Codes, Cryptography, and the McEliece Cryptosystem

Bethany L. Matsick

A Senior Thesis submitted in partial fulfillment
of the requirements for graduation
in the Honors Program
Liberty University
Spring 2020

Acceptance of Senior Honors Thesis

This Senior Honors Thesis is accepted in partial fulfillment of the requirements for graduation from the Honors Program of Liberty University.

____
Ethan Smith, Ph.D.
Thesis Chair

____
Scott Long, Ph.D.
Committee Member

____
David E. Schweitzer, Ph.D.
Assistant Honors Director

____
Date

Abstract

Over the past several decades, technology has continued to develop at an incredible rate, and the importance of properly securing information has increased significantly. While a variety of encryption schemes currently exist for this purpose, a number of them rely on problems, such as integer factorization, that are not resistant to quantum algorithms. With the reality of quantum computers approaching, it is critical that a quantum-resistant method of protecting information is found. After developing the proper background, we evaluate the potential of the McEliece cryptosystem for use in the post-quantum era by examining families of algebraic geometry codes that allow for increased security. Finally, we develop a family of *twisted* Hermitian codes that meets the criteria set forth for security.

Codes, Cryptography, and the McEliece Cryptosystem

Introduction

In 1978, Robert McEliece introduced a public key cryptosystem based on the difficult problem of decoding a random linear code. Due to its large key size, the McEliece cryptosystem has yet to see widespread use. However, since the McEliece cryptosystem does not appear to be susceptible to Shor's algorithm as is the case with the widely used RSA and elliptic curve cryptosystems, it is now being considered as a candidate for post-quantum cryptography. Ultimately, we desire a code with a Schur square that behaves like that of a random linear code, meaning the dimension of its Schur square is equal to that of a random linear code of the same dimension. Because most classical families of codes fall far short of this ideal, we develop a family of *twisted* Hermitian codes with a Schur square dimension comparable to that of random linear code. We show that, when constructed properly, these twisted Hermitian codes not only achieve a high dimensional Schur square but also maintain a reasonable data transfer rate. The twisted construction is a variant of that considered by Peter Beelen, Martin Bossert, Sven Puchinger, and Johan Rosenkilde (2018) and is joint work with Austin Allen, Keller Blackwell, Olivia Fiol, Rutuja Kshirsagar, Gretchen Matthews, and Zoe Nelson.

Mathematical Background

In order to effectively discuss coding theory, we introduce several important concepts from linear algebra, specifically relating to vector spaces. Vector spaces are foundational to linear algebra and have innumerous applications. However, considering specific subsets of vector spaces that conform to the same basic properties is also incredibly valuable. One such subset is the *span* of a given set of vectors.

Definition 1. Let $S = \{v_1, \ldots, v_n\} \subset \mathbb{F}^n$ be a finite set of n-vectors. Then $\text{span}(S)$ is the set of all linear combinations formed from vectors in S:

$$\text{span}\{v_1, \ldots, v_n\} = \Big\{ \sum_{i=1}^{n} c_i v_i \mid c_i \in \mathbb{F} \text{ for } i = 1, 2, \ldots, n \Big\}.$$

Let $W = \text{span}(S)$. We say that S is a generating set or spanning set for W (Curtis, 1984).

Given a set of vectors, it is also important to determine whether or not one vector can be written as a linear combination of the other vectors in the set.

Definition 2. Let $S \subseteq \mathbb{F}^n$. If there is some vector $v \in S$ such that v can be written as a finite linear combination of the other vectors in S, then we say that S is *linearly dependent*. If not, then S is said to be linearly independent.

Determining the minimum number of linearly independent vectors needed to span a given vector space leads to the notion of a basis.

Definition 3. A *basis* for a vector space V over $\mathbb{F}$ is an ordered set of vectors S such that

1. $V = \text{span}(S)$,

2. S is linearly independent.

Linear independence and bases are useful in a variety of contexts and are extremely relevant when examining coding theory in greater depth.

Finally, we provide the definition of a linear transformation as stated by Curtis (1984).

Definition 4. Let V and W be vector spaces over a field $\mathbb{F}$. A *linear transformation* of V into W is a function $T : V \rightarrow W$ which assigns to each vector $v \in V$ a unique vector $w = T(v) \in W$ such that

1. $T(v_1 + v_2) = T(v_1) + T(v_2)$ for all $v_1, v_2 \in V$, and

2. $T(\alpha v) = \alpha T(v)$ for all $v \in V$ and $\alpha \in \mathbb{F}$.

Linear transformations play an important role in the development of codes and will later allow us to define a family of codes known as *linear codes*.

Cryptography

Cryptography, at its core, deals with the sending and receiving of secret messages. Functionally, cryptography has five main purposes:

1. Privacy: Ensuring that only the intended receiver can read the sent message.

2. Authentication: Proving the identity of the sender.

3. Integrity: Ensuring that the received message has not been altered in any way from the original.

4. Non-repudiation: Proving that the sender truly sent the received message.

5. Key exchange: The way in which cryptographic keys are shared between the sender and the receiver (Kessler, 2019).

Throughout history, cryptography has played an important role in both military affairs and diplomatic endeavors (Conrad, 2013). However, as online activity increased during the late twentieth century, the importance of cryptography to the general public increased significantly. In the 1970s, techniques from number theory took cryptographic protocols into uncharted territory by providing a way for two people to communicate secret messages under the assumption that all of their communication is intercepted and read by an adversarial third party (Stein, 2017). Today, this practice is known as public key cryptography.

Though originally developed by GCHQ, modern public key cryptography was first introduced by Whitfield Diffie and Martin Hellman in 1976 (Diffie and Hellman, 1976). With the Diffie-Hellman encryption scheme, two parties cooperatively establish a secret shared key over an insecure channel. However, other systems allow for one-sided determination of the public and private keys. In 1984, Taher ElGamal presented the ElGamal cryptosystem (ElGamal, 1984). With this system, two parties had the ability to engage in secure communication over an unsecured channel without having a shared secret key (Kessler, 2019). In such a scenario, both the algorithm and the encryption key are made public for all users, but only those with a private key ultimately have the ability to carry out the decryption process.

In order to implement a public key cryptosystem, we need two mathematically related keys where knowledge of one key does not allow someone to easily determine the other key. To accomplish this, we implement trapdoor functions. A trapdoor function is one that is simple to compute on every input but difficult to invert given a random output without a secret key. In mathematical terms, if f is a trapdoor function, then there exists some private information t so that, given $f(x)$ and t, it is relatively simple to compute x.

For example, the cryptographic protocol known as RSA depends on the ease of multiplication paired with the computational complexity of factorization. Knowledge of an extremely large number does not easily lead to knowledge of that number’s factors. Yet, the extremely difficult problem of factorization becomes trivial when the receiver has access to the private key. Since its debut in 1977, RSA has seen widespread use and has been incredibly effective. However, the development of quantum computers would render RSA useless. As such, mathematicians and computer scientists now wish to uncover alternative methods based on some other hard problem.

Quantum Computing

Quantum computers do not function in the same way as existing computers since they rely on fundamentally different principles. Quantum computing takes advantage of the the ability of subatomic particles to exist in more than one state at any time. Universal quantum computers leverage this quantum mechanical phenomena to create states that scale exponentially with the number of quantum bits. As such, they are particularly suited to solve large mathematical problems.

For example, consider the problem of factoring very large numbers. Currently, enormous integer factorization problems are believed to be computationally infeasible with a regular computer. However, using Shor's algorithm, a quantum computer could quickly and simply find the desired factors. Shor's algorithm, developed in 1994 by mathematician Peter Shor, is a polynomial-time quantum computer algorithm for integer factorization (Gerjuoy, 2005). In particular, given an integer N, the algorithm finds its prime factors, and it does so in $O((\log N)^2(\log\log N)(\log\log\log N))$ time. Compared to the most efficient classical factoring algorithm, this is nearly exponentially faster.

Today, most public key systems are based on the difficulty of factoring integers or computing discrete logarithms (Stinson & Paterson, 2019). Given that both of these can be solved by Shor's algorithm, it is critical that cryptographic systems are developed that are resistant to quantum algorithms. While experimental quantum computers do not currently have enough processing power to truly break a cryptographic algorithm, cryptographers hope to design new algorithms to prepare for when quantum computing becomes a realistic threat. Consequently, the world of post-quantum cryptography has captured the attention of mathematicians and computer scientists around the world as they seek to create algorithms that are thought to be secure against attacks by quantum computers.

Recently, attention has been focused on cryptographic algorithms that are lattice-based, hash-based, or code-based (Stinson & Paterson, 2019). After developing basic ideas in coding theory, we discuss a cryptographic system based on error-correcting codes.

Coding Theory

Coding theory studies the properties of codes and their various applications. In coding theory the goal is not to hide messages but to ensure that they pass through a noisy channel without errors, i.e., to provide reliable communication. Codes are frequently used for data transmission, data storage, error-correction, and cryptography.

In order to lay a foundation for working with codes in the context of cryptography, we now make several definitions.

Definition 5. A *code* $\mathcal{C}$ of length n is defined to be a subset of $A^n = A \times A \times \ldots \times A$ (n copies) where A is an appropriately chosen alphabet (Walker, 2000).

An alphabet is a finite set of symbols called *letters*. In the given context, let $A = \mathbb{F}_q$ where $\mathbb{F}_q$ is a finite field with q elements, and q is prime or the power of a prime. If q is prime, then $\mathbb{Z}_q$ is a model for $\mathbb{F}_q$. If $q = p^t$ is a prime power, then $\mathbb{F}_q \cong \mathbb{Z}_q[x]/\langle p(x)\rangle$ where $p(x)$ is an irreducible polynomial of degree t.

If $\mathcal{C}$ is a vector subspace of $\mathbb{F}_q^n$, then $\mathcal{C}$ is said to be a *linear code*. The vector space structure of linear codes provides a known framework in which to work and allow specific parameters to be determined with relative ease. Typically, the parameters studied are *length*, *dimension*, and *minimum distance*.

First, we wish to define the minimum distance of a code. To do this, several ideas are necessary. Note that we refer to elements of a code as *codewords*. In order to find the *minimum* distance parameter, we need a method of determining distance between codewords.

Definition 6. For $\mathbf{x} = (x_1, x_2, ..., x_n)$, $\mathbf{y} = (y_1, y_2, ..., y_n) \in A^n$, the *Hamming distance* between $\mathbf{x}$ and $\mathbf{y}$ is

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

For $x \in \mathbb{F}^n$, the *Hamming weight* of $\mathbf{x}$ is

$$wt(\mathbf{x}) = d(\mathbf{x}, 0) = \#\{i : x_i \neq 0\}.$$

Given Definition 6, we now define the minimum distance parameter.

Definition 7. The *minimum distance* of a code $\mathcal{C}$ is

$$d_{min} = \min\{d(\mathbf{x}, \mathbf{y}) : \mathbf{x}, \mathbf{y} \in \mathcal{C} \text{ and } \mathbf{x} \neq \mathbf{y}\}.$$

If the meaning is clear from context, then the subscript may be omitted so that we simply use d to refer to minimum distance. In the case that $\mathcal{C}$ is a linear code, we can compute the minimum distance quite simply.

Theorem 8. Assume $\mathcal{C}$ is a linear code. Then the minimum distance is equal to the minimum weight, i.e.,

$$d_{min} = \min\{wt(\mathbf{x}) : \mathbf{x} \in \mathcal{C} \text{ and } \mathbf{x} \neq (0, 0, ..., 0)\}.$$

With this definition in hand, we now include definitions for the remaining parameters.

Definition 9. Assuming $\mathcal{C}$ is a linear code, we define the *length* of a code $\mathcal{C}$ to be the dimension of the ambient vector space $\mathbb{F}_q^n$. For $\mathcal{C} \subseteq \mathbb{F}_q^n$, the length of $\mathcal{C}$ is n. We define the *dimension*, k, of $\mathcal{C}$ to be the dimension of $\mathcal{C}$ as a vector space over $\mathbb{F}_q$. If $\mathcal{C}$ is a linear code that, as a vector space over $\mathbb{F}_q$, has dimension k and minimum distance d, then we say that $\mathcal{C}$ is an $[n, k, d]$ code.

One of the many benefits of working with linear codes is the ability to assign a basis to the space of codewords. Since $\mathcal{C}$ is a vector space, there exist linearly independent codewords $\mathbf{c}_1, ..., \mathbf{c}_k \in \mathcal{C}$ such that, for every $\mathbf{c} \in \mathcal{C}$, it follows that $\mathbf{c} = a_1\mathbf{c}_1 + ... + a_k\mathbf{c}_k$ for some field elements $a_1, ..., a_k \in \mathbb{F}_q$. Consequently, there exists a matrix G such that every codeword is a linear combination of the columns of G. We can write this matrix as

$$G = \begin{bmatrix} \mathbf{c}_1 & \mathbf{c}_2 & \ldots & \mathbf{c}_k \end{bmatrix},$$

where each codeword is of length n. The choice of G is not unique as any linearly independent set of k codewords may be chosen to form the columns of G. Additionally, if $\mathcal{C}$ is a linear code over $\mathbb{F}_q$, then $d_{min}(\mathcal{C}) = wt(\mathcal{C})$. In other words, the minimum distance of a linear code is the smallest weight of its non-zero codewords. As a result of all these things, the encoding and decoding procedures for a linear code are faster and simpler than those for arbitrary non-linear codes.

To check whether or not a specific word belongs in a code, we use a *parity-check* matrix.

Definition 10. A *parity check* matrix, H, of a linear code C is a generator matrix of the dual code, $C^{\perp}$. Thus, a codeword $\mathbf{c}$ is in C if and only if the matrix-vector product $H\mathbf{c}^{T} = 0$. For any row vector $\mathbf{x}$ of the ambient vector space, the *syndrome* of $\mathbf{x}$ is given by $\mathbf{s} = H\mathbf{x}^{\mathbf{T}}$. The vector $\mathbf{x}$ is a codeword if and only if $\mathbf{s} = 0$.

In order to illustrate several of these ideas, we will consider the Hamming(7,4) code. Not only can we use matrices to encode vectors, but we can also detect errors and correct them using the parity-check matrix and syndrome.

Example 11. Let

$$G = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \in \mathbb{F}_2^{7\times 4},$$

and let H be the parity-check matrix

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \in \mathbb{F}_2^{3\times 7}.$$

Using G, we will encode the nybble (half-byte) 1111 as an element of $\mathbb{F}_2^7$:

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

Using the parity-check matrix and the syndrome, we are able to detect and pinpoint errors in the encoded message. Suppose we introduce an error in position 2 of the encoded nybble. We then have $(1, 0, 1, 1, 1, 1, 1)$. Multiplying this vector by H, we obtain the syndrome:

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

Interpreting the syndrome as a 3-bit integer, we infer that we ought to correct position $0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 2$. Making this correction to the vector $(1, 0, 1, 1, 1, 1, 1)$, we obtain the original codeword $(1, 1, 1, 1, 1, 1, 1)$. Thus, we can detect an error in the encoded nybble and use the syndrome to retrieve the intended codeword.

McEliece Public Key Cryptosystem

As previously discussed, several families of cryptosystems have been proposed to secure communications in a quantum era, some of which are based on codes. In 1978, Robert McEliece introduced the McEliece cryptosystem based on binary Goppa codes. In this system the public key is an obfuscation of the underlying linear code, disguised to appear as a random code. The private key is an efficient decoding algorithm for the underlying code. Overall, the security of the McEliece cryptosystem is derived from the NP-hardness of decoding a random linear code. Though the McEliece cryptosystem remains unbroken to this day (even with quantum algorithms), its reliance on binary Goppa codes results in large key sizes that hinder practical implementation. As a result, many variants of the McEliece cryptosystem have been introduced, with other linear codes substituted within. Additional structure can lead to a reduction in key size but often at the cost of introducing vulnerabilities that allow an attacker to extract identifying characteristics of the underlying code from the public-key matrix. Once the attacker can identify the underlying code, the fundamental assumption securing the McEliece cryptosystem is no longer valid.

We will now look at the structure of the McEliece cryptosystem and at how messages are sent and received. Unlike RSA and elliptic curve cryptography, it does not rely on a computational problem known to be susceptible to Shor's algorithm.

Parameters

Let $n, k, t \in \mathbb{N}$, and define an $[n, k, \geq 2t + 1]$ code $\mathcal{C}$ over $\mathbb{F}_q$.

Key Generation

Let $G \in \mathbb{F}_q^{k \times n}$ be a generator matrix for the code $\mathcal{C}$, $S \in \mathbb{F}_q^{k \times k}$ be an invertible matrix, and $P \in \mathbb{F}_q^{n \times n}$ be a permutation matrix (i.e. has exactly one 1 in each row and each column). Set

$G^{pub} = SGP \in \mathbb{F}_q^{k\times n}$. We then have (G^{pub}, t) as the public key and $(S, D_{\mathcal{C}}, P)$ as the private key where $D_{\mathcal{C}}$ is the appropriate decoding algorithm.

Encryption

To send a private message $m = (m_1, ..., m_k) \in \mathbb{F}_q^k$, the vector m is encrypted as $m \mapsto mG^{pub} + z$ where z is a randomly chosen error vector of weight $wt(z) \leq t$. The encrypted message is then sent to the receiver.

Decryption

When $w = mG^{pub} + z = mSGP + z$ is received, the receiver uses his or her public/private key pair to decrypt the message. Multiplying by P^{-1}, we obtain

$$wP^{-1} = (mG^{pub} + z)P^{-1} = mSG + zP^{-1}.$$

Since wP^{-1} is viewed as a received word in the code, the decoding matrix can be applied. This results in the matrix mS as D_c eliminates the error vector zP^{-1}. Lastly, the receiver can multiply by S^{-1} on the right to obtain the original message m.

As previously discussed, the security of the McEliece cryptosystem is rooted in the problem of decoding a random linear code. Therefore, ideally, we desire a code that is highly structured but keeps the matrix G^{pub} indistinguishable from a random matrix. To accomplish this, the codes by which the generator matrix is determined must be chosen wisely. The concept of a Schur Square is deeply related to this issue.

Definition 12. Let $\mathbf{a}, \mathbf{b} \in \mathbb{F}^n$. The *Schur product* of vectors $\mathbf{a}$ and $\mathbf{b}$ is

$$\mathbf{a} * \mathbf{b} = (a_1b_1, a_2b_2, ..., a_nb_n).$$

Since the the components of **a** and **b** are multiplied component-wise, we observe that $\mathbf{a} * \mathbf{b} \in \mathbb{F}^n$. The Schur product is then used to define the Schur square of a given code.

Definition 13. Let $\mathcal{C}$ be an $[n, k, d]$ code. The *Schur square* of $\mathcal{C}$ is

$$\mathcal{C}^2 = \text{span}\{\mathbf{a} * \mathbf{b} : \mathbf{a}, \mathbf{b} \in \mathcal{C}\}.$$

Interpreting the above definition, we see that the Schur square is simply the set of linear combinations of $\mathbf{a} * \mathbf{b}$ where $\mathbf{a}, \mathbf{b} \in \mathcal{C}$.

Applying this idea to the basis of a code, we have the following: If $\mathcal{C} \subseteq \mathbb{F}^n$ has basis $\{\mathbf{b}_1, \mathbf{b}_2, ..., \mathbf{b}_k\}$, then $\mathbf{b}_i * \mathbf{b}_j \in \mathcal{C}^2$ for all i,j such that $1 \leq i,j \leq k$. Thus, $\mathcal{C}^2 =$ $\text{span}\{\mathbf{b}_i * \mathbf{b}_j : 1 \leq i,j \leq k\}$. Since each element can be multiplied with every other element and with itself, the largest possible dimension of $\mathcal{C}^2$ is $\binom{k+1}{2}$. Given that $\mathcal{C}^2 \subseteq \mathbb{F}^n$, its dimension cannot exceed n. Consequently, dim $\mathcal{C}^2 \leq \min\{n, \binom{k+1}{2}\}$.

Schur Squares and McEliece

We want G^{pub} to behave in the same way as a random code. If $\mathcal{C}$ is an $[n, k, d]$ code chosen at random from the set of all $[n, k, d]$ codes over $\mathbb{F}$ with $\binom{k+1}{2} < n$, then

$$\Pr\left[\dim \mathcal{C}^2 = \binom{k+1}{2}\right] = 1,$$

as determined by Pellikaan and Marquez-Corbella (2017). Therefore, if we desire G^{pub} to act as a random code, then we want to choose families of codes such that dim $\mathcal{C}^2 \approx \binom{k+1}{2}$. While there are many codes from which to choose, we will discuss two families of codes and their corresponding variants.

Reed-Solomon Codes and Variations

First, we will look at Reed-Solomon (RS) codes. Previously, we discussed the existence of a finite field $\mathbb{F}_q$ with $q = p^m$ elements for any prime p and integer $m \geq 1$. Reed-Solomon codes are $[n, k, d]$ linear codes constructed in such a finite field. Since they are constructed in $\mathbb{F}_q$, their length is limited to q. However, they are easily decoded and have a wide variety of uses (Wootters, 2018). We will define standard Reed-Solomon Codes then will discuss a variant proposed in 2017.

Reed-Solomon Codes

Definition 14. Label the $q-1$ nonzero elements of $\mathbb{F}_q$ as $\alpha_1, \ldots, \alpha_n$, and choose $k \in \mathbb{Z}$ so that $1 \leq k \leq n$. Define $\mathcal{L}_k = \{f \in \mathbb{F}_q[x] : \deg(f) \leq k-1\}$. Then the Reed-Solomon code is given by

$$\mathcal{C}_k = \{(f(\alpha_1), \ldots, f(\alpha_n)) : f \in \mathcal{L}_k\}.$$

In order to simplify the expression for $\mathcal{C}_k$, we write $\mathcal{C}_k = ev(\mathcal{L}_k)$, where $ev : \mathcal{L}_k \to \mathbb{F}_q^{q-1}$ is defined by the mapping

$$f \mapsto (f(\alpha_1), ..., f(\alpha_{q-1})).$$

Notice that $\mathcal{C}_k$ is a subset of $\mathbb{F}_q^{q-1} = \mathbb{F}_q \times \cdots \times \mathbb{F}_q$ ($q-1$ copies), so $\mathcal{C}_k$ is a code over the alphabet $\mathbb{F}_q$. Furthermore, since the map $ev : \mathcal{L}_k \to \mathbb{F}_q^{q-1}$ is a linear transformation and $\mathcal{C}_k$ is its image, $\mathcal{C}_k$ is a linear code (Walker, 2000). In order to better understand the basic construction of these codes, consider the following example.

Example 15. Let $q = 5$ and $k = 5$. Then $\mathbb{F}_5 = \mathbb{Z}/5\mathbb{Z}$. Using the definition of $\mathcal{L}_k$, we take each $f \in \mathbb{Z}_5[x]$ such that $deg(f) \leq 4$. As a result, we have $\mathcal{L}_5 = \text{span}\{1, x, x^2, x^3, x^4\}$. Evaluating this basis, the code itself is given by $\mathcal{C}_5 = ev(\mathcal{L}_5)$.

Parameters. Given the definition of a Reed-Solomon code, we wish to determine its basic parameters–length, dimension, and minimum distance. Recall that the length of a code corresponds to the number of coordinates in each codeword. Here, each codeword is generated by evaluating f at the $n = q - 1$ nonzero elements of $\mathbb{F}_q$, so the length must be $n = q - 1$. Similarly, the dimension is at most $\dim\mathcal{L}_{k-1} = k$. However, if $ev(f) = ev(g)$, then $f - g$ has at least $q - 1$ roots, so $f - g$ has degree at least $q - 1$. But $f - g \in \mathcal{L}_k$, which implies that $f = g$. Therefore, $\mathcal{C}_k$ has dimension exactly k (Walker, 2000). Finally, we will find the minimum distance using the minimum weight. Suppose $f \in \mathcal{L}_{k-1}$ and $wt(ev(f)) = d_{min} = d$. Then f has at least $n - d$ zeros, so it has degree at least $n - d$. Since $f \in \mathcal{L}_{k-1}$, this means that $n - d \leq k - 1$, so $d \geq n - k + 1$. It can be shown that Reed-Solomon codes also satisfy the Singleton Bound, i.e., $d \leq n - k + 1$. Therefore, $d = n - k + 1$. Any code with parameters that meet the Singleton Bound is called a *Maximum Distance Separable (MDS) code.*

Schur square. We now consider the Schur square of standard Reed-Solomon codes. By definition, the Schur square of a code $\mathcal{C}$ is given by $\mathcal{C}^2 = \text{span}\{\mathbf{a} * \mathbf{b} : \mathbf{a}, \mathbf{b} \in \mathcal{C}\}$. Applying this to the Reed-Solomon code, we have

$$\mathcal{C}_k^2 = \text{span}\{ev(f) * ev(g) : f, g \in \mathcal{L}_k\} = \text{span}\{(fg(\alpha_1), ..., fg(\alpha_n)) : f, g \in \mathcal{L}_k\}.$$

Recall that $\mathcal{L}_k = \text{span}\{1, x, ..., x^{k-1}\}$. As a result, a basis for the Schur square of a Reed-Solomon code is given by $\mathcal{L}_{2k-1} = \text{span}\{1, x, x^2, \ldots, x^{2k-2}\}$.

Previously, we determined that a code is suitable for use in the McEliece cryptosystem if $\dim\mathcal{C}^2 = \binom{k+1}{2}$. Here, $\dim\mathcal{C}^2 = 2k - 1 \leq \frac{k(k+1)}{2}$. Thus, standard Reed-Solomon codes do not perform well when implemented in McEliece. As k continues to grow large, the discrepancy between $2k-1$ and $\binom{k+1}{2}$ becomes greater and greater. For this reason, we wish to modify the code in some way that improves Schur sqaure dimension.

Single-Twist Reed-Solomon Codes

In order to increase the dimension of the Schur square, mathematicians Beelen, Puchinger, and Nielsen (2017) introduce the concept of *twisted* Reed-Solomon codes. We use the term *twist* to indicate an increase in the degree of the largest basis element.

Definition 16. Let $k,t,h,\eta \in \mathbb{Z}$ with $0 \leq h < k$, $\eta \geq 1$, and $t > 0$. Then

$$\begin{aligned}\mathcal{L}_{k,t,h,\eta} &= \left\{ \sum_{i=0, i\neq h}^{k-1} a_i x^i + a_h(x^h + \eta x^{k-1+t}) : a_i \in \mathbb{F} \right\} \\ &= \text{span}\{1, x, \ldots, x^{k-1}, \ldots, x^h + \eta x^{k-1+t}\},\end{aligned}$$

and the twisted Reed-Solomon code is defined to be $\mathcal{C}_{k,t,h,\eta} = ev(\mathcal{L}_{k,t,h,\eta})$.

Parameters. Twisted Reed-Solomon codes are not necessarily MDS for all $k, t, h,$ and η but can be MDS for certain parameters. Given that the highest degree is $n-(k-1+t)$, we can make a conjecture as to what the minimum distance may be. In order to find the minimum distance, we can determine the minimum weight of the evaluation of f over the subspace $\mathcal{L}_k$. Since the maximum degree is $k-1+t$, this is also the maximum possible number of distinct roots. Therefore, it is possible that $wt(ev(f)) = d \geq n-(k-1+t)$, but this has not yet been proven.

Examples of twisted Reed-Solomon codes. In order to demonstrate how twisting Reed-Solomon codes improves their suitability, consider the following series of examples.

Example 17. We will first compute the Schur square for a standard, untwisted Reed-Solomon code. Let $k = 5$. Since $\mathcal{L}_5 = \text{span}\{1, x, ..., x^4\}$, it follows that

$$\mathcal{C}_5^2 = ev(\text{span}\{1, x, x^2, x^3, x^4, x^5, x^6, x^7, x^8\}).$$

Observe that dim $\mathcal{C}_5^2 = 9$. However, because $k = 5$, we know that for the code to appear random, we need the dimension of the Schur square to be $\binom{k+1}{2} = \frac{(5)(6)}{2} = 15$. Obviously, $9 < 15$, and we fall short of this standard.

Example 18. Next, consider a twisted Reed-Solomon code as defined above. Let $\mathbb{F}_5 = \mathbb{Z}/5\mathbb{Z}$, $k = 5$, and $t = 2$, and consider the case when h is not used (indicated by a subscript of 0) and $\eta = 1$. Then

$$\mathcal{L}_{k,t,h,\eta} = \mathcal{L}_{5,2,0,1} = \text{span}\{1, x, x^2, x^3, x^4, x^{5-1+2}\} = \text{span}\{1, x, x^2, x^3, x^4, x^6\},$$

and $\mathcal{C}_{5,2,0,1} = ev(\mathcal{L}_{5,2,0,1})$. Computing the Schur square, we have

$$\mathcal{L}_{5,2,0,1}^2 = \text{span}\{1, x, x^2, x^3, x^4, x^5, x^6, x^7, x^8, x^9, x^{10}, x^{12}\}.$$

From this, we see that dim $\mathcal{C}_{5,2,0,1}^2 = 12$. Since $h = \eta = 0$, we do not have any elements added together in the basis. However, we still have a jump from x^4 to x^6. Given that $k = 5$, $2k - 1 = 9$, this is still an improvement. Unfortunately, it does not meet the ideal standard of $\binom{6}{2} = 15$.

Example 19. Now consider the twisted Reed-Solomon Code where $\mathbb{F}_5 = \mathbb{Z}/5\mathbb{Z}$, $k = 5$, $t = 2$, $h = 3$, and $\eta = 1$. Then $\mathcal{L}_{k,t,h,\eta} = \mathcal{L}_{5,2,3,1} = \text{span}\{1, x, x^2, x^4, x^3 + x^6\}$, and

$\mathcal{C}_{k,t,h,\eta} = \mathcal{C}_{5,2,3,1} = ev(\mathcal{L}_{5,2,3,1})$.

Computing the Schur square, we find that

$$\mathcal{L}_{5,4,3,1}^2 = \text{span}\{1, x, x^2, x^3, x^4, x^5, x^6, x^8, x^3 + x^6, x^4 + x^7, x^5 + x^8, x^7 + x^{10}, x^6 + 2x^9 + x^{12}\}.$$

Thus, dim $\mathcal{C}^2_{5,2,3,1} = 13$. We see that twisting the code, then, does improve the dimension of the Schur square over merely jumping up in degree. Again, however, it still does not meet ideal standard.

Reed-Solomon subcode and supercode relations. By observation, it is clear that the twisted codes have elements that are distinct from their untwisted counterparts. To understand the relationship between the two, we consider the following example.

Example 20. Let $n = 5, k = 5$, and $t = 2$. Then $\mathbb{F} = \mathbb{Z}_5$. Consider the code obtained by simply adding the values of k and t. Notice

$$\mathcal{L}_7 = \mathcal{L}_{5+2} = \text{span}\{1, x, x^2, x^3, x^4, x^5, x^6\},$$

where $\mathcal{C}_{k+t} = \mathcal{C}_7 = ev(\mathcal{L}_7)$.

Now, add the conditions $h = 0$ and $\eta = 1$, and consider the resulting twisted Reed-Solomon code. Previously, we saw that

$$\mathcal{L}_{k,t,h,\eta} = \mathcal{L}_{5,2,0,1} = \text{span}\{1, x, x^2, x^3, x^4, x^{5-1+2}\} = \text{span}\{1, x, x^2, x^3, x^4, x^6\},$$

with $\mathcal{C}_{k,t,h,\eta} = \mathcal{C}_{5,2,0,1} = ev(\mathcal{L}_{5,2,0,1})$. Therefore, since every element of the basis for $\mathcal{C}_{5,2,0,1}$ is also an element of the basis for $\mathcal{C}_7$, it follows that $\mathcal{C}_{5,2,0,1} \subseteq \mathcal{C}_7$.

Next, recall that

$$\mathcal{L}_{5,2,3,1} = \text{span}\{1, x, x^2, x^4, x^3 + x^6\},$$

and $\mathcal{C}_{k,t,h,\eta} = \mathcal{C}_{5,2,3,1} = ev(\mathcal{C}_{5,2,3,1})$. Again, every element of the basis for $\mathcal{C}_{5,2,3,1}$ is contained in $\mathcal{C}_{5,2,0,1}$. Thus, $\mathcal{C}_{5,2,3,1} \subseteq \mathcal{C}_{5,2,0,1} \subseteq \mathcal{C}_7$.

While it may seem as though we should be able to continue creating subset relations for $\eta \neq 1$, this is unfortunately not the case. Suppose that $\eta = c \in \mathbb{N}$ with $c \neq 0$ and $c \neq 1$. Then $\mathcal{L}_{k,t,h,\eta}$ will have a basis element of the form $x^h + cx^{k-1+t}$. At best, $\mathcal{L}_{k,t,h,1}$ contains $x^h + x^{k-1+t}$. Thus, $\mathcal{C}_{k,t,h,c} \not\subseteq \mathcal{C}_{k,t,h,1}$ for all $c \neq 0, 1$. However, we will always have that

$$\mathcal{C}_{k,t,h,\eta} \subseteq \mathcal{C}_{k,t,0,1} \subseteq \mathcal{C}_{k+t}.$$

Multi-Twist Reed-Solomon Codes

In a subsequent paper, Beelen et al. (2018) generalize further by introducing Reed-Solomon codes with multiple twists.

Definition 21. Let $\ell \in \mathbb{N}$, and let $\boldsymbol{\eta} = (\eta_i), \mathbf{t} = (t_i), \mathbf{h} = (h_i) \in \mathbb{Z}^\ell$. Require $0 < t_i \leq n - k$ and $0 \leq h_i < k$, with t_i's and h_i's distinct. Then

$$\mathcal{L}_{k,\mathbf{t},\mathbf{h},\boldsymbol{\eta}} = \left\{ \sum_{\substack{i=0 \\ i \neq \mathbf{h}}}^{k-1} a_i x^i + \sum_{j=1}^{l} a_{h_j}(x^{h_j} + \eta_j x^{k-1+t_j}) : a_i \in \mathbb{F} \right\}$$

$$= \operatorname{span}\{1, x, \ldots, x^{k-1}, \ldots, x^{h_1} + \eta_1 x^{k-1+t_1}, \ldots, x^{h_l} + \eta_l x^{k-1+t_l}\}.$$

Note that $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_\ell)$, $\mathbf{h} = (h_1, \ldots, h_\ell)$ and $\mathbf{t} = (t_1, \ldots, t_\ell)$.

Now, rather than simply letting $\eta, t, h \in \mathbb{Z}$, we have $\boldsymbol{\eta}, \mathbf{t}, \mathbf{h} \in \mathbb{Z}^\ell$, where ℓ is the desired number of twists in the code. To illustrate this concept, consider the following example.

Example 22. Let $q = 5, k = 5, \mathbf{t} = (2, 4), \mathbf{h} = (1, 3)$, and $\boldsymbol{\eta} = (2, 3)$. Then

$$\mathcal{L}_{5,(2,4),(1,3),(2,3)} = \operatorname{span}\{1, x^2, x^4, x + 2x^6, x^3 + 3x^8\},$$

and $\mathcal{C}_{5,(2,4),(1,3),(2,3)} = \operatorname{ev}(\mathcal{L}_{5,(2,4),(1,3),(2,3)})$.

Parameters. As with standard Reed-Solomon codes, multi-twist Reed-Solomon codes have length n and dimension k. Currently, the minimum distance has not been determined.

Suitability for the McEliece cryptosystem. As the number of twists increases, calculations become significantly more tedious by hand. Yet, Beelen et al. (2018) conclude that the Schur square of multi-twist Reed-Solomon codes is often large. However, the dimension does not necessarily reach the desired $\binom{k+1}{2}$, and so the codes may be susceptible to several different structural attacks. In search of a family of codes that may come closer to a Schur square dimension of $\binom{k+1}{2}$, we now consider an entirely different type of code.

Hermitian Codes

In order to define Hermitian codes, we must first define an equivalence relation of $\mathbb{F}_q^3 \backslash \{(0,0,0)\}$.

Definition 23. Given $(a,b,c),(e,f,g) \in \mathbb{F}_q^3$, we say $(a,b,c) \sim (e,f,g)$ if $(a,b,c) = (\lambda e, \lambda f, \lambda g)$ for some $\lambda \in \mathbb{F}_q \backslash \{0\}$. We write $(a:b:c)$ for the equivalence class of (a,b,c).

Definition 24. The projective plane is given by $\mathbb{P}^2(\mathbb{F}_q) = (\mathbb{F}_q^3 \backslash \{(0,0,0)\}) / \sim$. The affine plane, $\mathbb{F}_q^2$, is in 1-1 correspondence with the classes $(a:b:c) \in \mathbb{P}^2(\mathbb{F}_q)$ with $c \neq 0$ under the mapping

$$(a,b) \mapsto (a:b:1).$$

A class $(a:b:c) \in \mathbb{P}^2(\mathbb{F}_q)$ is said to be a point at infinity if $c = 0$.

In 1988 Stichtenoth introduced one-point Hermitian codes, which are algebraic geometry codes based on Hermitian curves. First, we define the *pole order* of $x^i y^j$ at P_∞ to be $\delta(x^i y^j) = iq + j(q+1)$. For future use, it is important to note that if the functions $f_1, \ldots, f_n$ have distinct pole orders, then they are linearly independent.

Definition 25. *Hermitian codes* are defined to be

$$\mathcal{C}(\alpha P_\infty) = \{(f(P_1),\dots,f(P_n)) : f \in \mathcal{L}(\mathcal{P}_\infty)\}, \text{ where}$$

$$\mathcal{L}(\alpha P_\infty) = \text{span}\{x^i y^j : 0 \le i,\ 0 \le j \le q-1,\ \delta(x^i y^j) \le \alpha\},$$

and $P_1,\dots,P_n$ are the points on the projective plane over $\mathbb{F}_{q^2}$ that satisfy the equation $y^q z + yz^q = x^{q+1}$. Thus, this code can be defined as the image of the evaluation map,

$$\text{ev} : \mathcal{L}(\alpha P_\infty) \to \mathbb{F}_{q^2}^n,$$

$$f \mapsto (f(P_1),\dots,f(P_n)).$$

Notice that $C(\alpha P_\infty)$ is then a code of length $n = q^3$, dimension at least $\alpha + 1 - \frac{q(q-1)}{2}$, with equality achieved when $\alpha \ge q(q-1)+1$. Note that if $\alpha < n$, then Hermitian codes have dimension exactly $k = \dim(\mathcal{L}(\alpha P))$ and have a minimum distance of $d \ge n - \alpha$.

Consider the following examples of Hermitian codes.

Example 26. Let $q = 3$ and $\alpha = 8$, then

$$\mathcal{L}(8P_\infty) = \text{span}\{1, x, y, x^2, xy, y^2\}.$$

We know that y^2 is the largest element to be included in the basis as $y^2 = x^0 y^2$ implies $i = 0$, $j = 2$, and $0(3) + 2(3+1) = 8 \le 8 = \alpha$. However, $x^3 = x^3 y^0$ implies $i = 3, j = 0$, and $3(3) + 0(3+1) = 9 \nleq 8 = \alpha$.

Example 27. Let $q = 3$ and $\alpha = 12$. Following the same pattern as the previous example, we have

$$\mathcal{L}(12P_\infty) = \text{span}\{1, x, y, x^2, xy, y^2, xy^2, x^3, x^2y\}.$$

Schur Square

Although one-point Hermitian codes provide valuable insight into the McEliece cryptosystem, they have many shortcomings. Most notably, the Schur Square dimension of a Hermitian code is significantly lower than the desired dimension of $\binom{k+1}{2}$. More specifically, $\dim C(\alpha P_\infty)^2 \leq 2\alpha + 1 - g$. This is due to the fact that the multiplications of certain elements do not always result in elements that are linearly independent from the already existing elements. To illustrate this point, consider the following example.

Example 28. Let $q = 5$, and consider $\mathcal{L}(12P) = \text{span}\{1, x, y, x^2, xy, y^2\}$. Counting the basis elements, we find $k = 6$. With this value of k, we have $\binom{k+1}{2} = 21$. Computing the Schur square, we have

$$\mathcal{L}(12P)^2 = \text{span}\{1, x, y, x^2, xy, y^2, \ldots, x^2y^2, xy^3, y^4\}.$$

Counting the basis elements, we find $k = 15$. While not terribly low, this dimension is clearly less than the desired Schur square dimension.

Schur square distinguishing, a structural attack introduced by Couvreur, Gaborit, Gauthier-Umaña, Otmani, and Tillich (2014), is effective against one-point Hermitian codes by exploiting the low Schur square dimension of one-point Hermitian codes. To retain many desirable qualities of one-point Hermitian codes while fortifying a Hermitian-based McEliece variant, we introduce a new family of codes called multi-twisted Hermitian codes.

Multi-Twisted Hermitian Codes

Inspired by the ideas of Beelen et al. (2017) we adapt one-point Hermitian codes and define multi-twisted Hermitian codes.

Definition 29. Let $\ell = 2n$ and $\mathbf{t} = ((r_1, s_1), \dots, (r_\ell, s_\ell)) \in \left((\mathbb{Z} \setminus \{0\})^2\right)^n$ be a vector of ℓ distinct non-zero ordered pairs of integers satisfying $uq + v(q+1) < (u + r_k)q + (v + s_k)(q+1)$, $\forall k = 1, \dots, \ell$. Let $\mathbf{h} = ((a_1, b_1), \dots, (r_\ell, s_\ell)) \in \left(\mathbb{Z}^2\right)^n$ be a vector of ℓ distinct ordered pairs of integers satisfying $a_k q + b_k(q+1) \leq uq + v(q+1)$ and $b_k \leq q - 1$. Let $\boldsymbol{\eta} = (\eta_1, \dots, \eta_\ell) \in \{\mathbb{F}_{q^2} \setminus 0\}^\ell$. The set of $(\mathbf{t}, \mathbf{h}, \boldsymbol{\eta})$-*twisted bivariate polynomials* is

$$B_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(\alpha P_\infty) = \left(B(\alpha P_\infty) \setminus \bigcup_{k=1}^{\ell} \{x^{a_k} y^{b_k}\} \right) \cup \bigcup_{k=1}^{\ell} \{x^{a_k} y^{b_k} + \eta_k x^{u+r_k} y^{v+s_k}\}.$$

Let $\mathcal{L}_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(\alpha P_\infty) = \operatorname{span}\{B_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(\alpha P_\infty)\}$.

Notice that

$$\mathcal{L}_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(\alpha P_\infty) = \left\{ \sum_{i=0}^{\lfloor \frac{\alpha}{q} \rfloor} \sum_{j=0}^{\lfloor \frac{\alpha - iq}{q+1} \rfloor} c_{i,j} x^i y^j + \sum_{k=1}^{\ell} \eta_k c_{a_k,b_k} x^{u+r_k} y^{v+s_k} \,\middle|\, c_{i,j} \in \mathbb{F}_{q^2} \right\}.$$

The ℓ-twisted Hermitian code $C_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(\alpha P_\infty)$ is $C_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(\alpha P_\infty) = \operatorname{ev}_{\beta_q}\left(\mathcal{L}_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}\right) \subseteq \mathbb{F}_{q^2}^n$.

To see how twists impact the dimension of the Schur square, we present a final example.

Example 30. As before, let $q = 5$. Begin with the untwisted basis $\mathcal{L}(12P) = \operatorname{span}\{1, x, y, x^2, xy, y^2\}$. Let $\mathbf{h} = ((2,0), (1,1), (0,2))$ and $\mathbf{t} = ((4,-1), (7,0), (10,1))$. We then have

$$\mathcal{L}_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(12P) = \operatorname{span}\{1, x, y, x^2 + x^4 y, xy + x^7 y^2, y^2 + x^{10} y^3\}.$$

Counting the basis elements, we again have $k = 6$. Thus, we desire a Schur square dimension of $\binom{6+1}{2} = 21$. While the calculation is tedious, we indeed find

$$\dim \mathcal{L}_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(12P)^2 = 21.$$

Schur Square Dimension

Recall that a one-point Hermitian code $C(\alpha P_\infty)$ has relatively low Schur square dimension. It can be shown that an ℓ-twisted Hermitian code $C^{n,k}_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(\alpha P_\infty)$ possesses a larger lower bound on Schur square dimension. Specifically,

$$\dim C^{n,k}_{\mathbf{t},\mathbf{h},\boldsymbol{\eta}}(\alpha P_\infty) \geq \left(\frac{k+1}{2}\right) - g,$$

where $g = \frac{q(q-1)}{2}$. Previously, we noted that $\dim C(\alpha P_\infty)^2 \leq 2\alpha + 1 - g$.

Conclusion

Overall, we have defined a new family of codes, multi-twist Hermitian codes, whose construction is based on the one-point Hermitian code. The length and dimension of the new codes is the same as the one-point Hermitian code. However, these new codes attain a Schur square dimension larger than that of the corresponding one-point Hermitian code. When more rigorous requirements are implemented, it is possible for a sub-family of these codes to achieve an extremely large Schur square dimension, close to that of a random linear code. Codes of this sub-family are resistant to Schur square distinguishing when implemented within the McEliece cryptosystem, yet they retain the same key size as when one-point Hermitian codes are used. Thus, twisted Hermitian codes can provide a higher level of security than standard Hermitian codes and have great potential for future use when implemented in the McEliece cryptosystem.

References

Adams, S. S. (2008). *Introduction to algebraic coding theory*. Ithaca, New York: Cornell University.

Beelen, P., Bossert, M., Puchinger, S., & Rosenkilde, J. (2018). Structural properties of twisted Reed-Solomon codes with applications to cryptography. *ArXiv e-prints*. arXiv: 1801.07003

Beelen, P., Puchinger, S., & Nielsen, J. R. (2017). Twisted Reed-Solomon codes. *ArXiv e-prints*. arXiv: 1701.01295

Bernstien, D., Buchmann, J., & Dachman, E. (2009). *Post-quantum cryptography*. New York, New York: Springer-Verlag Berlin Heidelberg.

Bolkema, J., Gluesing-Luerssen, H., Kelley, C., Lauter, K., Malmskog, B., & Rosenthal, J. (2017). Variations of the McEliece cryptosystem. *ArXiv e-prints*. arXiv: 1612.05085

Chen, E. (2016). *An infinitely large napkin*. Cambridge, Massachusetts: Massachusetts Institute of Technology.

Conrad, K. (2013). *Number theory and cryptography*. Storrs, Connecticut: University of Connecticut.

Couvreur, A., Gaborit, P., Gauthier-Umaña, V., Otmani, A., & Tillich, J.-P. (2014). Distinguisher-based attacks on public-key cryptosystems using Reed-Solomon codes. *Designs, Codes, and Cryptography*, *73*(2), 641–666. doi:10.1007/s10623-014-9967-z

Curtis, C. W. (1984). *Linear algebra: An introductory approach*. New York, New York: Springer.

Diffie, W., & Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, *22*(6), 644–654. doi:10.1109/TIT.1976.1055638

Dummit, D. S., & Foote, R. M. (2004). *Abstract algebra*. Hoboken, New Jersey: John Wiley and Sons, Inc.

ElGamal, T. (1984). A public key cryptosystem and a signature scheme based on discrete logarithms. *CRYPTO 84 on Advances in cryptology*.

Gerjuoy, E. (2005). Shor's factoring algorithm and modern cryptography: An illustration of the capabilities inherent in quantum computers. *American Journal of Physics*, *73*(521). doi:10.1119/1.1891170

Huffman, W., & Pless, V. (2003). *Fundamentals of error-correcting codes*. New York, New York: Cambridge University Press.

Kessler, G. C. (2019). *An overview of cryptography*. Daytona Beach, Florida: Embry-Riddle Aeronautical University.

McEliece, R. J. (1978). A public-key cryptosystem based on algebraic coding theory. *Deep Space Network Progress Report*, *44*, 114–116.

Paterson, D. R. S. M. B. (2019). *Cryptography: Theory and practice* (4th ed.). Boca Raton, Florida: Taylor Francis Group, LLC.

Pellikaan, R., & Márquez-Corbella, I. (2017). Error-correcting pairs for a public-key cryptosystem. *Journal of Physics: Conference Series*, *855*(1), 012032. Retrieved from http://stacks.iop.org/1742-6596/855/i=1/a=012032

Rosen, K. (2012). *Discrete mathematics and its applications* (7th ed.). New York, New York: McGraw-Hill Higher Education.

Schneier, B. (1996). *Applied cryptography: Protocols, algorithms, and source code in C* (2nd ed.). Indianapolis, Indiana: John Wiley & Sons, Inc.

Stein, W. (2017). *Elementary number theory: Primes, congruences, and secrets*. New York, New York: Springer.

Stichtenoth, H. (1988). A note on Hermitian codes over GF(q^2). *Institute of Electrical and Electronics Engineers: Transactions on Information Theory*, *34*(5, part 2), 1345–1348. doi:10.1109/18.21267

Walker, J. (2000). *Codes and curves*. Lincoln, Nebraska: The American Mathematical Society.

Wootters, M. (2018). *Algebraic error-correcting codes*. Stanford, California: Stanford University.