Chapter 7

Cyclic codes (2019-03-18)

Synopsis. We would like to study cyclic codes using a structure which is richer than just a vector space. An appropriate structure is a ring (which is a vector space over a field). But the ring \( \mathbb{F}_q[x] \) of polynomials is too big (infinite). We introduce the finite ring \( R_n \) of polynomials where multiplication is defined modulo the polynomial \( x^n - 1 \). In particular, cyclic codes coincide with ideals of the ring \( R_n \). We prove that every cyclic code \( C \) is an ideal generated by the generator polynomial of \( C \). We also define a check polynomial of \( C \). We can classify cyclic codes of length \( n \) by listing all monic divisors of the polynomial \( x^n - 1 \). We learn how to write a generator matrix of a cyclic code with a given generator polynomial. All types of codes we have considered so far will arise as cyclic codes. Finally, we define two new codes called Golay codes and give a complete classification of perfect codes over alphabets of prime power size, up to parameter equivalence.

Definition (cyclic shift, cyclic code). For a vector \( \overline{a} = (a_0, a_1, \ldots, a_{n-1}) \in \mathbb{F}_q^n \), we denote \( s(\overline{a}) = (a_{n-1}, a_0, \ldots, a_{n-2}) \) and call the vector \( s(\overline{a}) \) the cyclic shift of \( \overline{a} \).

A cyclic code in \( \mathbb{F}_q^n \) is a linear code \( C \) such that

\[
\overline{a} \in C \quad \implies \quad s(\overline{a}) \in C.
\]

Equivalently, a cyclic code is a linear code \( C \) such that \( s(C) = C \).

Remark: We can iterate the cyclic shift, so if a cyclic code \( C \) contains the vector \((a_0, a_1, \ldots, a_{n-1})\), then \( C \) also contains \((a_{n-2}, a_{n-1}, a_0, \ldots, a_{n-3}), \ldots, (a_1, \ldots, a_{n-1}, a_0)\).
Vectors as polynomials

To study cyclic codes, we will identify vectors of length \( n \) with polynomials of degree \(< n\) with coefficients in the field \( \mathbb{F}_q \):

\[
a = (a_0, a_1, \ldots, a_{n-1}) \mapsto a(x) = a_0 + a_1x + \ldots + a_{n-1}x^{n-1} \in \mathbb{F}_q[x]
\]

Here \( \mathbb{F}_q[x] \) is the ring of polynomials in one variable, \( x \), with coefficients in \( \mathbb{F}_q \).

**Example (\( E_3 \)).** Consider the binary even weight code

\[
E_3 = \{000, 110, 011, 101\} \subseteq \mathbb{F}_2^n.
\]

It is a linear code, and is closed under the cyclic shift: 000 is invariant under the cyclic shift, and 110 \( \rightarrow \) 011 \( \rightarrow \) 101. Hence \( E_3 \) is a cyclic code. It consists of the following code polynomials:

<table>
<thead>
<tr>
<th>Codevector</th>
<th>Code polynomial</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>1 + x</td>
<td></td>
</tr>
<tr>
<td>011</td>
<td>( x + x^2 )</td>
<td>= ( x(1 + x) )</td>
</tr>
<tr>
<td>101</td>
<td>( 1 + x^2 )</td>
<td>= ( (1 + x)(1 + x) )</td>
</tr>
</tbody>
</table>

Note that multiplication of polynomials is an extremely important operation on \( \mathbb{F}_q[x] \) which is not “visible” in \( \mathbb{F}_q^n \). Our goal in this chapter will be to make sense of this operation in coding theory — in particular, we will explain the mysterious fact that all the code polynomials of \( E_3 \) are multiples of \( 1 + x \).

The polynomial ring as an algebra

First of all, multiplication makes \( \mathbb{F}_q[x] \) a ring, as per the following

**Definition** (commutative ring). A (commutative) **ring** is an abelian group \((R, +)\) equipped with an extra operation \( \times \) (multiplication), such that for all \( a, b, c \in R \) (writing \( ab \) for \( a \times b \)):

\[
(ab)c = a(bc); \quad ab = ba; \quad \exists 1 \in R: \ 1 \neq 0, 1a = a; \quad a(b + c) = ab + ac.
\]

**Remark:** “commutative” refers to the commutativity of multiplication, \( ab = ba \).

**Remark:** the axiom “\( \forall a \in R \setminus \{0\} \exists a^{-1} \in R: aa^{-1} = 1 \)” is not part of the axioms of a commutative ring. Rings which satisfy this additional axiom (the inverse axiom) are **fields**. The ring \( \mathbb{F}_q[x] \) is not a field!

Besides being a ring, \( \mathbb{F}_q[x] \) is also a vector space over the field \( \mathbb{F}_q \). Combining the ring and the vector space structures together, we obtain the following structure:
**Definition** (algebra over the field $\mathbb{F}_q$). An algebra $A$ over the field $\mathbb{F}_q$ is a ring which is also a vector space over $\mathbb{F}_q$, such that

$$\forall \lambda \in \mathbb{F}_q, \ a, b \in A, \quad (\lambda a)b = a(\lambda b) = \lambda(ab).$$

That is, scaling one of the factors in a product $ab$ by $\lambda$ has the same effect as scaling the whole product.

**Remark.** Many mathematical objects are algebras over fields (not necessarily finite fields $\mathbb{F}_q$): for example,

- the algebra $K[x]$ of all polynomials in $x$ over a field $K$;
- the algebra $M_n(K)$ of all $n \times n$ matrices with entries in $K$;
- the algebra, over $\mathbb{R}$, of all functions $f : \mathbb{R} \to \mathbb{R}$,

etc. In the next Chapter we will consider a new example, a boolean algebra over the field $\mathbb{F}_2$.

**The Division Theorem for polynomials**

The algebra $\mathbb{F}_q[x]$ is not a field, hence in general we cannot divide $f(x)$ by $g(x)$ and expect to get a polynomial. However, just as the ring $\mathbb{Z}$ of integers, the algebra $\mathbb{F}_q[x]$ has an extra operation called **division with remainder**, as per the following

**Theorem 7.1** (Division Theorem for polynomials). For all $f(x) \in \mathbb{F}_q[x]$, $g(x) \in \mathbb{F}_q[x] \setminus \{0\}$, there exist unique $Q(x), r(x) \in \mathbb{F}_q[x]$ such that

$$f(x) = g(x)Q(x) + r(x) \quad \text{and} \quad \deg r(x) < \deg g(x) \quad \text{(possibly } r(x) = 0).$$

In this case the polynomial $Q(x)$ is the **quotient**, and $r(x)$ the **remainder**, of $f(x)$ when divided by $g(x)$.

We will **not** prove the Division Theorem but we will note and use the practical algorithm for finding the quotient and the remainder, known as **long division of polynomials**.

Example of long division: divide $x^5 + 1$ by $x^2 + x + 1$ in $\mathbb{F}_2[x]$, finding the quotient and the remainder,

$$x^2 + x + 1 \overline{) x^5 + x^2 + 1} \quad \text{(quotient)}$$

$$\begin{align*}
-x^5 &- \hspace{1cm} \text{(dividend)} \\
- \frac{-x^5 + x^4 + x^3}{x^4 + x^3} &+ 1 \\
- \frac{-x^4 + x^3 + x^2}{x^2 + x + 1} &+ 1 \\
- \frac{-x^2 + x + 1}{x} &\text{ (remainder)}
\end{align*}$$
Hence \( x^5 + 1 = (x^2 + x + 1)Q(x) + r(x) \) in \( \mathbb{F}_2[x] \), with \( Q(x) = x^3 + x^2 + 1 \) and \( r(x) = x \). This example shows long division of polynomials over \( \mathbb{F}_2 \). Division by a fixed binary polynomial is widely implemented in electronic circuits at hardware level, by means of shift registers. We will soon see why such implementations are needed.

**The finite algebra \( R_n \)**

The Division Theorem for polynomials allows us to construct the following finite algebra which will be of direct relevance to cyclic codes.

**Definition** (the algebra \( R_n \)). As a vector space, \( R_n \) consists of all polynomials of degree less than \( n \) in \( \mathbb{F}_q[x] \). Multiplication in \( R_n \) is defined by

\[
f(x), g(x) \in R_n \implies \text{remainder of } f(x)g(x) \text{ when divided by } x^n - 1,
\]

and is known as **multiplication modulo** \( x^n - 1 \).

**Fact.** The above operations make \( R_n \) an algebra over the field \( \mathbb{F}_q \).

We will not be verifying the axioms here but note that this construction is similar to \( \mathbb{Z}_m = \{0, 1, \ldots, m-1\} \) where the operations are modulo \( m \).

**Corollary 7.2** (identification of \( \mathbb{F}_q^n \) with \( R_n \)). The space \( \mathbb{F}_q^n \) is identified with \( R_n \) via a bijective linear map which sends the vector \( \underline{a} = (a_0, a_1, \ldots, a_{n-1}) \) to the element \( a(x) = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1} \) of \( R_n \). The cyclic shift \( s(\underline{a}) \) of the vector \( \underline{a} \) corresponds to \( xa(x) \) in \( R_n \).

**Proof.** The map \( \mathbb{F}_q^n \to R_n, \underline{a} \mapsto a(x) \) is obviously a linear map and has inverse, \( a_0 + a_1 x + \cdots + a_{n-1} x^{n-1} \mapsto (a_0, a_1, \ldots, a_{n-1}) \), hence is bijective. Now,

\[
xa(x) = a_0 x + a_1 x^2 + \cdots + a_{n-1} x^n = a_{n-1} (x^n - 1) + a_{n-1} + a_0 x + \cdots + a_{n-2} x^{n-1}
\]

showing that the remainder of \( xa(x) \) modulo \( x^n - 1 \) is indeed \( a_{n-1} + a_0 x + \cdots + a_{n-2} x^{n-1} \), which corresponds to the vector \( s(\underline{a}) \) as claimed.

What do the cyclic codes become when viewed as subsets of \( R_n \)? More than just subspaces:

**Definition** (ideal). An **ideal** of an algebra \( R \) is a subspace \( I \subseteq R \) such that \( RI \subseteq I \).

**Lemma 7.3** (Prange, 1957). Cyclic codes \( C \subseteq \mathbb{F}_q^n \) are ideals of the algebra \( R_n \).
Cyclic codes (2019-03-18)

Proof. Let $C \subseteq \mathbb{F}_q^n$ be a cyclic code. By definition, $C$ is linear, so is a subspace $R_n$. Furthermore, $xC \subseteq C$ because $C$ is closed under the cyclic shift. Iterating, we obtain $x^2C = x(xC) \subseteq xC \subseteq C$, $x^3C \subseteq C$, $\ldots$ $x^{n-1}C \subseteq C$. Since $1, x, \ldots, x^{n-1}$ span $R_n$, this shows that $R_nC \subseteq C$ hence $C$ is an ideal. \hfill \Box

Cyclic codes in $\mathbb{F}_q^n$, or, the same, ideals of $R_n$, can be classified using the following notion.

Definition. We say that $g(x) \in \mathbb{F}_q[x]$ is a generator polynomial of a cyclic code $C$ if $g(x)$ is monic, and $C$ consists of all multiples of $g(x)$ of degree less than $n$. We say that the generator polynomial of the zero code, $\{0\}$, is $x^n - 1$.

(Monic means: the coefficient of the highest power of $x$ in $g$ is 1.)

Theorem 7.4. Every cyclic code $C$ has a generator polynomial $g(x)$. The generator polynomial of $C$ is unique and is a factor of $x^n - 1$ in $\mathbb{F}_q[x]$.

Proof. If $C = \{0\}$, by definition $x^n - 1$ is the unique generator polynomial of $\{0\}$ and is obviously a factor of itself. So we assume $C \neq \{0\}$.

Existence of $g(x)$: let $g(x) \in C$ be a non-zero polynomial of lowest degree. Make $g(x)$ monic by dividing it by its leading coefficient. We now claim that $g(x)$ is a generator polynomial of $C$.

All multiples of $g(x)$ are in $C$ because $C$ is an ideal. Vice versa, if $f(x) \in C$, apply the Division Theorem for polynomials to write $r(x) = f(x) - g(x)Q(x)$ where $\deg r(x) < \deg g(x)$. Being an ideal, $C$ is closed under “−” and under multiplication by $Q(x)$, so $r(x) \in C$. But $g(x)$ had lowest degree among non-zero polynomials with this property, hence $r(x) = 0$ and $f(x) = g(x)Q(x)$. We proved that all polynomials in $C$ are multiples of $g(x)$.

$g(x)$ is a factor of $x^n - 1$: by the Division Theorem, write $r(x) = (x^n - 1) - g(x)Q(x)$ where $\deg r(x) < \deg g(x)$. Taking both sides modulo $x^n - 1$, we obtain $r(x) = g(x)(-Q(x))$ in $R_n$. Since $g(x) \in C$ and $C$ is an ideal, $r(x) \in C$, so again since $g(x)$ is a non-zero polynomial of lowest degree in $C$, we have $r(x) = 0$ and $x^n - 1 = g(x)Q(x)$, showing that $g(x)$ is a factor of $x^n - 1$.

Uniqueness: let $g_1(x) \in C$ be another generator polynomial, then by definition $g_1(x)$ must be a multiple of $g(x)$, and $g(x)$ must be a multiple of $g_1(x)$, so, given that both polynomials are monic, $g_1(x) = g(x)$. \hfill \Box

Definition (check polynomial). Let $g(x)$ be the generator polynomial of a cyclic code $C$. The polynomial $h(x)$ such that $g(x)h(x) = x^n - 1$ is called the check polynomial of $C$. If $\deg g(x) = r$, then $\deg h(x) = n - r$, and $h$ is monic.
Example \((E_3\) (continued)). Recall that the code \(E_3\), viewed as an ideal of \(R_3\), consists of polynomials \(0, 1 + x, x + x^2 = x(1 + x)\) and \(1 + x^2 = (1 + x)^2\), of degree \(< 3\). The monic code polynomial of lowest degree is \(1 + x\). Observe that all the code polynomials are multiples of \(1 + x\) which is therefore the generator polynomial of the cyclic code \(E_3\).

Error detection by a cyclic code

Theorem 7.4 means that if \(C\) is a cyclic code, there is no need to store a check matrix for error detection. To determine whether the received vector \(y\) is a codevector, divide the polynomial \(y(x)\) by the generator polynomial \(g(x)\); the remainder is 0, if and only if \(y \in C\). This is how error detection is implemented in practice for binary codes (e.g., in Ethernet networks). Long division by \(g(x)\) is implemented by circuitry.

Nevertheless, for theoretical purposes we would like to have generator and check matrices for a cyclic code with a given generator polynomial.

**Theorem 7.5** (a generator matrix and a check matrix for a cyclic code). Let \(C \subseteq \mathbb{F}_q^n\) be a cyclic code with generator polynomial \(g(x) = g_0 + g_1 x + \ldots + g_r x^r\) and check polynomial \(h(x) = h_0 + h_1 x + \ldots + h_k x^k\). We note that \(k = n - r\) and that \(g_r = h_k = 1\) (the polynomials are monic). Then:

- the vector \(\underline{g} = [g_0 \ g_1 \ \ldots \ g_r \ 0 \ \ldots \ 0] \in \mathbb{F}_q^n\) which corresponds to \(g(x)\), and its next \(k - 1\) cyclic shifts, form a generator matrix for \(C\):

\[
G = \begin{bmatrix}
g_0 & g_1 & \ldots & g_{r-1} & 1 & 0 & \ldots & 0 \\
0 & g_0 & g_1 & \ldots & g_{r-1} & 1 & \ddots & 0 \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\
0 & \ldots & 0 & g_0 & \ldots & \ldots & g_{r-1} & 1 \\
\end{bmatrix} \quad \text{\(k\) rows};
\]

- the vector corresponding to \(h(x)\) written backwards, i.e., the vector \(\underline{h}_{\text{backwards}} = [0 \ \ldots \ 0 \ h_k \ h_{k-1} \ \ldots \ h_0]\), and its next \(r - 1\) shifts to the left form a check matrix for \(C\):

\[
H = \begin{bmatrix}
0 & \ldots & 0 & 1 & \ldots & \ldots & h_1 & h_0 \\
0 & \ddots & 1 & h_{k-1} & \ldots & \ldots & h_1 & h_0 \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\
1 & h_{k-1} & \ldots & \ldots & h_1 & h_0 & 0 & \ldots & 0 \\
\end{bmatrix} \quad \text{\(r\) rows}.
\]

**Proof.** By definition, \(C = \{u(x)g(x) : \deg(u(x)g(x)) < n\} = \{u(x)g(x) : \deg u(x) < n - r = k\}\). So a basis of \(C\) is given by \(g(x), xg(x), \ldots, x^{k-1}g(x)\). Written as vectors, these are exactly the cyclic shifts of \(g\) listed above. We proved that \(G\) is a generator matrix for \(C\).
Now we write down the inner product of the first row of $G$ and the first row of $H$:
\[ g \cdot h_{\text{backwards}} = g_0h_{n-1} + g_1h_{n-2} + \ldots + g_{n-1}h_0 \]
is precisely the coefficient of $x^{n-1}$ in the polynomial $g(x)h(x)$. Yet this polynomial is $x^n - 1$ hence $g \cdot h_{\text{backwards}} = 0$. More generally, the inner product of the $i$th row of $G$ and the $j$th row of $H$ is the coefficient of $x^{n-1}$ in the polynomial $(x^{i-1}g(x))(x^{j-1}h(x)) = x^{n+i+j-2} - x^{i+j-2}$ which does not contain $x^{n-1}$ because $i + j - 2 \leq n - 2$. Hence the $r$-dimensional code generated by $H$ is orthogonal to $C$ and is contained in $C^\perp$; but $\text{dim } C^\perp = r$ so $H$ generates $C^\perp$.

**Remark.**
1. This is not the only generator matrix (resp., check matrix) for $C$. As we know, a generator matrix is not unique. Moreover, these matrices are not usually in standard form. Note that a generator polynomial of $C$ is unique.
2. Strictly speaking, Theorem 7.5 does not show that $C^\perp$ is also a cyclic code, but this is proved in the exercises to this chapter.

**Example.** Use Theorem 7.4 and Theorem 7.5 to find all the cyclic binary codes of length 3.

**Solution.** Generator polynomials are monic factors of $x^n - 1$ in $\mathbb{F}_q[x]$. The first step is to factorise $x^n - 1$ into irreducible monic polynomials in $\mathbb{F}_q[x]$. A polynomial is irreducible if it cannot be written as a product of two polynomials of positive degree. 

Note that the polynomial $x^n - 1$ is not irreducible in $\mathbb{F}_q[x]$, for all $n > 1$ and for all $q$. Indeed, $x^n - 1 = (x - 1)(x^{n-1} + \ldots + x + 1)$.

We work over the field $\mathbb{F}_2$ and observe:
\[ x^3 - 1 = (x - 1)(x^2 + x + 1). \]
The polynomial $x - 1 = x + 1$ is irreducible, because it is of degree 1.

Can we factorise the polynomial $x^2 + x + 1$ in $\mathbb{F}_2[x]$? If we could, we would have a factorisation $(x + a)(x + b)$. But then $ab = 1$ which means $a = b = 1$ in $\mathbb{F}_2$. Note that $(x + 1)^2 = x^2 + 1$ in $\mathbb{F}_2[x]$. We have shown that $x^2 + x + 1$ is irreducible in $\mathbb{F}_2[x]$.

So the possible monic factors of $x^3 - 1$ in $\mathbb{F}_2[x]$ are:
\[ 1; \quad 1 + x; \quad 1 + x + x^2; \quad 1 + x^3. \]

We can now list all the cyclic codes in $\mathbb{F}_2^3$ as ideals of $R_3$ generated by each of the above polynomials. For each code we give a generator matrix $G$, state the minimum distance $d$ and a well-known name of the code, and point out its dual code (which is also cyclic).
• $g(x) = 1$, $G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ which corresponds to the trivial binary code of length 3: 
\{000, 100, 010, 001, 110, 101, 011, 111\} = $\mathbb{F}_2^3$ with $d = 1$. The dual code of $\mathbb{F}_2^3$ is the zero code (see below).

• $g(x) = 1 + x$, $G = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$. This is \{000, 110, 011\} = $E_3$, the binary even weight code of length 3 which has $d = 2$. The dual of $E_3$ is $\text{Rep}(3, 2)$ (see below).

• $g(x) = 1 + x + x^2$, $G = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$. This is \{000, 111\} = $\text{Rep}(3, 2)$, the binary repetition code of length 3 with $d = 3$. This code is $(E_3)^\perp$.

• $g(x) = 1 + x^3$. Theorem 7.5 returns matrix $G$ with $k = 3 - 3 = 0$ rows, $G = \begin{bmatrix} \end{bmatrix}$. And indeed, by definition $1 + x^3$ is the generator polynomial of the zero code, \{000\}, which has empty generator matrix. It is a useless code but formally it is a linear and cyclic code, so we have to allow it for reasons of consistency. The minimum distance of the zero code is undefined. This code is $(\mathbb{F}_2^3)^\perp$.

The following two remarks aim to highlight useful features of cyclic codes.

**Remark.** Recall that:

• the only way to specify a general non-linear code in $\mathbb{F}_q^n$ is to list all the codewords, which consist of a total of $q^k \times n$ symbols;

• a linear code can be specified by a generator matrix, which has $k \times n$ entries;

• a cyclic code can be specified in an even more compact way — by giving its generator polynomial, which corresponds to a single codeword! We only need to specify $n - k$ coefficients of the generator polynomial (its degree is $n - k$ and its leading coefficient is 1).

**Remark.** What do we use check matrices for? For example, to find the minimum distance of a linear code.

**Strategy of searching for interesting/perfect/etc codes:** Look for divisors of $x^n - 1$ and hope that the cyclic codes they generate have a large minimum distance.

**Example 7.6** (two new perfect codes — the Golay codes). The following two codes were found by Marcel Golay in 1949. They are known as the binary Golay code and the ternary Golay code, respectively.
The binary Golay code $G_{23}$

In $\mathbb{F}_2[x]$, $x^{23} - 1 = g(x)h(x)$, where $g(x) = x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1$ and $h(x) = x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^5 + x^2 + 1$. (Exercise: check this!)

Denote the cyclic code generated by $g$ by $G_{23}$.

**Claim.** $G_{23}$ is a perfect $[23, 12, 7]_2$-code.

**Partial proof.** The code is binary of length $n = 23$ by construction. The dimension is $\deg h = 12$. We will not prove that $d = 7$ but will only observe that $d \leq 7$: indeed, the vector $g \in G_{23}$ is 10101110001100000000000, of weight 7.

To show that $G_{23}$ is perfect, write the Hamming bound for a binary code in logarithmic form: $k \leq n - \log_2 \left( \binom{n}{0} + \cdots + \binom{n}{t} \right)$. Here $t = \lceil (7-1)/2 \rceil = 3$ so the expression under the logarithm is $1 + \binom{23}{1} + \binom{23}{2} + \binom{23}{3} = 1 + 23 + 23 \times \frac{22}{2} + 23 \times \frac{22}{2} \times \frac{21}{3} = 1 + 23(1 + 11 + 77) = 2048$. One has $12 = 23 - \log_2 2048$ hence the Hamming bound is attained.

**Trivia**

The code $G_{23}$ was used by Voyager 1 & 2 spaceships (NASA, Jupiter and Saturn, 1979–81). More precisely, the Golay code was used in a version extended to 24 bits by adding a parity check bit to each codeword. This increased the minimum distance to 8 thereby improving error detection (not affecting error correction). But the extended 24 bit code $G_{24} := \hat{G}_{23}$ is no longer perfect.

The ternary Golay code $G_{11}$

In $\mathbb{F}_3[x]$, $x^{11} - 1 = g(x)h(x)$ where $g(x) = x^5 + x^4 + 2x^3 + x^2 + 2$ and $h(x) = x^6 + 2x^5 + 2x^3 + 2x^2 + x^2 + 1$.

The ternary Golay code $G_{11}$ is the cyclic code of length 11 generated by $g$. It is an $[11, 6, 5]_3$ code. It is perfect (see the calculation of the Hamming bound in the “Football Pool” exercise to this Chapter).

**Historical notes**

Golay found his two perfect codes in 1949, before cyclic codes were discovered. He defined the codes $G_{23}$ and $G_{11}$ by writing their check matrices. Crucially, Golay observed that $\binom{23}{0} + \binom{23}{1} + \binom{23}{2} + \binom{23}{3}$ is a power of two. From the proof of perfectness above one can see that the condition $\binom{n}{0} + \cdots + \binom{n}{t} = 2^r$ is necessary for the existence of a perfect $t$-error-correcting binary code of length $n$. This condition is not sufficient: e.g., in his
1949 paper Golay also observes that \( \binom{90}{0} + \binom{90}{1} + \binom{90}{2} = 2^{12} \) but this does not lead to any perfect binary code of length 90.

Amazingly, Golay’s 1949 paper where he constructs all the Hamming codes and the two Golay codes, is barely half a page long. A copy of the paper is displayed on the course website.

Now we can state the classification theorem about perfect codes. It was proved in 1973, more than twenty years since Golay gave a conjecturally complete list of perfect codes in alphabets of prime power size. We will not give its proof here, but you should learn the statement of the theorem.

**Definition** (parameter equivalence). We say that two codes are *parameter equivalent*, if they both are \([n, k, d]_q\)-codes for some \(n, k, d\) and \(q\).

**Theorem 7.7** (Tietäväinen – van Lint, 1973; classification of perfect codes where \(q\) is a prime power). Let \(q\) be a power of a prime number. A perfect \([n, k, d]_q\)-code is parameter equivalent to one of the following:

- a trivial code: \(n\) arbitrary, \(k = n\), \(d = 1\), \(q\) any prime power;
- a binary repetition code of odd length: \(n\) odd, \(k = 1\), \(d = n\), \(q = 2\);
- a Hamming code \(\text{Ham}(r, q)\): \(n = \frac{q^r - 1}{q - 1}\), \(k = n - r\), \(d = 3\), \(q\) any prime power;
- the Golay code \(G_{23}\), which is a \([23, 12, 7]_2\)-code;
- the Golay code \(G_{11}\) which is an \([11, 6, 5]_3\)-code.

**Remark.** Classification of perfect codes over alphabets of size not equal to a prime power is, in general, an open problem.