THEORY OF ERROR CORRECTING CODES MT461/MT5461

MARK WILDON

These notes cover the part of the syllabus for MT461/MT5461 that is not part of MT361. Further installments will be issued as they are ready. All handouts and problem sheets will be put on the MT361 Moodle page, marked **MSc/MSci**.

I would very much appreciate being told of any corrections or possible improvements to these notes.

You are warmly encouraged to ask questions in lectures, and to talk to me after lectures and in my office hours. I am also happy to answer questions about the lectures or problem sheets by email. My email address is `mark.wildon@rhul.ac.uk`.

Extra lecture for MT461/MT5461: Thursday noon (ABLT2).

Office hours in McCrea 240: Tuesday 11am, Thursday 3pm, Friday 3pm.

Date: Second term 2011/12.

Overview

The extra content on the syllabus for MT5461 is on Reed–Solomon codes and cyclic codes over finite fields. These codes are examples of the linear codes that will be covered in Part C of the main lectures.

We will first look at the original definition of Reed–Solomon codes, and see one efficient decoding algorithm. Then in the second half of the term we will look at cyclic codes in general, and make the connection with Reed–Solomon codes. We will end by defining the important family of BCH codes.

1. Revision of fields and polynomials

Most good codes make use of the algebraic structure of finite fields and polynomial rings. For example, the Reed–Solomon code used on compact discs has as its alphabet the finite field of order 2^8. This section gives the minimum background knowledge required for the course: it will allow us to define Reed–Solomon codes over fields of prime order, and to prove the results we need on cyclic codes.[1]

Fields. Roughly put, fields are sets in which one can add, subtract and multiply any two elements, and also divide by non-zero elements. Examples of familiar infinite fields are the rational numbers $\mathbf{Q}$ and the real numbers $\mathbf{R}$. If p is a prime, then the set $\mathbf{F}_p = \{0, 1, \ldots, p-1\}$, with addition and multiplication defined modulo p is a finite field (see Theorem 1.3).

Definition 1.1. A *field* is a set of elements $\mathbf{F}$ with two operations, $+$ (addition) and $\times$ (multiplication), and two special elements $0,\ 1 \in \mathbf{F}$ such that $0 \neq 1$ and

(1) $a + b = b + a$ for all $a, b \in \mathbf{F}$;
(2) $0 + a = a + 0 = a$ for all $a \in \mathbf{F}$;
(3) for all $a \in \mathbf{F}$ there exists $b \in \mathbf{F}$ such that $a + b = 0$;
(4) $a + (b + c) = (a + b) + c$ for all $a, b, c \in \mathbf{F}$;

(5) $a \times b = b \times a$ for all $a, b \in \mathbf{F}$;
(6) $1 \times a = a \times 1 = a$ for all $a \in \mathbf{F}$;
(7) for all non-zero $a \in \mathbf{F}$ there exists $b \in \mathbf{F}$ such that $a \times b = 1$;
(8) $a \times (b \times c) = (a \times b) \times c$ for all $a, b, c \in \mathbf{F}$;

(9) $a \times (b + c) = a \times b + a \times c$ for all $a, b, c \in \mathbf{F}$.

[1] If you have not seen finite fields of prime power order then you will have to take one or two results on trust towards the end of the course. The examination will only require finite fields of prime order.

It may be helpful to note that (1)–(4) imply that $\mathbf{F}$ is an abelian group under addition, and that (5)–(8) imply that $(\mathbf{F}\backslash\{0\}, \times)$ is an abelian group under multiplication. The final axiom (9) is the *distributive law* relating addition and multiplication.

It is usual to write $-a$ for the element b in (4), and a^{-1} for the element b in (8). These are the additive and multiplicative *inverses* of a and b, respectively: by the first exercise below they are unique. We usually write ab rather than $a \times b$.

Definition 1.2. The *order* of a finite field $\mathbf{F}$ is defined to be the number of elements in $\mathbf{F}$.

Exercise: show from the axioms for a field that if $\mathbf{F}$ is a field then $a \times 0 = 0$ for all $a \in \mathbf{F}$. Show that if $x \in \mathbf{F}$ then x has a unique additive inverse, and that if $x \neq 0$ then x has a unique multiplicative inverse.

Exercise: show from the axioms for a field that if $\mathbf{F}$ is a field and a, $b \in \mathbf{F}$ are such that $a \times b = 0$, then either $a = 0$ or $b = 0$.

Theorem 1.3. *Let p be a prime. The set $\mathbf{F}_p = \{0, 1, \ldots, p-1\}$ with addition and multiplication defined modulo p is a field.*

Example 1.4. The addition and multiplication tables for the finite field $\mathbf{F}_4 = \{0, 1, \alpha, 1+\alpha\}$ of order 4 are shown below.

$+$	0	1	α	$1+\alpha$
0	0	1	α	$1+\alpha$
1	1	0	$1+\alpha$	α
α	α	$1+\alpha$	0	1
$1+\alpha$	$1+\alpha$	α	1	0

$\times$	1	α	$1+\alpha$
1	1	α	$1+\alpha$
α	α	$1+\alpha$	1
$1+\alpha$	$1+\alpha$	1	α

Probably the most important thing to realise is that $\mathbf{F}_4$ **is not the integers modulo** 4. Indeed, in $\mathbf{Z}_4 = \{0, 1, 2, 3\}$ we have $2 \times 2 = 0$, but if $a \in \mathbf{F}_4$ then $a \times a \neq 0$, as can be seen from the multiplication table. (Alternatively this follows from the second exercise above.)

Polynomials. Let $\mathbf{F}$ be a field. Let $\mathbf{F}[x]$ denote the set of all polynomials

$$f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_dx^d$$

where $d \in \mathbf{N}_0$ and $a_0, a_1, a_2, \ldots, a_d \in \mathbf{F}$.

If $f(x) = a_0 + a_1x + a_2 + \cdots + a_dx^d$ where $a_d \neq 0$, then we say that d is the *degree* of the polynomial $f(x)$, and write $\deg f = d$. Note that a polynomial is a non-zero constant if and only if it has degree 0. (The degree of the zero polynomial $f(x) = 0$ is unspecified.)

Polynomials are added and multiplied in the natural way.

Lemma 1.5 (Division algorithm)**.** *Let $\mathbf{F}$ be a field, let $f(x) \in \mathbf{F}[x]$ be a non-zero polynomial and let $g(x) \in \mathbf{F}[x]$. There exist polynomials $s(x), r(x) \in \mathbf{F}[x]$ such that*

$$g(x) = s(x)f(x) + r(x)$$

and either $r(x) = 0$ or $\deg r(x) < \deg f(x)$.

We say that $s(x)$ is the *quotient* and $r(x)$ is the *remainder* when $g(x)$ is divided by $f(x)$.

Exercise: Let $f(x) = x^3 + x + 1 \in \mathbf{F}_2[x]$. Find the quotient and remainder when $g(x) = x^5 + x^2 + x$ is divided by $f(x)$.

For Reed–Solomon codes we shall need the following properties of polynomials.

Lemma 1.6. *Let $\mathbf{F}$ be a field.*

(i) *If $f(x) \in \mathbf{F}[x]$ has $a \in \mathbf{F}$ as a root, i.e. $f(a) = 0$, then there is a polynomial $g(x) \in \mathbf{F}[x]$ such that $f(x) = (x - a)g(x)$.*

(ii) *If $f(x) \in \mathbf{F}[x]$ has degree d then $f(x)$ has at most d distinct roots in $\mathbf{F}$.*

(iii) *If f, $g \in \mathbf{F}[x]$ both have degree $< n$ and there exist distinct $a_1, \ldots, a_n \in \mathbf{F}$ such that $f(a_i) = g(a_i)$ for each $i \in \{1, \ldots, n\}$ then $f(x) = g(x)$.*

Lemma 1.7 (Polynomial interpolation)**.** *Let $\mathbf{F}$ be a field. Let*

$$a_1, a_2, \ldots, a_k \in \mathbf{F}$$

be distinct and let $y_1, y_2, \ldots, y_k \in \mathbf{F}$. The unique polynomial $f(x) \in \mathbf{F}[x]$ of degree $< k$ such that $f(a_i) = y_i$ for all i is

$$f(x) = \sum_{i=1}^{n} y_i \frac{\prod_{j \neq i}(x - a_j)}{\prod_{j \neq i}(a_i - a_j)}.$$

Part 1: Reed–Solomon Codes

2. Definition and basic properties of Reed–Solomon codes

In this section we give the original definition of Reed–Solomon codes over finite fields. We will work over the fields $\mathbf{F}_p$ of prime degree constructed in Theorem 1.3, but it is easy to see that the definition and all the results extend to a general finite field.

Definition 2.1. Let p be a prime and let k, $n \in \mathbf{N}$ be such that $k \leq n \leq p$. Let

$$a_1, a_2, \ldots, a_n$$

be distinct elements of $\mathbf{F}_p$. For each polynomial $f(x) \in \mathbf{F}_p[x]$ we define a word $u(f) \in \mathbf{F}_p^n$ by

$$u(f) = (f(a_1), f(a_2), \ldots, f(a_n)).$$

The *Reed–Solomon code* associated to the parameters p, n, k and the field elements $a_1, a_2, \ldots, a_n$ is the length n code over $\mathbf{F}_p$ with codewords

$$\{u(f) : f \in \mathbf{F}_p[x],\ f = 0 \text{ or } \deg f \leq k-1\}.$$

Correction: Original version had $\{u(f) : f \in \mathbf{F}_p[x], \deg f \leq k-1\}$. This wrongly excluded the zero polynomial, which has undefined degree, but still needs to be included.

It is worth bearing in mind Remark 1.6(3) from the main lectures: the parameters p, n, k and the field elements $a_1, a_2, \ldots, a_n$ are part of the specification of a Reed–Solomon code, and should be assumed to be known to everyone.

For instance, the Reed–Solomon code used on compact discs is defined over the finite field $\mathbf{F}_{2^8}$ (using the obvious extension of Definition 2.1) and has parameters $n = 255$ and $k = 223$.

Example 2.2. Let $p = 5$ and let $k = 2$.

(1) If $n = 3$ and we take $a_1 = 0$, $a_2 = 1$ and $a_3 = 2$, then the associated Reed–Solomon code has a codeword

$$(f(0), f(1), f(2))$$

for each $f(x) \in \mathbf{F}_p[x]$ of degree ≤ 1. If $f(x) = bx + c$ then

$$u(f) = (c, b+c, 2b+c)$$

so the full set of codewords is

$$\{(c, b+c, 2b+c) : b, c \in \mathbf{F}_5\}.$$

This code is 1-error detecting, but not 2-error detecting.

(2) If $n = 4$ and we take a_1, a_2, a_3 as before, and $a_4 = 3$ then we get an extension of the code in (1). The set of codewords is

$$\{(c, b + c, 2b + c, 3b + c) : b, c \in \mathbf{F}_5\}.$$

This code has the same size as the previous code, but longer length, so one might expect it to have better error-detecting and error-correcting properties.

The next exercise will be rapidly subsumed by more general results proved using the theory developed in Part A of the main course. But you will find it very instructive to find a direct proof.

Exercise: Show that if $C = \{(c, b+c, 2b+c, 3b+c) : b, c \in \mathbf{F}_5\}$ then C is 2-error detecting and 1-error correcting. (*Hint:* Question 4 on Sheet 1 will help, particularly for the latter part.)

For the rest of this section, fix parameters p, n, k and field elements $a_1, a_2, \ldots, a_n$. Let $RS_{p,n,k}$ denote the associated Reed–Solomon code over $\mathbf{F}_p$.

Lemma 2.3. *The Reed–Solomon code $RS_{p,n,k}$ has size p^k.*

Possibly you have realised that the Hamming distances between codewords in a code controls how many errors the code can detect and correct. The next lemma gives a lower bound on these distances for the Reed–Solomon code.

Lemma 2.4. *If f, $g \in \mathbf{F}_p[x]$ are distinct polynomials of degree $\leq k-1$ then*

$$d(u(f), u(g)) \geq n - k + 1.$$

To find the minimum distance (as defined in Definition 3.1 of the main notes) of $RS_{p,n,k}$ we must also show that there are two codewords in $RS_{p,n,k}$ at distance $n - k + 1$. This can be done using Lemma 1.7 on polynomial interpolation.

Theorem 2.5. *The minimum distance of $RS_{p,n,k}$ is $n - k + 1$.*

The Singleton Bound (to be proved in Part B of the main course) states that any p-ary code of length n and minimum distance d has at most p^{n-d+1} codewords. By Lemma 2.3 and Theorem 2.5, the Reed–Solomon codes meet this bound, and so have the largest possible size for their length and minimum distance.

Corollary 2.6. *Let p be a prime. If k, $e \in \mathbf{N}$ are such that $k + 2e \leq p$ then the Reed–Solomon code $RS_{p,k+2e,k}$ is e-error correcting.*

We now discuss encoding and decoding for Reed–Solomon codes. The code $RS_{p,n,k}$ has size p^k so it can encode p^k different messages. We saw in the first proof of Lemma 2.3 (by polynomial interpolation) that given any $(b_1, b_2, \ldots, b_k) \in \mathbf{F}_p^k$, there exists a polynomial $f \in \mathbf{F}_p[x]$ of degree $< k$ such that $f(a_i) = b_i$ for $1 \leq i \leq k$, and so

$$u(f) = (b_1, b_2, \ldots, b_k, \ldots).$$

Hence if we agree to identify messages with $\mathbf{F}_p^k$, then we can use polynomial interpolation to define a suitable encoder. (One advantage of this encoder is that if a message is received without any errors, then the message can be read off from the first k positions.)

Decoding is a much trickier issue. We would like to use nearest neighbour decoding, but the naïve algorithm where we search through the entire code looking for the nearest codeword is completely impractical once p and k are large. For example, the Reed–Solomon code used on compact disks has size $k = 223$ and size 256^{223}.

The original decoder proposed by Reed and Solomon used polynomial interpolation to find all codewords that agreed with the received word in at least k positions. The nearest one would then be taken as the sent codeword. (So the decoder implements nearest neighbour decoding.) However, there are $\binom{n}{k}$ choices of k positions from n, so while an improvement, this would still be impractical for large codes.

Example 2.7. Suppose we use the Reed–Solomon code with $p = 5$, $n = 4$ and $k = 2$ evaluating at $a_1 = 0$, $a_2 = 1$, $a_3 = 2$, $a_4 = 3$, as in Example 2.2(2). By Corollary 2.6, this code is 1-error correcting. Suppose we receive $v = (4, 0, 3, 0)$.

Given any two positions i and j, it follows from Lemma 1.7 that there is a unique polynomial g of degree < 2 such that $g(a_i) = v_i$ and $g(a_j) = v_j$.

The table below shows the interpolating polynomials for each pair of positions and the corresponding codewords. For example, to find $f(x)$ such that $f(0) = 4$ and $f(2) = 3$, we use Lemma 1.7 and get

$$f(x) = 4\frac{x-2}{0-2} + 3\frac{x-0}{2-0} = 3(x-2) - x = 2x + 4.$$

Conditions on f	Solution	Codeword $u(f)$
$f(0) = 4,\ f(1) = 0$	$f(x) = 4 + x$	$(4, 0, 1, 2)$
$f(0) = 4,\ f(2) = 3$	$f(x) = 4 + 2x$	$(4, 1, 3, 0)$
$f(1) = 0,\ f(2) = 3$	$f(x) = 2 + 3x$	$(2, 0, 3, 1)$
$f(0) = 4,\ f(3) = 0$	$f(x) = 4 + 2x$	$(4, 1, 3, 0)$
$f(1) = 0,\ f(3) = 0$	$f(x) = 0$	$(0, 0, 0, 0)$
$f(2) = 3,\ f(3) = 0$	$f(x) = 4 + 2x$	$(4, 1, 3, 0)$

In practice, we would stop as soon as we found the codeword $(4, 1, 3, 0)$ since $d(4130, 4030) = 1$, and by the exercise on page 19 of the main lecture notes, there is at most one codeword within distance 1 of any given word.

3. Efficient decoding of Reed–Solomon codes

In this section we shall see an efficient algorithm for decoding Reed–Solomon codes invented by Berlekamp and Welch in 1983. As usual we work with the Reed–Solomon code $RS_{p,n,k}$ where p is prime and n, $k \in \mathbf{N}$, and polynomials are evaluated at $a_1, a_2, \ldots, a_n$. Assume that $n = k + 2e$, so by Corollary 2.6 the code is e-error correcting.

Theorem 3.1 (Key Equation)**.** *Suppose that the codeword*

$$u(f) = (f(a_1), \ldots, f(a_n))$$

is transmitted and the word $(v_1, \ldots, v_n)$ is received. If there are $\leq e$ errors in transmission then there exist polynomials

- *$Q(x)$ of degree $\leq k + e - 1$*
- *$E(x)$ of degree $\leq e$,*

such that the Key Equation

$$Q(a_i) = v_i E(a_i)$$

holds for each $i \in \{1, 2, \ldots, n\}$.

It is not at all obvious why the Key Equation is helpful. We first show that any solution to it can be used to decode a received word.

Lemma 3.2. *Suppose that the codeword*

$$u(f) = (f(a_1), \ldots, f(a_n))$$

is transmitted and the word $(v_1, \ldots, v_n)$ is received. If $E(x)$ and $Q(x)$ satisfy the Key Equation, and the number of errors in transmission is $\leq e$, then $Q(x) = f(x)E(x)$ and so $f(x) = Q(x)/E(x)$.

We saw in the proof of Theorem 3.1 that we could solve the Key Equation if somehow we knew where the errors had occurred. But this does not help us in practice. Instead we proceed by solving linear equations. (Unlike the naïve approach in §2, we only have to solve one system, not $\binom{n}{k}$ separate systems!)

Lemma 3.3. *Suppose that the word $(v_1, \ldots, v_n)$ is received. The polynomials*

$$\begin{aligned} Q(x) &= Q_0 + Q_1 x + \cdots + Q_{k+e-1} x^{k+e-1} \\ E(x) &= E_0 + E_1 x + \cdots + E_e x^e \end{aligned}$$

in $\mathbf{F}_p[x]$ satisfy the Key Equation if and only if

$$\begin{aligned} Q_0 + a_i Q_1 + a_i^2 Q_2 + \cdots + a_i^{k+e-1} Q_{k+e-1} & \\ &= v_i(E_0 + a_i E_1 + a_i^2 E_2 + \cdots + a_i^e E_e) \end{aligned}$$

for each $i \in \{1, \ldots, n\}$. An equivalent condition is that

$$\begin{pmatrix} 1 & a_1 & a_1^2 & \cdots & a_1^{k+e-1} & -v_1 & -v_1 a_1 & \cdots & -v_1 a_1^e \\ 1 & a_2 & a_2^2 & \cdots & a_2^{k+e-1} & -v_2 & -v_2 a_2 & \cdots & -v_2 a_2^e \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & a_n & a_n^2 & \cdots & a_n^{k+e-1} & -v_n & -v_n a_n & \cdots & -v_n a_n^e \end{pmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ \vdots \\ Q_{k+e-1} \\ E_0 \\ E_1 \\ \vdots \\ E_e \end{pmatrix} = 0$$

The matrix above is an $n \times (n+1)$ matrix. So we can solve the Key Equation by solving an $n \times (n+1)$ system of linear equations.

Example 3.4. We shall use the code of Example 2.2(2) (also seen in Example 2.7). Let $p = 5$, let $k = 2$, let $e = 1$ (so $n = 4$) and let $a_1 = 0$, $a_2 = 1$, $a_3 = 2$, $a_4 = 3$. With these parameters, the Key Equation for the polynomials $Q(x) = Q_0 + Q_1 x + Q_2 x^2$ and $E(x) = E_0 + E_1 x$ is

$$\begin{pmatrix} 1 & 0 & 0 & 4v_1 & 0 \\ 1 & 1 & 1 & 4v_2 & 4v_2 \\ 1 & 2 & 4 & 4v_3 & 3v_3 \\ 1 & 3 & 4 & 4v_4 & 2v_4 \end{pmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ E_0 \\ E_1 \end{pmatrix} = 0.$$

(1) Suppose we receive the word 4130. (This is the codeword for $f(x) = 4+2x$.) Then $v_1 = 4$, $v_2 = 1$, $v_3 = 3$, $v_4 = 0$ and we must solve

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 4 & 4 \\ 1 & 2 & 4 & 2 & 4 \\ 1 & 3 & 4 & 0 & 0 \end{pmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ E_0 \\ E_1 \end{pmatrix} = 0.$$

The kernel is two dimensional, spanned by the vectors

$$(0,4,2,0,1)^t, \quad (4,2,0,1,0)^t.$$

The first vector gives $Q(x) = 4x + 2x^2$ and $E(x) = x$, so we decode using $f(x) = Q(x)/E(x) = 4 + 2x$ to get $u(f) = 4130$. (The second vector gives the same answer even more quickly.)

(2) Suppose we receive the word 4030. Then $v_1 = 4$, $v_2 = 0$, $v_3 = 3$, $v_4 = 0$ and we must solve

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 2 & 4 & 2 & 4 \\ 1 & 3 & 4 & 0 & 0 \end{pmatrix} \begin{pmatrix} Q_0 \\ Q_1 \\ Q_2 \\ E_0 \\ E_1 \end{pmatrix} = 0.$$

The kernel is one dimension spanned by $(1,2,2,4,1)^t$. So we take $Q(x) = 1 + 2x + 2x^2$ and $E(x) = 4 + x$. Polynomial division gives

$$Q(x)/E(x) = 2x + 4$$

so we decode using $f(x) = 2x + 4$ to get $u(f) = 4130$.

(3) Finally suppose we receive 4020. Then the kernel is one dimensional, spanned by $(4,3,3,1,0)$. So we take $Q(x) = 4 + 3x + 3x^2$ and $E(x) = 1$, but $Q(x)/E(x)$ does not have degree ≤ 1, so we are unable to decode. Since the Key Equation method always works when $\leq e$ errors occur, we know that ≥ 2 errors have occurred, but we are unable to correct them.

When more than e errors occur it can also happen that the received word is decoded incorrectly. For example, this would happen in the setup of Example 3.4 if we received 4000. Another possibility is that $E(x)$ does not divide $Q(x)$: in this case we detect an error but are unable to correct it.

Final remarks: (1) When preparing this section I used §4 of these notes: `http://math.berkeley.edu/~mhaiman/math55/reed-solomon.pdf`.

The next section in Haiman's notes explains a refinement to the Key-Equation method that reduces the problem to solving an $e \times e$ system of equations and then doing a one-off polynomial interpolation. Since e is much less than n (for the code used on compact discs, $n = 255$ and $e = 16$) this is a big improvement.

(2) A MATHEMATICA notebook for solving the Key Equation is available on Moodle. Unless you really want to do it by hand, I suggest you use it (or another computer algebra program) to do Question 4 on Sheet 4.

Part 2: Cyclic codes

4. Cyclic codes

Cyclic codes are a special type of linear code. In Part C of the main course we will consider linear codes over the binary alphabet. Here we work more generally over a finite field $\mathbf{F}_p$ of prime order. (As for Reed–Solomon codes, all our results extend to a general finite field.)

Definition 4.1. Let p be prime. A code C over $\mathbf{F}_p$ is *linear* if

(i) for all $u \in C$ and $a \in \mathbf{F}_p$ we have $au \in C$;
(ii) for all u, $w \in C$ we have $u + w \in C$.

Here au is the word defined by $(au)_i = au_i$ and $u + v$ is defined by $(u + w)_i = u_i + w_i$. Equivalently, a code C over $\mathbf{F}_p$ is linear if C is a vector subspace of $\mathbf{F}_p^n$.

Exercise: Show that any Reed–Solomon code is linear.

Definition 4.2. Let p be a prime. A code C over $\mathbf{F}_p$ is said to be *cyclic* if C is linear and

$$(u_0, u_1, \ldots, u_{n-1}) \implies (u_{n-1}, u_0, \ldots, u_{n-2}) \in C.$$

The reason for numbering positions from 0 will be seen shortly. Note that we can apply the shift in the definition many times, so a cyclic code is closed under arbitrary cyclic shifts.

Example 4.3.

(1) Let p be prime and let $n \in \mathbf{N}$. The repetition code of length n over $\mathbf{F}_p$ is cyclic.

(2) Let C be the set of binary words of length $n \in \mathbf{N}$ with evenly many 1s. We may define C using addition in $\mathbf{F}_2$ by
$$C = \{(u_0, \ldots, u_{n-1}) : u_i \in \mathbf{F}_2, u_0 + \cdots + u_{n-1} = 0\}.$$
Then C is a cyclic code.

(3) Let D be the binary code $\{0000, 1010, 0101, 1111\}$. *Exercise:* check that D is linear. The shift map acts on D by fixing 0000 and 1111 and swapping 1010 and 0101, so D is cyclic.

There is a very helpful correspondence between codewords in a cyclic code and polynomials.

Definition 4.4. Let p be prime. Given a codeword

$$u = (u_0, u_1, \ldots, u_{n-1}) \in \mathbf{F}_p^n.$$

we define the *polynomial corresponding to u* to be

$$u_0 + u_1 x + \cdots + u_{n-1} x^{n-1}$$

and write

$$u \longleftrightarrow u_0 + u_1 x + \cdots + u_{n-1} x^{n-1}.$$

For example, the polynomials corresponding to codewords in the binary code D in Example 4.3(3) are as shown below:

$$\begin{aligned} 0000 &\longleftrightarrow 0 \\ 1010 &\longleftrightarrow 1 + x^2 \\ 0101 &\longleftrightarrow x + x^3 \\ 1111 &\longleftrightarrow 1 + x + x^2 + x^3 \end{aligned}$$

Notice that $1+x^2$ corresponds to 1010, and if we multiply $1+x^2$ by x we get $x + x^3$ which corresponds to 0101. If we multiply by x again we get $x^2 + x^4$; however, the shift of 0101 is 1010, which corresponds to $1 + x^2$. The same problem arises with 1111.

If somehow we could identify x^4 with 1 then in all cases multiplication by x would correspond to the cyclic shift $(u_0, u_1, u_2, u_3) \mapsto (u_3, u_0, u_1, u_2)$ of codewords.

Definition 4.5. Let p be prime. The ring $\mathbf{F}_p[x]/(x^n - 1)$, read as '$\mathbf{F}_p[x]$ modulo $x^n - 1$' has elements all polynomials in $\mathbf{F}_p[x]$ of degree $< n$. Given

$$\begin{aligned} f(x) &= a_0 + a_1 x + \cdots + a_{n-1} x^{n-1} \\ g(x) &= b_0 + b_1 x + \cdots + b_{n-1} x^{n-1} \end{aligned}$$

in $\mathbf{F}_p[x]/(x^n - 1)$ we define their sum, in the obvious way, to be

$$f(x) + g(x) = (a_0 + b_0) + (a_1 + b_1)x + \cdots + (a_{n-1} + b_{n-1})x^{n-1}.$$

The product $f(x)g(x) \in \mathbf{F}_p[x]/(x^n - 1)$ is defined by taking the normal product $f(x)g(x) \in \mathbf{F}_p[x]$ and then taking the remainder on division by $x^n - 1$.

In Definition 4.4, the polynomial corresponding to a codeword $u \in \mathbf{F}_p^n$ should be taken to be an element of $\mathbf{F}_p[x]/(x^n - 1)$.

Remarks 4.6.

(1) This definition is analogous to the earlier definition (see Theorem 1.3) of the finite field $\mathbf{F}_p$ as the set of numbers
$$\{0, 1, \ldots, p-1\}$$
with addition and multiplication defined by performing these operations in $\mathbf{Z}$, and then taking the remainder after division by p.

(2) We will assume that $\mathbf{F}[x]/(x^n - 1)$ is a ring, i.e. it satisfies all the axioms, except (7), on page 2. It is routine but time-consuming to check they all hold.

This result also follow from the general theory of quotient rings. Defined this way, $\mathbf{F}[x]/(x^n - 1)$ is the set of cosets
$$f(x) + \langle (x^n - 1) \rangle$$
of the ideal in $\mathbf{F}[x]$ generated by $(x^n - 1)$. Our definition makes a specific choice of coset representatives.

Exercise: Check that if $f(x) = x + x^3 \in \mathbf{F}_2[x]/(x^4 - 1)$ is the polynomial corresponding to 0101 then $xf(x) = 1 + x^2$ and so
$$xf(x) \longleftrightarrow 1010$$
as we wanted. Also show that $f(x)^2 = 0 \in \mathbf{F}_2[x]/(x^4 - 1)$.

We defined the ring $\mathbf{F}_p[x]/(x^n - 1)$ so that the following lemma would hold.

Lemma 4.7. *Let p be a prime and let $u = (u_0, u_1, \ldots, u_{n-1}) \in \mathbf{F}_p^n$. Let*
$$f(x) = u_0 + u_1 x + \cdots + u_{n-1} x^{n-1} \in \mathbf{F}_p[x]/(x^n - 1)$$
be the polynomial corresponding to u. The polynomial corresponding to $(u_{n-1}, u_0, \ldots, u_{n-2})$ is $xf(x) \in \mathbf{F}[x]/(x^n - 1)$.

From now on we will usually identify a cyclic code of length n over $\mathbf{F}_p$ with the corresponding set of polynomials in $\mathbf{F}_p[x]/(x^n - 1)$. By Lemma 4.7, cyclic shifts of codewords correspond to multiplication by x. The next exercise gives a more general property.

Exercise: Let $C \subseteq \mathbf{F}_p[x]/(x^n - 1)$ be a cyclic code. Show that if $f(x) \in C$ and $h(x) \in \mathbf{F}_p/(x^n - 1)$ then $h(x)f(x) \in C$.

The next definition will lead to a way to find all cyclic codes over $\mathbf{F}_p$ of a specified length.

Definition 4.8. Let p be a prime. Let C be a cyclic code of length n over the finite field $\mathbf{F}_p$, identified with a subset of $\mathbf{F}_p[x]/(x^n-1)$. A *generator polynomial* for C is a polynomial $g(x) \in \mathbf{F}_p[x]$ of degree $< n$ such that $g(x)$ divides $x^n - 1$ and

$$C = \left\{\bar{f}(x)\bar{g}(x) : \bar{f}(x) \in \mathbf{F}[x]/(x^n-1)\right\}.$$

Here a bar over a polynomial means that it should be considered as an element of $\mathbf{F}_p[x]/(x^n-1)$. Thus the product $\bar{f}(x)\bar{g}(x)$ takes place in $\mathbf{F}_p[x]/(x^n-1)$, not in $\mathbf{F}_p[x]$.

Example 4.9. Let $C = \{0, 1+x^2, x+x^3, 1+x+x^2+x^3\} \subseteq \mathbf{F}_2[x]/(x^4-1)$ be the polynomial version of the code in Example 6.2(2). We claim that $g(x) = 1+x^2$ is a generator polynomial for C.

Since $g(x)^2 = (1+x^2)^2 = 1+x^4 = x^4-1$, the polynomial $g(x)$ divides x^4-1. Every polynomial in C is a multiple of $1+x^2$ since we have

$$\begin{aligned} 0\bar{g}(x) &= 0 \\ 1\bar{g}(x) &= 1+x^2 \\ x\bar{g}(x) &= x+x^3 \\ (1+x)\bar{g}(x) &= 1+x+x^2+x^3. \end{aligned}$$

Finally, suppose $\bar{f}(x) \in \mathbf{F}_2[x]/(x^4-1)$. Dividing $f(x)$ by $1+x^2$ we can write

$$f(x) = s(x)(1+x^2) + r(x)$$

where the degree of $r(x)$ is < 2. So $r(x) \in \{0, 1, x, 1+x\}$ and

$$f(x)(1+x^2) = s(x)(1+x^2)^2 + r(x)(1+x^2).$$

Hence, taking products in $\mathbf{F}_2[x]/(x^4-1)$ we have

$$\bar{f}(x)(1+x^2) = r(x)(1+x^2) \in C.$$

Exercise: Consider the code over $\mathbf{F}_3$ with codewords $\{(a,b,c,a,b,c) : a,b,c \in \mathbf{F}_3\}$. The corresponding subset of $\mathbf{F}_3[x]/(x^6-1)$ is

$$C = \{a + bx + cx^2 + ax^3 + bx^4 + cx^5 : a,b,c \in \mathbf{F}_3\}.$$

Find a generator polynomial for C.

Theorem 4.10. *Let $\mathbf{F}$ be a finite field and let $C \subseteq \mathbf{F}[x]/(x^n-1)$ be a cyclic code of length n. Then C has a generator polynomial.*

Conversely, we can start with a generator polynomial and construct a cyclic code.

Theorem 4.11. *Let p be a prime, let $n \in \mathbf{N}$ and let $g(x) \in \mathbf{F}_p[x]/(x^n - 1)$ be a divisor of $x^n - 1$. If $g(x)$ has degree $r < n$ then*

$$\{g(x), xg(x), \dots, x^{n-r-1}g(x)\}.$$

is a basis for the cyclic code $C \subseteq \mathbf{F}_p[x]/(x^n - 1)$ with generator polynomial $g(x)$.

Note that Theorem 4.11 shows that a cyclic code of length n with a generator polynomial of degree r over the finite field $\mathbf{F}_p$ has dimension $n - r$ and size p^{n-r}.

The following example shows how to construct all cyclic codes of a given length over the finite field $\mathbf{F}_p$, provided we have to hand the factorization of $x^n - 1$ in $\mathbf{F}_p$. (Finding these factorizations requires finite field theory beyond the scope of this course: they will be provided if required in an exam question.)

Example 4.12. In $\mathbf{F}_2[x]$ we have

$$x^7 - 1 = (1 + x)(1 + x + x^3)(1 + x^2 + x^3)$$

where each factor is *irreducible*, i.e. the factors cannot be written as products of polynomials of small degree. The polynomial divisors of $x^7 - 1$ are therefore

$$\begin{aligned}&1,\ 1 + x,\ 1 + x + x^3,\ 1 + x^2 + x^3,\ (1 + x)(1 + x + x^3),\\ &(1 + x)(1 + x^2 + x^3),\ (1 + x + x^3)(1 + x^2 + x^3)\end{aligned}$$

(1) The code with generator polynomial $1 + x$ is the parity check extension of the code consisting of all binary words of length 6.

(2) Since $(1 + x + x^3)(1 + x^2 + x^3) = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6$ the code with this generator polynomial is the binary repetition code of length 7.

(3) The code with generator polynomial $1 + x + x^3$ has generator matrix

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

it is equivalent to the Hamming $[7, 4, 3]$-code. *Exercise:* prove this.

Theorem 4.13. *Let C be a cyclic code of length n over $\mathbf{F}$ with generator polynomial $g(x) \in \mathbf{F}_p[x]$ of degree r. If $g(x) = a_0 + a_1x + \cdots + a_rx^r$ then the $(n-r) \times n$ matrix*

$$G = \begin{pmatrix} a_0 & a_1 & a_2 & \dots & a_r & 0 & \dots & 0 \\ 0 & a_0 & a_1 & \dots & a_{r-1} & a_r & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & a_0 & \dots & \dots & a_{r-1} & a_r \end{pmatrix}$$

is a generator matrix for C.

The encoding scheme on page 37 of the main notes would encode the number represented by $(b_0, b_1, \dots, b_{n-r-1})$ in binary as

$$(b_0, b_1, \dots, b_{n-r-1})G$$

As a polynomial, this codeword is

$$b_0f(x) + b_1xf(x) + \cdots + b_{n-r-1}x^{n-r-1}f(x) = \\ (b_0 + b_1x + \cdots + b_{n-r-1}x^{n-r-1})f(x).$$

So we can encode messages in a cyclic code by polynomial multiplication. This can be performed more quickly than matrix multiplication.

We end by finding the parity check matrix of an important family of cyclic codes, which includes suitable cyclic Reed–Solomon codes (see Question 5 on Sheet 8 and Question 11 on Sheet 9). This gives a way to decode these codes using syndrome decoding.

Theorem 4.14. *Let C be a cyclic code over $\mathbf{F}_p$ of length n. Suppose that C has generator polynomial $g(x) \in \mathbf{F}_p$ of degree r and that g has distinct roots $c_1, \dots, c_r \in \mathbf{F}_p$. Then the matrix*

$$H = \begin{pmatrix} 1 & c_1 & c_1^2 & \cdots & c_1^{n-1} \\ 1 & c_2 & c_2^2 & \cdots & c_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & c_r & c_r^2 & \cdots & c_r^{n-1} \end{pmatrix}$$

is a parity check matrix for C. The syndrome of a received word $v \in \mathbf{F}_p^n$ corresponding to the polynomial $k(x) \in \mathbf{F}_p[x]/(x^n - 1)$ is equal to

$$(k(c_1), \dots, k(c_r)).$$

In particular, we note that the syndromes of the n possible errors affecting only a single bit are distinct, and so C is at least 1-error correcting. In fact C has minimum distance $r + 1$: see the end of the proof of Theorem 5.4 below.

5. A brief look at BCH codes

This section may be considered non-examinable, and is included for interest only. Some knowledge of finite fields of prime power order is needed.

Hamming codes as cyclic codes. We start by giving a more algebraic way to construct the Hamming code of length 7. This construction generalizes to Hamming codes of any length.

Let $\mathbf{F}_8$ be the finite field field of order 8. Let $\alpha \in \mathbf{F}_8$ be such that every non-zero element of $\mathbf{F}_8$ is a power of α. (It is a general theorem that any finite field contains such a *primitive element.*) Let

$$C = \{f(x) \in \mathbf{F}_2[x]/(x^7 - 1) : f(\alpha) = 0\}.$$

Exercise: show from this definition that C is a cyclic code.

Using the correspondence between polynomials and codewords, an equivalent definition of C is

$$C = \{(a_0, a_1, \ldots, a_6) \in \mathbf{F}_2^7 : a_0 + a_1\alpha + \cdots + a_6\alpha^6 = 0\}$$

Lemma 5.1. *The code C is equivalent to the Hamming $[7, 4, 3]$-code by a permutation of its positions.*

Proof. We may define the finite field $\mathbf{F}_8$ by $\mathbf{F}_8 = \mathbf{F}_2(\alpha)$ where α is a root of the irreducible primitive polynomial

$$g(x) = x^3 + x + 1 \in \mathbf{F}_2[x].$$

If $f(x) \in \mathbf{F}_2[x]$ is a polynomial such that $f(\alpha) = 0$ then, since $g(x)$ is the minimum polynomial of α, $g(x)$ divides $f(x)$. Hence

$$C = \{f(x) \in \mathbf{F}_2[x]/(x^7 - 1) : f(x) \text{ is divisible by } g(x)\}.$$

Thus C has generator polynomial $g(x)$. It now follows from Example 4.12(3) that C is equivalent to the Hamming code of length 7. $\square$

BCH codes. BCH codes (named after Bose and Ray-Chaudhuri) are the generalization of cyclic Hamming codes in which the polynomials corresponding to codewords are required to have roots at several different powers of the primitive element α.

Definition 5.2. Let $t \in \mathbf{N}$ be given and let $2^r - 1 > 2t + 1$. Let $\alpha \in \mathbf{F}_{2^r}$ be a primitive root. The *BCH-code* with *design distance* $2t+1$ and length $n = 2^r - 1$ is the cyclic code C defined over $\mathbf{F}_2$ by

$$C = \{f(x) \in \mathbf{F}_2[x]/(x^n - 1) : f(\alpha) = f(\alpha^2) = \ldots = f(\alpha^{2t+1}) = 0\}.$$

Equivalently, using the same idea as in Theorem 4.14, we may define

$$C = \{(u_0, u_1, \ldots, u_{n-1}) \in \mathbf{F}_2^n : (u_0, u_1, \ldots, u_{n-1})K^{tr} = 0\}$$

where

$$K = \begin{pmatrix} 1 & \alpha & \alpha^2 & \ldots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \ldots & \alpha^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{2t} & a^{2(2t)} & \ldots & a^{2t(n-1)} \end{pmatrix}$$

Note that while K behaves like a parity matrix for C, it is not one in the strict sense of Definition 14.1, because the entries of K are not in $\mathbf{F}_2$.

Example 5.3. Let r, $t \in \mathbf{N}$ and let C be the BCH-code of length $2^r - 1$ with design distance $2t + 1$.

(1) If $r = 3$ and $t = 1$ then since the minimum polynomial of α^2 if $x^3 + x + 1$, as for α, it follows from Lemma 5.1 that C is the Hamming $[7, 4, 3]$-code.

(2) If $r = 3$ and $t = 2$ then

$$C = \{f(x) \in \mathbf{F}_2[x] : f(\alpha) = f(\alpha^3) = 0, \ \deg f \leq 6.\}$$

The minimum polynomials of α, α^2 and α^4 are $x^3 + x + 1$ and the minimum polynomial of α^3 is $x^3 + x^2 + 1$. Hence, C has generator polynomial

$$(x^3 + x + 1)(x^3 + x^2 + 1) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1.$$

In this case we merely succeed in giving a complicated construction of the repetition code. Note that the minimum distance of this code is 7, which is strictly more than the design distance 5.

(3) Let $r = 4$ and let $t = 2$. Let α be a root of $x^4 + x + 1$; this is a primitive polynomial so α has multiplicative order 15. The minimum polynomial of α^3 is $x^4 + x^3 + x^2 + x + 1$ so the BCH code for these parameters is cyclic with minimum polynomial $x^8 + x^7 + x^6 + x^4 + 1$. By Theorem 4.13 the code is

$$C = \{f(x)(x^8 + x^7 + x^6 + x^4 + 1) : f(x) \in \mathbf{F}_2[x], \ \deg f \leq 6\}$$

and so $\dim C = 15 - 8 = 7$. Since the generator polynomial has weight 5, the code contains words of weight 5. Hence C has minimum distance at most 5. It therefore follows from Theorem 5.4 below that C is a 2-error correcting binary $[15, 7, 5]$-code.

For comparison, the Hamming code of length 15 is a $[15, 11, 3]$-code. The Hadamard codes constructed in Question 2 of Sheet 6 are linear: taking a linear Hadamard $(16, 32, 8)$ and puncturing it in its final position gives a $[15, 5, 7]$-code.

Working with linear codes of length 15 we have the following table of good codes.

1-error correcting (Hamming)	size $2^{11} = 2048$	[15,11,3]
2-error correcting (BCH)	size $2^7 = 128$	[15,7,5]
3-error corr. (punctured Hadamard)	size $2^5 = 32$	[15,5,7]

By the exercise below, the Hamming and punctured Hadamard codes are optimal. The BCH code has the largest possible size of a *linear* binary code of length 15 and minimum distance 5, but there is a larger non-linear code, of size 256. (See Theorem 7.4.5 in Van Lint, *Introduction to coding theory.*)

Exercise: Show from the Hamming Packing Bound that $A_2(15,3) = 2^{11}$. Use Theorem 11.6 and Corollary 9.7 in the main notes to show that $A_2(15,7) = 2^5$.

Theorem 5.4. *Let C be the BCH code of length $n = 2^r - 1$ and design distance $2t+1$. Then C is a cyclic code of dimension $\geq n - rt$ and minimum distance $\geq 2t+1$.*

Outline proof. The product of the minimum polynomials of α^i for $1 \leq i \leq 2t+1$ is a generator polynomial for C. Each minimum polynomial has degree $\leq$, since $\alpha \in \mathbf{F}_2^r$. Moreover, if $f \in \mathbf{F}_2[x]$ then $f(\alpha) = 0$ if and only if $f(\alpha^2) = 0$. Hence the minimum polynomial of α^{2i} is the same as the minimum polynomial of α^i, and so we need only consider odd powers of α. It follows that g has degree $\leq rt$ and so the dimension of C is at least $n - rt$, by Theorem 4.11.

To show that the minimum distance of C is at least $2t+1$ it suffices, by the same idea used in Question 10 on Sheet 9, to show that no $2t$ columns of the matrix K following Definition 5.2 are linearly dependent over $\mathbf{F}_2$. This can be proved using Lemma 1.7 on polynomial interpolation, or by using the Vandermonde determinant. $\square$

BCH codes can be decoded by syndrome decoding, as described at the end of §4. There are also more efficient decoders based on the Berlekamp–Massey algorithm.