langle (u|u+v), (u'+v'|v') \rangle_a \\ &= \langle u, u' \rangle_a + \langle u, v' \rangle_a + \langle u, v' \rangle_a + \langle v, v' \rangle_a \\ &= 0. \end{aligned}$$

$C_2^{\perp_a} \subseteq C_1^{\perp_a}$. Hence, $\langle u, v' \rangle_a = 0$. □

Lemma 31. *Let C_1 be an $[n, q^{n-k_1}]$ self-orthogonal code corresponding to an $[[n, k_1, d_1]]_q$ code and C_2 be an $[n, q^{n-k_2}]$ self-orthogonal code corresponding to an $[[n, k_2, d_2]]_q$ code, such that $C_1 \subseteq C_2$. Then there exists a $[[2n, k_1 - k_2, d]]$ code, where $d = min\{2d_1, d_2\}$ and suppose that q is even.*

Proof. Take C to be the $[2n, q^{2n-k_1+k_2}]$ additive code consisting of vectors $u|u+v$ such that $u \in C_2^{\perp_a}$ and $v \in C_1$, where bar denotes concatenation. Then $C^{\perp_a} = \{u|u+v : u \in C_1^{\perp_a}, v \in C_2\}$ that has minimum distance $\min\{2d_1, d_2\}$. Let $v_1 = (u|u+v) \in C$ and $v_2 = (u'|u'+v') \in C^{\perp_a}$. These vectors remain self-orthogonal because

$$\begin{aligned} \langle v_1, v_2 \rangle_a &= \langle (u|u+v), (u'|u'+v') \rangle_a \\ &= \langle u, u' \rangle_a + \langle u, u' \rangle_a + \langle u, v' \rangle_a + \langle v, u' \rangle_a + \langle v, v' \rangle_a \\ &= 2\langle u, u' \rangle_a \\ &= 0 \end{aligned}$$

for even q. We have, $\langle v, v' \rangle_a = 0$ because $C_1 \subseteq C_2 \subseteq {C_2}^{\perp_a} \subseteq {C_1}^{\perp_a}$. All vectors in C_2 are thus in ${C_1}^{\perp_a}$, and hence, orthogonal to all vectors in C_1. Further, the minimum distance is $\min\{2d_1, d_2\}$. Hence, proved by Theorem 22. □

These code constructions help in improving the lower bounds for nonbinary quan-

tum stabilizer codes. We illustrate this by a small part of our tables. We make use of some existing quantum codes along with these code constructions.

CHAPTER V

IMPROVING THE LOWER BOUNDS ON THE MINIMUM DISTANCE

This chapter explains how new quantum codes are constructed from the existing ones. These code constructions help improve the lower bounds for nonbinary quantum stabilizer codes. This is illustrated by a small part of our tables. Some existing quantum codes are used along with these code constructions to achieve the improvements. An improvement in a single place in the table, spreads in all the directions in the table using these constructions. Thus, these construction play a very important role in the whole process of improving the lower bounds.

A. A Small Example

Table VII shows a small part of the lower and upper bounds' pairs for codes over $\mathbf{F}_3$. The existing codes are shown in **bold** font in the table. How the lower bounds get spread from these existing codes is shown using different superscripts. The explanation of these superscripts is now given.

- α: Using Theorem 11
- β: Using Theorem 12
- γ: Using Theorem 14
- δ: Using Theorem 13
- p: Product Code
- g: Using [13]

Table VII. General code constructions

n/k	1	2	3	4	5	6
3	$\mathbf{2-2}^{\beta}$	$1-1^{\leftarrow}$	$\mathbf{1-1}^{\gamma}$	-	-	-
4	$2-2^{\leftarrow}$	$\mathbf{2-2}^{\beta}$	$1-1^{\leftarrow}$	$\mathbf{1-1}^{\gamma}$	-	-
5	$\mathbf{3-3}^{\beta}$	$2-2^{\leftarrow}$	$\mathbf{2-2}^{\beta}$	$1-1^{\leftarrow}$	$\mathbf{1-1}^{\gamma}$	-
6	$3-3^{\leftarrow}$	$\mathbf{3-3}^{\beta}$	$2-2^{\leftarrow}$	$\mathbf{2-2}^{\beta}$	$1-1^{\leftarrow}$	$\mathbf{1-1}^{\gamma}$
7	$3-4^{\leftarrow}$	$3-3^{\leftarrow}$	$\mathbf{3-3}^{\beta}$	$2-2^{\leftarrow}$	$\mathbf{2-2}^{\beta}$	$1-1^{\leftarrow}$
8	$3-4^{\leftarrow}$	$3-4^{d}$	$3-3^{\leftarrow}$	$\mathbf{3-3}^{\alpha}$	$2-2^{\leftarrow}$	$\mathbf{2-2}^{\alpha}$
9	$\mathbf{4-5^{g}}$	$3-4^{\leftarrow}$	$3-4^{\leftarrow}$	$3-3^{\leftarrow}$	$\mathbf{3-3}^{\alpha}$	$2-2^{\leftarrow}$
10	$4-5^{\downarrow}$	$3-5^{\leftarrow}$	$3-4^{\leftarrow}$	$3-4^{\leftarrow}$	$3-3^{\leftarrow}$	$\mathbf{3-3}^{\delta}$
11	$4-6^{\downarrow}$	$3-5^{\leftarrow}$	$3-5^{d}$	$3-4^{\downarrow}$	$3-4^{\downarrow}$	$3-3^{\downarrow}$
12	$4-6^{\downarrow}$	$\mathbf{4-6^{p}}$	$3-5^{d}$	$3-5^{\downarrow}$	$3-4^{\downarrow}$	$3-3^{\downarrow}$
13	$4-7^{\downarrow}$	$4-6^{\downarrow}$	$4-6^{d}$	$4-5^{\downarrow}$	$3-4^{\downarrow}$	$3-4^{\downarrow}$
14	$5-7^{\leftarrow}$	$5-7^{d}$	$5-6^{\downarrow}$	$4-5^{\downarrow}$	$3-5^{\downarrow}$	$3-4^{\downarrow}$
15	$\mathbf{6-8^{p}}$	$6-7^{\downarrow}$	$5-6^{\downarrow}$	$4-6^{\downarrow}$	$3-5^{\downarrow}$	$3-5^{\downarrow}$

- $\leftarrow$: Using Lemma 25
- $\downarrow$: Using Lemma 23
- d: Using Lemma 24

The following tables explain how the constructions help improve the lower bounds on the minimum distance of codes.

B. The Larger Tables

The larger tables again make use of the known codes, newly constructed codes and all the above lemmas. Different lemmas might give same improvement on the lower bounds. Hence to reduce the complexity, the code construction procedure is not shown in the tables. Tables VIII, IX, X, XI, XII and XIII only show the lower bound-upper bound pairs for all combinations of (n, k).

Table VIII.: Bounds on the minimum distance for codes over $\mathbf{F}_3$ (k:1 to 10)

n/k	1	2	3	4	5	6	7	8	9	10
3	2-2	1-1	1-1	-	-	-	-	-	-	-
4	2-2	2-2	1-1	1-1	-	-	-	-	-	-
5	3-3	2-2	2-2	1-1	1-1	-	-	-	-	-
6	3-3	3-3	2-2	2-2	1-1	1-1	-	-	-	-
7	3-4	3-3	3-3	2-2	2-2	1-1	1-1	-	-	-
8	3-4	3-4	3-3	3-3	2-2	2-2	1-1	1-1	-	-
9	4-5	3-4	3-4	3-3	3-3	2-2	2-2	1-1	1-1	-

Continued on next page

Table VIII – Continued from previous page

n/k	1	2	3	4	5	6	7	8	9	10
10	4-5	3-5	3-4	3-4	3-3	3-3	2-2	1-2	1-1	1-1
11	4-6	3-5	3-5	3-4	3-4	3-3	2-2	1-2	1-2	1-1
12	4-6	4-6	3-5	3-5	3-4	3-3	2-3	2-2	1-2	1-2
13	4-7	4-6	4-6	4-5	3-4	3-4	3-3	2-3	1-2	1-2
14	5-7	5-7	5-6	4-5	3-5	3-4	3-4	2-3	1-3	1-2
15	6-8	6-7	5-6	4-6	3-5	3-5	3-4	2-4	1-3	1-3
16	6-8	6-7	5-7	4-6	3-6	3-5	3-5	2-4	2-4	2-3
17	6-8	6-8	5-7	4-7	3-6	3-6	3-5	2-5	2-4	2-4
18	6-9	6-8	5-8	4-7	3-7	3-6	3-6	2-5	2-5	2-4
19	6-9	6-9	5-8	4-8	4-7	3-7	3-6	2-6	2-5	2-5
20	6-10	6-9	5-9	5-8	4-8	4-7	3-7	2-6	2-6	2-5
21	6-10	6-10	6-9	5-9	5-8	4-8	3-7	3-7	2-6	2-6
22	6-11	6-10	6-10	6-9	5-9	4-8	4-8	3-7	2-7	2-6
23	7-11	7-11	7-10	6-10	5-9	5-9	4-8	3-8	2-7	2-7
24	8-11	8-11	7-11	6-10	6-10	5-9	4-9	3-8	2-8	2-7
25	9-12	8-11	7-11	6-11	6-10	5-10	4-9	3-9	2-8	2-8
26	9-12	8-12	7-11	6-11	6-10	5-10	4-10	3-9	2-9	2-8
27	9-13	8-12	7-12	6-11	6-11	5-10	4-10	3-10	3-9	3-9
28	9-13	8-13	7-12	7-12	6-11	5-11	4-10	3-10	3-9	3-9
29	9-14	8-13	8-13	7-12	6-12	5-11	4-11	3-10	3-10	3-9
30	9-14	9-14	8-13	7-13	6-12	6-12	4-11	4-11	4-10	4-10

Table IX. Bounds on the minimum distance for codes over $\mathbf{F}_3$ (k:11 to 20)

n/k	11	12	13	14	15	16	17	18	19	20
10	-	-	-	-	-	-	-	-	-	-
11	1-1	-	-	-	-	-	-	-	-	-
12	1-1	1-1	-	-	-	-	-	-	-	-
13	1-2	1-1	1-1	-	-	-	-	-	-	-
14	1-2	1-2	1-1	1-1	-	-	-	-	-	-
15	1-2	1-2	1-2	1-1	1-1	-	-	-	-	-
16	2-3	2-2	2-2	2-2	1-1	1-1	-	-	-	-
17	2-3	2-3	2-2	2-2	1-2	1-1	1-1	-	-	-
18	2-4	2-3	2-3	2-2	1-2	1-2	1-1	1-1	-	-
19	2-4	2-4	2-3	2-3	1-2	1-2	1-2	1-1	1-1	-
20	2-5	2-4	2-4	2-3	1-3	1-2	1-2	1-2	1-1	1-1
21	2-5	2-5	2-4	2-4	1-3	1-3	1-2	1-2	1-2	1-1
22	2-6	2-5	2-5	2-4	1-4	1-3	1-3	1-2	1-2	1-2
23	2-6	2-6	2-5	2-5	1-4	1-4	1-3	1-3	1-2	1-2
24	2-7	2-6	2-5	2-5	1-4	1-4	1-4	1-3	1-3	1-2
25	2-7	2-6	2-6	2-5	1-5	1-4	1-4	1-4	1-3	1-3
26	2-7	2-7	2-6	2-6	2-5	2-5	2-4	2-4	2-4	2-3
27	3-8	3-7	3-7	3-6	3-6	3-5	3-5	3-4	3-4	3-4
28	3-8	3-8	3-7	3-7	3-6	3-6	3-5	3-5	3-4	3-4
29	3-9	3-8	3-8	3-7	3-7	3-6	3-6	3-5	3-5	3-4
30	4-9	4-9	4-8	4-8	4-7	4-7	4-6	4-6	4-5	4-5

Table X. Bounds on the minimum distance for codes over $\mathbf{F}_3$ (k:21 to 30)

n/k	21	22	23	24	25	26	27	28	29	30
20	-	-	-	-	-	-	-	-	-	-
21	1-1	-	-	-	-	-	-	-	-	-
22	1-1	1-1	-	-	-	-	-	-	-	-
23	1-2	1-1	1-1	-	-	-	-	-	-	-
24	1-2	1-2	1-1	1-1	-	-	-	-	-	-
25	1-2	1-2	1-2	1-1	1-1	-	-	-	-	-
26	2-3	2-2	2-2	1-2	1-1	1-1	-	-	-	-
27	3-3	3-3	2-2	1-2	1-2	1-1	1-1	-	-	-
28	3-4	3-3	2-3	2-2	1-2	1-2	1-1	1-1	-	-
29	3-4	3-4	3-3	2-3	1-2	1-2	1-2	1-1	1-1	-
30	4-4	4-4	3-3	2-3	2-2	2-2	1-2	1-2	1-1	1-1

Table XI.: Bounds on the minimum distance for codes over $\mathbf{F}_4$ (k:1 to 10)

n/k	1	2	3	4	5	6	7	8	9	10
3	2-2	1-1	1-1	-	-	-	-	-	-	-
4	2-2	2-2	1-1	1-1	-	-	-	-	-	-
5	3-3	2-2	2-2	1-1	1-1	-	-	-	-	-
6	3-3	3-3	2-2	2-2	1-1	1-1	-	-	-	-
7	3-4	3-3	3-3	2-2	2-2	1-1	1-1	-	-	-
8	4-4	4-4	3-3	3-3	2-2	2-2	1-1	1-1	-	-
9	4-5	4-4	3-4	3-3	3-3	2-2	2-2	1-1	1-1	-
10	4-5	4-5	4-4	4-4	3-3	3-3	2-2	2-2	1-1	1-1
11	4-6	4-5	4-5	4-4	3-4	3-3	3-3	2-2	2-2	1-1
12	4-6	4-6	4-5	4-5	4-4	4-4	3-3	3-3	2-2	2-2
13	4-7	4-6	4-6	4-5	4-5	4-4	3-4	3-3	3-3	2-2
14	5-7	5-7	4-6	4-6	4-5	4-5	4-4	4-4	3-3	3-3
15	6-8	5-7	4-7	4-6	4-6	4-5	4-5	4-4	3-4	3-3
16	6-8	5-8	4-7	4-7	4-6	4-6	4-5	4-5	4-4	4-4
17	6-9	5-8	5-8	4-7	4-7	4-6	4-6	4-5	4-5	4-4
18	6-9	6-9	5-8	4-8	4-7	4-7	4-6	4-6	4-5	4-5
19	6-10	6-9	5-9	4-8	4-8	4-7	4-7	4-6	4-6	4-5
20	6-10	6-10	5-9	5-9	4-8	4-8	4-7	4-7	4-6	4-6
21	6-11	6-10	6-10	5-9	5-9	4-8	4-8	4-7	4-7	4-6
22	6-11	6-11	6-10	6-10	5-9	4-9	4-8	4-8	4-7	4-6

Continued on next page

Table XI – Continued from previous page

n/k	1	2	3	4	5	6	7	8	9	10
23	7-12	7-11	7-11	6-10	5-10	4-9	4-9	4-8	4-7	4-7
24	8-12	8-12	7-11	6-11	5-10	4-10	4-9	4-8	4-8	4-7
25	9-13	8-12	7-12	6-11	5-11	4-10	4-9	4-9	4-8	4-8
26	9-13	8-13	7-12	6-12	5-11	5-10	4-10	4-9	4-9	4-8
27	9-14	8-13	7-13	6-12	6-11	5-11	5-10	4-10	4-9	4-9
28	9-14	8-14	7-13	7-12	6-12	6-11	5-11	4-10	4-10	4-9
29	9-15	8-14	8-13	7-13	7-12	6-12	5-11	4-11	4-10	4-10
30	9-15	9-14	8-14	8-13	7-13	6-12	5-12	4-11	4-11	4-10

Table XII. Bounds on the minimum distance for codes over $\mathbf{F}_4$ (k:11 to 20)

n/k	11	12	13	14	15	16	17	18	19	20
10	-	-	-	-	-	-	-	-	-	-
11	1-1	-	-	-	-	-	-	-	-	-
12	1-1	1-1	-	-	-	-	-	-	-	-
13	2-2	1-1	1-1	-	-	-	-	-	-	-
14	2-2	2-2	1-1	1-1	-	-	-	-	-	-
15	3-3	2-2	2-2	1-1	1-1	-	-	-	-	-
16	3-3	3-3	2-2	2-2	1-1	1-1	-	-	-	-
17	3-4	3-3	3-3	2-2	1-2	1-1	1-1	-	-	-
18	3-4	3-4	3-3	2-2	1-2	1-2	1-1	1-1	-	-
19	3-5	3-4	3-3	2-3	1-2	1-2	1-2	1-1	1-1	-
20	3-5	3-4	3-4	2-3	1-3	1-2	1-2	1-2	1-1	1-1
21	3-5	3-5	3-4	2-4	1-3	1-3	1-2	1-2	1-2	1-1
22	3-6	3-5	3-5	2-4	1-4	1-3	1-3	1-2	1-2	1-2
23	3-6	3-6	3-5	2-5	1-4	1-4	1-3	1-3	1-2	1-2
24	3-7	3-6	3-6	2-5	1-5	1-4	1-4	1-3	1-3	1-2
25	3-7	3-7	3-6	2-6	1-5	1-5	1-4	1-4	1-3	1-3
26	3-8	3-7	3-7	2-6	1-6	1-5	1-5	1-4	1-4	1-3
27	3-8	3-8	3-7	2-7	1-6	1-6	1-5	1-5	1-4	1-4
28	3-9	3-8	3-8	2-7	1-7	1-6	1-6	1-5	1-5	1-4
29	3-9	3-9	3-8	2-8	1-7	1-7	1-6	1-6	1-5	1-5
30	3-10	3-9	3-9	2-8	1-8	1-7	1-7	1-6	1-6	1-5

Table XIII. Bounds on the minimum distance for codes over $\mathbf{F}_4$ (k:21 to 30)

n/k	21	22	23	24	25	26	27	28	29	30
20	-	-	-	-	-	-	-	-	-	-
21	1-1	-	-	-	-	-	-	-	-	-
22	1-1	1-1	-	-	-	-	-	-	-	-
23	1-2	1-1	1-1	-	-	-	-	-	-	-
24	1-2	1-2	1-1	1-1	-	-	-	-	-	-
25	1-2	1-2	1-2	1-1	1-1	-	-	-	-	-
26	1-3	1-2	1-2	1-2	1-1	1-1	-	-	-	-
27	1-3	1-3	1-2	1-2	1-2	1-1	1-1	-	-	-
28	1-4	1-3	1-3	1-2	1-2	1-2	1-1	1-1	-	-
29	1-4	1-4	1-3	1-3	1-2	1-2	1-2	1-1	1-1	-
30	1-5	1-4	1-4	1-3	1-3	1-2	1-2	1-2	1-1	1-1

CHAPTER VI

FUTURE WORK

The tables in Chapter IV show that the lower bounds and the upper bounds on the minimum distance of nonbinary stabilizer codes are very close to each other. However, the picture is still not complete. Even though completing the tables depends upon how tight the proposed upper bounds are, it also depends upon how good the quantum codes are. More nonbinary quantum stabilizer code families need to be constructed. Various code families that exist in classical theory such as Reed-Muller codes, Hamming codes, etc. need to be used for finding nonbinary quantum stabilizer codes. The current implementation of the exhaustive search program can be optimized further. The various things that might help include

- Exploring the generator matrix properties of the dual code.
- Eliminating vectors for being code vectors using the weight distribution.
- Eliminating some vectors for being generators depending upon the weight enumerators.

These optimizations might directly give some codes with their generators.

Another important aspect of error correction is actually encoding and decoding the data words using the encoding and decoding circuits. Some work has already been done in this area. How to reduce the generator matrix to a standard form by using Gaussian elimination and algorithms for constructing the encoding and decoding circuits for the corresponding quantum error correcting codes is discussed in [16, 17, 4]. These results can be generalized to the nonbinary quantum stabilizer codes. The standard form will change for the F_p-linear codes. It is a vast area that can be explored thoroughly for constructing the encoding and decoding circuits for these

already found codes, as well as the new codes. This will help in practically realizing the error correcting codes. To make these circuits feasible to implement and use, the concept of fault tolerant quantum computing comes into the picture. The encoding and decoding circuits should then be made fault tolerant where fault tolerance means elegant handling of errors in the error correction procedure itself. Studying all these sub areas in this research area will make it well established.

CHAPTER VII

CONCLUSION

This research work explored the relatively unstudied area of nonbinary stabilizer codes. It can be concluded that most of the concepts of binary error correcting codes or particularly binary stabilizer codes can be generalized to nonbinary stabilizer codes. Existing literature in the area of the nonbinary stabilizer codes proved to be useful. While exhaustively searching for codes, it is observed that the typical forms of the generator matrices and weight enumerators helped reducing the search space. In spite of these improvements, the searching approach did not help much because of the limited optimizations implemented and machine power. A new quantum code family called quantum BCH codes is found that helped us in building some MDS codes. However, the distances obtained by these codes are not very impressive in general. It is also proved that new quantum codes can be constructed from the existing ones. These general code constructions gave a huge set of nonbinary quantum stabilizer codes.

The upper bounds on the minimum distance of nonbinary stabilizer codes are established in [14]. The main aim of this work was to improve the lower bounds in such a way that ultimately they go as close as possible to the proposed upper bounds. The lower bounds in the large part of the table *met* the upper bounds and nearly met the upper bounds in other parts. This work not only improved the lower bounds on the minimum distance but also made a significant contribution to the area of constructing nonbinary quantum stabilizer codes. The small or no gaps in the upper bounds and lower bounds table illustrate this fact. The work can be continued by finding more code families, giving more code constructions and providing the encoding circuits for these codes.

REFERENCES

[1] M.Neilson, I. Chuang, *Quantum Computation and Quantum Information*, Cambridge: Cambridge University Press, 2000.

[2] E. Knill,"Nonbinary unitary error bases and quantum codes," submitted to *Las Alamos National Laboratory Archives*, vol. quant-ph/9608048, 1996.

[3] R. Matsumoto, T. Uyematsu,"Constructing quantum error-correcting codes for p^m-state systems from classical error-correcting codes," *IEICE Trans. Fundamentals*, vol. E83-A, pp. 1878-1883, Oct. 2000.

[4] A.R. Calderbank, E.M. Rains, P.W. Shor, N.J.A Sloane,"Quantum error correction via codes over GF(4)," *IEEE. Trans. Inf. Theory*, vol. 44, pp. 1369-1387, Jul. 1998.

[5] A. Ashikhmin, E. Knill, "Nonbinary quantum stabilizer codes," *IEEE Trans. Inf. Theory*, vol. 47, pp. 3065-3072, Nov. 2001.

[6] J. Bierbrauer and Y.Edel, "Quantum twisted codes," *Journal of Combinatorial Designs*, vol. 8, pp. 174-188, 2000.

[7] J. Kim and J. Walker,"Nonbinary quantum error-correcting codes from algebraic curves," submitted to *IEEE Trans. Info. Theory*, Aug. 2003.

[8] A. Klappenecker and M. Roetteler,"Beyond stabilizer codes 1: Nice error bases," *IEEE Trans. Info. Theory*, vol. 48, pp. 2392-2395, Aug. 2002.

[9] M. Grassl, T. Beth, M. Roetteler,"On optimal quantum codes," *Internal Journal on Quantum Computation*, vol. 2, pp. 757-775, 2003.

[10] S. Wolfram, *The Mathematica Book*, Third Edition, Cambridge: Cambridge University, 1996.

[11] A.Thangaraj and S. McLaughlin," Quantum codes from cyclic codes over $GF(4^m)$," *IEEE Trans. Info. Theory*, vol. 47, pp. 1176-1178, 2001.

[12] A. Klappenecker, "A short introduction to stabilizer codes," Jan. 2003, Department of Computer Science, Texas A&M University.

[13] K.Feng, Z.Ma,"A finite Gilbert-Varshamov bound for pure stabilizer quantum codes," submitted to *IEEE Trans. Info. Theory*, 2003.

[14] S. Kumar,"Upper bounds on the minimum distance of nonbinary stabilizer codes," M.S. thesis, Texas A&M University, 2004.

[15] A. Ashikhmin, S. Litsyn, M. Tsfasman, "Asymptotically good quantum codes," *Phys. Rev. A*, vol. 63, pp. 311-315, 2001.

[16] R. Brylinski and G. Chen, *Mathematics of Quantum Computation*, Los Angeles: Chapman & Hall, Feb. 2002.

[17] R. Cleve and D. Gottesman,"Efficient computations of encodings for quantum error correction," *Phys. Rev. A*, vol. 56, pp. 76-82, 1997.

[18] S. Wicker, *Error Control Systems for Digital Communication and Storage*, First Edition, Englewood Cliffs: Prentice Hall, 1994.

APPENDIX A

IMPLEMENTATION DETAILS

In the course of this research, several implementations were done. I played around with several mathematical tools like GAP, MOSEK, Mathematica [10] and so on. I also used the standard programming languages like C, Lex & Yacc, C#, Java and so on. The following programs were implemented at the beginning:

- A Java Applet that implemented Grover's Search algorithm.
- A Quantum Circuit Simulator in Lex & Yacc.
- Small programs in GAP and MOSEK to learn about group theory and linear programming bounds.

The following implementations are done in order to find and verify some results:

- A program that takes upper limit on n and a text file of upper bounds on these codes as input. Using the existing quantum codes and the general code constructions, the program gives a text file as output that has pairs of lower bound-upper bound for all the (n,k) combinations. This code is developed in C#.
- A program that takes a text file of lower-bound-upper bound pairs as input and generates corresponding latex file for the same. It allows us to choose parts of the table. This code is developed in C#.
- A program that exhaustively searches for classical self orthogonal codes. It makes use of the standard form of the generator matrix to reduce the search space. This program is developed in Mathematica.

- A program that searches for self-orthogonal BCH codes to arrive at quantum BCH codes. It calculates the roots of the dual code and the BCH code and checks if the roots of the dual code are contained in that of the original code. If the containment holds, the code is self-orthogonal. The actual code is given below. This program is developed in Mathematica.

- A program similar to the previous one, that searches for self-orthogonal BCH codes to arrive at quantum BCH codes over higher fields. It calculates the roots of the dual code and the BCH code and checks if the roots of the dual code are contained in that of the original code. If it is, the code is self-orthogonal. Code over the higher field is self-orthogonal with respect to the standard inner product. The code is similar to the one included. This program is developed in Mathematica.

- A similar program that searches for self-orthogonal BCH codes to arrive at quantum BCH codes over higher fields. It calculates the roots of the dual code and the BCH code and checks if the roots of the dual code are contained in that of the original code. If they do, the code is self-orthogonal. Code over the lower field is self-orthogonal with respect to the Hermitian product. The code is similar to the one included. This program is developed in Mathematica.

```
(*********************************************************************)
(* Quantum BCH codes *)
(*********************************************************************)
(* Input: n,d,q values *)
(* Output: The zeros of the BCH code over F_q and zeros *)
(* of the alternating Hermitian dual code. *)
(* The program first finds the value of m such that n|q^m - 1. *)
(* It then chooses n roots of unity from F_(q^m) and orders them. *)
(* Then it chooses d-1 consecutive roots from this set as roots *)
(* of dual of C. *)
(* These roots are then conjugated m-1 times and the zero *)
(* set of the dual code *)
(* is formed. All these roots are then raised to the power of -Sqrt[q] *)
(* to form a set of conjugates. If these conjugates lie in the complementary *)
(* set of the previous roots, the corresponding BCH code is self orthogonal. *)
(* The corresponding quantum BCH code is then n, 2 * degree of g(x) - n, d *)
(*********************************************************************)
(* Include finite field package *)
<< AlgebraFiniteFields`
(* This function calculates the Hermitian product of two vectors. It can be *)
(* used to verify that two vectors are Hermitian orthogonal*)
Hermitian[a_, b_, n_, q_]:=(
    p=0;
    For[i=1,i < n+1,i++,
    {
        p=p+a[[i]]*Power[b[[i]],Sqrt[q]];
```

```
    }];
    Print[p];
    )
(* This function gives the roots of code polynomial and its dual *)
inc[n_, d_, q_]:=(
    For[m=1,m < 50,m++,
    {
        If[Mod[Power[q,m]-1,n]==0,Break[]];
    }
    ];

    (* Exit if m exceeds 50 *)
    If [ m==50,
    {
        Print[failed];
        Abort[]
    }];
    Print[m];
    (* Use the sequential labeling in finite fields *)
    SetFieldFormat[GF[Power[q,m]],FormatType→FunctionOfCode[fqm]];
    (* Find the primitive element *)
    For[i=1,i<Power[q,m]+1,i++,
    {
        flag=0;
        For[j=1,j<Power[q,m]-1,j++,
        {
```

```
        t=Power[fqm[i],j];
        If[t=fqm[1],flag=1;];
    }]
    If[flag=0,Break[];];
}];
pe=fqm[i];
(* Find the n^th roots of unity in the field F_q^m *)
myroots={};
For[i=1,i<Power[q,m]+1,i++,
{
    If[Power[fqm[i],n]==fqm[1],
    myroots=Append[myroots,fqm[i]];
    ];
}
];
(* Find the generating element to order the roots *)
For[i=1,i<Length[myroots]+1,i++,
{
    flag=0;
    For[j=1,j<n,j++,
    {
        t=Power[myroots[[i]],j];
        If[t==fqm[1],flag=1;];
    }]
    If[flag=0,Break[];];
}];
```

```
orderedroots={};
orderedroots=Append[orderedroots,myroots[[i]]];
For[j=2,j<n+1,j++,{
    orderedroots=Append[orderedroots,Power[myroots[[i]],j]];
}];
myroots=orderedroots;
    le=Length[myroots];
(* This loop allows the wrap around choice of sequential roots *)
For[i=1,i<d-1,i++,
{     myroots=Append[myroots,myroots[[i]]];
}
];
For[j=1,j<le+1,j++,
{
    exist={};
    For[i=1,i<n+1,i++,exist=Append[exist,0]];
    For[k=j,k<j+d-1,k++,
    {
    b={};
    If[k<le+1,exist[[k]]=1;,exist[[k-le]]=1;];
    t=myroots[[k]];
    (* Calculate m-1 conjugates *)
    For[o=1,o<m,o++,
    {
        s=Power[t,q];
        b=Position[myroots,s];
```

```
            c=b[[1]][[1]];
            exist[[c]]=1;
            t=s;
        }];
        }
];
(* Collect the roots of the dual code *)
zerosdual=Position[exist,1];
smallset={};
For[g=1,g<Length[zerosdual]+1,g++,
{
    x=myroots[[zerosdual[[g]]]];
    smallset=Append[smallset,x];
}
];
conjus={};
For[i=1,i<Length[zerosdual]+1,i++,
{
    y=myroots[[zerosdual[[i]]]];
    (* Raise the roots to the power -Sqrt(q) *)
    t=Power[y,-Sqrt[q]];
    conjus=Append[conjus,t];
}
];
cs={};
For[i=1,i¡Length[conjus]+1,i++,
```

```
cs=Append[cs,conjus[[i]][[1]]];
cc=Complement[orderedroots,cs];
ss={};
For[i=1,i<Length[smallset]+1,i++,
ss=Append[ss,smallset[[i]][[1]]];
countsmall=Length[ss];
countinter=Length[Intersection[cc,ss]];
(* Print the result if the containment holds *)
If[countsmall=countinter,
{
    Print[CODE];
    Print[cc];
    (* Calculate the powers of the primitive element *)
    For[jk=1,jk<Length[cc]+1,jk++,
    {
    For[ik=1,ik<Power[q,m],ik++,
    {
        If[Power[pe,ik]==cc[[jk]],
        If[ik=Power[q,m]-1,Print[0],
        Print[ik];];
    ];
    }
];
}];
Print[ss];
For[jk=1,jk<Length[ss]+1,jk++,
```

```
{
    For[ik=1,ik<Power[q,m],ik++,
    {
    If[Power[pe,ik]=ss[[jk]],
    If[ik==Power[q,m]-1,Print[0],
    Print[ik];];
    ];
    }
    ];
    }];
(* Print the dimensions of the corresponding QBCH code *)
Print[n, __, 2*Length[cc]−n, __, d];
}];
}];
)
```

VITA

Name: Avanti Ketkar

Educational Background: B.E. (Computer Science) - Pune University

Permenant Address: 9-1, B-1, Shubha apts, Pune - 411004, India