Ho Chi Minh University of Science HCMUS

Update: April 16, 2009

Finite fields: some applications

Michel Waldschmidt [2]

Second course

April 10, 2009

1.4 Proof of the irreducibility of the cyclotomic polynomial Φ_n for any $n \geq 1$.

Proof of Theorem 7. Let $f \in \mathbf{Z}[X]$ be an irreducible factor of Φ_n and let g satisfy $fg = \Phi_n$. Our goal is to prove $f = \Phi_n$ and $g = 1$.

Since Φ_n is monic, the same is true for f and g. Let ζ be a root of f in $\mathbf{C}$ and let p be a prime number which does not divide n. Since ζ^p is a primitive n-th root of unity, it is a zero of Φ_n.

The first and main step of the proof is to check that $f(\zeta^p) = 0$. If ζ^p is not a root of f, then it is a root of g. We assume $g(\zeta^p) = 0$ and we shall reach a contradiction.

Since f is irreducible, f is the minimal polynomial of ζ, hence from $g(\zeta^p) = 0$ we infer that $f(X)$ divides $g(X^p)$. Write $g(X^p) = f(X)h(X)$ and consider the morphism Ψ_p of reduction modulo p already introduced in (9):

$$\Psi_p : \mathbf{Z}[X] \longrightarrow \mathbf{F}_p[X].$$

Denote by F, G, H the images of f, g, h. Recall that $fg = \Phi_n$ in $\mathbf{Z}[X]$, hence $F(X)G(X)$ divides $X^n - 1$ in $\mathbf{F}_p[X]$. The assumption that p does not divide n implies that $X^n - 1$ has no square factor in $\mathbf{F}_p[X]$.

Let $P \in \mathbf{Z}[X]$ be an irreducible factor of F. From $G(X^p) = F(X)H(X)$ it follows that $P(X)$ divides $G(X^p)$. But $G \in \mathbf{F}_p[X]$, hence (see Lemma 17) $G(X^p) = G(X)^p$ and therefore P divides $G(X)$. Now P^2 divides the product FG, which is a contradiction.

We have checked that for any root ζ of f in $\mathbf{C}$ and any prime number p which does not divide n, the number ζ^p is again a root of f. By induction on the number of prime factors of m, it follows that for any integer m with

[2] This text is accessible on the author's web site
http://www.math.jussieu.fr/~miw/coursVietnam2009.html

$\gcd(m,n) = 1$ the number ζ^m is a root of f. Now f vanishes at all the primitive roots of unity, hence $f = \Phi_n$ and $g = 1$.

$\square$

2 Error correcting codes

2.1 Preliminary definitions

A *code* of *length* n on a finite alphabet A with q elements is a subset $\mathcal{C}$ of A^n. A *word* is an element of A^n, a *codeword* is an element of $\mathcal{C}$.

A *linear code* over a finite field $\mathbf{F}_q$ of length n and *dimension* r is a $\mathbf{F}_q$–vector subspace of $\mathbf{F}_q^n$ of dimension r (such a code is also called a (n,r)–code). A subspace $\mathcal{C}$ of $\mathbf{F}_q^n$ of dimension r can be described by giving a basis $e_1,\dots,e_r$ of $\mathcal{C}$ over $\mathbf{F}_q$, so that

$$\mathcal{C} = \{m_1e_1 + \cdots + m_re_r \ ; \ (m_1,\dots,m_r) \in \mathbf{F}_q^r\}.$$

An alternative description of a subspace $\mathcal{C}$ of $\mathbf{F}_q^n$ of codimension $n-r$ is by giving $n-r$ linearly independent linear forms $L_1,\dots,L_{n-r}$ in n variables $\underline{x} = (x_1,\dots,x_n)$ with coefficients in $\mathbf{F}_q$, such that

$$\mathcal{C} = \ker L_1 \cap \cdots \cap \ker L_{n-r}.$$

The sender replaces his message $(m_1,\dots,m_r) \in \mathbf{F}_q^r$ of length r by the longer message $m_1e_1 + \cdots + m_re_r \in \mathcal{C} \subset \mathbf{F}_q^n$ of length n. The receiver checks whether the message $\underline{x} = (x_1,\dots,x_n) \in \mathbf{F}_q^n$ belongs to $\mathcal{C}$ by computing the $n-r$–tuple $\underline{L}(\underline{x}) = (L_1(\underline{x}),\dots,L_{n-r}(\underline{x})) \in \mathbf{F}_q^{n-r}$. If there is no error during the transmission, then $\underline{x} \in \mathcal{C}$ and $L_1(\underline{x}) = \cdots = L_{n-r}(\underline{x}) = 0$. On the opposite, if the receiver observes that some $L_i(\underline{x})$ is non–zero, he knows that the received message has at least one error. The message with was sent was an element $\underline{c}$ of the code $\mathcal{C}$, the message received $\underline{x}$ is not in $\mathcal{C}$, the error is $\underline{\epsilon} = \underline{x} - \underline{c}$. The values of $\underline{L}(\underline{x})$ may enable him to correct the errors in case there are not too many of them. We only give examples today. For simplicity we take $q = 2$: we consider *binary codes*.

2.2 Examples

Trivial codes of length n are $\mathcal{C} = \{0\}$ of dimension 0 and $\mathcal{C} = \mathbf{F}_q^n$ of dimension n.

The two first examples below are *repetition codes*. The next one is a *parity bit code* detecting one error. The following ones use the parity bit idea but are 1–error correcting codes.

Example 22. $n=2$, $r=1$, rate $=1/2$, *detects one error.*

$$\mathcal{C}=\{(0,0),\ (1,1)\},\qquad e_1=(1,1),\qquad L_1(x_1,x_2)=x_1+x_2.$$

Example 23. $n=3$, $r=1$, rate $=1/3$, *corrects one error.*

$$\mathcal{C}=\{(0,0,0),\ (1,1,1)\},\qquad e_1=(1,1,1),$$

$$L_1(\underline{x})=x_1+x_3,\ L_2(\underline{x})=x_2+x_3.$$

If the message which is received is correct, it is either $(0,0,0)$ or $(1,1,1)$, and the two numbers $L_1(\underline{x})$ and $L_2(\underline{x})$ are 0 (in $\mathbf{F}_2$). If there is exactly one mistake, then the message which is received is either one of

$$(0,0,1),\ (0,1,0),\ (1,0,0),$$

or else one of

$$(1,1,0),\ (1,0,1),\ (0,1,1).$$

In the first case the message which was sent was $(0,0,0)$, in the second case it was $(1,1,1)$.

A message with a single error is obtained by adding to a codeword one of the three possible errors

$$(1,0,0),\ (0,1,0),\ (0,0,1).$$

If the mistake was on x_1, which means that $\underline{x}=\underline{c}+\underline{\epsilon}$ with $\underline{\epsilon}=(1,0,0)$ and $\underline{c}\in\mathcal{C}$ a codeword, then $L_1(\underline{x})=1$ and $L_2(\underline{x})=0$. If the mistake was on x_2, then $\underline{\epsilon}=(0,1,0)$ and $L_1(\underline{x})=0$ and $L_2(\underline{x})=1$. Finally if the mistake was on x_3, then $\underline{\epsilon}=(0,0,1)$ and $L_1(\underline{x})=L_2(\underline{x})=1$. Therefore the three possible values for the pair $\underline{L}(\underline{x})=(L_1(\underline{x}),L_2(\underline{x}))$ other than $(0,0)$ correspond to the three possible positions for a mistake. We shall see that this is a perfect one error correcting code.

Example 24. $n=3$, $r=2$, rate $=2/3$, *detects one error.*

$$\mathcal{C}=\{(m_1,m_2,m_1+m_2)\ ;\ (m_1,m_2)\in\mathbf{F}_2^2\}$$

$$e_1=(1,0,1),\ e_2=(0,1,1),\qquad L_1(x_1,x_2,x_3)=x_1+x_2+x_3.$$

This is the easiest example of the *bit parity check.*

Example 25. $n = 5$, $r = 2$, rate $= 2/5$, *corrects one error.*

$$\mathcal{C} = \left\{(m_1, m_2, m_1, m_2, m_1 + m_2)\ ;\ (m_1, m_2) \in \mathbf{F}_2^2\right\}$$

$$e_1 = (1, 0, 1, 0, 1),\ e_2 = (0, 1, 0, 1, 1),$$

$$L_1(\underline{x}) = x_1 + x_3,\ L_2(\underline{x}) = x_2 + x_4,\ L_3(\underline{x}) = x_1 + x_2 + x_5,$$

The possible values for the triple $\underline{L}(\underline{x})$ corresponding to a single error are displayed in the following table.

$\underline{x}$	x_1	x_2	x_3	x_4	x_5
$\underline{L}(\underline{x})$	$(1, 0, 1)$	$(0, 1, 1)$	$(1, 0, 0)$	$(0, 1, 0)$	$(0, 0, 1)$

Therefore when there is a single error, the value of $\underline{L}(\underline{x})$ enables one to correct the error.

One may observe that a single error will never produce the triple $(1, 1, 0)$ nor $(1, 1, 1)$ for $\underline{L}(\underline{x})$: there are 8 elements $\underline{x} \in \mathbf{F}_2^5$ which cannot be received starting from a codeword and adding at most one mistake, namely $(x_1, x_2, x_1 + 1, x_2 + 1, x_5)$, with $(x_1, x_2, x_5) \in \mathbf{F}_2^3$.

Example 26. $n = 6$, $r = 3$, rate $= 1/2$, *corrects one error.*

$$\mathcal{C} = \left\{(m_1, m_2, m_3, m_2 + m_3, m_1 + m_3, m_1 + m_2)\ ;\ (m_1, m_2, m_3) \in \mathbf{F}_2^3\right\}$$

$$e_1 = (1, 0, 0, 0, 1, 1),\ e_2 = (0, 1, 0, 1, 0, 1),\ e_3 = (0, 0, 1, 1, 1, 0),$$

$$L_1(\underline{x}) = x_2 + x_3 + x_4,\ L_2(\underline{x}) = x_1 + x_3 + x_5,\ L_3(\underline{x}) = x_1 + x_2 + x_6.$$

The possible values for the triple $\underline{L}(\underline{x})$ corresponding to a single error are displayed in the following table.

$\underline{x}$	x_1	x_2	x_3	x_4	x_5	x_6
$\underline{L}(\underline{x})$	$(0, 1, 1)$	$(1, 0, 1)$	$(1, 1, 0)$	$(1, 0, 0)$	$(0, 1, 0)$	$(0, 0, 1)$

Therefore when there is a single error, the value of $\underline{L}(\underline{x})$ enables one to correct the error.

One may observe that a single error will never produce the triple $(1, 1, 1)$ for $\underline{L}(\underline{x})$: there are 8 elements $\underline{x} \in \mathbf{F}_2^5$ which cannot be received starting from a codeword and adding at most one mistake, namely:

$$(x_1, x_2, x_3, x_2 + x_3 + 1, x_1 + x_3 + 1, x_1 + x_2 + 1) \quad \text{with} \quad (x_1, x_2, x_3) \in \mathbf{F}_2^3.$$

Example 27 (Hamming Code of dimension 4 and length 7 over $\mathbf{F}_2$).

$n = 7$, $r = 4$, rate $= 7/4$, *corrects one error.*

$\mathcal{C}$ is the set of

$$(m_1, m_2, m_3, m_4, m_1 + m_2 + m_4, m_1 + m_3 + m_4, m_2 + m_3 + m_4) \in \mathbf{F}_2^7$$

where (m_1, m_2, m_3, m_4) ranges over $\mathbf{F}_2^4$. A basis of $\mathcal{C}$ is

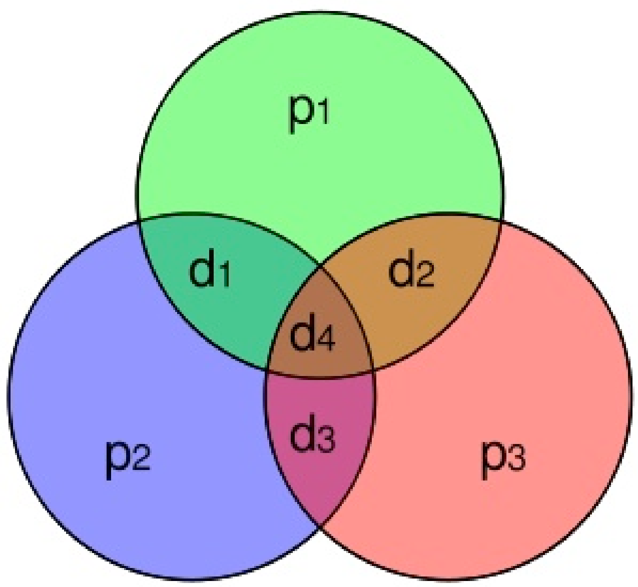
$$e_1 = (1, 0, 0, 0, 1, 1, 0), \quad e_2 = (0, 1, 0, 0, 1, 0, 1),$$
$$e_3 = (0, 0, 1, 0, 0, 1, 1), \quad e_4 = (0, 0, 0, 1, 1, 1, 1)$$

and $\mathcal{C}$ is also the intersection of the hyperplanes defined as the kernels of the linear forms

$$L_1(\underline{x}) = x_1+x_2+x_4+x_5,\ L_2(\underline{x}) = x_1+x_3+x_4+x_6,\ L_3(\underline{x}) = x_2+x_3+x_4+x_7.$$

This corresponds to the next picture from

http://en.wikipedia.org/wiki/Hamming_code



Hamming (7,4) code

The possible values for the triple $\underline{L}(\underline{x})$ corresponding to a single error are displayed in the following table.

$\underline{x}$	x_1	x_2	x_3	x_4	x_5	x_6	x_7
$\underline{L}(\underline{x})$	$(1,1,0)$	$(1,0,1)$	$(0,1,1)$	$(1,1,1)$	$(1,0,0)$	$(0,1,0)$	$(0,0,1)$

This table gives a bijective map between the set $\{1,2,3,4,5,6,7\}$ of indices of the unique wrong letter in the word $\underline{x}$ which is received with a single mistake on the one hand, the set of values of the triple

$$\underline{L}(\underline{x}) = \big(L_1(\underline{x}), L_2(\underline{x}), L_3(\underline{x})\big) \in \mathbf{F}_2^3 \setminus \{0\}$$

on the second hand. This is a *perfect* 1*–error correcting code.*