An Introduction to Algebraic Coding Theory

Ramsey Rossmann

April 28, 2019

Abstract

This paper looks at the algebra of simple error-detecting and error-correcting codes and how algebraic tools can be used to create and understand such codes. Without the probabilistic and combinatorial characteristics of these codes taken into consideration, we rely on a few basic measures of codes to assess their usefulness. Sections 1 and 2 frame the issue and present the terminology. Section 3 explores linear codes. Section 4 extends the results from linear codes by viewing them as subgroups of polynomials instead of vector spaces, opening up the tools of ring theory to these codes.

1 Introduction

In many settings, there is a small but positive probability that errors will occur in the transmission of a message from one party to another. To limit the impact of such errors, one can implement a scheme so that, if errors occur, they can be identified and possibly corrected in a reliable manner.

A simple scheme could be to send the information multiple times. For example, suppose Alice wants to send Bob the message $u_0u_1\cdots u_k$. Alice and Bob talk beforehand and agree that Alice will send $u_0u_0u_0u_1u_1u_1\cdots u_ku_ku_k$ so that Bob has a better chance of recognizing if an error occurs. If the message Bob receives is in the correct format (every three bits are the same), Bob can be pretty sure that there are no errors in the message. On the other hand, if he sees, for example, that the first three bits are not the same, he has *detected* some error. This gives him two options: he could ask Alice to resend the message, or he could attempt to *correct* the error. If two of the three bits are the same, then he has a good guess as to what the original bit, and could probably feel good about making that correction himself.

This scheme could be taken a step further by Alice sending 5 or 7 or perhaps dramatically more copies of the original message. By simple probabilistic measures, it is easy to see that such a scheme would detect and correct errors with a high degree of accuracy. However, there is a big price to be paid by sending a code that is many times the length of the original message.

Here we have illustrated a few key features of coding theory that are important to note now and that we will explore further in this paper. First, we are only concerned with errors that occur on a noisy channel. That is, we are not worried about Alice encoding the message incorrectly or Bob decoding it incorrectly; the errors the codes will address occur after encoding and before decoding. We will also assume that these errors are relatively rare and occur randomly and independently. We also can already see a tradeoff between the accuracy of a coding scheme and its length, where high accuracy and short length are good characteristics.

In order to assess how "good" a code is, we will consider the following questions:

- How many errors will it detect?
- How many errors will it correct?
- How accurate are the corrections?
- How efficient is the code? That is, how many bits of code are devoted to a message versus the number devoted to checking?
- How easy is encoding and decoding?

Coding theorists have developed many different codes and families of codes that answer each of these questions differently. While there is no best code, some codes are better than others in different settings. In this paper, we will explore a few of these families of codes and present specific examples of them along the way. We will focus more on the algebra of codes and less on the probabilistic and combinatorial features. We will also make many simplifying assumptions in an effort to offer a clean introduction to the material.

2 Basics

Here is our setup: Alice has a k-digit message (as a binary string) that she will encode into an n-digit codeword (also a binary string). The codeword will be transmitted across a noisy channel where some bits may change (we assume these changes are unlikely, randomly occuring, and independent of each other). The codeword arrives at the receiver as an n-digit binary string and is then decoded into a k-digit binary string for Bob. The goal is for Alice's

k-digit message be the same as Bob's, and the encoding and decoding methods can make that more likely. This gives rise to the following definitions.

2.1 Definitions

Here, we introduce some ideas that will be vital to working with error-detection and correction techniques.

Definition 1. *A* **coding scheme** *consists of an encoding function $E : \mathbb{Z}_2^k \to \mathbb{Z}_2^n$ and a decoding function $D : \mathbb{Z}_2^n \to \mathbb{Z}_2^k$.*

Definition 2. *A* **codeword** *is an element of the image of an encoding function $E : \mathbb{Z}_2^k \to \mathbb{Z}_2^n$.*

Definition 3. *A* **code** *$\mathscr{C}$ is the image of an encoding function.*

Definition 4. *A code is an $(n, k)-$***block code** *if it encodes messages of length k into codewords of length n.*

Using an $(n, k)-$block code, we can take a message of any length, break it into k-length pieces and encode each piece into n-digit codewords. As is standard, we will refer to a digit of a binary string as a *bit*.

Definition 5. *The* **rate** *of an $(n, k)-$ block code is $r = k/n$.*

This is a basic measure of the efficiency of a code.

Definition 6. *The* **distance** *between two codewords $\mathbf{x}$ and $\mathbf{y}$, denoted $d(\mathbf{x}, \mathbf{y})$, is the number of bits in which $\mathbf{x}$ and $\mathbf{y}$ differ. This is also the minimum number of transmission errors required to change $\mathbf{x}$ into $\mathbf{y}$ or vice versa.*

Definition 7. *The* **minimum distance** *of a code $\mathscr{C}$, denoted $d_{min}(\mathscr{C})$, is the minimum of all distances $d(\mathbf{x}, \mathbf{y})$ for all distinct codewords $\mathbf{x}$ and $\mathbf{y}$ in $\mathscr{C}$.*

Definition 8. *The* **weight** *of a codeword $\mathbf{x}$, denoted $w(\mathbf{c})$, is the number of 1s in $\mathbf{x}$.*

Definition 9. *A code is* **t-error-detecting** *if, whenever there are at most t errors and at least 1 error in a codeword, the resulting word is not a codeword. A code $\mathscr{C}$ is* **exactly t-error-detecting** *if it is t-error-detecting and not $(t + 1)$-error-detecting.*

Here, the idea is that if a code is t-error-detecting, a codeword requires more than t errors to be decoded incorrectly.

Definition 10. *A decoding function uses* **maximum-likelihood decoding** *if it decodes a received word $\mathbf{x}$ into a codeword $\mathbf{y}$ such that $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z})$ for all codewords $\mathbf{z} \neq \mathbf{y}$.*

In other words, a maximum-likelihood decoding function picks a codeword that is closest to received word. Clearly, it would be nice if there were only one such codeword.

Definition 11. *A code is* **t-error-correcting** *if maximum-likelihood decoding corrects all errors of size t or less, assuming all ties are considered errors. A tie occurs when, for a received word* $\mathbf{x}$*, there exist distinct codewords* $\mathbf{y}$ *and* $\mathbf{z}$ *each with minimum distance from from* $\mathbf{x}$.

Definition 12. *Given a binary string $x_1x_2\cdots x_n$, the bit x_n is a* **parity check bit** *if* $x_1 + x_2 + \ldots + x_{n-1} \equiv x_n \pmod 2$.

From this definition, it is evident that if x_n is a parity check bit, then $x_1+x_2+\ldots+x_n \equiv 0 \pmod 2$. This is a simple, commonly used error-detection technique.

2.2 Preliminary Results

We now state some characteristics of these definitions.

Theorem 1. *A code $\mathscr{C}$ is exactly t-error-detecting if and only if $d_{min}(\mathscr{C}) = t + 1$.*

Proof. ($\Rightarrow$) Suppose a code $\mathscr{C}$ is exactly t-error-detecting. If $d_{\min}(\mathscr{C}) < t + 1$, then there exist codewords $\mathbf{x}$ and $\mathbf{y}$ such that t or fewer errors are required to change $\mathbf{x}$ to $\mathbf{y}$. That is, $\mathbf{x}$ could be encoded, encounter t errors such that what was sent as $\mathbf{x}$ is decoded as $\mathbf{y}$. Thus, $\mathscr{C}$ would not be t-error detecting, so $d_{\min}(\mathscr{C}) \geq t+1$. If $d_{\min}(\mathscr{C}) > t+1$, then any codeword that experiences $t + 1$ errors is no longer a codeword, so $t + 1$ errors can be detected. This is also a contradiction, so $d_{\min}(\mathscr{C}) = t + 1$.

($\Leftarrow$) Suppose $d_{\min}(\mathscr{C}) = t+1$. Then any codeword that experiences t or fewer errors is not a codeword, so the error can be detected. Let $\mathbf{x}$ and $\mathbf{y}$ be codewords such that $d(\mathbf{x}, \mathbf{y}) = t+1$. Then there are exactly $t + 1$ bits that are differently between $\mathbf{x}$ and $\mathbf{y}$. Thus, there is a set of $t + 1$ errors that can occur such that what is sent as $\mathbf{x}$ is decoded as $\mathbf{y}$. Therefore, $\mathscr{C}$ is not $t + 1$-error-detecting. By definition, $\mathscr{C}$ is exactly t-error-detecting. □

Theorem 2. *A code $\mathscr{C}$ is t-error-correcting if and only if $d_{min}(\mathscr{C}) = 2t + 1$ or $2t + 2$.*

A proof can be found in [6, 131].

3 Linear Codes

Here we will explore a simple, powerful family of single error-correcting codes called linear codes. These codes rely heavily on linear algebra and abstract algebra techniques. First, we will begin with an example.

3.1 Example

Suppose we want to create a block code that uses parity check bits to encode 4-bit messages into 7-bit codes. First, we will consider an m-bit binary string $u_1 u_2 \cdots u_m$ to be equivalent to the vector $\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix} = (u_1, u_2, \ldots, u_m)$ in $\mathbb{Z}_2^m$. Let $E : \mathbb{Z}_2^4 \to \mathbb{Z}_2^7$ be an encoding function with $E(u_1, u_2, u_3, u_4) = (x_1, x_2, x_3, x_4, x_5, x_6, x_7)$, defined as follows with arithmetic modulo 2:

$$\begin{aligned} x_1 &= u_1 & \qquad x_5 &= u_1 + u_2 \\ x_2 &= u_2 & x_6 &= u_2 + u_3 \\ x_3 &= u_3 & x_7 &= u_3 + u_4 \\ x_4 &= u_4 \end{aligned}$$

In other words, bit 5 checks bits 1 and 2, bit 6 checks bits 2 and 3, and bit 7 checks bits 3 and 4. How can we determine the code $\mathscr{C}$ that E creates? Luckily, matrices are helpful here. Let

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \text{ and } \mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

Then, for any $\mathbf{u} \in \mathbb{Z}_2^4$, $\mathbf{Gu} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_1 + u_2 \\ u_2 + u_3 \\ u_3 + u_4 \end{bmatrix} \in \mathbb{Z}_2^7$ is a codeword of $\mathscr{C}$. Thus, we will call $\mathbf{G}$ the generator matrix for $\mathscr{C}$.

For some $\mathbf{x} \in \mathbb{Z}_2^7$, we see that $\mathbf{H}$ has the property that $\mathbf{Hx} = \begin{bmatrix} x_1 + x_2 + x_5 \\ x_2 + x_3 + x_6 \\ x_3 + x_4 + x_7 \end{bmatrix} = \mathbf{0}$ if and only if

$$x_1 + x_2 \equiv x_5 \pmod 2 \tag{1}$$
$$x_2 + x_3 \equiv x_6 \pmod 2 \tag{2}$$
$$x_3 + x_4 \equiv x_7 \pmod 2. \tag{3}$$

By our definition of E, (1), (2), and (3) are simultaneously true if and only if $\mathbf{x}$ is a codeword of $\mathscr{C}$. Thus, $\mathbf{Hx} = \mathbf{0}$ if and only if $\mathbf{x} \in \mathscr{C}$. For this reason, we will refer to the matrix $\mathbf{H}$ as the parity-check matrix of $\mathscr{C}$. The matrix $\mathbf{H}$ also defines the code $\mathscr{C}$ since $\mathscr{C} = \mathcal{N}(\mathbf{H})$.

Here, we have shown an example of a linear $(7,4)-$block code. We will now generalize this kind of code, adapting the definition given by Lidl and Pilz in [4, p.193].

3.2 Generalization

Definition 13. *Let $\mathbf{H}$ be an $(n-k) \times n$ binary matrix of rank $n-k$. The null space of $\mathbf{H}$, $\mathcal{N}(\mathbf{H}) \subset \mathbb{Z}_2^n$, forms a code $\mathscr{C}$ called a* **linear** $(n,k)-$**code** *with* **parity-check matrix H**.

Theorem 3. *Let $\mathscr{C}$ be a linear $(n,k)-$code with parity-check matrix $\mathbf{H}$. Then $\mathscr{C}$ is an additive group.*

Proof. Since $\mathbb{Z}_2^n$ is an additive group, we will show $\mathscr{C}$ is a subgroup of $\mathbb{Z}_2^n$ using a subgroup test. First, since $\mathbf{H}$ is a matrix, $\mathbf{H0} = \mathbf{0}$. Thus $\mathbf{0} \in \mathscr{C}$, so $\mathscr{C}$ is non-empty. Let $\mathbf{x}, \mathbf{y} \in \mathscr{C}$. Then $\mathbf{H}(\mathbf{x} - \mathbf{y}) = \mathbf{Hx} + \mathbf{H}(-\mathbf{y}) = \mathbf{0} - \mathbf{Hy} = -\mathbf{0} = \mathbf{0}$, so $\mathbf{x} - \mathbf{y} \in \mathscr{C}$. Thus, $\mathscr{C}$ is a subgroup of $\mathbb{Z}_2^n$, so $\mathscr{C}$ is a group. □

In the previous example, we can see that $\mathbf{H}$ is of the form $\mathbf{H} = \left[\mathbf{A} \mid \mathbf{I}_{n-k}\right]$. When this is the case, the first k bits are the message bits (also called **information bits**) and the last $n-k$ bits are the **parity-check bits**. The rank of such a matrix $\mathbf{H}$ is at least $n-k$ since it contains $\mathbf{I}_{n-k}$. Since there are $n-k$ rows, the rank is also at most $n-k$, so the rank is $n-k$.

3.3 Coset Decoding

Part of the beauty of linear codes is that encoding only requires matrix multiplication, which is computational straightforward. For decoding, it is a different story. We can easily check to see if a received word $\mathbf{x}$ is a codeword by matrix multiplication with $\mathbf{H}$. If $\mathbf{Hx} = \mathbf{0}$, then we have a codeword in the first k bits of $\mathbf{x}$. However, if $\mathbf{Hx} \neq \mathbf{0}$, we do not have an quick way to find out what error occured. This will be our next task.

We can consider a received vector $\mathbf{x}$ as the sum of two vectors: the transmitted codeword $\mathbf{c}$ and the transmission error $\mathbf{e}$ (any change in a bit corresponds to adding 1 to that bit). Specifically,

$$\mathbf{x} = \mathbf{c} + \mathbf{e}. \tag{4}$$

Since $\mathbf{c}$ is in $\mathscr{C}$, we know $\mathbf{Hc} = \mathbf{0}$. Then,

$$\mathbf{Hx} = \mathbf{H}(\mathbf{c} + \mathbf{e}) = \mathbf{Hc} + \mathbf{He} = \mathbf{0} + \mathbf{He} = \mathbf{He}.$$

Using maximum likelihood decoding, we want to find the vector that is closest to $\mathbf{x}$. In other words, we assume the error is smallest possible, so we want the error vector $\mathbf{e}$ with the least weight.

Since $\mathscr{C}$ is a subgroup of $\mathbb{Z}_2^n$, we can consider its cosets, which are of the form $\mathbf{x} + \mathscr{C}$ for $\mathbf{x} \in \mathbb{Z}_2^n$. We know there is a one-to-one correspondence given by $\mathbf{x} + \mathscr{C} \mapsto \mathbf{Hx}$. Thus, we know $\mathbf{x}$ and $\mathbf{e}$ are in the same coset since $\mathbf{Hx} = \mathbf{He}$. This means that, instead of searching all potential error vectors for the smallest, we need only search the vectors in the coset $\mathbf{x}+\mathscr{C}$. We will let $\mathbf{e}$ be the vector of least weight in this coset. Once we have $\mathbf{e}$, decoding $\mathbf{x} + \mathbf{e}$ gives the most likely original message.

Since $|\mathbb{Z}_2^n| = 2^n$ and $|\mathscr{C}| = 2^k$, Lagrange's Theorem says there are 2^{n-k} distinct cosets of $\mathscr{C}$ in $\mathbb{Z}_2^n$, each with size 2^k. This means that if $n - k$ (the number of parity-check bits) is large, searching for the correct coset becomes very slow if not impossible. This means we will need to find a scheme with a more efficient decoding method.

3.4 Assessment

As we have seen, linear codes are simple to implement on a small scale but run into trouble for as they get bigger. Here, we will take a moment to note some other characteristics of linear codes to get a sense of how they compare with others we will encounter.

The linearity of a linear code comes from the following theorem:

Theorem 4. *Let $\mathscr{C}$ be a linear code with parity-check matrix $\mathbf{H}$ and codewords $\mathbf{x}$ and $\mathbf{y}$. Let $c \in \mathbb{Z}_2$.*[1] *Then $\mathbf{x} + \mathbf{y}$ and $c\mathbf{x}$ are codewords as well.*

Proof. Since $\mathbf{x}$ and $\mathbf{y}$ are codewords, we know $\mathbf{Hx} = \mathbf{0}$ and $\mathbf{Hy} = \mathbf{0}$. Then

$$\mathbf{H}(\mathbf{x} + \mathbf{y}) = \mathbf{Hx} + \mathbf{Hy} = \mathbf{0} + \mathbf{0} = \mathbf{0}.$$

Thus, $\mathbf{x} + \mathbf{y} \in \mathscr{C}$. Similarly,

$$\mathbf{H}(c\mathbf{x}) = c\mathbf{Hx} = c\mathbf{0} = \mathbf{0},$$

so $c\mathbf{x} \in \mathscr{C}$. □

[1]While this result holds for any field, we are only working in $\mathbb{Z}_2$ for this paper, so we will ignore the more general claim.

This factor is useful in determining the minimum distance of a linear code.

Theorem 5. *The minimum distance d_{min} of a linear code $\mathscr{C}$ is the minimum weight of all nonzero codewords.*

Proof. For some codeword nonzero $\mathbf{c_0} \in \mathscr{C}$, we have the following

$$\begin{aligned} \min_{\mathbf{c} \in \mathscr{C} \setminus \{\mathbf{c}_0\}} d(\mathbf{c}, \mathbf{c}_0) &= \min_{\mathbf{c} \in \mathscr{C} \setminus \{\mathbf{c}_0\}} d(\mathbf{c} - \mathbf{c}_0, 0) \\ &= \min_{\mathbf{c} \in \mathscr{C} \setminus \{\mathbf{0}\}} d(\mathbf{c}, 0) && (\text{since } \mathbf{c} - \mathbf{c}_0 \in \mathscr{C}) \\ &= \min_{\mathbf{c} \in \mathscr{C} \setminus \{\mathbf{0}\}} \mathrm{w}(\mathbf{c}). \end{aligned}$$

Then,

$$d_{\min} = \min_{\mathbf{c}_0 \in \mathscr{C} \setminus \{\mathbf{0}\}} \left(\min_{\mathbf{c} \in \mathscr{C} \setminus \{\mathbf{c}_0\}} d(\mathbf{c}, \mathbf{c}_0) \right) = \min_{\mathbf{c}_0 \in \mathscr{C} \setminus \{\mathbf{0}\}} \left(\min_{\mathbf{c} \in \mathscr{C} \setminus \{\mathbf{0}\}} \mathrm{w}(\mathbf{c}) \right) = \min_{\mathbf{c} \in \mathscr{C} \setminus \{\mathbf{0}\}} \mathrm{w}(\mathbf{c}).$$

Thus, the minimum distance of $\mathscr{C}$ is the minimum weight of all nonzero codewords. □

As a result, linear codes can constructed to detect more than double erros and correct more than single errors. However, for linear $(n, k)-$codes, removing low-weight words decreases the number of codewords, meaning the code cannot pass as many messages. This pressure could be alleviated by increasing n, but this comes at a significant decoding cost, as we have seen.

4 Polynomial Codes

Now we will look at a specific kind of linear code with much more structure than we have seen by viewing codes as subsets of polynomial rings. Specifically, we will let $\mathbf{u} = u_0 u_1 \cdots u_{n-1}$ correspond to the polynomial $f(x) = u_0 + u_1 x + \cdots + u_{n-1} x^{n-1} \in R_n = \mathbb{Z}_2[x]/\langle x^n - 1 \rangle$. We will also be interested in the following definition.

Definition 14. *A code $\mathscr{C}$ is a* **cyclic code** *if, for all $\mathbf{u} = u_0 u_1 \ldots u_{n-1} \in \mathscr{C}$, $u_{n-1} u_0 u_1 \ldots u_{n-2} \in \mathscr{C}$.*

Now we can connect these two new ideas.

Theorem 6. *A linear code $\mathscr{C}$ in $\mathbb{Z}_2^n$ is cyclic if and only if it is an ideal in $R_n = \mathbb{Z}[x]/\langle x^n - 1 \rangle$.*[2]

[2]This theorem and proof are from Judson, [3, p. 353].

Proof. ($\Rightarrow$) Let $\mathscr{C}$ be a linear cyclic code, and suppose $f(t)$ is in $\mathscr{C}$. Then $tf(t)$ must also be in $\mathscr{C}$ since $\mathscr{C}$ is cyclic. Then, $t^k f(t) \in \mathscr{C}$ for all $k \in \mathbb{Z}^+$. Since $\mathscr{C}$ is linear, any linear combination of the codewords $f(t), tf(t), t^2 f(t), \ldots, t^{n-1} f(t)$ is also a codeword by Theorem 4. Since any polynomial $p(t)$ in $\mathbb{Z}[x]/\langle x^n - 1\rangle$ is of the form $p(t) = a_0 + a_1 t + \cdots + a_{n-1} t^{n-1}$, $p(t)f(t)$ is in $\mathscr{C}$ for every polynomial $p(t)$ in R_n. Thus, $\mathscr{C}$ is an ideal.
($\Leftarrow$) Let $\mathscr{C}$ be an ideal in $\mathbb{Z}[x]/\langle x^n - 1\rangle$. Suppose that $f(t) = a_0 + a_1 t + a_2 t^2 + \cdots + a_{n-1} t^{n-1}$ is a codeword in $\mathscr{C}$. Since $\mathscr{C}$ is an ideal, $tf(t)$ is a codeword in $\mathscr{C}$ as well. Thus, $\mathscr{C}$ is cyclic. $\square$

4.1 Matrices

Since we are very familiar with ideals of a ring, we now have many more tools to describe these codes.[3] For instance, we can now define generator and parity-check matrices for cyclic codes. Since R_n is a principal ideal domain, we know any cyclic code $\mathscr{C}$ has a unique monic polynomial generator $g(t)$. This generator can be used to create the generator matrix $\mathbf{G}$ for $\mathscr{C}$. We also know that there exists a polynomial $h(x)$ such that $g(x)h(x) = x^n - 1$. The polynomial $h(x)$ can be used create the parity-check matrix $\mathbf{H}$ for $\mathscr{C}$. For $g(x) = g_0 + g_1 x + \cdots + g_{n-k} x^{n-k}$ and $h(x) = h_0 + h_1 x + \cdots + h_k x^k$, we have:

$$G = \begin{bmatrix} g_0 & 0 & \cdots & 0 \\ g_1 & g_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ g_{n-k} & g_{n-k-1} & \cdots & g_0 \\ 0 & g_{n-k} & \cdots & g_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g_{n-k} \end{bmatrix}, \text{ and } H_{(n-k)\times n} = \begin{bmatrix} 0 & \cdots & 0 & 0 & h_k & \cdots & h_0 \\ 0 & \cdots & 0 & h_k & \cdots & h_0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ h_k & \cdots & h_0 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

Though we will leave it unproven, these matrices serve the same purposes that the generator and parity-check matrices for linear codes from earlier did.

5 Conclusion

While there is much more to be said about coding theory, it is clear that algebraic techniques are important to developing simple coding schemes. Linear codes offer an intuitive approach and connection to linear algebra that is extended by the theory of cosets and ideals to create more structured, powerful codes. Tools from probability and combinatorics offer more insight into the benefits of cyclic codes and other algebraic extensions and encourage exploring different topics in algebra to satisfy other constraints or curiosities.

[3]This section is heavily based on Judson's exposition of polynomial codes in [3, 354-5]

References

[1] Richard W. Hamming. *Coding and Information Theory.* Prentice-Hall, Inc., 1980.

[2] Raymond Hill. *A First Course in Coding Theory.* Clarendon Press, 1999.

[3] Thomas W. Judson. *Abstract Algebra: Theory and Applications.* Orthogonal Publishing L3C, 2018.

[4] Rudolf Lidl and Gunter Pilz. *Applied Abstract Algebra.* Springer, 2008.

[5] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes.* Elsevier Science Publishers B.V., 1988.

[6] Steven Roman. *Coding and Information Theory.* Springer-Verlag, 1992.