École Polytechnique Fédérale de Lausanne

School of Computer and Communication Sciences

Handout 23 Principles of Digital Communications

Notes on Coding November 24, 2020

These notes complement the lectures on classical coding theory, by considering non-binary codes and giving a more extensive treatment of bounds on minimum distance. The section regarding polynomials over finite fields is included only for the sake of completeness. Nonetheless, I imagine some of you may find the material of interest.

Codes, Minimum distance

In this note, we will be concerned with codes for a channel whose input alphabet is of size q; to fix ideas let the input alphabet be $X = \{0, 1, \dots, q-1\}$. Codes for the case $q = 2$ are called 'binary codes,' on which we mostly restricted our attention to in the class. Codes for the alphabet X above are called q-ary codes.

A block code of block length n with M codewords is a set of M sequences of length n from the alphabet X. For example the code $\mathcal{C} = \{000, 011, 110, 101\}$ is a code of block length 3 with 4 codewords on the alphabet $\{0, 1\}$.

Given two sequences $\mathbf{x} = (x_1, \dots, x_n) \in X^n$ and $\mathbf{y} = (y_1, \dots, y_n) \in X^n$, the Hamming distance $d_H(\mathbf{x}, \mathbf{y})$ is the number of positions in which they differ: $d_H(\mathbf{x}, \mathbf{y}) = |\{i : x_i \neq y_i\}|$. For example, $d_H(010, 111) = 2$. Note that (i) $d_H(\mathbf{x}, \mathbf{y}) = 0$ if and only if $\mathbf{x} = \mathbf{y}$, (ii) $d_H(\mathbf{x}, \mathbf{y}) = d_H(\mathbf{y}, \mathbf{x})$, and (iii) $d_H(\mathbf{x}, \mathbf{z}) \le d_H(\mathbf{x}, \mathbf{y}) + d_H(\mathbf{y}, \mathbf{z})$, so d_H is a metric. The property (iii) is called the triangle inequality.

Given a code $\mathcal{C}$, the minimum distance $d_{\min}(\mathcal{C})$ of the code is defined by

$$d_{\min}(\mathcal{C}) = \min_{\mathbf{x}\in\mathcal{C}, \mathbf{y}\in\mathcal{C}, \mathbf{x}\neq\mathbf{y}} d_H(\mathbf{x}, \mathbf{y}),$$

the smallest of the Hamming distances between pairs of codewords. For the code $\mathcal{C}$ in the example above, $d_{\min}(\mathcal{C}) = 2$.

Consider a binary symmetric channel with crossover probability $\epsilon < 1/2$. For this channel the probability $P(\mathbf{y}|\mathbf{x})$ of receiving a sequence $\mathbf{y} \in \{0, 1\}^n$ when a sequence $\mathbf{x} \in \{0, 1\}^n$ is transmitted is given by $\epsilon^{d_H(\mathbf{x},\mathbf{y})}(1-\epsilon)^{n-d_H(\mathbf{x},\mathbf{y})}$, and so $P(\mathbf{y}|\mathbf{x})$ is determined completely by $d_H(\mathbf{x}, \mathbf{y})$ and furthermore it is a decreasing function of it. Thus, when a particular $\mathbf{y}$ is received, the maximum likelihood decoder will search for the codeword that is closest to $\mathbf{y}$ (with distance measured by the Hamming distance). Also, given a code with only two codewords, $\mathbf{x}_1$ and $\mathbf{x}_2$, the probability of error of the maximum likelihood decoding rule only depends on $d_H(\mathbf{x}_1, \mathbf{x}_2)$. Therefore, the minimum distance of a code emerges as a good figure of merit for a code: larger the minimum distance, in general (but not necessarily), better the code will perform on a binary symmetric channel.

For input alphabets with more than 2 symbols, one can also consider a more general kind of symmetric channel: a channel with transition probabilities $P(i|i) = (1-\epsilon)$ and $P(j|i) = \epsilon/(q-1)$ for $i \neq j$. One can again see that for a code with only two codewords, $d_{\min}(\mathcal{C})$ determines the code performance, and that $d_{\min}(\mathcal{C})$ is a good figure of merit for arbitrary codes when the code is meant to be used on a symmetric channel. The corresponding figure of merit for codes meant to be used on a Gaussian channel is the minimum Euclidean distance. We will not consider the Gaussian channel in this note.

If a code $\mathcal{C}$ has odd minimum distance $d = 2e + 1$, then it is possible for a decoder to correct up to e channel errors. This is a simple consequence of the triangle inequality:

if no more than e errors are made, the received (corrupted) sequence is closer to the transmitted codeword than any other codeword. Similarly, if a code has even minimum distance $d = 2e+2$, then it is possible for a decoder to correct all errors up to e, and detect (and maybe even correct) $e+1$ errors: again, if no more than e errors are made the received sequence is closer to the transmitted codeword than any other; if on the other hand $e+1$ errors are made, then we can only be sure that no other codeword is closer.

Since $d_{\min}(\mathcal{C})$ emerges as a good indicator of code performance it is important to find bounds for it. In various problems in the homework sets we already did this for the case of the binary codes. Here, we will show them in this more general setting:

Theorem 1 (Sphere packing bound). *Any q-ary block code of block length n and minimum distance d has at most*

$$M = \left\lfloor q^n \Bigg/ \sum_{i=0}^{\lfloor (d-1)/2 \rfloor} \binom{n}{i} (q-1)^i \right\rfloor$$

codewords.

Proof. This follows from the same line of argument as for the binary case discussed in class: Since the code has minimum distance d, it can correct $e = \lfloor (d-1)/2 \rfloor$ errors. Thus, the "spheres" of radius e around each codeword are disjoint. Since each such sphere contains $\sum_{i=0}^{e} \binom{n}{i} (q-1)^i$ sequences, and the total number of sequences is q^n the bound follows. □

Note that the theorem above is of a negative kind, it disallows the existence of codes of high rates and large minimum distance. The next theorem is in the opposite direction; it says that for a given minimum distance there are codes of at least a certain rate.

Theorem 2 (Gilbert and Varshamov). *Given integers n and d, there exists a code with block length n, minimum distance $d_{\min} \geq d$, with at least*

$$M = \left\lceil q^n \Bigg/ \sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i \right\rceil$$

codewords.

Proof. Again, the argument follows the reasoning for the binary case done in class: from the list of all q^n available sequences start picking codewords. When we pick a sequence as a codeword, delete from the available list all sequences at distance less than d from it. The procedure is guaranteed to yield a code with minimum distance at least d, and since for each codeword picked we delete at most $\sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i$ sequences from the available list, we can pick at least M codewords before running out of sequences to pick from. □

Note that in the two theorems above the sums go up to different limits, that there is essentially a factor of two difference in d between the two theorems.

The next two theorems are again of a negative nature. They say that codes with large minimum distance do not exist.

Theorem 3 (Singleton). *A q-ary block code of block length n with $M > q^k$ codewords has minimum distance at most $n-k$.*

Proof. Suppose $x_1, \ldots, x_M$ are the codewords. For each $i = 1, \ldots, M$, let y_i be the first k letters of x_i, so that x_i is y_i followed by $n-k$ letters. If $M > q^k$, since there are only q^k distinct q-ary sequence of length k, not all the y's can be distinct, and there must be i and j such that $y_i = y_j$. But then, the corresponding x's can disagree only on their last $n-k$ letters, and thus have Hamming distance at most $n-k$. □

Theorem 4 (Plotkin). *The minimum distance d of a q-ary block code of block length n with M codewords satisfies*

$$d \leq n\Big(1 - \frac{1}{q}\Big)\frac{M}{M-1}.$$

Furthermore, if $k = \lfloor \log_q M \rfloor$, for any $0 \leq j \leq k$,

$$d \leq (n + j - k)\Big(1 - \frac{1}{q}\Big)\frac{q^j}{q^j - 1}.$$

Proof. Since $d \leq d_H(\mathbf{x}, \mathbf{y})$ for any $\mathbf{x} \in \mathcal{C}$, $\mathbf{y} \in \mathcal{C}$, $\mathbf{x} \neq \mathbf{y}$, we have

$$M(M-1)d \leq \sum_{\mathbf{x} \in C} \sum_{\substack{\mathbf{y} \in C \\ \mathbf{y} \neq \mathbf{x}}} d_H(\mathbf{x}, \mathbf{y}) = \sum_{\mathbf{x} \in C} \sum_{\mathbf{y} \in C} d_H(\mathbf{x}, \mathbf{y}).$$

Now, $d_H(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} d_H(x_i, y_i)$ where x_i and y_i are the ith digit of $\mathbf{x}$ and $\mathbf{y}$ and $d_H(x_i, y_i)$ is equal to 1 if x_i and y_i are different and zero otherwise. Thus,

$$M(M-1)d \leq \sum_{i=1}^{n} \sum_{\mathbf{x} \in C} \sum_{\mathbf{y} \in C} d_H(x_i, y_i).$$

Suppose out of the M codewords $M_{i,m}$ of them have the letter m as their ith symbol. Let $S_{i,m}$ denote these codewords. Note that

$$\begin{aligned}
\sum_{\mathbf{x} \in C} \sum_{\mathbf{y} \in C} d(x_i, y_i) &= \sum_{m=0}^{q-1} \sum_{\mathbf{x} \in S_{i,m}} \sum_{\mathbf{y} \in C} d(x_i, y_i) \\
&= \sum_{m=0}^{q-1} \sum_{\mathbf{x} \in S_{i,m}} \sum_{\mathbf{y} \in S_{i,m}^c} 1 \\
&= \sum_{m=0}^{q-1} M_{i,m}(M - M_{i,m}).
\end{aligned}$$

Note that

$$\max_{\substack{z_0, \ldots, z_{q-1} \\ \sum_i z_i = M}} \sum_{m=0}^{q-1} z_m(M - z_m) = M^2(1 - 1/q),$$

and thus $\sum_{\mathbf{x} \in C} \sum_{\mathbf{y} \in C} d(x_i, y_i) \leq M^2(1 - 1/q)$. Thus,

$$M(M-1)d_{\min} \leq nM^2(1 - 1/q),$$

yielding the first part of the theorem.

For the second part, consider the first q^k codewords. Classify these codewords grouping them according to their initial $k - j$ digits. Since there are only q^{k-j} groups, one of the groups will contain more than $q^k/q^{k-j} = q^j$ codewords. These codewords, by construction agree on their initial $k-j$ digits, so the minimum distance within this group will not change if we remove these letters. Applying the first bound to this subset of codewords we obtain the bound in the second part. □

Linear Codes

Consider a code of block length n and rate R. This code has 2^{nR} codewords, and each codeword is a sequence of n channel symbols. Just to store the codewords at the encoder then requires space for $n2^{nR}$ letters, which is extremely large even for modest values of n. The task of the decoder is even harder: given an output sequence $\mathbf{y}$, it has to decide which of these 2^{nR} codewords is likely to have been input to the channel.

To get around these difficulties, we need to introduce some structure to our codes so that the codewords can be described not just as a set of 2^{nR} sequences, but in a way that allows a simpler representation.

One technique that permits simpler representations involves the idea of algebraic fields. The key idea is to make sure that $\mathcal{C}$ is not just an arbitrary subset of X^n, but is a vector space. Recall the definition of vector spaces in $\mathbb{R}^n$: A subset S of $\mathbb{R}^n$ is a vector space if for any two elements $\mathbf{x} = (x_1, \dots, x_n)$ and $\mathbf{y} = (y_1, \dots, y_n)$ of S and for any two real numbers a and b, $a\mathbf{x} + b\mathbf{y}$ is also an element of S. We thus see that the notion of vector space is entangled with the notion of addition and multiplication of real numbers. If we can somehow introduce these notions of addition and multiplication to the set X of channel inputs, then we can talk about codes that are vector spaces. The proper algebraic structure to impose on X is that of a field.

To talk about fields, one first needs to talk about groups: That a set X together with an operation $+$ forms a group means the following: given x and y from X, one can form $x + y$ which is also in X and this 'addition' operation has the following properties:

Associativity: $(x + y) + z = x + (y + z)$.

Identity: there exists an element in X, denoted by 0 and called the *identity element*, such that $x + 0 = 0 + x = x$ for all $x \in X$.

Inverse: for all $x \in X$ there exists $(-x) \in X$ such that $x + (-x) = (-x) + x = 0$.

A group is said to be Abelian (or commutative) if in addition to the above properties we also have that $x + y = y + x$ for all x and y.

A finite field is a finite set X of at least two elements, and two operations called addition $(+)$ and multiplication $(\cdot)$, with the the following properties:

The set X and the operation $+$ form an Abelian group.

The set of nonzero elements (i.e., X without the identity element 0 of the operation $+$) and $\cdot$ form an Abelian group.

The distributive law is satisfied: for all x, y, z in X

$$(x + y) \cdot z = x \cdot z + y \cdot z.$$

We will denote the additive identity element by 0 and the multiplicative identity element by 1. The additive inverse of a will be denoted by $(-a)$ and the multiplicative inverse of a by a^{-1}. Note that the set of real numbers with the usual addition and multiplication satisfies the above properties. Also note that the set $\{0, 1\}$ with modulo-2 addition and multiplication also satisfies the above properties. From the properties above, one can easily derive some others:

$x \cdot 0 = 0 \cdot x = 0$ for all $x \in X$,

$x \cdot y \neq 0$ for $x \neq 0$ and $y \neq 0$,

$-(x \cdot y) = (-x) \cdot y = x \cdot (-y)$ for all x and y,

$x \neq 0$ and $x \cdot y = x \cdot y'$ imply $y = y'$.

Suppose now that the input alphabet X is equipped with operations $+$ and $\cdot$ so that $(X, +, \cdot)$ form a field. In the class we considered the case $X = \{0, 1\}$ with modulo-2 addition and multiplication.

Given a field $(X, +, \cdot)$ we can consider X^n as a vector space by defining vector addition and scalar multiplication: for vectors $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$ of X^n define their sum $\mathbf{x} + \mathbf{y}$ as $\mathbf{z} = (z_1, \ldots, z_n)$ where $z_i = x_i + y_i$; for the vector $\mathbf{x} = (x_1, \ldots, x_n) \in X^n$ and $a \in X$ define $a \cdot \mathbf{x}$ as the vector $\mathbf{z} = (z_1, \ldots, z_n)$ with $z_i = a \cdot x_i$.

A block code $\mathcal{C}$ of block length n is said to be a linear code, if it is a vector space: for any two elements $\mathbf{x}$ and $\mathbf{y}$ of $\mathcal{C}$ and any two elements a and b of X, $a \cdot \mathbf{x} + b \cdot \mathbf{y}$ is also in $\mathcal{C}$.

In a homework problem we proved that a binary linear block code contains 2^k codewords for some integer k. This result generalizes to q-ary linear block codes:

Theorem 5. *If $X = \{0, \ldots, q-1\}$ is a field, and $\mathcal{C}$ is a linear q-ary block code, Then $\mathcal{C}$ has q^k elements for some integer k.*

One way to generate a linear block code of length n with q^k elements is to use a linear transformation: The encoder stores an $n \times k$ matrix G whose elements are in X. (Note the difference in storage requirement: nk versus nq^k.) It accepts messages of the form $\mathbf{u} = (u_1, \ldots, u_k)$ where each u_i belongs to X. Given a message $\mathbf{u}$ to convey, the encoder computes $\mathbf{x} = G\mathbf{u}$, i.e.,

$$x_i = \sum_{j=1}^{k} G_{ij} \cdot u_j, \quad i = 1, \ldots, n,$$

and transmits the sequence $\mathbf{x} = (x_1, \ldots, x_n)$. In the above equation, all sums and products are done using the field operations. One can prove that for any linear code $\mathcal{C}$ there exists such a matrix G.

For a sequence $\mathbf{x} = (x_1, \ldots, x_n)$ we define the Hamming weight of this sequence $w_H(\mathbf{x})$ as the number of non-zero x_i's: $w_H(\mathbf{x}) = |\{i : x_i \neq 0\}|$. If $(X, +)$ form a group (which is the case when $(X, +, \cdot)$ is a field) the Hamming distance between $\mathbf{x}$ and $\mathbf{y}$ can be represented as $d_H(\mathbf{x}, \mathbf{y}) = w_H(\mathbf{x} - \mathbf{y})$. This is the case since $x_i = y_i$ if and only if $x_i - y_i$ is zero.

Theorem 6. *For a linear code $\mathcal{C}$,*

$$d_{\min}(\mathcal{C}) = \min_{\mathbf{x} \in \mathcal{C}, \mathbf{x} \neq (0, \ldots, 0)} w_H(\mathbf{x}).$$

The right hand side is called the minimum weight of the code.

Proof. By linearity $\mathbf{0} = (0, \ldots, 0)$ is a codeword. Since for any sequence $\mathbf{x}$, $w_H(\mathbf{x}) = d_H(\mathbf{x}, \mathbf{0})$, the weight of any non-zero codeword is an upper bound on $d_{\min}(\mathcal{C})$ and thus the left hand side is less than or equal to the right hand side. On the other hand, for the two codewords $\mathbf{x}$, $\mathbf{y}$, at distance $d_{\min}(\mathcal{C})$ from each other, $\mathbf{z} = \mathbf{x} - \mathbf{y}$ is also a codeword (by linearity), $\mathbf{z} \neq \mathbf{0}$ and, $d_{\min}(\mathcal{C}) = d_H(\mathbf{x}, \mathbf{y}) = w_H(\mathbf{z})$. Thus, the left hand side is greater than or equal to the right hand side. Combining the two statements we obtain the result. □

It is clear that the negative results of the previous section about codes in general (Theorems 1, 3 and 4) apply also to linear codes.

It is somewhat of a surprise that the positive theorem (Theorem 2) is also valid for linear codes.

Theorem 7. *There exists a linear code with blocklength n, with $M \geq q^k$ codewords and minimum distance $d_{\min} \geq d$ provided that*

$$\sum_{i=0}^{d-2} \binom{n-1}{i} (q-1)^i < q^{n-k}.$$

Proof. Let $r = n - k$. We will find an $r \times n$ matrix H with the property that any set of $d-1$ of its columns is linearly independent. We will define the code as the set of $\mathbf{x} \in X^n$ such that $H\mathbf{x} = 0$. It is easy to check that such a code is linear. Observe further: (i) an $r \times n$ matrix H is of rank at most r, the space of solutions have is of dimension at least $n - r$ and thus the code has at least q^{n-r} codewords. (ii) The code has minimum distance d or more: for any nonzero $\mathbf{x}$ of weight $d-1$ or less, $H\mathbf{x}$ is a linear combination of $d-1$ or fewer columns of H and thus cannot equal to zero by the linear independence of such a set of columns. Thus any nonzero codeword has weight d or more, so $d_{\min} \geq d$.

It remains to construct a matrix with the lineary independence propery. We will do this by choosing the columns one at a time. Suppose i columns have already been chosen such that any subset of $d-1$ or fewer columns are linearly independent. The number of distinct linear combinations of these i columns taken $d-2$ or fewer at a time is

$$\binom{i}{1}(q-1) + \binom{i}{2}(q-1)^2 + \cdots + \binom{i}{d-2}(q-1)^{d-2}$$

(the binomial coefficient $\binom{i}{k}$ gives the number of ways to choose k columns and $(q-1)^k$ gives the number of ways to choose the non-zero coefficients of these k columns in the combination.) If this number if strictly less than $q^r - 1$, then we can find a non-zero r vector that is distinct from all these combinations and let it be the $i + 1$th column. Since the new column is different than the $d-2$ combinations of the previous columns the set of $i+1$ columns have the property that any $d-1$ of them are linearly independent. Thus we can iterating to pick all n columns as long as the condition of the theorem is satisfied. □

We will now show that restricting ourselves to linear codes does not prevent us from reaching the capacity of symmetric channels:

Theorem 8. *If the input alphabet $X = \{0, \ldots, q-1\}$ of a symmetric channel with capacity C is a field, then for any rate $R < C$, and any $\epsilon > 0$, there is a* linear *code with rate at least R and error probability at most ϵ.*

Proof. Recall the random coding proof of the fact that there exist codes which achieve capacity: In that proof, one generated M codewords $\mathbf{X}(1), \ldots, \mathbf{X}(M)$, each picked independently according to the distribution $p(\mathbf{x}) = p(x_1) \ldots p(x_n)$, and p is chosen to maximize the mutual information. The proof then proceeded to analyze the probability of error by assuming that $\mathbf{X}(m)$ is the transmitted sequence, $\mathbf{Y}$ the received sequence and bounding the probability that for $m' \neq m$, the pair $(\mathbf{X}(m'), \mathbf{Y})$ is jointly typical. What made the proof work was that for any m and $m' \neq m$, the codewords $\mathbf{X}(m)$ and $\mathbf{X}(m')$ were chosen independently; i.e., that the codewords were *pairwise independent.* The *full independence* of the M codewords was not necessary in the proof.

We will now show that capacity approaching linear codes exist by a random coding argument which generates $\mathbf{X}(1), \ldots, \mathbf{X}(M)$ as the codewords of a random *linear* code, and showing that $\Pr(\mathbf{X}(m) = \mathbf{x}) = p(\mathbf{x})$ where $p(\mathbf{x}) = p(x_1) \ldots p(x_n)$, and p maximizes mutual information, and the codewords are pairwise independent. By the remark above, this is sufficient to prove our theorem, as we can simply refer to the proof of the original random coding theorem. To do this, first observe

(i) for a q-ary symmetric channel, the input distribution that achieves capacity is the uniform distribution on the inputs. For this distribution, $p(\mathbf{x}) = q^{-n}$.

(ii) if $\mathbf{x}(1), \ldots, \mathbf{x}(M)$ are the codewords of a code $\mathcal{C}$, then, for any fixed vector $\mathbf{v}$, the code $\tilde{\mathcal{C}}$ whose codewords are $\tilde{\mathbf{x}}(m) = \mathbf{x}(m) + \mathbf{v}$ is equally good as the code $\mathcal{C}$. (If the code $\tilde{\mathcal{C}}$ is used in a communication system, then, at the receiver, we can first subtract $\mathbf{v}$ from the received sequence $\mathbf{y}$, and this would make things as if the code $\mathcal{C}$ were being used.)

Secondly, choose a $n \times k$ matrix G whose entries are chosen independently and uniformly from the q-ary alphabet, and a vector $\mathbf{v}$ whose entries are chosen independently and uniformly over the q-ary alphabet. Consider now a code whose codewords are $\{G\mathbf{u} + \mathbf{v} : \mathbf{u} \in X^k\}$, and denote the codeword that corresponds to $\mathbf{u}$ by $\mathbf{X}(\mathbf{u})$. Note that because of observation (ii) above, this code has the same performance as a linear code with generator matrix G. To complete the proof we only have to show that for each $\mathbf{u}$, and $\mathbf{x}$,

$$\Pr(\mathbf{X}(\mathbf{u}) = \mathbf{x}) = q^{-n},$$

and for each $\mathbf{u}$, $\mathbf{x}$, $\mathbf{u}'$ and $\mathbf{x}'$ with $\mathbf{u}' \neq \mathbf{u}$,

$$\Pr(\mathbf{X}(\mathbf{u}) = \mathbf{x}, \mathbf{X}(\mathbf{u}') = \mathbf{x}') = q^{-2n}.$$

To see these, first note that there are q^{nk+n} equally likely ways of choosing G and $\mathbf{v}$, and for each choice of G there is exactly one choice of $\mathbf{v}$ which will make $\mathbf{X}(\mathbf{u})$ equal to any particular value $\mathbf{x}$ (i.e., $\mathbf{v} = \mathbf{x} - G\mathbf{u}$). Since there are q^{nk} choices of G this gives $\Pr(\mathbf{X}(\mathbf{u}) = \mathbf{x}) = q^{nk}/q^{nk+n} = q^{-n}$. Next, suppose $\mathbf{u} \neq \mathbf{u}'$, and suppose they differ (possibly among others) in the ith position. Observe that

$$\mathbf{X}(\mathbf{u}) - \mathbf{X}(\mathbf{u}') = G(\mathbf{u} - \mathbf{u}'),$$

and for any choice of the $k - 1$ columns $\mathbf{g}_1, \ldots, \mathbf{g}_{i-1}, \mathbf{g}_{i+1}, \ldots, \mathbf{g}_k$ of the matrix G there is exactly one choice of the column $\mathbf{g}_i$ which will make the above difference any fixed value $\mathbf{w}$, (i.e., $\mathbf{g}_i = (u_i - u_i')^{-1}\big[\mathbf{w} - \sum_{j \neq i} \mathbf{g}_j(u_j - u_j')\big]$). Now, given $\mathbf{x}$ and $\mathbf{x}'$, let $\mathbf{w} = \mathbf{x} - \mathbf{x}'$ and observe

$$\Pr(\mathbf{X}(\mathbf{u}) = \mathbf{x}, \mathbf{X}(\mathbf{u}') = \mathbf{x}') = \Pr(\mathbf{X}(\mathbf{u}) = \mathbf{x}, \mathbf{X}(\mathbf{u}') - \mathbf{X}(\mathbf{u}) = \mathbf{w}),$$

and there are $q^{n(k-1)}$ ways of choosing the columns $\mathbf{g}_1, \ldots, \mathbf{g}_{i-1}, \mathbf{g}_{i+1}, \ldots, \mathbf{g}_k$, and once they are chosen, $\mathbf{g}_i$ is determined uniquely by $\mathbf{w}$, and then, $\mathbf{v}$ is determined uniquely by $\mathbf{x}$. Thus, for any $\mathbf{x}$ and $\mathbf{x}'$ there are $q^{n(k-1)}$ choices of G and $\mathbf{v}$ to make $\mathbf{X}(\mathbf{u}) = \mathbf{x}$ and $\mathbf{X}(\mathbf{u}') = \mathbf{x}'$, and so

$$\Pr(\mathbf{X}(\mathbf{u}) = \mathbf{x}, \mathbf{X}(\mathbf{u}') = \mathbf{x}') = q^{n(k-1)}/q^{nk+n} = q^{-2n},$$

as we were required to show. □

Reed–Solomon Codes

From Theorem 3 we know that any q-ary block code of block length n with q^k codewords has minimum distance d at most $n + 1 - k$.

One can easily see that the code $\mathcal{C} = \{0\ldots0, 1\ldots1, \ldots, (q-1)\ldots(q-1)\}$ meets the bound with equality ($k = 1$, $d = n$). This particular code is called the repetition code. Also, the code $\mathcal{C} = X^n$ also meets the bound ($k = n$, $d = 1$). Are there any codes apart from these trivial examples which meet the Singleton bound?

The answer to the question above turns out to be yes. There is a class of codes called the Reed–Solomon Codes which meet the bound with equality for arbitrary n and k, with one caveat: the block length n needs to be less than or equal to q. In this section we will see how to construct Reed–Solomon codes.

Consider and encoder that operates as follows: Fix n distinct values in X, say, $\alpha_1, \dots, \alpha_n$. To send a given message $\mathbf{u} = (u_1, \dots, u_k) \in X^k$, the encoder constructs the polynomial $u(D) = u_1 + u_2 D + \cdots + u_k D^{k-1}$ and evaluates this polynomial at $\alpha_1, \dots, \alpha_n$. The sequence sent over the channel is the result of these evaluations, $x_1 = u(\alpha_1), \dots, x_n = u(\alpha_n)$.

Let us see if this code is linear. Given two codewords $\mathbf{x}$ and $\mathbf{y}$ generated by messages $\mathbf{u}$ and $\mathbf{v}$, it is easy to see that the message $a \cdot \mathbf{u} + b \cdot \mathbf{v}$ yields the codeword $a \cdot \mathbf{x} + b \cdot \mathbf{y}$, and thus the code is linear.

Let us see what the minimum distance of this code is. Suppose $\mathbf{u}$ is a non-zero sequence, so that $u(D)$ is not the zero polynomial. Note that $u(D)$ is of degree at most $k-1$ so it can have at most $k-1$ roots. This means that at most $k-1$ of the x_i's are zero, and thus at least $n-k+1$ of the x_i's are non-zero. Thus, the minimum weight of the code is at least $n-k+1$, and so, $d_{\min}(\mathcal{C}) \geq n-k+1$. From the Singleton bound we conclude that $d_{\min}(\mathcal{C}) = n-k+1$.

The only step in doubt in the above paragraph is that a degree k polynomial has at most k roots. We know this to be true for polynomials with complex coefficients, but we did not prove this result for polynomials over finite fields. Nonetheless, the result is true, as the next section will show.

Polynomials over Finite Fields

In this section we will prove just enough things about finite fields that will suffice to conclude that a polynomial of degree k can have at most k roots. For a full treatment of finite fields in the smallest number of pages, one can refer to E. Artin, *Galois Theory,* Notre Dame Mathematical Lectures, 2nd ed., 1946, also published by Dover Publications.

An expression of the form $f_n D^n + f_{n-1} D^{n-1} + \cdots + f_0$, denoted $f(D)$, is called a *polynomial* of degree n over X if the coefficients $f_n, \dots, f_0$ are all elements of X and if the leading coefficient f_n is nonzero. It is convenient to consider the coefficients f_i associated with a degree n polynomial to be defined for all $i \geq 0$ but to satisfy $f_i = 0$ for $i > n$. Thus, we can consider a polynomial over X as being a way to represent an infinite sequence of field elements $f_0, f_1, \dots$ when only finitely many terms in the sequence are nonzero. The degree of the polynomial is the largest n for which f_n is nonzero. In the special case of the $\mathbf{0}$ polynomial when all f_n are zero, we say that the $\mathbf{0}$ polynomial has degree zero.

Two polynomials are said to be equal if they correspond to the same sequence of coefficients. For example, over the binary field $X = \{0, 1\}$ with modulo-2 addition and multiplication, the polynomials $f(D) = D + D^2 + D^3$ and $g(D) = D$ are different, even though $f(0) = g(0) = 0$ and $f(1) = g(1) = 1$. Thus, as functions over X, f and g are the same, but as polynomials they are different.

The sum and product of two polynomials are defined in the usual way:

$$(f+g)(D) = \sum_i (f_i + g_i) D^i,$$

$$(f \cdot g)(D) = \sum_i \Bigl(\sum_j f_j \cdot g_{i-j} \Bigr) D^i.$$

It is easy to see that the degree of $f+g$ is less than or equal to the maximum of the degrees of f and g, and for nonzero polynomials f and g, $f \cdot g$ is nonzero and the degree of $f \cdot g$

is equal to the sum of the degrees of f and g. It is also easy to see that $f \cdot \mathbf{0} = \mathbf{0}$ for any polynomial f.

The multiplication of a polynomial f by a field element $a \in X$ is defined by

$$(a \cdot f)(D) = \sum_i (a \cdot f_i)D^i,$$

and similarly, the negative of a polynomial is defined by

$$(-f)(D) = \sum_i (-f_i)D^i.$$

It is easy to verify that under addition, the set of polynomials forms an Abelian group. It can also be seen that polynomial multiplication is both associative and commutative and that the distributive law $(f + g) \cdot h = f \cdot h + g \cdot h$ holds. The polynomials over a field is not a field, however, because of the lack of a multiplicative inverse.

Some elementary properties of polynomial arithmetic that will be useful are:

$$-(f \cdot g) = (-f) \cdot g = f \cdot (-g), \tag{1}$$

$$f \cdot g = f \cdot h \text{ and } f \neq \mathbf{0} \text{ imply } g = h. \tag{2}$$

From the above it seems that the polynomials have properties just like that of integers. The analogy goes even further: even though in general we cannot divide one polynomial by another and get a polynomial quotient, we can perform division if we are willing to tolerate a remainder term.

Theorem 9 (Euclidean Division Algorithm). *Let $f(D)$ and $g(D)$ be two polynomials over X and let $g(D)$ have degree at least 1. Then, there exist unique polynomials $h(D)$ and $r(D)$ over X for which*

$$f(D) = g(D) \cdot h(D) + r(D)$$

where degree of $r(D)$ is less than that of $g(D)$.

Proof. We first show how to find h and r and then show that they are unique. Let $f(D)$ have degree n and $g(D)$ have degree m. If $n < m$, set $h(D) = \mathbf{0}$ and $r(D) = f(D)$. If $n \geq m$, we divide $g(D)$ into $f(D)$ by the same procedure we use for ordinary polynomials over the real-number field, letting $h(D)$ to be the quotient, and $r(D)$ the remainder. That is, the leading term of $h(D)$ is $f_n g_m^{-1} D^{n-m}$. This leading term is multiplied by $g(D)$ and subtracted from $f(D)$, and the next term of $h(D)$ is found by starting to divide $g(D)$ into this remainder. When the remainder has smaller degree than $g(D)$ it is taken as $r(D)$. More formally, the algorithm works as follows:

(i) Initialize: set $h(D) = \mathbf{0}$,

(ii) Let n be the degree of f and m the degree of g. If $n < m$, set $r = f$, stop.

(iii) Set $t(D) = f_n \cdot g_m^{-1} D^{n-m}$. Let $h \leftarrow h + t$, $f \leftarrow f - t \cdot g$. Goto (ii).

Since the degree of f is decreasing at each iteration, the algorithm is guaranteed to stop. It is also clear that at the end of the procedure h and r satisfy the equation of the theorem.

Suppose now that there were two solutions the equation given by $h(D), r(D)$ and $h'(D), r'(D)$. Then

$$g(D) \cdot h(D) + r(D) = g(D) \cdot h'(D) + r'(D)$$
$$g(D) \cdot [h(D) - h'(D)] = r'(D) - r(D).$$

Now, $r' - r$ has a degree less than that of g, and thus $h - h' = \mathbf{0}$. But this implies $r' - r = \mathbf{0}$, and it follows that the solution is unique. □

We can use the Euclidean division algorithm to investigate the existence of factors and roots of a polynomial $f(D)$ over X. A polynomial $g(D)$ is said to be *a factor of* (or *divides*) another polynomial $f(D)$ if there is a polynomial $h(D)$ over X such that

$$f(D) = g(D) \cdot h(D).$$

In other words g divides f if the Euclidean division algorithm stops with remainder term $r = \mathbf{0}$. A polynomial f over X is called *reducible* if there are two polynomials g and h over X, each of degree at least 1, for which $f = g \cdot h$. A polynomial is called *irreducible* if is not reducible. The irreducible polynomials play the role the prime numbers play among integers.

It is often useful to factor a polynomial into irreducible factors. In doing so, there is a certain amount of ambiguity, since given one factorization, we can always multiply one of the factors by an arbitrary non-zero field element and multiply another factor by the inverse of the field element, not changing the product. A *monic* polynomial is defined as a polynomial whose leading coefficient is 1, and we avoid the above ambiguity by factoring a polynomial into a field element times a product of monic irreducible factors.

Theorem 10 (Unique Factorization). *A polynomial $f(D)$ over a given field X has a unique factorization into a field element times a product of monic irreducible polynomials over the field, each of degree at least 1.*

Proof. Clearly, the field element is unique, and is just f_n where n is the degree of $f(D)$. Thus we can restrict our attention to monic polynomials. Assume that the theorem is not true, and that there is some lowest degree monic polynomial f that can be factored in two ways, as

$$f(D) = a_1(D) \dots a_k(D) = b_1(D) \dots b_j(D)$$

where the a's and b's are monic and irreducible.

All the a's must be different from all the b's, otherwise a polynomial could be factored out, leaving two factorizations of a lower degree monic polynomial than f. Suppose without loss of generality, that the degree of $b_1(D)$ is less than or equal to that of $a_1(D)$. We then have

$$a_1(D) = b_1(D) \cdot h(D) + r(D)$$

where $r(D)$ is nonzero (since a_1 is irreducible) has degree less than $b_1(D)$ and $a_1(D)$. Substituting this in the factorization of f above, we have

$$r(D) \cdot a_2(D) \dots a_k(D) = b_1(D) \cdot [b_2(D) \dots b_j(D) - h(D) \cdot a_2(D) \dots a_k(D)].$$

Factoring r and multiplying by a field element to make the factors monic, we have two factorizations of a lower degree monic polynomial than $f(D)$, and the irreducible polynomial b_1 does not appear on the left hand side. Thus we have a contradiction and the theorem is true. □

An element α of the field X is said to be a *root* of a polynomial f over that field if $f(\alpha) = 0$.

Theorem 11. *An element α of a field X is a root of a non-zero polynomial $f(D)$ over X if and only if $D - \alpha$ is a factor of $f(D)$. Furthermore, if $f(D)$ has degree n, at most n field elements are roots of $f(D)$.*

Proof. From the Euclidean division algorithm, we have

$$f(D) = (D - \alpha) \cdot h(D) + r(D).$$

Since $D - \alpha$ has degree 1, $r(D)$ has degree zero, and is just a field element r_0. Substituting α for D, we have $f(\alpha) = r_0$. Thus, if $f(\alpha) = 0$ then $r_0 = 0$ and $D - \alpha$ is a factor of $f(D)$. Conversely, if $D - \alpha$ is a factor of $f(D)$, we have $r_0 = 0$ and $f(\alpha) = 0$.

Now, factor $f(D)$ into a field element times a product of irreducible factors of degree at least 1. Since the degree of $f(D)$ is the sum of the degrees of the factors, there are at most n factors. These are unique, and hence $f(D)$ has at most n roots in the field. □

With this theorem, the only doubt about the minimum distance of Reed–Solomon codes is cleared.

The theory of finite fields establishes that fields of all sizes of the form p^m where p is a prime and m is a non-negative integer exist, and that these are the only finite fields. Thus, one can construct Reed-Solomon codes for any X which has p^m elements for some prime p, and integer $m > 0$.

The usual application of Reed-Solomon codes is to binary channels. Here, one considers m uses of the binary channel as one use of a channel with input alphabet $X = \{0, 1\}^m$. This alphabet has $q = 2^m$ elements, and thus can be made into a field by equipping it with proper addition and multiplication operations. One then constructs a Reed–Solomon code for this field.