Classical Channel Coding, I
Finite Fields, Linear Block Code Basics, Perfect Codes

John MacLaren Walsh, Ph.D.

ECET 602, Winter Quarter, 2013

1 References

- *Error Control Coding*, 2nd Ed. Shu Lin and Daniel J. Costello, Jr. Pearson Prentice Hall, 2004.
- *Error Control Systems for Digital Communication and Storage*, Prentice Hall, S. B. Wicker, 1995.

2 Galois Fields

A *field* is a set $\mathcal{F}$ together with two binary operations $\square$ "addition" and $\circ$ "multiplication" such that

- $\square$ and $\circ$ are commutative ($a\square b = b\square a$ and $a \circ b = b \circ a$) and associative ($a\square(b\square c) = (a\square b)\square c$ and $a \circ (b \circ c) = (a \circ b) \circ c$)
- $\circ$ distributes over $\square$: $c \circ (a\square b) = c \circ a \square c \circ b$
- There exists a (unique) element $0 \in \mathcal{F}$ such that for all $a \in \mathcal{F}$ $a\square 0 = a$.
- For every $a \in \mathcal{F}$ there exists a (unique) additive inverse $b \in \mathcal{F}$ such that $a\square b = 0$
- There exists a (unique) element $1 \in \mathcal{F}$ such that for all $a \in \mathcal{F} \setminus \{0\}$, $1 \circ a = a$.
- For every $a \in \mathcal{F} \setminus \{0\}$ there exists a (unique) multiplicative inverse $b \in \mathcal{F} \setminus \{0\}$ such that $a \circ b = 1$.

This can be alternately stated as $\mathcal{F}$ and $\square$ yield a commutative group, and $\mathcal{F} \setminus \{0\}$ and $\circ$ form a commutative group, and $\circ$distributes over $\square$.

You are already familiar with the field of complex numbers and real numbers with normal multiplication and addition. These fields have an uncountably infinite number of elements in them. There are also fields for which $|\mathcal{F}|$ is finite, and in this case, $|\mathcal{F}|$ is known as the *order* of the field. These are called Galois fields, after Evariste Galois, who in 1830 published a related paper. The only finite fields have $|\mathcal{F}| = p^m$, where p is a prime number (a positive integer which can not be factored as the product of any two smaller integers), and m is a positive integer. The finite field of order q is typically denoted by $GF(q)$.

2.1 Galois Fields of Prime Order

The prime order Galois Fields $GF(p)$ can be represented as the integers $\{0, 1, \ldots, p-1\}$ under modular p ordinary arithmetic. Here $a\square b = a + b \mod p$ and $a \circ b = ab \mod p$. (Recall that $c \mod p$ is the remainder when dividing the positive integer c by the positive integer p.) We will work extensively with $GF(2)$.

2.2 Galois Fields of Prime Power Order

The prime-power order Galois Fields $GF(p^m)$ can be represented as the set of $m-1$th or lower degree polynomials $a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \cdots + a_1 x + a_0$ with coefficients $a_{m-1}, a_{m-2}, \ldots, a_1, a_0$ in $GF(p)$. Addition is carried out as usual with polynomials, with the coefficients being added with modular p

addition, i.e. $a(x)\square b(x) = ((a_{m-1} + b_{m-1}) \mod p)x^{m-1} + \cdots + ((a_0 + b_0) \mod p)$. Multiplication can be carried out as normal polynomial multiplication modulo a *primitive polynomial* $r(x)$.

$$a(x) \circ b(x) = c(x) \mod r(x), \quad c(x) = \sum_{n=0}^{2(m-1)} \underbrace{\left(\sum_{k=0}^{n} a_k b_{n-k} \right)}_{\text{arithmetic integer mod } p} x^n \tag{1}$$

The $\mod r(x)$ means remainder after polynomial long division by $r(x)$.

The primitive polynomial $r(x)$ is a polynomial for which the smallest positive integer n for which $r(x)$ divides the polynomial $x^n - 1$ (i.e. with zero remainder) is $n = p^m - 1$, and must be irreducible over $GF(p)[x]$ (i.e. it can not be factorable into two lower degree polynomials with coefficients in $GF(p)$). Equivalently, primitive polynomials are those irreducible polynomials that also have a *primitive* element α of $GF(p^m)$ among their roots, and a primitive element has the property that its successive powers enumerate all of the non-zero elements of the field $GF(p^m)$, i.e. $\mathcal{F} = \{0, 1, \alpha, \alpha^2, \alpha^3, \ldots, \alpha^{p^m-2}\}$, with $\alpha^{p^m-1} = 1$.

This property of a primitive element motivates another representation of $GF(p^m)$ in terms of 0 and the powers of a primitive element α. Multiplication in this representation is simpler than in the polynomial representation, as $\alpha^i \circ \alpha^j = \alpha^{i+j \mod p^m - 1}$.

As addition is simple in the polynomial representation, and multiplication is easy in the primitive power representation, it is desirable to have a map between the polynomial representation and the primitive power representation, so that the operations can be carried out easily. This can be done as follows. Polynomial long divide x^i by a primitive polynomial $r(x)$ for each $i \in \{0, \ldots, p^m - 2\}$ to get the representation

$$x^i = q_i(x)r(x) + a_i(x), \quad a_i(x) = a_{i,m-1}x^{m-1} + a_{i,m-2}x^{m-1} + \cdots + a_{i,0} \tag{2}$$

if we substitute a primitive element α that is a root of $r(x)$ into this expression, we observe that

$$\alpha^i = q_i(\alpha)r(\alpha) + a_i(\alpha) = a_i(\alpha) = a_{i,m-1}\alpha^{m-1} + a_{i,m-2}\alpha^{m-1} + \cdots + a_{i,0} \tag{3}$$

In this manner for each power i in α^i, $i \in \{0, 1, \ldots, p^m - 1\}$, we can determine an associated polynomial $a_i(x)$.

Hence α^i and α^j can be added by adding the corresponding polynomials $a_i(x)$ and $a_j(x)$ (by adding the integer coefficients mod p as usual), and selecting the power k in α^k having the resulting polynomial $a_k(x) = a_i(x) + a_j(x)$ as its representation.

3 Vector Spaces over Finite Fields

We can create a vector space $\mathcal{V}$ over a finite field $GF(q)$ the usual way $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$ with $x_i \in GF(q), i \in \{1, \ldots, n\}$, and defining normal vector addition

$$\mathbf{u} + \mathbf{v} = (u_1 + v_1, \ldots, u_n + v_n)^T \tag{4}$$

and scalar multiplication.

$$a\mathbf{u} = (au_1, au_2, \ldots, au_n)^T \tag{5}$$

A vector subspace $\mathcal{S}$ of a vector space is a collection of vectors that is closed under addition and scalar multiplication, that is if $\mathbf{u}, \mathbf{v} \in \mathcal{S}$, then $\mathbf{u} + \mathbf{v} \in \mathcal{S}$ and $a\mathbf{u} \in \mathcal{S}$ for any $a \in GF(q)$.

If the set of all linear combinations $a_1\mathbf{v}_1 + \cdots + a_k\mathbf{v}_k$ of a collection of vectors $\mathcal{G} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ gives the vector subspace $\mathcal{S}$, then $\mathcal{G}$ is known as a *spanning set* for $\mathcal{S}$. A spanning set of minimal cardinality is called a *basis*, and contains vectors which are linearly independent. The cardinality of the basis is known as the dimension of the vector space $\dim(\mathcal{S})$.

The dual space $\mathcal{S}^\perp$ to a vector subspace $\mathcal{S}$ of a vector space $\mathcal{V}$ is the set of vectors

$$\mathcal{S}^\perp = \left\{ \mathbf{v} \in \mathcal{V} \,\middle|\, \mathbf{u}^T\mathbf{v} = 0 \quad \forall \mathbf{u} \in \mathbf{S} \right\} \tag{6}$$

The following relationship holds between the dimensions of $\mathcal{S}, \mathcal{S}^\perp, \mathcal{V}$:

$$\dim(\mathcal{S}) + \dim(\mathcal{S}^\perp) = \dim(\mathcal{V}) = n \tag{7}$$

Note: it is tempting to think of $\mathbf{u}^T\mathbf{v} = \sum_{i=1}^{n} u_i v_i$ as an inner product (indeed, both of the texts above have the misfortune to call this an inner product). The reader should be forewarned, however, that the required property $\langle \mathbf{x}, \mathbf{x} \rangle \geq 0$ with equality if and only if $\mathbf{x} = 0$, of a usual inner product *does not hold for* $\mathbf{u}^T\mathbf{v}$ *in Galois Fields*! For instance, there are self-orthogonal vectors! Indeed, any vector $\mathbf{x} \in GF(2)^n$ having an even number of 1s has the property $\mathbf{x}^T\mathbf{x} = 0$. Just think of the implications of this fact for ideas like Gram-Schmidt orthogonalization! Be forewarned, then, that while many linear algebra ideas from the reals translate to general finite field linear algebra, not all of them do.

4 Introduction to Linear Block Codes

A linear code has the property that it forms a vector subspace of $GF(q)^n$, that is, any linear combination of codewords is also itself a codeword.

Linear block codes operate by taking a message vector $\mathbf{x} \in GF(q)^k$ and mapping it to a length n codeword $\mathbf{c} \in GF(q)^n$ through a linear operation:

$$\mathbf{c} = \mathbf{x}\mathbf{G} \tag{8}$$

were we have introduced the generator matrix $\mathbf{G}$. For a systematic code, the generator matrix will take the form $\mathbf{G} = [\mathbf{P}|\mathbf{I}_k]$ (i.e., systematic means that the original message $\mathbf{x}$ appears in the encoded codeword). From this generator matrix, we can form the parity check matrix $\mathbf{H} = \left[\mathbf{I}_{n-k}| - \mathbf{P}^T\right]$, having the property that $\mathbf{c}\mathbf{H}^T = \mathbf{0}$ for any codeword $\mathbf{c}$.

Algebraic codes (the subject of our first two lectures on coding theory) tend to focus on channel models 1) that return symbols in the same alphabet as the input and, 2) that are memoryless with the property that, for a given symbol, each error pattern is equally likely. This class of channel models allow the probability of a given error pattern $\mathbf{e}$ to which the transmitted codeword $\mathbf{c}$ is added to get the channel output $\mathbf{r} = \mathbf{c} + \mathbf{e}$, to be written as a decreasing function of the number of non-zero positions in $\mathbf{e}$, or *Hamming weight* $\mathrm{wt}(\mathbf{e})$.

In this case, the goal of a decoder is to select the error pattern of minimum Hamming weight (minimum number of non-zero positions) consistent with the received codeword. This error pattern can be found using the syndrome decoder

$$\mathbf{e} = \arg \min_{\mathbf{e}|\mathbf{e}\mathbf{H}^T = \mathbf{r}\mathbf{H}^T} w(\mathbf{e}) \tag{9}$$

5 Error Detection & Error Correction

There are two criteria of interest in channel coding, the ability to *detect* errors, i.e. to determine when the codeword at the output of the channel differed from the code word at the input the channel, and the ability to *correct* errors, i.e. when an error has occurred, determine the original codeword.

The only errors which can not be detected, that is, the only *undetectable* error patterns $\mathbf{e}$, are those for which the received word $\mathbf{r}$ is itself a codeword $\mathbf{c}' \neq \mathbf{c}$, so that $\mathbf{e} = \mathbf{c}' - \mathbf{c}$. As the likelihood of an error pattern is assumed a function of its Hamming weight, this means that the most likely undetectable error patterns are between those codewords which are closest in Hamming distance:

$$d_{min} = \min_{\mathbf{c},\mathbf{c}' \in \mathcal{C}, \mathbf{c}' \neq \mathbf{c}} \mathrm{wt}(\mathbf{c}' - \mathbf{c}) \tag{10}$$

This quantity is called the *minimum distance* of the code. *Any error pattern which has Hamming weight less than the minimum distance of the code can be detected.* For a linear code, the difference between any two codewords is itself a codeword (because the any linear combination of two codewords is itself a codeword). Hence the minimum distance of a linear code is given by the minimum non-zero Hamming weight codeword

$$d_{min} = \min_{\mathbf{c} \in \mathcal{C}, \mathbf{c} \neq \mathbf{0}} \mathrm{wt}(\mathbf{c}) \tag{11}$$

An error pattern must be detectable in order to be correctable, and there are many error patterns which can be detected which may not be corrected. The minimum distance of a code also determines the maximum number t of errors that the code is guaranteed to be able to correct. Put another way, the minimum distance also determines the number t such that any error pattern of Hamming weight t

or less can be corrected. Indeed, a code with minimum distance d_{min} can correct all error patterns with $t = \lfloor (d_{min} - 1)/2 \rfloor$ or fewer errors. This is because any two codewords are separated by at least a Hamming weight of d_{min}, and hence a Hamming ball

$$B_t(\mathbf{c}) = \{\mathbf{x} | \mathrm{wt}(\mathbf{x} - \mathbf{c}) \leq t\} \tag{12}$$

of radius t can be placed at each codeword $\mathbf{c} \in \mathcal{C}$, and these balls will not intersect. As the decoder selects the codeword which is nearest to $\mathbf{r}$ in Hamming weight, all of the received words within a given radius t Hamming ball will be decoded to the codeword at its center. Hence any error pattern having Hamming weight less than or equal to t can be corrected.

When the union of these disjoint radius t Hamming balls centered at codewords gives all possible received vectors $\mathbf{r}$, we say that the code is *perfect*. We will discuss perfect codes in the next section.

When a code is *not* perfect (we will see momentarily that this is usually the case), then there can be *some* error patterns with Hamming weight greater than t which can be corrected. However, not *all* error patterns of this higher weight can be corrected. While these patterns have a lower probability (by assumption) than smaller error patterns, they can contribute significantly to the average error behavior because they have a large weight. This shows that the minimum distance is not the only metric of interest when designing a code. Add to this the channel model under consideration when considering the Hamming weight of an error pattern as a sufficient statistic for the associated channel probability, and one observes how limited the scope of coding theories that focus exclusively on the minimum distance as the sole metric is. We will introduce other channel models and codes in the "modern coding" theory lectures.

6 The Perfect Codes

A perfect code has the property that $GF(q)^N$ can be represented as the disjoint union of equal size Hamming balls centered at the codewords. Such codes obey the Hamming bound

$$MV_q(n,t) \leq q^n \tag{13}$$

with equality, where $V_q(n,t) = \sum_{j=0}^{t} \binom{n}{j} (q-1)^j$ is the size of the n dimensional Hamming ball of radius t. This bound simply observes that if a Hamming sphere of size t is to be placed around every codeword, and there are M such balls, then the total volume (number of points) in these balls can not exceed the total volume (number of points) of the space.

Only a handful of perfect codes exist. All of them fall into one of the following five categories (t-error correcting code with $M = q^k$ codewords of length n)

- **No code:** $\{q, n, k = n, t = 0\}$
- **Repetition Code:** $\{q = 2, n \text{ odd}, k = 1, t = (n-1)/2\}$
- **Hamming Codes:** $\{q = 2, n = 2^m - 1, k = 2^m - m - 1, t = 1\}$. The linear codes with these parameters have parity check matrices whose columns are all non-zero length m binary sequences, and are called Hamming codes. (also there are several nonlinear codes with these parameters)
- **Binary Golay:** $\{q = 2, n = 23, k = 12, t = 3\}$
- **Ternary Golay:** $\{q = 3, n = 11, k = 6, t = 2\}$