Coding Theory: Linear-Error Correcting Codes

Anna Dovzhik

Math 420: Advanced Linear Algebra Spring 2014

Sharing data across channels, such as satellite, television, or compact disc, often comes at the risk of error due to noise. A well-known example is the task of relaying images of planets from space; given the incredible distance that this data must travel, it is be to expected that interference will occur. Since about 1948, coding theory has been utilized to help detect and correct corrupted messages such as these, by introducing redundancy into an encoded message, which provides a means by which to detect errors. Although non-linear codes exist, the focus here will be on algebraic coding, which is efficient and often used in practice.

1 Basic Definitions

The following build up a basic vocabulary of coding theory.

Definition 1.1 If $A = a_1, a_2, \ldots, a_q$, then A is a **code alphabet** of size q and $a_n \in A$ is a code symbol. For our purposes, A will be a finite field $\mathbf{F}_q$.

Definition 1.2 A **q-ary word w** of length n is a vector that has each of its components in the code alphabet.

Definition 1.3 A **q-ary block code** is a set C over an alphabet A, where each element, or **codeword**, is a q-ary word of length n. Note that $|C|$ is the size of C. A code of length n and size M is called an (n, M)-code.

Example 1.1 [3, p.6]
$C = \{00, 01, 10, 11\}$ is a binary (2,4)-code taken over the code alphabet $\mathbf{F}_2 = \{0, 1\}$.

Definition 1.4 For two codewords, $\mathbf{w_1}, \mathbf{w_2}$, of length n over an alphabet A, the **Hamming distance**, denoted $d(\mathbf{w_1}, \mathbf{w_2})$, is the number of places where the two vectors differ.

Example 1.2 If $A = \{0, 1\}$, and we have two codewords $\mathbf{w_1} = 101010, \mathbf{w_2} = 100100$, then their Hamming distance is simply $d(\mathbf{w_1}, \mathbf{w_2}) = 3$.

Definition 1.5 For a code C, the **minimum distance** is denoted $d(C) = \min\{d(\mathbf{w_1 w_2}) : \mathbf{w_1}, \mathbf{w_2} \in C, \mathbf{w_1} \neq \mathbf{w_2}\}$.

Definition 1.6 For a codeword $\mathbf{w}$, the **Hamming weight** of $\mathbf{w}$, or $wt(\mathbf{w})$, is the number of nonzero places in $\mathbf{w}$. That is, $wt(\mathbf{w}) = d(\mathbf{w}, \mathbf{0})$.

Notational note A code of length n, size M, and minimum distance d, where n, M, d are parameters, is called an (n, M, d)-code. The **main coding theory problem** refers to the need of optimizing one of these parameters when the others are given. When n is smaller,

faster transmission can occur. When n is fixed, M is a measure of the efficiency of the code, and d indicates its error-correcting potential. A code can correct more errors when d is larger.

Errors are detected when a vector received is not a codeword. If a codeword $\mathbf{x}$ is sent and $\mathbf{y}$ is received instead, then the error that occurred can be expressed as $e = \mathbf{x} + \mathbf{y}$. Thus, in order to detect a certain error pattern e, $\mathbf{x} + e$ cannot be a codeword.

Example 1.3 [2, p.20]
Suppose we have the binary (3,3)-code $C = \{001, 101, 110\}$. Then the error pattern $e_1 = 010$ can be detected because for all $\mathbf{x} \in C$, $\mathbf{x} + e_1 \notin C$: $001 + 010 = 011, 101 + 010 = 111, 110 + 010 = 100$. However, when $e_2 = 100$, $001 + 100 = 101 \in C$, which is enough to say that C does not detect e_2.

Definition 1.7 A code is **u-error-detecting** if when a codeword incurs between one to u errors, the resulting word is not a codeword.

Theorem 1.1 A code is u-error-detecting if and only if $d(C) \geq u + 1$.

Proof: [5, p.17], [6, p.5]
($\Rightarrow$) Suppose $d(C) \geq u + 1$. Then any error pattern of weight at most u will alter a codeword into a non-codeword, which can be detected.

($\Leftarrow$) Now suppose that a code is u-error-detecting. Then for any error pattern e with $wt(e) \leq u$ and codeword $\mathbf{x}$, $\mathbf{x}+e$ is not a codeword. Now suppose that for $\mathbf{x}$, $\mathbf{y} \in C$, $d(\mathbf{x}, \mathbf{y}) \leq u$. Let $e = \mathbf{x} + \mathbf{y}$. Then $wt(e) \leq u$ and $\mathbf{x} + e = \mathbf{x} + \mathbf{x} + \mathbf{y} = \mathbf{y}$, which is a codeword. Therefore, e cannot be detected. This is a contradiction, which can be fixed by conditioning $d(\mathbf{x}, \mathbf{y}) \geq u + 1$. □

An error pattern e can be corrected if there is a codeword $\mathbf{x}$ where $e + \mathbf{x}$ is closer to $\mathbf{x}$ than any other codeword. This comparison can be done with minimum distance.

Definition 1.8 A code is **v-error-correcting** if v or fewer errors can be corrected by decoding a transmitted word based on minimum distance.

Example 1.4 [3, p.13]
If we have the simple binary (3, 2)-code $C = \{000, 111\}$, then C is single-error-correcting. If 000 is sent and either 100, 010, or 001 is received, then changing one digit would accurately decode the word to 000. However, if 000 is sent and 110 is received, the code would inaccurately be decoded to 111, based on the evaluation of minimum distances between the received vector and the codewords of C. Thus, this code cannot correct two or more erroneous digits.

Theorem 1.2 A code is v-error-correcting if and only if $d(C) \geq 2v + 1$. That is, if C has a distance d, it corrects $\frac{d-1}{2}$ errors.

Proof: [3, p.13]

($\Leftarrow$) Suppose $d(C) \geq 2v+1$. If $\mathbf{x}$ is sent, but $\mathbf{y}$ is received and v or less errors occur, then $d(\mathbf{x}, \mathbf{y}) \leq v$. To prove that the code is v-error-correcting, observe that for another codeword $\mathbf{x}' \in C$,

$$\begin{aligned} d(\mathbf{y}, \mathbf{x}') &\geq d(\mathbf{x}, \mathbf{x}') - d(\mathbf{y}, \mathbf{x}) \\ &\geq 2v + 1 - v \\ &= v + 1 \\ &> d(\mathbf{y}, \mathbf{x}). \end{aligned}$$

Since the distance between the sent codeword $\mathbf{x}$ and the received word $\mathbf{y}$ is less than the distance between $\mathbf{y}$ and any other codeword in C, $\mathbf{y}$ will be accurately corrected to $\mathbf{x}$. Recall that for this case, v or less errors occurred, which means C is v-error-correcting.

($\Rightarrow$) Now suppose that C is v-error-correcting. If now instead $d(C) < 2v + 1$, then there exist codewords $\mathbf{x}, \mathbf{x}'$ such that $d(\mathbf{x}, \mathbf{x}') = d(C) \leq 2v$. Consider the case when $\mathbf{x}$ is sent and v or less errors occur.
Since this code is v-error-correcting, $d(\mathbf{x}, \mathbf{x}') \geq v + 1$. If $\mathbf{x}$ and $\mathbf{x}'$ differ in $d = d(C)$ positions, note that $v + 1 \leq d \leq 2v$. For a received word $\mathbf{y}$ of length n where v digits of $\mathbf{y}$ match with $\mathbf{x}'$, $d - v$ digits match with $\mathbf{x}$, and $n - d$ match with both $\mathbf{x}$ and $\mathbf{x}'$, then $d(\mathbf{y}, \mathbf{x}') = d - v \leq v = d(\mathbf{y}, \mathbf{x})$.

So either $\mathbf{x}'$ is incorrectly chosen as the word closest to $\mathbf{y}$, or the two distances are equal. In either case, this implies that the minimum distance must be redefined. Since it was stated to be $d(C) < 2v + 1$, the only option now is $d(C) \geq 2v + 1$. □

Definition 1.9 A binary number system, or **binary representation** of numbers, is often used in due to the popularity of base 2 in computing. For instance, the number *abcde*, where $a, b, c, d, e \in \{0, 1\}$, would represent $a \times 2^4 + b \times 2^3 + c \times 2^2 + d \times 2^1 + e \times 2^0$.

Example 1.5 [6, p.39] The binary representation of 10101 is:

$$\begin{aligned} 10101 &= 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\ &= 16 + 4 + 1 \\ &= 21 \end{aligned}$$

2 Finite Fields

Although the examples of the previous section were done over the binary field, knowledge of larger finite fields is useful for several important functions of error-correcting codes. Being able to correct more than one error is desirable, which is commonly done by taking powers in a finite field. Since taking powers in a binary field is trivial ($1^n = 1, 0^n = 0$), a larger

finite field would prove more fruitful. Being able to correct error bursts, which is when errors occur consecutively, and finding new codes also are both easier with finite fields [5, p.95].

Definition 2.1 A field is a nonempty set F of elements equipped with the operations addition and multiplication. A field must satisfy eight axioms under the following categories: closure under addition and multiplication, commutativity of addition and multiplication, associativity of addition and multiplication, distributivity of multiplication over addition, additive and multiplicative identities, and additive and multiplicative inverses.

In order to begin discussing finite fields, modular arithmetic must be introduced.

Definition 2.2 Let a, b, and $m > 1$ be integers. Then a is congruent to b modulo m, or $a \equiv b \pmod{m}$, if $m|(a-b)$. This means that the remainder of $\frac{m}{a}$ is b, or $(a-b)$ can be divided by m. As a result, $a = mq + b$. In addition, b is known as the principal remainder and can be denoted as $a \pmod{m}$.

Modular addition and multiplication is defined as follows.

If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then

$$a + c \equiv b + d \pmod{m}$$

$$a \cdot c \equiv b \cdot d \pmod{m}.$$

Note that the set of integers $0, 1, \ldots, m-1$, denoted $\mathbf{Z}_m$, $\mathbf{Z}/m$, or $GF(m)$, forms a commutative ring when using the following definitions of addition and multiplication in $\mathbf{Z}_m$:

$$\text{Addition of } e \text{ and } f\text{: } (e + f \pmod{m})$$
$$\text{Multiplication of } e \text{ and } f\text{: } (ef \pmod{m}).$$

The only axiom $\mathbf{Z}_m$ does not satisfy in order to be classified as a field is the existence of a multiplicative inverse.

Example 2.1 The following tables define binary arithmetic $(+)$ and multiplication $(\cdot)$.

$+$	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

Example 2.2 For $\mathbf{Z}_4$, $3+2 = 3$. Using the definitions of modular arithmetic, $3 \pmod{4} = 1$ and $2 \pmod{4} = 2$, so $(2+1) \pmod{4} = 1$. Using the definition of addition in $\mathbf{Z}_4$, $1 \pmod{4} = 3$.

The following are some more advanced definitions and theorems that are useful for relevant work in finite fields.

Theorem 2.1 $\mathbf{Z}_p$ is a field if and only if p is a prime.

Definition 2.3 Denote the multiplicative identity of a field F as 1. Then **characteristic** of F is the least positive integer p such that 1 added to itself p times is equal to 0. We can prove that this characteristic must be either 0 or a prime number.

Theorem 2.2 A finite field F of characteristic p contains p^n elements for some integer $n \geq 1$.

There are many analogues between an integral ring and a polynomial ring. The following definitions are very similar to those just described.

Definition 2.4 A polynomial $f(x)$ of positive degree is said to be reducible over a field F if there exist two polynomials $g(x)$ and $h(x)$ such that the degree of $g(x)$ is less than that of $f(x)$, the degree of $h(x)$ is also less than that of $f(x)$, and $f(x) = g(x)h(x)$. Irreducible polynomials are the polynomial version of prime numbers.

Definition 2.5 If the coefficients of a polynomial are in a field F, let $f(x) \in F[x]$ be a polynomial of degree $n \geq 1$. Then for any polynomial $g(x) \in F[x]$, there exists a unique pair $s(x), r(x)$ of polynomials, where the degree of $r(x)$ is less than that of $f(x)$, and $g(x) = s(x)f(x) + r(x)$. Very similarly to the integer case, $r(x)$ is known as the principal remainder of $g(x)$ divided by $f(x)$, and can be denoted as $g(x) \,(\mathrm{mod} f(x))$.

Definition 2.6 For two nonzero polynomials $f(x), g(x) \in F[x]$, the **greatest common divisor**, $\gcd(f(x), g(x))$, is the monic polynomial (polynomial with the leading coefficient equal to 1) of the highest degree that divides into both $f(x)$ and $g(x)$.

Euclid's algorithm is a method used to find such highest common factors as mentioned above, and applies to both integers and polynomials. This provides an efficient way of dealing with errors in codes such as BCH and Reed-Solomon.

Definition 2.7 For n polynomials in $\mathbf{F}_q[x]$, denoted $f(x_1), f_2(x), \ldots, f_n(x)$, the **least common multiple**, denoted $\mathrm{lcm}(f(x_1), f_2(x), \ldots, f_n(x))$ is the lowest degree monic polynomial that is a multiple of all the polynomials.

Definition 2.8 A **minimal polynomial** of an element in a finite field $\mathbf{F}_p$ is a nonzero monic polynomial of the least degree possible such that the element is a root.

Definition 2.9 A **primitive element** or **generator** of $\mathbf{F}_p$ is an α such that $\mathbf{F}_q = \{0, \alpha, \alpha^2, \ldots, \alpha^{p-1}\}$ Every finite field has at least one primitive element, and primitive elements are not unique.

3 Linear Codes

A code C is a linear code if every codeword can be expressed as a linear combination of codewords. A linear code of length n over a finite field $\mathbf{F}_q$ is a subspace of the vector space $\mathbf{F}_q^n$, where n denotes the length of each vector. Thus, C satisfies the 10 properties that define a vector space.

Definition 3.1 If C is a linear code in $\mathbf{F}_q^n$, then the **dual code** of C is $C^\perp$.

Definition 3.2 The matrix whose rows are the basis vectors of a linear code is a **generator matrix**.

Example 3.1 [4, p.6]
For the binary $(5, 3)$ code, one example of a generator matrix G is

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

Definition 3.3 A **parity-check matrix** for a linear code C is often used for decoding. It is defined as a generator matrix for the dual code $C^\perp$.

Note that if C is a (n, k)-code, then a generator matrix G must be $k \times n$ and a parity-check matrix H must be an $(n - k) \times n$ matrix.

Definition 3.4 Two q-ary codes are **equivalent** if one can be obtained from the other using a combination of the operations (i) permutation of the positions of the code and (ii) multiplication of the symbols appearing in a fixed position. When applied to a generator matrix, (i) is a column swap, and (ii) is a row operation.

Consider the following example for an alternative way of conceptualizing the parity-check matrix.

Example 3.2 [4, p.7]
If the generator matrix of a code C is given by G of **Example 3.1**, then it is clear that the last two columns of G can be expressed as linear combinations of the first three columns of G. If the columns are denoted $v_1, v_2, \ldots, v_5$, then $v_1 + v_3 = v_4$, and $v_1 + v_2 + v_3 = v_5$. These equations are known as parity-check equations. The corresponding parity-check matrix is formed as

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

Thus, the parity-check matrix can be viewed as a coefficient matrix for the parity-check equations. In this example, the generator matrix for the (n, M)-code had n linearly independent columns, and the remaining columns $M - n$ columns of the generator matrix were

expressed as linear combinations of those independent columns.

Notational note For an (n, k)-code, the standard form of a generator matrix G is $(I_k|A)$ and the standard form of a parity-check matrix H is $(B|I_{n-k})$. The standard form of G can be achieved by row operations. Once G is in reduced-row-echelon form, the columns can be rearranged so that an identity matrix is on the left.

For an (n, k, d)-code C over $\mathbf{F}_q$, there are q^k distinct codewords, each of which can be expressed as a linear combination of the basis vectors of C (or of the rows of the generator matrix). In order to transform a vector $\mathbf{u} \in \mathbf{F}_q^k$ into a codeword of C, you treat $\mathbf{u}$ as a $1 \times k$ matrix and multiply it on the right by G. So to acquire a codeword $\mathbf{v}$ based off of $\mathbf{u}$, $\mathbf{v} = \mathbf{u}G$.

Example 3.3 [3, p.58]
Suppose C is the binary $(5, 3)$-code with

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

and a vector $\mathbf{u} = 101$ is to be encoded. Then the codeword $\mathbf{v}$ can simply be determined through matrix multiplication as follows.

$$\mathbf{v} = (101) \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix} = (10011).$$

When G is in standard form, the vector $\mathbf{v}$ will be $n + k$ digits long, and the first k digits will be the vector that was initially encoded. This is because any matrix multiplied by the identity matrix remains unchanged. The first k digits are known as the **message digits** and the last $n - k$ digits are known as **check digits**. Check digits represent redundancy, which greatly helps protect against noise.

Theorem 3.1 If C is a (n, k)-code over $\mathbf{F}_p$, then $\mathbf{v}$ is a codeword of C if and only if it is orthogonal to every row of the parity-check matrix H, or equivalently, $\mathbf{v}H^T = \mathbf{0}$. This also means that G is a generator matrix for C if and only if the rows of G are linearly independent and $GH^T = O$.

The above theorem is a direct consequence of the definitions of the generator and parity-check matrices. For any $\mathbf{v} \in C$, then $\mathbf{v} \cdot c = 0$ for all $c \in C$, which is the definition of orthogonality. Since the rows of G are in C and the rows of H are in $C^\perp$, all rows of G and H must be orthogonal to each other [3, p.53].

Theorem 3.2 If $G = (I_k|A)$ is the standard form of the generator matrix for an (n, k)-code C, then a parity-check matrix for C is $H = (-A^T|I_{n-k})$.

Overview of Proof: [1, p.71]
H is therefore of size $(n - k) \times n$, and its rows are linearly independent. Due to the

placement of the identity matrices in each standard form, the inner product of any row of H with any row of G sums to 0. Note that if the code is binary, then A does not need to be negated. □

Theorem 3.3 For a linear code C and a parity-check matrix H, C has distance $\geq d$ if and only if any $d-1$ columns of H are linearly independent, and C has distance $\leq d$ if and only if H has d columns that are linear dependent. In other words, when C has distance d, any $d-1$ columns of H are linearly independent and H has d columns that are linearly dependent [3, p.54].

Recall the **main coding theory problem** from **Section 1**. Determining how many words a linear code of length n and distance d can have involves finding different bounds on the size M of the code, given n and d.

Here is one cursory overview of such a bound. Imagine fitting disjoint spheres, each of a fixed radius, in a space A^n, where A is an alphabet of size q with codewords of length n. This is known as the sphere-packing problem, and the following provides one well-known upper bound as a consequence.

Definition 3.5 A q-ary code is a **perfect code** if it attains the Hamming, or sphere-packing bound. For $q > 1$ and $1 \leq d \leq n$, this is defined as having

$$\frac{q^n}{\sum_{i=0}^{[(d-1)/2]} \binom{n}{i}(q-1)^i}$$

codewords.

Theorem 3.4 When q is a prime power, the paramters (n, k, d) of a linear code over $\mathbf{F}_q$ satisfy $k + d \leq n + 1$. This upper bound is known as the **Singleton bound**

Definition 3.6 A (n, k, d) code where $k+d = n+1$ is a **maximum distance separable code** (MDS) code.

Theorem 3.5 If a linear code C over F_q with parameters (n, k, d) is MDS, then $C^\perp$ is MDS, every set of $n-k$ columns of the parity-check matrix are linearly independent, and every set of k columns of the generator matrix is linearly independent.

4 Hamming codes

Our first example of a class of linear codes will be Hamming codes, which are single error-correcting and double error-detecting codes that are easy to encode and decode.

Definition 4.1 A binary code of length $2^r - 1$, $r \geq 2$, with a parity-check matrix H whose columns consist of all nonzero binary codewords of length r, is known as a **binary**

Hamming code of length $2^r - 1$, and is denoted $\text{Ham}(r, 2)$. Note that the columns may be in any order, which means that all binary Hamming codes of a given length n are equivalent (see **Definition 3.4**). For the finite field $\mathbf{F}_q$, the q-ary Hamming code of length $\frac{q^r-1}{q-1} - r$ is denoted as $\text{Ham}(r, q)$.

Hamming codes are perfect codes. For those of the form $\text{Ham}(r, 2)$, with dimension/size k and distance d, some properties are: $k = 2^r - 1 - r$ and $d = 3$ (which, by **Theorem 1.1**, means they are exactly single-error-correcting). For the $\text{Ham}(r, q)$ case, $k = \frac{q^r-1}{q-1}$.

In order to decode with a binary Hamming code, an incoming vector $\mathbf{v}$'s syndrome must be calculated. This is simply $S(\mathbf{v}) = \mathbf{v}H^T$, when v is treated as a row vector. If v is treated as a column vector, then $S(\mathbf{v}) = H\mathbf{v}$. If $S(\mathbf{v}) = \mathbf{0}$, then it can be assumed that v was a legitimate codeword. Otherwise, for some j such that $1 \leq j \leq 2^r - 1$, $S(\mathbf{v})$ is the binary representation of j (see **Definition 1.9**). If there was a single error, this means that $\mathbf{e}_j$ was the error, and the actual word sent was $\mathbf{v} - \mathbf{e}_j$.

Example 4.1 For Ham(3, 2), the parity-check matrix H_1 can be defined as follows:

$$H_1 = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$$

A common decoding algorithm for binary Hamming codes is as follows, using our example of the Ham(3, 2) code. Recall **Definition 1.9**. Since all binary Hamming codes of a given length are equivalent, arrange the columns of H_1 in order of increasing binary numbers. In our case, this means that the first column (001) is the binary representation of 1 and the last column (111) is a binary representation of the number 7.

$$H_2 = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}$$

Suppose that the vector (or 1×7 matrix) $\mathbf{y} = (1101011)$ is received. Then

$$\mathbf{y}H_2^T = (1101011) \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} = (110)$$

Since $S(\mathbf{y}) \neq 0$, we note that (110) is the binary representation of the number 6, which means that the single error that occurred happened in the sixth position of $\mathbf{y}$. Thus, $\mathbf{y}$ is corrected to (1101001), and the original message that was encoded is 1101 [1, p.84].

As a check and review, let us form the generator matrix, G_2, of H_2 and encode $\mathbf{x} = 1101$ with it. To derive G_2, arrange the columns of H_2 into standard form and then use **Theorem 3.2**. Then observe that

$$\mathbf{x}G_2 = (1101) \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} = (1101001)$$

Thus, $\mathbf{x}$ is encoded to (1101001), matching the first part of this example.

5 BCH Codes

For a Hamming code to be able to correct two errors, its minimum distance would have to be increased from 3 to 5, which means either lengthening the codewords or eliminating some of them. BCH codes are a subcode of the Hamming codes where certain codewords in the Hamming code are eliminated [5, Ch. 13]. However, BCH codes are cyclic codes, which means that they can be determined from a generator polynomial.

Definition 5.1 Suppose α is a primitive element of a finite field $\mathbf{F}_q^m$ and $M^i(x)$ is the minimal polynomial of α^i with respect to $\mathbf{F}_q$. Then a **primitive BCH code** over $\mathbf{F}_q$ of length $n = q^m - 1$ and distance d is a q-ary cyclic code that is generated by the polynomial defined as $\text{lcm}(M^a(x), M^{a+1}(x), \ldots, M^{a+d-2}(x))$ for some a.

One way to represent a codeword c is with a binary polynomial $c(x)$, where α is a primitive element and $c(\alpha^k) = 0$. Given a codeword $\mathbf{c}$ of length n, let the digits of $\mathbf{c}$ be denoted $\mathbf{c} = c_{n-1}, \ldots, c_1, c_0$, and define the polynomial $c(x)$ as

$$c(x) = \sum_{i=0}^{n-1} c_i x^i.$$

Example 5.1 [5, Ch. 14] Utilizing the above formula, the BCH code of length 15,

$$00001 \quad 11011 \quad 00101,$$

corresponds to the polynomial $x^{10} + x^9 + x^8 + x^6 + x^5 + x^2 + 1$.

Encoding a message space of polynomials simply involves polynomial multiplication.

Example 5.2 [5, p.219] Encoding a codeword such as $\mathbf{w} = 10111$ involves representing it as the polynomial $b(x) = x^4 + x^2 + x^1 + x^0$, and then multiplying it by the generator polynomial,

$$g(x) = x^{10} + x^9 + x^8 + x^6 + x^5 + x^2 + 1.$$

Since the coefficients of these polynomials are only 0 or 1, polynomial multiplication is the same as multiplication of the binary representations of these polynomials. Thus,

$$g(x)b(x) = 11000 \quad 10011 \quad 01011.$$

This can be equivalently expressed as the polynomial $x^{14} + x^{13} + x^9 + x^6 + x^5 + x^3 + x + 1$.

Since a generator matrix simply has codewords as its rows, the coefficients of a generator polynomial can be used to build the rows of such a generator matrix. Note that just as with the previous example, this will only involve coefficients of values 0 or 1.

Decoding BCH codes involves calculating syndromes (as done with the Hamming codes), finding the error locator polynomial, and finding its roots.

6 Reed-Solomon Codes

Moving away from relatively simplistic binary examples, Reed-Solomon codes are a popular, important subclass of BCH codes that can handle error-bursts (as opposed to random errors that do not happen in blocks).

Definition 6.1 A q-ary **Reed-Solomon code** (RS code) is a q-ary BCH code of length $q-1$ that is generated by $g(x) = (x-\alpha^{a+1})(x-\alpha^{a+2})\dots(x-\alpha^{a+d-1})$, where $a \geq 0$, $2 \leq d \leq q-1$, and α is a primitive element of $\mathbf{F}_q$. Since the length of a binary RS code would be $2-1=1$, this type of code is never considered.

Example 6.1 [3, p.172] For a 7-ary RS code of length 6 and generator polynomial $g(x) = (x-3)(x-3^2)(x-3^3) = 6+x+3x^2+x^3$, a generator matrix G and parity-check matrix H can be formed:

$$G = \begin{pmatrix} 6 & 1 & 3 & 1 & 0 & 0 \\ 0 & 6 & 1 & 3 & 1 & 0 \\ 0 & 0 & 6 & 1 & 3 & 1 \end{pmatrix}$$

$$H = \begin{pmatrix} 1 & 4 & 1 & 1 & 0 & 0 \\ 0 & 1 & 4 & 1 & 1 & 0 \\ 0 & 0 & 1 & 4 & 1 & 1 \end{pmatrix}$$

It can be proved that RS codes are MDS codes.

7 Conclusion

The process of encoding data, transmitting it across a channel, and decoding that data in way that allows for discovering errors is the essence of coding theory. The reason that linear codes are constructed is because vector spaces and finite fields provide a great deal of structure. Without the assurance that every codeword can be expressed as a linear combination of other codewords, tools such as the parity-check matrix would no longer serve a purpose. Hamming, BCH, and RS codes have all been used in practice, and combine wide areas of algebra to best solve different issues that arise in encoding and decoding. Many more codes with different advantages (both linear and non-linear) exist, using similar tools to those already discussed.

References

[1] Hill, Raymond. *A first course in coding theory*. Oxford Oxfordshire: Clarendon Press, 1986. Print.

[2] Hoffman, D. G., D. A. Leonard, C. C. Lindner, K. T. Phelps, C. A. Rodger, and J. R. Wall. *Coding theory: the essentials*. New York etc.: Marcel Dekker, 1991. Print

[3] Ling, San, and Chaoping Xing. *Coding theory: a first course*. Cambridge, UK: Cambridge University Press, 2004. Print.

[4] Pless, Vera. *Introduction to the theory of error-correcting codes*. 3^{rd} ed. New York: Wiley, 1998. Print.

[5] Pretzel, Oliver. *Error-correcting codes and finite fields*. Oxford: Clarendon Press, 2000. Print.

[6] Vermani, L. R.. Elements of algebraic coding theory. London: Chapman & Hall, 1996. Print.