Universidad de Oviedo

Facultad de Ciencias

Grado en Matemáticas

Cuerpos Finitos y Aplicaciones

Samuel José García García

Junio de 2024

Tutora:

Consuelo Martínez López

Índice general

Introducción **3**

0.1. Introducción 3

0.2. Preliminares 5

1. Teoría de cuerpos finitos **7**

1.1. Estructura de los cuerpos finitos 7

1.2. Polinomios sobre cuerpos finitos 13

1.2.1. Polinomios irreducibles 13

1.2.2. Polinomios ciclotómicos 20

1.2.3. Polinomios primitivos 23

1.2.4. Polinomios linealizados: búsqueda de raíces y divisibilidad simbólica. 27

2. Factorización de polinomios sobre cuerpos finitos **33**

2.1. El Algoritmo de Berlekamp 34

2.2. Factorización en cuerpos finitos grandes 40

3. Aplicaciones **43**

3.1. Códigos Correctores 44

3.1.1. Motivación y definiciones. Códigos lineales. 44

3.1.2. Códigos Goppa 51

3.2. Criptografía de clave pública 58

3.2.1. Funciones de una vía: el problema del logaritmo discreto 59
3.2.2. Códigos en criptografía postcuántica: el esquema de McEliece . . . 63
3.3. Un ejemplo de cifrado de McEliece 66

Conclusiones **72**

Bibiografía **73**

A. Algoritmos **76**

Introducción y preliminares

0.1. Introducción

El interés por los cuerpos finitos se remonta hasta el siglo XVII. Si bien en aquel entonces el concepto de cuerpo no estaba propiamente definido, resultados clásicos de Teoría de Números pueden ser entendidos en este marco. El **Pequeño Teorema de Fermat**, propuesto en 1636 por **Pierre de Fermat** (1601-1665) y demostrado independientemente por **Leonard Euler** (1707-1783) en 1736 y **Leibniz** (1646-1716) en 1683, afirma que si p es un número primo y $a > 0$ es un número natural tal que $mcd(p, a) = 1$, entonces:

$$a^p \equiv a \pmod{p} \tag{1}$$

También se tiene el **Teorema de Wilson**, demostrado por **Lagrange** en 1771 y ya utilizado por **Alhacén** en el siglo XI. Si p es un número primo, entonces:

$$(p-1)! \equiv -1 \pmod{p} \tag{2}$$

Ambos resultados, al involucrar una identidad en congruencias módulo un número primo p, son fácilmente comprensibles si se atiende a la estructura de grupo multiplicativo del anillo de restos $\mathbb{Z}_p$. Esta familia de anillos constituye el primer ejemplo de cuerpos finitos, y son la base sobre la que se contruyen el resto.

Fue **Carl Friedrich Gauss** (1777-1855) quien introdujo la noción de congruencia propiamente dicha y adoptó el símbolo $\equiv$ para denotarla. En sus obras *Disquisitiones*

Arithmeticae y *Disquisitiones Generales de Congruentiis*, Gauss ahonda en estas cuestiones dándole un especial valor al rigor y a la demostración matemática. Fue capaz de probar que el polinomio $x^{p^n} - x$ es el producto de todos los polinomios mónicos irreducibles de grado $d \mid n$ con coeficientes módulo p, y dio una fórmula para obtener el número de dichos polinomios. Es decir, Gauss ya fue capaz de demostrar los resultados importantes de la Teoría de Cuerpos Finitos (ver la Proposición 1.9 y el Corolario 1.23) para la familia de cuerpos finitos primos.

Sin embargo, hoy sabemos que para cada primo p y natural n existe una extensión de $\mathbb{Z}_p$ con p^n elementos. En analogía con la introducción de la unidad imaginaria $i = \sqrt{-1}$ para poder resolver ecuaciones polinomiales en el nuevo cuerpo $\mathbb{C}$ que no poseían solución en los reales, **Évariste Galois** (1811-1832) tuvo la idea de introducir raíces *imaginarias* para ecuaciones de la forma

$$F(x) \equiv 0 \pmod{p}$$

donde F es un polinomio irreducible módulo p de grado n. Además, fue capaz de establecer muchas de las herramientas que utilizaremos en el trabajo: la relación entre las conjugadas de una raíz de un polinomio irreducible con el resto de raíces, o el hecho de que los elementos de la extensión construida son precisamente las raíces de $x^{p^n} - x$. Aquí se inicia el estudio de los Cuerpos Finitos como entidad propia, y en honor a Galois se suele denotar al cuerpo finito de q elementos como $GF(q)$ (por las siglas en inglés de *Galois Field*). **Eliakim Hastings Moore** (1862-1932) introdujo la noción abstracta de cuerpo finito, con la axiomatización que utilizamos hoy en día.

Es a lo largo del siglo XX cuando los cuerpos finitos, que parecían ser una de las ramas más puras de las matemáticas, se vuelven una pieza central en el desarrollo de dos disciplinas imprescindibles a día de hoy: la **Criptografía** y la **Teoría de Códigos Detectores y Correctores de Errores**.

En este trabajo recopilaremos los aspectos teóricos más importantes de los cuerpos finitos y nos haremos una idea precisa de su estructura subyacente. Para ello, será necesario

conocer en profundidad distintas clases de polinomios. A ello dedicamos el **Capítulo** 1. En este capítulo se recogerán también resultados adicionales necesarios para justificar las aplicaciones del **Capítulo** 3, donde daremos una muestra de cómo se utilizan los cuerpos finitos para construir códigos (en especial los Códigos Goppa en 3.1.2), funciones de una vía (a partir del problema del logaritmo discreto de 3.2.1), y criptosistemas resistentes frente a ordenadores cuánticos en 3.2.2. Estas aplicaciones tan solo pueden resultar útiles si se dispone de algoritmos eficientes de factorización de polinomios. Por ello, en el **Capítulo** 2 presentamos los **algoritmos de Berlekamp y Zassenhaus**, utilizados para esta labor. Por último, a modo de ejemplo, cifraremos y descifraremos un pequeño mensaje con el esquema de McEliece, ilustrando en un caso sencillo varios de los procesos descritos en el trabajo.

El texto primordial del que se han obtenido los contenidos del trabajo ha sido *Introduction to Finite Fields and their Applications*, de Rudolf Lidl y Harald Niederreiter [Lidl and Niederreiter, 1994]. Los apuntes históricos proceden del *Handbook of Finite Fields* [Mullen, 2013].

0.2. Preliminares

El trabajo se fundamentará principalmente en los contenidos impartidos en las asignaturas Álgebra lineal y Geometría, Álgebra I y II y Códigos Correctores de Errores y Criptografía.

En particular, haremos uso de la siguiente lista de conceptos y resultados.

1. La definición de las estructuras algebraicas básicas de grupo y anillo, así como ejemplos elementales de ellas. Recordamos que un anillo conmutativo $(K, +, \cdot)$ es un **cuerpo** si todo elemento no nulo de K posee un inverso multipicativo, esto es, si $(K\backslash\{0\}, \cdot)$ conforma a su vez un grupo, llamado **grupo multiplicativo** de K y denotado K^*.

2. **La estructura de grupos abelianos** finitamente generados y de grupos cíclicos.

3. **El teorema Chino de los Restos**, en el contexto de anillos de enteros y de polinomios;

4. Cuestiones básicas de polinomios sobre un cuerpo: por ejemplo, $f \in K[x]$ de grado n tiene a lo sumo n-raíces, y f tiene raíces múltiples en alguna extensión de K donde f se escinda si y solo si $mcd(f, f') \neq 1$

5. Resultados sobre **series formales de potencias**, en concreto, del hecho de que una serie formal sobre un anillo es inversible si y solo si su término independiente es inversible.

6. Propiedades básicas de teoría de cuerpos y de las **extensiones algebraicas** de cuerpos.

7. El teorema de **existencia y unicidad de cuerpos de escisión**.

8. El hecho de que $\mathbb{Z}_p$ es un cuerpo si y solo si p es primo.

9. El cuerpo primo de cualquier cuerpo de característica p es isomorfo a $\mathbb{Z}_p$.

10. En un anillo conmutativo R de característica p se cumple $(a+b)^p = a^p + b^p$ para todo $a, b \in R$. La aplicación $a \in R \to a^p \in R$ se denomina **automorfismo de Frobenius**.

En adelante, denotaremos a un cuerpo de $q = p^n$ elementos como $\mathbb{F}_q$. La ambigüedad de esta notación se verá salvada gracias al Teorema 1.4. Es útil notar que $\mathbb{F}_q$ puede verse como una extensión finita (por ejemplo, grado n) sobre su cuerpo primo $\mathbb{Z}_p$. Es decir, $\mathbb{F}_q$ es un $\mathbb{Z}_p$-espacio vectorial de dimensión n. Se sigue que necesariamente $q = p^n$: **todos los cuerpos finitos tienen un número de elementos potencia de primo**. Es más, el cardinal de cualquier extensión finita F de $\mathbb{F}_q$ es una potencia de q, al ser F un $\mathbb{F}_q$-espacio vectorial.

Capítulo 1

Teoría de cuerpos finitos

En la primera sección de este capítulo, nos interesamos por la estructura aditiva y multiplicativa de los cuerpos finitos, así como de su retículo de subcuerpos. Veremos que cuerpos de orden $q = p^n$ se construyen como cuerpos de escisión sobre $\mathbb{Z}_p$. Es por ello que el estudio de la estructura de los cuerpos finitos está íntimamente relacionado con el estudio de ciertas clases de polinomios: irreducibles, primitivos, ciclotómicos y linealizados. Expondremos los resultados más importantes de estos polinomios en la segunda sección del capítulo.

1.1. Estructura de los cuerpos finitos

La estructura de cualquier $\mathbb{F}_q$ ($q = p^n$ con p primo) como grupo aditivo nos es conocida, gracias a su condición de $\mathbb{Z}_p$-espacio vectorial: no es más que la suma directa de n copias de $\mathbb{Z}_p$. Sobre la multiplicación en $\mathbb{F}_q$ se tiene el siguiente resultado:

Teorema 1.1. *El grupo multiplicativo de un cuerpo con q elementos es cíclico, de orden* $q - 1$.

Demostración. Usamos la estructura de los grupos abelianos finitamente generados aplicada al grupo multiplicativo $\mathbb{F}_q^*$. Denotemos por $o(x)$ como el orden de un elemento en

un grupo. Sabemos que existe un $a \in \mathbb{F}_q^*$ tal que $o(b) \mid o(a)$ para todo $b \in \mathbb{F}_q^*$, esto es, tenemos $q-1$ raíces del polinomio $x^{o(a)} - 1 \in \mathbb{Z}_p[x]$, que tiene a lo sumo $o(a)$ raíces. Por tanto, $q - 1 \leq o(a) \leq |\mathbb{F}_q^*| = q - 1$. □

Definición 1.2. *Un elemento a generador de $\mathbb{F}_q^*$ se dirá **primitivo**. Notar que si Φ denota la función phi de Euler, existen $\Phi(q-1)$ elementos primitivos en $\mathbb{F}_q$.*

Observación 1.3. *Como siempre existen elementos primitivos $a \in \mathbb{F}_q$, y es inmediato que $F_q = K(a)$ para cualquier subcuerpo K de F, $F : K$ es una extensión simple.*

Se acaba de ver que dado un cuerpo de q elementos $F = \mathbb{F}_q$, las raíces del polinomio $x^{q-1} - 1$ (visto sobre cualquier subcuerpo K suyo) son exactamente los elementos de $F \backslash \{0\}$. Así:

$$f = x^q - x = \prod_{\alpha \in F} (x - \alpha) \tag{1.1}$$

y F es inmediatamente cuerpo de escisión sobre K, lo cual nos garantiza su unicidad salvo isomorfismo por el **Teorema de existencia y unicidad de cuerpos de escisión**. El recíproco de este razonamiento permite concluir un resultado fundamental:

Teorema 1.4. *Para toda potencia $q = p^n$ de un primo p, existe un único (salvo isomorfismo) cuerpo de q elementos.*

Demostración. Ya hemos razonado la unicidad. Sea F un cuerpo de escisión de $f(x) = x^q - x$ sobre $\mathbb{Z}_p$. Es fácil comprobar, gracias al automorfismo de Froebenius, que el conjunto de raíces de f es un subcuerpo de F que contiene a $\mathbb{Z}_p$, luego ha de coincidir con F. Pero f tiene q raíces distintas en F, por $f' = -1$. Por tanto F tiene q elementos. □

Se verá ahora que existe una correspondencia unívoca entre los subcuerpos de $\mathbb{F}_{p^n}$ y los divisores de n:

Teorema 1.5. *Sea $\mathbb{F}_{p^n}$ un cuerpo finito de p^n elementos. Entonces $\mathbb{F}_q \subseteq \mathbb{F}_{p^n}$ si y solo si $q = p^m$ con $m \mid n$.*

Demostración. Sea $\mathbb{F}_q \subseteq \mathbb{F}_{p^n}$. Se tiene que $p^n = q^k$ para algún k, pero por definición de cuerpo primo $\mathbb{Z}_p \subseteq \mathbb{F}_q$, luego $q = p^m$ con $1 \leq m \leq n$. Entonces $p^n = p^{mk}$ y resulta $m \mid n$. Recíprocamente, supongamos que $n = mk$. Notar que para todo $x, r \in \mathbb{N}$, $x - 1 \mid x^r - 1$, así que en particular, $p^m - 1 \mid (p^m)^k - 1 = p^n - 1$. Aplicando de nuevo la misma propiedad se obtiene que $x^{p^m-1} - 1 \mid x^{p^n-1} - 1$, luego $x^{p^m} - x \mid x^{p^n} - x = \prod_{\alpha \in \mathbb{F}_{p^n}} (x - \alpha)$. Así pues, $x^{p^m} - x$ se escinde en $\mathbb{F}_p^n$, y razonando de igual manera que en el Teorema 1.4 se deduce que sus p^m raíces son distintas y forman un cuerpo. □

Observación 1.6. *La notación $\mathbb{F}_q \subseteq \mathbb{F}_{p^n}$ no genera ninguna ambigüedad: si existe un subcuerpo de q elementos dentro de $\mathbb{F}_{p^m}$ este es único, puesto que $x^q - x$ no puede tener más de las q raíces que conforman $\mathbb{F}_q$.*

Observación 1.7. *Se utilizará regularmente el siguiente criterio. Dado $\alpha \in \mathbb{F}_{p^n}$ y $\mathbb{F}_q \subseteq \mathbb{F}_{p^n}$, $\alpha \in \mathbb{F}_q$ si y solo si $\alpha^q = \alpha$.*

Se ha visto que $\mathbb{F}_q^n$ **es siempre cuerpo de escisión de** $x^{q^n} - x$ **sobre cualquiera de sus subcuerpos**. Gracias al teorema anterior podemos afinar un poco más:

Corolario 1.8. *$\mathbb{F}_{q^n}$ es el cuerpo de escisión de cualquier polinomio $g \in \mathbb{F}_q[x]$ irreducible de grado n.*

Demostración. Sean $\alpha_1, ..., \alpha_n$ las raíces de g en su cuerpo de escisión F sobre $\mathbb{F}_q$. Entonces, como g es irreducible, g es el polinomio mínimo de α_i sobre $\mathbb{F}_q$ para cada $i = 1, ..., n$, luego $[\mathbb{F}_q(\alpha_i) : \mathbb{F}_q] = n$. Además, por la Observación 1.6 tenemos $\mathbb{F}_q(\alpha_i) = \mathbb{F}_{q^n} \subseteq F$. Por tanto $\mathbb{F}(\alpha_1, ..., \alpha_n) = \mathbb{F}_{q^n} \subseteq F$, y se tiene que necesariamente $\mathbb{F}_{q^n} = F$. □

El siguiente resultado también nos será útil:

Proposición 1.9. *Sea $n \in \mathbb{N}$, y sea P el conjunto de los polinomios mónicos irreducibles en $\mathbb{F}_q[x]$ de grado m divisor de n. Entonces $x^{q^n} - x = \prod_{f \in P} f(x)$.*

Demostración. El corolario anterior implica que cualquier polinomio irreducible de grado $m \mid n$ sobre $\mathbb{F}_q[x]$ se escinde en $\mathbb{F}_{q^m} \subseteq \mathbb{F}_{q^n}$, luego todos los polinomios de P dividen a

$x^{q^n} - x$. Recíprocamente, si f es un polinomio mónico que divide a $x^{q^n} - x$, f se escinde en $\mathbb{F}_{q^n}$, luego el cuerpo de escisión de f, $\mathbb{F}_{q^m}$, está contenido en $\mathbb{F}_{q^n}$. Por el Teorema 1.5, $m \mid n$, y por tanto $f \in P$. Para concluir el enunciado, notamos que la factorización en irreducibles de $x^{q^n} - x$ en $\mathbb{F}_q[x]$ no puede tener factores múltiples, dado que no tiene raíces múltiples. Por tanto dichos factores irreducibles han de coincidir exactamente con los elementos de P.

Observación 1.10. *Esta proposición nos dice, en particular, que si $f \in \mathbb{F}_q[x]$ es irreducible de grado m, $f \mid x^{q^n} - x$ si y solo si $m \mid n$.*

□

A pesar de la unicidad lograda en el Teorema 1.4, existen múltiples formas de representar los elementos de un cuerpo finito con diferencias notables a la hora de trabajar su aritmética. Como $\mathbb{F}_{p^n} : \mathbb{Z}_p$ es una extensión simple con $[\mathbb{F}_{p^n} : \mathbb{Z}_p] = n$, sabemos entonces que $\mathbb{F}_{p^n} \cong \mathbb{Z}_p[x]/(g)$ con $g \in \mathbb{Z}_p[x]$ un polinomio irreducible de grado n. Podemos escribir entonces cualquier elemento $a \in \mathbb{F}_q$ de forma única como $a = h(x)$ con $gr(h) < n$, y la suma y producto de elementos en $\mathbb{F}_{p^n}$ vendrá dada por la aritmética modular conocida de $\mathbb{Z}_p[x]/(g)$. Aquí radica la importancia de conocer métodos que nos permitan encontrar polinomios irreducibles de cualquier grado que nos proporcionen esta construcción.

Si, en cambio, conviene facilitar la multiplicación de elementos, desearíamos poder escribir $\mathbb{F}_{p^n} = \{0, 1, \eta, \eta^2, ..., \eta^{p^n-1}\}$, con algún $\eta \in \mathbb{F}_{p^n}$ primitivo. En términos de η, la exponenciación y toma de logaritmos discretos resultaría inmediata. Veremos que una construcción así es siempre posible. Para ello, recordamos la noción de cuerpo ciclotómico:

Definición 1.11. *Sea $n \in \mathbb{N}$ y K un cuerpo. Al cuerpo de escisión del polinomio $x^n - 1$ se le llama **enésimo cuerpo ciclotómico sobre** K, y se denota $K^{(n)}$. Las raíces de $x^n - 1$ se dicen raíces enésimas de la unidad. Denotamos al conjunto de dichas raíces como $E^{(n)}$.*

Teorema 1.12. *Sea $\mathbb{F}_q$ un cuerpo finito de característica p y $m \in \mathbb{N}$ con $m = np^e$, siendo n tal que $mcd(n,p) = 1$. Entonces:*

i) $E^{(n)}$ es ciclíco de orden n con respecto al producto.

ii) $\mathbb{F}_q^{(m)} = \mathbb{F}_q^{(n)}$ y $E^{(m)} = E^{(n)}$

iii) Las raíces de $x^m - 1$ son los elementos de $E^{(n)}$ con multiplicidad p^e.

Demostración. $x^n - 1$ no tiene raíces múltiples, puesto que su derivada nx^{n-1} tiene como raíz, a lo sumo, el 0, que no es raíz de $x^n - 1$. Así pues, $E^{(n)}$ consta de n elementos, los cuales forman un grupo bajo la multiplicación: en efecto, dados $\alpha, \beta \in E^{(n)}$, $(\alpha\beta^{-1})^n - 1 = \alpha^n\beta^{-n} - 1 = 0$. Pero $E^{(n)} \leq K^{(n)^*}$, luego $E^{(n)}$ es también cíclico y tenemos i). ii) y iii) son consecuencia de la factorización $x^m - 1 = x^{p^e n} - 1 = (x^n - 1)^{p^e}$. □

Definición 1.13. *En las condiciones anteriores, un generador de $E^{(n)}$ se dice **n-raíz primitiva de la unidad sobre** $\mathbb{F}_q$. Al polinomio que tiene por raíces todas las n-raíces primitivas de la unidad se le llama **enésimo polinomio ciclotómico**, y se denota por $Q_n(x)$.*

Observación 1.14. *Si α es una n-raíz primitiva de la unidad, se puede desarrollar $Q_n = \prod_{mcd(s,n)=1}(x - \alpha^s)$. Además, Q_n es un polinomio mónico de grado $\Phi(n)$.*

En principio, la definición del polinomio ciclotómico tan solo permite afirmar que $Q_n \in \mathbb{F}_q^{(n)}[x]$. El hecho de que siempre podamos encontrarlo en $\mathbb{Z}_p[x]$ dota de una importancia vital al polinomio ciclotómico. Lo vemos en el siguiente teorema:

Teorema 1.15. *Sea F un cuerpo finito de característica p y $n \in \mathbb{N}$ con $mcd(p,n) = 1$. Entonces:*

i) $x^n - 1 = \prod_{d|n} Q_d(x)$.

ii) $Q_n(x) \in \mathbb{Z}_p[x]$.

Demostración. i) Para cada $d \mid n$, es claro que $E^{(d)} \leq E^{(n)}$, luego $Q_d(x)$ es el polinomio cuyas raíces son exactamente todos los elementos de orden d en $E^{(n)}$. Recorriendo todos

los órdenes posibles en $E^{(n)}$, obtenemos que $\prod_{d|n} Q_d(x)$ es el polinomio que tiene por raíces exactamente las n-raíces de la unidad, es decir, $x^n - 1$.

ii) Procedemos por inducción sobre n. $Q_1(x) = x - 1 \in \mathbb{Z}_p[x]$. Supongamos ahora que $Q_k(x) \in \mathbb{Z}_p[x]$ para todo $k < n$. Podemos reescribir i) como:

$$x^n - 1 = \left(\prod_{d|n, d<n} Q_d(x) \right) Q_n(x) = f(x) Q_n(x)$$

con $f(x) \in \mathbb{Z}_p[x]$ gracias a la hipótesis de inducción. Necesariamente ha de ser también $Q_n(x) \in \mathbb{Z}_p[x]$. En efecto, por el algoritmo de la división euclídea , sabemos que Q_n es el único polinomio $q(x)$ en $F^{(n)}[x]$ tal que $x^n - 1 = f(x)q(x) + r(x)$ con algún $r \in F^{(n)}[x]$ y $r = 0$ o $gr(r) < gr(f)$. Basta repetir la misma división euclídea en $\mathbb{Z}_p[x]$ y notar que el cociente ha de ser (por la unicidad mencionada) Q_n. □

Terminamos esta sección caracterizando el n-cuerpo ciclotómico de un cuerpo finito:

Teorema 1.16. *Sea $\mathbb{F}_q$ un cuerpo de característica p, $n \in \mathbb{N}$ tal que $mcd(p, n) = 1$ y d el orden multiplicativo de q módulo n, esto es, el menor $k \geq 1$ tal que $q^k \equiv 1 \pmod{n}$. Entonces $\mathbb{F}_q^{(n)} = \mathbb{F}_{q^d}$. Además, es el cuerpo de escisión de cualquiera de los $\Phi(n)/d$ factores irreducibles de Q_n en $\mathbb{F}_q[x]$.*

Demostración. Por un lado, $x^n - 1$ se escinde en $\mathbb{F}_{q^d}$: como $n \mid q^d - 1 = |\mathbb{F}_{q^d}{}^*|$, $\mathbb{F}_{q^d}{}^*$ posee un subgrupo cíclico de orden n, es decir, $E^{(n)}$. Por tanto $\mathbb{F}_q^{(n)} \subseteq \mathbb{F}_{q^d}$. Supongamos que dicho contenido no es estricto. Entonces $|\mathbb{F}_q^{(n)}| = q^k$ con $k < d$. Pero ha de darse que $E^{(n)} \subseteq \mathbb{F}_q^{(n)}$, luego $E^{(n)} \leq \mathbb{F}_q^{(n)^*}$ y por tanto $n \mid q^k - 1$, o equivalentemente $q^k \equiv 1\ mod(n)$ con $0 < k < d$, entrando en contradicción con la definición de d. En definitiva, se tiene que $\mathbb{F}_q^{(n)} = \mathbb{F}_{q^d}$.

Sea ahora η una raíz de Q_n, esto es, una n-raíz primitiva de la unidad. Es claro que $\mathbb{F}_q^{(n)} = \mathbb{F}_{q^d} = \mathbb{F}_q(\eta)$, luego el polinomio mínimo de η ha de tener grado d y es, además, uno de los factores irreducibles de Q_n. Se sigue que todos los factores irreducibles de Q_n tienen grado d y, como $gr(Q_n) = \Phi(n)$, ha de haber $\Phi(n)/d$ factores. El cuerpo de escisión de cualquiera de ellos sobre $\mathbb{F}_q$ ha de tener dimensión d y ser por tanto $\mathbb{F}_{q^d}$. □

Por la Observación 1.7, se tiene que un cuerpo finito $\mathbb{F}_{p^n}$ es cuerpo de escisión de $x^{p^n-1}-1$ sobre $\mathbb{Z}_p$, luego es el (p^n-1)-ésimo cuerpo ciclotómico de $\mathbb{Z}_p$. Además, siempre se da que $mcd(p^n-1,p)=1$, luego podemos aplicar el teorema anterior para obtener una construcción de la forma anticipada al introducir los cuerpos ciclotómicos. En primer lugar, calcularemos Q_{p^n-1} en $\mathbb{Z}_p[x]$, lo cual siempre es posible gracias al Teorema 1.15, y trataremos de obtener su factorización en irreducibles. Cualquiera de dichos factores, digamos g, nos sirve para realizar la construcción $\mathbb{Z}_p[x]/(g) \cong \mathbb{F}_{p^n}$, y el elemento $\eta = x+(g)$ es una raíz de g, y por lo tanto una (p^n-1)-raíz primitiva de la unidad. Es decir, $\mathbb{F}_{p^n} = \{0, 1, \eta, \eta^2, \eta^3, ..., \eta^{p^n-1}\}$. Sin embargo, para poder realizar este proceso en la práctica necesitamos saber calcular polinomios ciclotómicos, así como tener algoritmos eficientes de factorización. La sección 1.2.2 y el Capítulo 2 abordan estos problemas con mayor detalle.

1.2. Polinomios sobre cuerpos finitos

Hemos visto como el estudio de la estructura de los cuerpos finitos está íntimamente relacionada con el estudio y la factorización de polinomios. Es por ello que dedicamos esta sección a recabar resultados importantes sobre distintas clases de polinomios.

1.2.1. Polinomios irreducibles

Estudiamos ahora las propiedades básicas de los polinomios irreducibles sobre un cuerpo finito $\mathbb{F}_q$. Además, introduciremos la función de Moebius para dar una fórmula explícita del número de polinomios irreducibles para un grado dado cualquiera. El conocimiento de este número cobrará relevancia en el criptosistema de McEliece estudiado en el último capítulo. Por último, introducimos el concepto de orden de un polinomio para obtener resultados adicionales.

Empezamos con una relación fundamental entre las raíces de un polinomio irreducible, que nos permite conocerlas todas a partir de una de ellas.

Teorema 1.17. *Sea $g \in \mathbb{F}_q[x]$ un polinomio irreducible de grado n, y $\alpha \in \mathbb{F}_{q^n}$ una raíz suya. Entonces las raíces de g son exactamente $\{\alpha, \alpha^q, \alpha^{q^2}, ..., \alpha^{q^{n-1}}\}$.*

Demostración. Si α es una raíz de g, α^{q^k} también, puesto que $g(\alpha^{q^k}) = g(\alpha)^{q^k}$ por el automorfismo de Frobenius. Veamos ahora que los elementos de $\{\alpha, \alpha^q, \alpha^{q^2}, ..., \alpha^{q^{n-1}}\}$ son distintos dos a dos. Supongamos que $\alpha^{q^i} = \alpha^{q^j}$ con $0 \leq i < j \leq n-1$. Aprovechando el hecho de que $\alpha^{q^n} = \alpha$ por $\alpha \in \mathbb{F}_{q^n}$ y el criterio de 1.7, elevar la identidad supuesta a la potencia q^{n-j} nos conduce a $\alpha = \alpha^{q^n} = \alpha^{q^{n+i-j}}$. Así pues, α es raíz de $x^{q^{n+i-j}} - x$, luego $g \mid x^{q^{n+i-j}} - x$ por definición de polinomio mínimo. Como consecuencia de la Observación 1.10, se tiene que $n \mid n+i-j$, lo cual es imposible dado que $n+i-j < n$ por hipótesis. En definitiva, $\alpha, \alpha^q, \alpha^{q^2}, ..., \alpha^{q^{n-1}}$ son las n raíces simples de g.

□

Definición 1.18. *Sea $\alpha \in \mathbb{F}_{q^n}$. A los elementos $\alpha, \alpha^q, \alpha^{q^2}, ..., \alpha^{q^{n-1}}$ se les llama **conjugados de** α **con respecto a** $\mathbb{F}_q$. En el teorema anterior hemos visto que si el polinomio mínimo g de α tiene grado n, entonces todos sus conjugados son distintos dos a dos. En cambio, si su grado es $m \mid n$, entonces $\alpha, \alpha^q, \alpha^{q^2}, ..., \alpha^{q^{m-1}}$ son distintos dos a dos por el mismo motivo, luego los conjugados de α son precisamente estos elementos repetidos de forma cíclica, apareciendo cada uno n/m veces. En tal caso, al polinomio $f = g^{n/m}$ se le llama **polinomio característico de** α **sobre** $\mathbb{F}_q$, y sus raíces son todas las conjugadas de α.*

Ya sabemos que para cualquier $n \in \mathbb{N}$, existe al menos un polinomio mónico irreducible en $\mathbb{F}_q[x]$. Destacamos dos motivos por los cuales resultaría útil conocer cuántos hay con exactitud. En primer lugar, nos permitiría saber cuando hemos terminado de calcularlos todos; en segundo, nos permite estimar la eficacia de métodos probabilísticos de generación de polinomios irreducibles. Es decir, nos permite responder a la pregunta: ¿cuál es la

probabilidad de que, tomando aleatoriamente un polinomio de grado n, este resulte ser irreducible? Si dicha probabilidad no se reduce drásticamente según crece n, parece viable tomar polinomios de forma aleatoria, discernir si se trata de un polinomio irreducible mediante, por ejemplo, alguno de los algoritmos de factorización del Capítulo 2, y en caso de que no lo sea, repetir el proceso. Este procedimiento puede llegar a ser en la práctica mucho más efectivo que tratar una construcción exacta. Denotemos de ahora en adelante por $N_q(d)$ al número de polinomios mónicos irreducibles sobre $\mathbb{F}_q[x]$ de grado d. La Proposición 1.9 otorga de manera inmediata una fórmula recursiva:

Corolario 1.19. $q^n = \sum_{d|n} dN_q(d)$.

Demostración. Denotando como en la Proposición 1.9 P al conjunto de los polinomios mónicos irreducibles en $\mathbb{F}_q[x]$ de grado m divisor de n, tenemos que $x^{q^n} - x = \prod_{f \in P} f(x)$. Basta tomar grados y clasificar los elementos de P para concluir

$$q^n = \sum_{f \in P} gr(f) = \sum_{d|n} dN_q(d) \tag{1.2}$$

. □

Gracias a esta fórmula, el conocimiento de $N_q(d)$ para todos los divisores propios de n permite deducir con facilidad $N_q(n)$. No obstante, nos resultará muy útil introducir la **Fórmula de Inversión de Möbius**, no solo para obtener una fórmula explícita de $N_q(n)$ a partir de la anterior, sino que también recurriremos a ella cuando estudiemos en más detalle los polinomios ciclotómicos.

Definición 1.20. *Se define la* ***función de Möbius*** $\mu : \mathbb{N} \longrightarrow \{-1, 0, 1\}$ *como:*

$$\mu(n) = \begin{cases} 1 & si \quad n = 1 \\ (-1)^k & si \quad n \text{ es producto de } k \text{ primos distintos dos a dos} \\ 0 & si \quad \text{algún primo en la factorización de } n \text{ se repite} \end{cases}$$

Para demostrar la fórmula de inversión de Möbius necesitamos el siguiente lema:

Lema 1.21. *Para todo $n > 1$ se verifica $\sum_{d|n} \mu(d) = 0$.*

Demostración. Supongamos $n = {p_1}^{k_1}...p_m^{k_n}$. Dado que si algún primo se repite en la factorización de $d \mid n$ el sumando $\mu(d)$ se anula, podemos considerar en la suma solamente los divisores de $p_1...p_m$. Por tanto, sin más que desarrollar y aplicar la definición de μ obtenemos: $\sum_{d|n} \mu(d) = \mu(1) + \sum_{i=1}^{m} \mu(p_i) + \sum_{i<j} \mu(p_i p_j) + ... + \mu(p_1...p_m) = 1 - \binom{m}{1} + \binom{m}{2} + ... + (-1)^m \binom{m}{m} = \sum_{k=0}^{m} (-1)^k \binom{m}{k}$. Pero esta última expresión es el desarrollo del binomio de Newton $(1 + (-1))^k$, que es 0.

□

El teorema de la Fórmula de Inversión se tiene en términos de grupos abelianos en general, luego admite una expresión aditiva o multiplicativa. En nuestro caso actual nos conviene adoptar la notación aditiva. No obstante en la Proposición 1.24 y en el Teorema 1.32 usaremos el mismo resultado en su forma multiplicativa.

Teorema 1.22. *Fórmula de Inversión de Möbius.*

Sea $(G, +)$ un grupo abeliano, y $h, H : \mathbb{N} \longrightarrow G$ dos aplicaciones cualesquiera. Entonces la igualdad

$$H(n) = \sum_{d|n} h(d) \tag{1.3}$$

es cierta para todo $n \in \mathbb{N}$ si y solo si

$$h(n) = \sum_{d|n} \mu(\frac{n}{d}) H(d) \tag{1.4}$$

se da para todo $n \in \mathbb{N}$ (o equivalentemente $h(n) = \sum_{d|n} \mu(d) H(\frac{n}{d})$, recorriendo el sumatorio de manera inversa) .

Demostración. $\Rightarrow$: Supongamos cierta (1.3) y fijemos $n \in \mathbb{N}$. Entonces tenemos que

$$\sum_{d|n} \mu(d) H(\frac{n}{d}) = \sum_{d|n} \mu(d) \sum_{d' | \frac{n}{d}} h(d') = \sum_{d|n} \sum_{d' | \frac{n}{d}} \mu(d) h(d').$$

Usando ahora la igualdad $\{(d,d') : d \mid n \text{ y } d' \mid \frac{n}{d}\} = \{(d,d') : d' \mid n \text{ y } d \mid \frac{n}{d'}\}$ llegamos a que la expresión anterior es igual a

$$\sum_{d'|n} \sum_{d|\frac{n}{d'}} \mu(d)h(d') = \sum_{d'|n} h(d') \sum_{d|\frac{n}{d'}} \mu(d) = h(n).$$

La última igualdad se debe al Lema 1.21: $\sum_{d|\frac{n}{d'}} \mu(d) = 1$ si y solo si $n = d'$, y en otro caso es igual a 0.

$\Leftarrow$: Recíprocamente, supongamos que para todo $m \in \mathbb{N}$ se tiene (1.4) y fijamos $n \in \mathbb{N}$. Entonces:

$$\sum_{d|n} h(d) = \sum_{d|n} \sum_{d'|d} \mu\left(\frac{d}{d'}\right) H(d') = \sum_{d'|n} H(d') \sum_{d \text{ tal que } d'|d|n} \mu\left(\frac{d}{d'}\right).$$

Teniendo ahora en cuenta que si $d' \mid n$ y $d' \mid d$, entonces $d' \mid d \mid n \Leftrightarrow d \mid n \Leftrightarrow d/d' \mid n/d'$, y llamando $a = \dfrac{d}{d'}$, obtenemos

$$\sum_{d \text{ tal que } d'|d|n} \mu\left(\frac{d}{d'}\right) = \sum_{a|\frac{n}{d'}} \mu(a) = 1 \text{ si y solo si } n = d'$$

y en otro caso igual a 0, debido nuevamente al Lema 1.21. Concluimos que $\sum_{d|n} h(d) = H(n)$.

□

Considerando $H(n) = q^n$ y $h(n) = nN_q(d)$ como funciones de $\mathbb{N}$ en $(\mathbb{Z}, +)$, la fórmula de inversión de Möbius nos permite transformar la identidad del Corolario 1.19 en $nN_q(n) = \sum_{d|n} \mu(n/d)q^d$, obteniendo el resultado deseado:

Corolario 1.23. *El número de polinomios irreducibles mónicos de grado n en $\mathbb{F}_q[x]$ es:*

$$N_q(n) = \frac{1}{n} \sum_{d|n} \mu(n/d)q^d = \frac{1}{n} \sum_{d|n} \mu(d)q^{n/d}.$$

La gran generalidad en la que se establece la Fórmula de Inversión nos permite obtener más expresiones explícitas. En la Proposición 1.9 hemos concluido que el producto de todos

los polinomios irreducibles mónicos con grado divisor de n en $\mathbb{F}_q[x]$ es $x^{q^n} - x$. Ahora podemos obtener de forma inmediata el **producto de todos los polinomios mónicos irreducibles para cualquier grado dado**. En efecto, si denotamos $P_q(n) = \{f \in \mathbb{F}_q[x] : f \text{ es mónico e irreducible con } gr(f) = n\}$ y $I_q(n) = \prod P_q(n)$, entonces $x^{q^n} - x = \prod_{d|n} I_q(d)$. Para aplicar 1.22 hemos de considerar $H(n) = x^{q^n} - x$ y $h(n) = I_q(n)$ como funciones de $\mathbb{N}$ en el grupo multiplicativo del cuerpo de fracciones de $\mathbb{F}_q[x]$, esto es, del cuerpo de funciones algebraicas $\mathbb{F}_q(x)$. Teniendo en cuenta que la ecuación (1.3) se escribe multitplicativamente como $h(n) = \prod_{d|n} H(d)^{\mu(n/d)}$, obtenemos:

Proposición 1.24.

$$I_q(n) = \prod_{d|n} (x^{q^d} - x)^{\mu(n/d)}$$

Así pues, para conocer todos los polinomios irreducibles de un cierto grado, basta calcular $I_q(n)$ y factorizarlo, por ejemplo, mediante las técnicas tratadas en el Capítulo 2. No obstante, en el Teorema 1.33 lograremos una factorización parcial de este polinomio en términos de polinomios ciclotómicos.

Para seguir profundizando en el conocimiento de los polinomios sobre un cuerpo finito, necesitamos introducir el concepto de orden de un polinomio. Dicha definición se basa en el siguiente resultado:

Lema 1.25. *Sea $f \in \mathbb{F}_q[x]$ con $f(0) \neq 0$ y $gr(f) = n > 0$. Entonces existe $0 < e \leq q^n - 1$ tal que $f \mid x^e - 1$.*

Demostración. Consideramos las q^n clases $x^j + (f)$ con $j = 0, ..., q^n - 1$ del anillo de restos $\mathbb{F}_q[x]/(f)$. Dichas clases son todas no nulas, puesto que $x^j + (f) = 0 + (f)$ implica $f \mid x^j$ lo cual es imposible por $f(0) \neq 0$ si $j > 0$ y por $gr(f) > 0$ si $j = 0$. Por otro lado, el anillo $\mathbb{F}_q[x]/(f)$ tiene exactamente $q^n - 1$ elementos no nulos, luego existen $0 \leq s < j \leq q^n - 1$ tales que $x^j + (f) = x^s + (f)$ o, equivalentemente, $f \mid x^j - x^s = x^s(x^{j-s} - 1)$. Como $f(0) \neq 0$, ha de darse $f \mid x^{j-s} - 1$ y basta tomar $e = j - s \leq q^n - 1$. □

Definición 1.26. *Sea $f \in \mathbb{F}_q[q]$ un polinomio no nulo. Entonces, si $f(0) \neq 0$, llamamos* ***orden de*** *f al menor $e \in \mathbb{N}$ en las condiciones del lema anterior, y se denota $ord(f)$. Si por el contrario el 0 es raíz de f con multiplicidad s, definimos $ord(f) = ord(g)$ donde $f = x^s g(x)$ y $g(0) \neq 0$.*

Para polinomios irreducibles, tenemos una muy buena caracterización de su orden:

Teorema 1.27. *Sea $f \in \mathbb{F}_q[x]$ un polinomio irreducible distinto de x. Entonces el orden de f coincide con el orden de cualquiera de sus raíces en el grupo multiplicativo del cuerpo de escisión de f.*

Demostración. Supongamos que $gr(f) = n$, $e = ord(f)$, y sea α una raíz de f en $\mathbb{F}_{q^n}$. Como $f \mid x^e - 1$, se tiene que $\alpha^e - 1 = 0$, y por tanto $o(\alpha) \mid e$. Pero por otro lado, α también es raíz de $x^{o(\alpha)} - 1 = 0$, luego por ser f el polinomio mínimo de α tenemos que $f \mid x^{o(\alpha)} - 1$. La definición de orden implica que $e \leq o(\alpha)$, y tenemos finalmente $o(\alpha) = e$. □

Corolario 1.28. *Sea $f \in \mathbb{F}_q[x]$ un polinomio irreducible de grado n. Entonces $ord(f) \mid q^n - 1$.*

Contar los polinomios de $P_q(n)$ (los mónicos e irreducibles sobre $\mathbb{F}_q[x]$ de grado d) con un cierto orden fijo es especialmente sencillo. Llamemos $P_{(q,e)}(n) = \{f \in P_q(n) : ord(f) = e\}$ y $N_{(q,e)}(n)$ al número de elementos en $P_{(q,e)}(n)$. Tenemos el siguiente resultado:

Teorema 1.29. *Sean $e, n \in \mathbb{N}$ y m el orden multiplicativo de q módulo e. Entonces:*

$$N_{(q,e)}(n) = \begin{cases} 2 & si \quad n = e = 1 \\ \Phi(e)/n & si \quad e \geq 2 \text{ y } n = m \\ 0 & en\ otro\ caso \end{cases}$$

Demostración. Supongamos $e \geq 2$. Gracias al teorema anterior, sabemos que los polinomios mónicos irreducibles en $\mathbb{F}_q[x]$ son exactamente los polinomios mínimos de las e-raíces

de la unidad sobre $\mathbb{F}_q$. Por el Teorema 1.16, estos han de ser los $\Phi(e)/m$ factores irreducibles de Q_e, todos ellos de grado m necesariamente. Por otro lado, x y $x-1$ son los únicos polinomios mónicos irreducibles de orden 1. □

Ahondaremos en el estudio del orden en la sección dedicada a los polinomios primitivos.

Sabemos que el cuerpo de escisión de un polinomio irreducible $f \in \mathbb{F}_q[x]$ de grado n es $\mathbb{F}_{q^n}$. Nos preguntamos ahora por la cuestión contraria: ¿sobre qué extensiones finitas de $\mathbb{F}_q$ el polinomio f sigue siendo irreducible? Utilizando los resultados precedentes podemos demostrar el siguiente teorema, que nos da una respuesta completa a la pregunta:

Teorema 1.30. *Sea $\mathbb{F}_{q^k} : \mathbb{F}_q$ una extensión de cuerpos finitos, $f \in \mathbb{F}_q[x]$ un polinomio irreducible de grado n y $d = mcd(n,k)$. Entonces f se factoriza en $\mathbb{F}_{q^n}[x]$ como producto de d polinomios irreducibles de grado n/d.*

Demostración. El caso $n = 1$ es trivial, luego supongamos $n \geq 2$. Sea g un factor de f irreducible en $\mathbb{F}_{q^k}[x]$ y $e = ord(f)$. Por el teorema anterior, ha de darse que n coincida con el orden multiplicativo de q módulo e, y además $e \geq 2$. Por otro lado, gracias al Teorema 1.27 deducimos que $ord(g) = ord(f) = e$, luego, de nuevo por el teorema anterior, el grado de g es el orden multiplicativo de q^k módulo $ord(g) = e$. Pero dicho orden es $\frac{n}{mcd(n,k)} = n/d$. Se sigue fácilmente que f tiene n/d factores irreducibles $\mathbb{F}_{q^n}[x]$. □

En particular:

Corolario 1.31. *$f \in P_q(n)$ es irreducible en $\mathbb{F}_{q^k}[x]$ si y solo si $mcd(n,k) = 1$.*

1.2.2. Polinomios ciclotómicos

Pretendendemos profundizar un poco más en el estudio de los polinomios ciclotómicos. Daremos una fórmula general para su cálculo, así como otras más sencillas para casos

particulares. También establecemos su utilidad para la obtención de todos los polinomios irreducibles de un grado dado.

Recordamos que si n es coprimo con la característica de $\mathbb{F}_q$, entonces:

$$x^n - 1 = \prod_{d|n} Q_d(x).$$

Si tomamos una potencia r^k de un número primo r con $mcd(p, r) = 1$ y despejando, obtendremos una expresión muy sencilla:

$$Q_{r^k} = \frac{x^{r^k} - 1}{\prod_{0 \leq i \leq k-1} Q_{r^i}} = \frac{x^{r^k} - 1}{x^{r^{k-1}} - 1}.$$

Haciendo el cambio $y = x^{r^{k-1}}$ se ve fácilmente que:

$$Q_{r^k} = \frac{y^r - 1}{y - 1} = 1 + y + y^2 + ... + y^{r-1} = 1 + x^{r^{k-1}} + x^{2r^{k-1}} + ... + x^{(r-1)r^{k-1}}.$$

En particular:

$$Q_r(x) = 1 + x + x^2 + ... + x^{r-1}.$$

La fórmula de inversión de Möbius nos proporciona una fórmula general:

Teorema 1.32. *Sea $\mathbb{F}_{p^m}$ un cuerpo finito de característica p y $n \in \mathbb{N}$ tal que $mcd(p, n) = 1$. Entonces el enésimo polinomio ciclotómico se escribe:*

$$Q_n(x) = \prod_{d|n} (x^d - 1)^{\mu(n/d)}$$

Demostración. Basta tomar $h(d) = Q_d(x)$ y $H(d) = x^d - 1$ como funciones de $\mathbb{N}$ en el grupo multiplicativo del cuerpo de funciones algebraicas sobre $\mathbb{F}_{p^m}$ y aplicar la fórmula de inversión. □

El siguiente resultado ofrece una factorización parcial de los polinomios $I_q(n)$ de la Proposición 1.24 en términos de polinomios ciclotómicos:

Teorema 1.33. *$I_q(n) = \prod_m Q_m$ para todo $n > 1$, donde m recorre los divisores de $q^n - 1$ para los cuales n es el orden multiplicativo de q módulo m.*

Demostración. Sea S el conjunto de todos los elementos de $\mathbb{F}_{q^n}$ cuyo polinomio mínimo sobre $\mathbb{F}_q$ tiene grado n. Entonces es claro que $I_q(n) = \prod_{\alpha \in S}(x - \alpha)$. Dado que el caso $n = 1$ es trivial, podemos suponer $n > 1$, y entonces $0 \notin S \subseteq \mathbb{F}_{q^n}^*$. Así pues, $\alpha \in S$ implica $o(\alpha) \mid q^n - 1$. Pero α no puede pertenecer a ningún subcuerpo propio $\mathbb{F}_{q^d} \subseteq \mathbb{F}_{q^n}$, por tanto $o(\alpha)$ no divide a $q^d - 1$ para ningún $d < n$ por 1.7. Es decir, n es el orden multiplicativo de q módulo $o(\alpha)$.

Gracias a este razonamiento, podemos partir S en clases S_m donde cada S_m es el conjunto de elementos en S de orden m y m recorre los naturales para los cuales n es el orden multiplicativo de q módulo m. Dicha partición permite factorizar:

$$I_q(n) = \prod_m \prod_{\alpha \in S_m} (x - \alpha).$$

Pero cada S_m es, en realidad, el conjunto de todas las m-raíces primitivas de la unidad sobre $\mathbb{F}_q$. En efecto, si el polinomio mínimo de una m-raíz primitiva de la unidad $\beta \in \mathbb{F}_{q^n}$ no tuviera grado n, entonces β pertenecería a un subcuerpo propio de $\mathbb{F}_{q^n}$, luego n no podría ser el orden multiplicativo de q módulo m, y entraríamos en contradicción con la condición impuesta sobre cada m. Por tanto se tiene por definición que $\prod_{\alpha \in S_m}(x - \alpha) = Q_m(x)$, completando el resultado.

□

Podemos aprovechar los argumentos usados en esta demostración para obtener un hecho que resultará útil más adelante: si tomamos $e > 1$ con $mcd(e, q) = 1$, existe n el orden multiplicativo de q módulo e, y $n \geq 1$. Usando la misma notación de antes, tendríamos que $\prod_{\alpha \in S_e}(x - \alpha) = Q_m(x)$. Pero por otro lado $\prod_{\alpha \in S_e}(x - \alpha)$ también ha de ser igual al producto de todos los polinomios mónicos irreducibles de grado n y de orden e en $\mathbb{F}_q[x]$ gracias a la demostración de 1.33 y al Teorema 1.27. Por lo tanto se tiene:

Corolario 1.34. *Sean $e \geq 2$ con $mcd(e, q) = 1$ y n el orden multiplicativo de q módulo e. Entonces el producto de todos los polinomios mónicos irreducibles en $\mathbb{F}_q[x]$ de grado n y orden e es igual a Q_e.*

1.2.3. Polinomios primitivos

En esta sección abordamos una clase de polinomios que resulta fundamental para las aplicaciones de los cuerpos finitos. Además resultan ser los polinomios cuya obtención ya anticipábamos útil para construir extensiones a partir de un elemento generador del grupo multiplicativo del cuerpo.

Definición 1.35. *Un polinomio $f \in \mathbb{F}_q[x]$ de grado n se dice* ***primitivo*** *si es el polinomio mínimo sobre $\mathbb{F}_q$ de un elemento primitivo de $\mathbb{F}_{q^n}$.*

Gracias al Teorema 1.27 esta definición es equivalente a afirmar que f es irreducible y $ord(f) = q^n - 1$. Es decir, los polinomios primitivos son aquellos irreducibles de orden máximo entre los de su mismo grado. Nuestro siguiente objetivo consiste en acortar dicha definición ofreciendo, además, una condición suficiente de irreducibilidad:

Teorema 1.36. *Sea $f \in \mathbb{F}_q[x]$ de grado n. Son equivalentes:*

i) f es primitivo.

ii) f es mónico, $f(0) \neq 0$ y $ord(f) = q^n - 1$.

La primera implicación es obvia por el Teorema 1.27. Sin embargo, para demostrar la segunda implicación necesitaremos varios lemas y resultados previos de interés independiente para el estudio del orden. Concretamente nos proporcionarán la relación entre el orden de un polinomio y el de sus factores primos.

Lema 1.37. *Sea $c \in \mathbb{Z}$ con $c > 0$ y $f \in \mathbb{F}_q[x]$. Si $f(0) \neq 0$, entonces $f \mid x^c - 1$ si y solo si $ord(f) \mid c$.*

Demostración. Recordamos que si $ord(f) = e$ y $f(0) \neq 0$, entonces $f \mid x^e - 1$. Por tanto, dado que si $e \mid c$ se tiene $x^e - 1 \mid x^c - 1$ (como vimos en la demostración de 1.5), se da $f \mid x^c - 1$. Recíprocamente, si $f \mid x^c - 1$, se tiene por definición de orden que $e \leq c$, luego podemos aplicar la división euclídea para tener $c = se + r$ con $0 \leq r \leq e - 1$. Entonces

$$f \mid x^c - 1 = x^{se+r} - 1 = x^r x^{se} - 1 = x^r x^{se} - x^r + x^r - 1 = x^r(x^{se} - 1) + (x^r - 1).$$

Puesto que también $f \mid x^e - 1 \mid x^{se} - 1 \mid x^r(x^{se} - 1)$, ha de darse $f \mid x^r - 1$. Para no entrar en contradicción con la definición de e, la única posibilidad es $x^r - 1 = 0$ y por tanto $r = 0$, luego $e \mid c$. □

Los dos siguientes proposiciones nos permiten obtener el orden de un polinomio arbitrario en función de sus factores irreducibles.

Proposición 1.38. *Sea $g \in \mathbb{F}_q[x]$ irreducible distinto de x, $p = car(\mathbb{F}_q)$ y k un entero positivo. Entonces $ord(g^k) = ord(g)p^n$ donde n es el menor entero t tal que $p^t \geq k$.*

Demostración. Llamemos $e = ord(g)$ y $c = ord(g^k)$. Veamos que $e \mid c \mid ep^n$. Por el lema anterior, $g \mid x^e - 1$, luego $g^k \mid (x^e - 1)^k \mid (x^e - 1)^{p^n} = x^{ep^n} - 1$, y de nuevo por el lema, $c \mid ep^n$. Además, como $g \mid g^k \mid x^c - 1$, se tiene $e \mid c \mid ep^n$, lo cual es solo posible si c es de la forma $c = ep^t$ con $0 \leq t \leq n$. Veamos que $t = n$. Como e es el orden multiplicativo de las raíces de g, y este no puede ser múltiplo de p, entonces $x^e - 1$ no tiene raíces múltiples, de donde se sigue que las raíces de $x^{ep^t} - 1$ tienen multiplicidad p^t. Pero $g^k \mid x^{ep^t} - 1$, luego $k \leq p^t$. La definición de n implica $n \leq t$. □

Proposición 1.39. *Sean $g_1, ..., g_k \in \mathbb{F}_q[x]$ coprimos dos a dos. Entonces*

$$ord(g_1, ..., g_k) = mcm(ord(g_1), ..., ord(g_k))$$

Demostración. Basta verlo para $k = 2$. Sean $e_1 = ord(g_1)$, $e_2 = ord(g_2)$, $f = g_1 g_2$ y $c = mcm(e_1, e_2)$. Usando que $e_1, e_2 \mid c$ se tiene $x^{e_1} - 1, x^{e_2} - 1 \mid x^c - 1$. El Lema 1.37 implica entonces que $g_1, g_2 \mid x^c - 1$. Como $mcd(g_1, g_2) = 1$, se tiene $f \mid x^c - 1$. De nuevo, el lema implica $ord(f) \mid c$. Razonando de manera inversa, $f \mid x^{ord(f)} - 1$ implica $g_1, g_2 \mid x^{ord(f)} - 1$, luego por el lema $e_1, e_2 \mid ord(f)$ y finalmente $c = mcd(e_1, e_2) \mid ord(f)$. □

Observación 1.40. *Estas dos últimas proposiciones permiten conocer el orden de un polinomio arbitrario a partir de los órdenes de sus factores irreducibles, órdenes que ya hemos caracterizado en el Teorema 1.27. Recordamos que esta factorización puede ser lograda con los métodos expuestos en el capítulo siguiente. No obstante, conviene ser*

capaces de conocer el orden de un polinomio irreducible a priori del cálculo de sus raíces. Esto puede lograrse de la siguiente manera. Sea $f \in \mathbb{F}_q[x]$ *un polinomio irreducible de grado* n*, y* $e = ord(f)$*. Entonces sabemos por el Corolario 1.28 que* $e \mid q^n - 1$*. Podemos proceder eficientemente si conocemos* $q^n - 1 = \prod_{j=1}^{m} p_j^{r_j}$ *como producto de factores primos: se tiene que* $e = \prod_{j=0}^{m} p_j^{s_j}$ *para ciertos* $0 \leq s_j \leq r_j$ *,* $j = 1, ..., m$*. Además, por definición de orden, sabemos que* $t \mid \prod_{j=0}^{m} p_j^{r_j}$ *verifica* $e \mid t$ *si y solo si* $x^t \equiv 1 \pmod{f}$*. Por ello, si* $x^{\frac{q^n-1}{p_1}} \not\equiv 1 \pmod{f}$ *entonces* $e \nmid \frac{q^n-1}{p_1} = p_1^{r_1-1} \prod_{j=2}^{m} p_j^{r_j}$*, lo cual es solo posible si* $r_1 \leq s_1$*, luego* $s_1 = r_1$*. En caso contrario se tiene* $s_1 \leq r_1 - 1$*, y se repite el proceso calculando* $x^{\frac{q^n-1}{p_1^2}} \pmod{f}$*, y así sucesivamente hasta encontrar el valor de* s_1*. Se procede de igual manera para calcular* $s_2, ..., s_m$*.*

Ahora ya podemos terminar de probar el Teorema 1.36.

Demostración del Teorema 1.36:

Demostración. Hay que ver la segunda implicación. Partimos de $f \in \mathbb{F}_q[x]$ mónico de grado n, con $f(0) \neq 0$ y $e = ord(f) = q^n - 1$. Por el Teorema 1.27, nos basta ver que f es irreducible. Supongamos que no lo es. Entonces se presentan dos alternativas:

- a) $f = g^k$ con $g(0) \neq 0$ y $k > 1$. En tal caso, si p es la característica de $\mathbb{F}_q$, la Proposición 1.38 implica que $p \mid e = q^n - 1$, lo cual es imposible por ser q potencia de p.

- b) $f = g_1 g_2$ con $mcd(g_1, g_2) = 1$ de respectivos grados $n_1, n_2 \geq 1$. Entonces tiene sentido considerar $ord(g_1), ord(g_2) > 0$, y la Proposición 1.39 implica que $q^n - 1 = e \mid mcm(ord(g_1), ord(g_2)) \mid ord(g_1)ord(g_2) \leq (q^{n_1} - 1)(q^{n_2} - 1) = q^n - q^{n_1} - q^{n_2} - 1 < q^n - 1$. Como los términos involucrados son no nulos, esto implica $q^n - 1 < q^n - 1$, lo cual es absurdo.

□

Antes de sugerir un método de construcción de polinomios primitivos, podemos ejemplificar la utilidad del teorema demostrado introduciendo la importante noción de recíproco de un polinomio.

Ejemplo 1.41. *Sea* $f(x) = \sum_{i=0}^{n} a_i x^i \in \mathbb{F}_q[x]$ *con* $a_n \neq 0$. *El* ***polinomio recíproco de*** f *es:*

$$f^*(x) = \sum_{i=0}^{n} a_{n-i} x^i = x^n f\Big(\frac{1}{x}\Big).$$

Se cumplen propiedades ciertamente interesantes. Supongamos que f *divide a un cierto polinomio* g. *Entonces* $g(x) = f(x)h(x)$ *para algún* h *de grado* m, *y por tanto:*

$$x^{m+n} g\Big(\frac{1}{x}\Big) = x^n f\Big(\frac{1}{x}\Big) x^m h\Big(\frac{1}{x}\Big) = f^*(x)h^*(x) \mid g^*(x).$$

Además, notamos que si $f(0) \neq 0$, *esto es,* $a_0 \neq 0$, *entonces* $(f^*)^* = f$. *En este caso, si tomamos en particular* $g(x) = x^e - 1$ *se tiene* $g^*(x) = -g(x)$. *Por tanto* $f \mid x^e - 1 \Rightarrow f^* \mid x^e - 1 \Rightarrow f = (f^*)^* \mid x^e - 1$, *es decir* $f \mid x^e - 1$ *si y solo si* $f^* \mid x^e - 1$, *lo cual deja claro por definición que* $ord(f) = ord(f^*)$. *Si en cambio* $f(0) = 0$, $(f^*)^* \neq f$. *No obstante, en este caso* $f(x) = x^k q(x)$ *con* $q(0) \neq 0$ *y es fácil comprobar que* $f^* = q^*$. *Se deduce, por lo anterior y de nuevo por definición, que* $ord(f^*) = ord(q^*) = ord(q) = ord(f)$. *En todo caso se tiene que* $ord(f) = ord(f^*)$. *Por último, es consecuencia de este hecho y de la caracterización dada en el Teorema 1.36 que:*

$$\frac{1}{f(0)} f(x) \textbf{ es primitivo si y solo si } \frac{1}{f^*(0)} f^*(x) \textbf{ es primitivo,}$$

hecho que nos proporciona una manera sorprendentemente simple de obtener un polinomio primitivo a partir de otro. En el contexto de las aplicaciones cuerpos finitos, este polinomio ejemplo cobra importancia, por ejemplo, al introducir el concepto de dual de un código cíclico en Teoría Algebraica de Códigos.

Existen diferentes métodos para la construcción de polinomios primitivos de grado n, algunos de ellos basados en la obtención del polinomio mínimo de un elemento primitivo

de $\mathbb{F}_{q^n}$. Nosotros aprovecharemos los resultados obtenidos hasta el momento, notando que todos los polinomios primitivos de grado n pueden obtenerse de la siguiente manera:

1. Se calcula $e = q^n - 1$ y Q_e. Esto puede lograrse, por ejemplo, con la fórmula explícita del Teorema 1.32.

2. Se factoriza Q_e mediante alguno de los métodos que serán expuestos en el capítulo siguiente. Como $mcd(q^n - 1, q) = 1$ y n es, obviamente, el orden multiplicativo de q módulo $q^n - 1 = e$, el Corolario 1.34 y el Teorema 1.36 garantizan que dicha factorización está formada precisamente por todos los polinomios primitivos sobre $\mathbb{F}_q$ de grado n.

1.2.4. Polinomios linealizados: búsqueda de raíces y divisibilidad simbólica.

Dedicamos esta sección a estudiar los polinomios linealizados respecto de un cuerpo finito $\mathbb{F}_q$ fijo a lo largo de toda la sección.

Definición 1.42. *Sea $\mathbb{F}_{q^m} : \mathbb{F}_q$ una extensión de cuerpos finitos. Los **polinomios linealizados sobre** $\mathbb{F}_{q^m}$ son aquellos $L \in \mathbb{F}_{q^m}[x]$ de la forma*

$$L(x) = \sum_{i=0}^{n} a_i x^{q^i}$$

con $a_0, ..., a_n \in \mathbb{F}_{q^m}$.

Observación 1.43. *Si a su vez $F : \mathbb{F}_{q^m}$ es otra extensión de cuerpos, podemos considerar F como un $\mathbb{F}_q$-espacio vectorial. En virtud de 1.7 tenemos que $c^{q^i=c}$ $\forall i \in \mathbb{N}, \forall c \in \mathbb{F}_q$. Usando el automorfismo de Frobenius es evidente que la evaluación $L : \alpha \in F \to L(\alpha) \in F$ es un $\mathbb{F}_q$-endomorfismo.*

Esta propiedad convierte a los polinomios linealizados en un objeto digno de estudio. A nosotros nos interesan especialmente porque nos permiten introducir técnicas de Álgebra

Lineal aplicables al problema de la búsqueda de las raíces de un polinomio arbitrario. Además, nos ayudan a profundizar en el entendimiento de la estructura de los cuerpos finitos. Uno de los ejemplos más sencillos de polinomio linealizado resulta ser de especial importancia:

Ejemplo 1.44. *Sean $F = \mathbb{F}_{q^m} : \mathbb{F}_q = K$. El polinomio $Tr_{F|K}(x) := x+x^q+x^{q^2}+...+x^{q^{m-1}}$ es linealizado. Hemos visto que entonces $Tr_{F|K}$ es una aplicación lineal de F en F, pero además se tiene que $Tr_{F|K}(F) \subseteq K$. En efecto, sea $\alpha \in F$, y $f \in K[x]$ su polinomio característico sobre K, esto es,*

$$f(x) = (x-\alpha)(x-\alpha^q)(x-\alpha^{q^2})...(x-\alpha^{q^{m-1}})$$

Entonces el coeficiente que acompaña al término x^{m-1} de f es

$$-\sum_{i=0}^{m-1} \alpha^{q^i} = -Tr_{F|K}(\alpha),$$

luego $Tr_{F|K}(\alpha) \in K$. Como $Tr_{F|K}$ tiene a lo sumo q^{m-1} raíces y F tiene q^m elementos, la aplicación $Tr_{F|K}$ no puede ser nula, y es por tanto un epimorfismo entre los K- espacios vectoriales F y K. A $Tr_{F|K}(\alpha)$ se le llama ***traza de*** *α* ***respecto de*** *K. Es sencillo probar que cualquier otra transformación lineal de F en K es de la forma $L_\beta(\alpha) = Tr_{F|K}(\beta\alpha)$ con $\beta \in K$. Otras propiedades de la traza pueden ser obtenidas en el estudio más general de los polinomios linealizados.*

Nos centramos ahora en el estudio de las raíces de un polinomio linealizado.

Teorema 1.45. *Sea $L(x) = \sum_{i=0}^{n} a_i x^{q^i} \in \mathbb{F}_{q^m}[x]$ un polinomio linealizado y F un cuerpo finito donde se escinda. Sea U el conjunto de las raíces de L en F. Entonces:*

i) Todas las raíces de L tienen la misma multiplicidad q^k para cierto $k \geq 0$.

ii) U es un $\mathbb{F}_q$-subespacio de F de dimensión $n-k$.

Demostración. Para i) notar que $L'(x) = a_0$, luego por 0.2 L tiene raíces múltiples si y solo si $a_0 = 0$. Si $a_0 = 0$, L puede escribirse $L(x) = (\sum_{i=1}^{n} a_i x^{q^{i-1}})^q$, y en caso necesario repetimos el razonamiento hasta alcanzar $L(x) = g(x)^{q^k}$ con g sin raíces múltiples.

Para ii) notar que U no es más que el núcleo de la evaluación $L : F \to \mathbb{F}$. Además contiene $\frac{q^n}{q^k} = q^{n-k}$ elementos. □

La propiedad establecida en este teorema es exclusiva de los polinomios linearizados. Enunciamos sin demostración este hecho.

Teorema 1.46. *Si $L \in \mathbb{F}_{q^m}[x]$, F su cuerpo de escisión y U su conjunto de raíces verifican i) y ii) en el Teorema anterior, entonces L es linealizado.*

Demostración. Ver [Lidl and Niederreiter, 1994]. □

Notamos que en el Teorema 1.45 no se necesita realmente que L se escinda en F para determinar que el conjunto de raíces de L en F sea un subespacio vectorial. Es por ello que la búsqueda de raíces de un polinomio linealizado sobre $\mathbb{F}_{q^m}$ en cualquier extensión $F : \mathbb{F}_{q^m}$ es tarea sencilla. Basta calcular la matriz B asociada a L respecto de alguna base de F y resolver un sistema homogéneo. Pongamos que $F = \mathbb{F}_q{}^r$ con $m \mid r$. Por simplicidad, supongamos que hemos construido F mediante un polinomio irreducible $f \in \mathbb{F}_q[x]$ y $\theta \in F$ una raíz suya. Entonces $\{1, \theta, \theta^2, ..., \theta^{s-1}\}$ es una base de F. Para cada $j = 0, ..., s-1$ calculamos $L(\theta^j)$, y lo reducimos módulo $f(\theta) = 0$ para obtener una expresión de la forma $L(\theta^j) = \sum_{i=0}^{s-1} b_{ij}\theta^j$. Tomando $B = (b_{ij})_{i,j=0,...,s-1}$, basta calcular el núcleo de B resolviendo un sistema homogéneo para obtener todas sus raíces.

De la misma manera, puede observarse que si tomamos un elemento $c \in \mathbb{F}_{q^m} \subseteq F_{q^s}$ con $c = \sum_{j=0}^{s-1} c_j\theta^j$ y $c_j \in \mathbb{F}_q$ para $j = 0, ..., s-1$, y si denotamos $w = (c_0, ..., c_{s-1})^t$, entonces una solución del sistema $Bv = w$, digamos $v = (d_0, ..., d_{s-1})^t \in \mathbb{F}_{q^m}^s$, proporciona un elemento $\alpha = \sum_{j=0}^{s-1} d_j\theta^j$ verificando que $L(\alpha) = c$. Dicho de otro modo, α es raíz del polinomio $A(x) = L(x) - c$. Obtenemos una familia un poco más grande de polinomios para la cual sabemos calcular sus raíces:

Definición 1.47. *Un polinomio $A(x) = L(x) - c \in \mathbb{F}_{q^m}[x]$ donde L es linealizado y $c \in \mathbb{F}_q$ se llama* ***polinomio afín****.*

A priori no parece una mejora muy significativa. El siguiente teorema de demostración constructiva es clave para dotar a los polinomios afines de una importancia vital en nuestra tarea:

Teorema 1.48. *Todo polinomio sobre $\mathbb{F}_{q^m}$ no trivial es divisor de un polinomio afín sobre $\mathbb{F}_{q^m}$.*

Demostración. Sea $f \in \mathbb{F}_{q^m}[x]$ con $n = grado(f) \geq 1$. Aplicando el algoritmo de la división euclidea, tomamos para cada $i = 0, ..., n-1$ el único polinomio mónico de grado menor que n, $r_i(x)$, tal que $f \mid x^{q^i} - r_i(x)$, esto es, $x^{q^i} \equiv r_i(x) \pmod{f}$. Escribimos estos polinomios como $r_i(x) = \sum_{j=0}^{n-1} s_{ij} x^j$ con $s_{ij} \in \mathbb{F}_{q^m}$. Consideremos el sistema de ecuaciones:

$$\sum_{i=0}^{n-1} s_{ij} y_i = 0 \qquad \text{para todo } j \in \{1, ..., n-1\}.$$

Dicho sistema tiene n incógnitas $y_0, ..., y_{n-1}$ y $n-1$ restricciones, luego existe una solución no trivial, digamos $(\alpha_0, ..., \alpha_{n-1})$. Entonces el polinomio $r(x) = \sum_{i=0}^{n-1} \alpha_i r_i(x)$ es una constante. Sustituyendo:

$$r(x) = \sum_{i=0}^{n-1} \alpha_i \sum_{j=0}^{n-1} s_{ij} x^j = \sum_{j=0}^{n-1} \left(\sum_{i=0}^{n-1} s_{ij} \alpha_i \right) x^j = \sum_{i=0}^{n-1} s_{i0} \alpha_i =: c \in \mathbb{F}_{q^m}.$$

Haciendo una combinación lineal de las congruencias $x^{q^i} \equiv r_i(x) \pmod{f}$ obtenemos:

$$L(x) := \sum_{i=0}^{n-1} \alpha_i x^{q^i} \equiv r(x) = c \pmod{f},$$

estando el polinomio L así definido linealizado. Por tanto $f \mid L(x) - c =: A(x)$. □

El método para obtener las raíces de f es directo: basta calcular mediante el procedimiento descrito en la demostración anterior un múltiplo afín suyo, A, y resolver el sistema lineal asociado. Las soluciones de dicho sistema conformarán el conjunto de raíces de A, en el cual están contenidas las raíces de f. Basta aplicar un filtrado para comprobar cuales de entre todas ellas anulan en efecto a f.

Este método de obtención de raíces justifica la introducción de esta clase de polinomios por su importancia en el problema de factorización que trataremos en el capítulo próximo. Sin embargo, no podemos terminar esta sección dedicada a los polinomios linealizados obviando un hecho particularmente interesante:

Proposición 1.49. *Sea el conjunto $\mathcal{L}_q$ de polinomios linealizados sobre $\mathbb{F}_q$. Entonces $\mathcal{L}_q$ forma un dominio de integridad junto con la suma y la composición.*

Demostración. Sean $L_1, L_2 \in \mathcal{L}_q$ con $L_1(x) = \sum_{i=0}^{n} a_i x^{q^i}$ y $L_2(x) = \sum_{j=0}^{m} b_j x^{q^i}$. Usando que $a^q = a \ \forall a \in \mathbb{F}_q$ junto con el automorfismo de Frobenius, se puede llegar fácilmente a la fórmula:

$$L_2(L_1(x)) = \sum_{j=0}^{m} \sum_{i=0}^{n} a_i b_j x^{q^{i+j}}.$$

A partir de ella, demostrar la proposición se trata de comprobaciones simples. □

Por tanto en $\mathcal{L}_q$ conviven dos nociones distintas de divisibilidad:

Definición 1.50. *Sean $L_1, L_2 \in \mathcal{L}_q$. Denotaremos $L_2(L_1(x)) = L_2 \otimes L_1$. Llamamos a esta operación* ***producto simbólico****. Si existe $L_3 \in \mathcal{L}_q$ tal que $L_1 = L_2 \otimes L_3$, diremos que L_2* ***divide simbólicamente*** *a L_1.*

Se abre de esta manera la puerta al estudio de la divisibilidad simbólica en $\mathcal{L}_q$. Dicha tarea no es trivial, puesto que el producto habitual y el producto simbólico son esencialmente distintos. Sin embargo, podemos establecer un nexo de unión natural a partir de la siguiente definición:

Definición 1.51. *Sea $l(x) = \sum_{i=0}^{n} a_i x^i \in \mathbb{F}_q[x]$. El polinomio $L(x) = \sum_{i=0}^{n} a_i x^{q^i}$ se dice* ***asociado lineal de*** *l, y l se dice* ***asociado convencional de*** *L. Se dice que ambos polinomios son* ***asociados****.*

El producto simbólico se relaciona con el producto convencional a través de los asociados:

Proposición 1.52. *Sean $l_1, l_2 \in F_q[x]$ y L_1, L_2 sus respectivos asociados lineales. Entonces el asociado lineal de $l = l_1 l_2$ es $L = L_1 \otimes L_2$. Como consecuencia, dos polinomios linealizados sobre $\mathbb{F}_q$ se dividen simbólicamente si y solo si sus asociados convencionales se dividen.*

Demostración. Si $l_1(x) = \sum_{i=0}^{n} a_i x^i$ y $l_2(x) = \sum_{j=0}^{m} b_j x^j$, entonces por un lado:

$$l(x) = l_1 l_2(x) = \sum_{k=0}^{m+n} (\sum_{i+j=k} a_i b_j) x^k$$

y por otro, reescribiendo la expresión descrita en la Proposición 1.49:

$$L(x) = L_1 \otimes L_2(x) = \sum_{k=0}^{m+n} \sum_{i+j=k} a_i b_j x^{q^k},$$

luego L es el asociado lineal de l. La consecuencia del enunciado es obvia. □

Ejemplo 1.53. *Por último, veamos un ejemplo de cómo la relación entre las dos nociones de divisibilidad sirve para relacionar las raíces de un cierto tipo de polinomio, entre los que se encuentra la traza, con las imágenes de otro más sencillo. Consideramos $F(x) = \prod_{\beta \in \mathbb{F}_{q^m}} (x - \beta)$ para alguna extensión $\mathbb{F}_{q^m} : \mathbb{F}_q$. Sabemos que $F(x) = x^{q^m} - x$, luego F es un polinomio linealizado sobre $\mathbb{F}_q$. Sea un divisor simbólico suyo L. Entonces existe $G \in \mathcal{L}_q$ tal que $F(x) = G \otimes L = G(L(x))$. Así pues, las raíces de F, es decir, los elementos de $\mathbb{F}_{q^m}$, coinciden con las raíces β de $L(x) - \alpha$ donde α es una raíz de G. Por tanto las raíces α de G son el conjunto $\{L(\beta) : \beta \in \mathbb{F}_{q^m}\}$. La proposición anterior entra en juego a la hora de calcular el polinomio G a partir de L y viceversa, pues G no es más que el asociado lineal de $\dfrac{x^m - 1}{l(x)}$ donde l es el asociado convencional de L.*

Concretemos para el caso de $Tr_{\mathbb{F}_{q^m} : \mathbb{F}_q}(x) = x + x^q + ... + x^{q^{m-1}}$. Como $x^m - 1 = (x-1)(1+x+...+x^{m-1})$, tomando asociados obtenemos $x^{q^m} - x = (x^q - x) \otimes Tr_{\mathbb{F}_{q^m} : \mathbb{F}_q}(x)$, y aplicando el razonamiento anterior se tiene que $KerTr_{\mathbb{F}_{q^m} : \mathbb{F}_q} = \{\beta^q - \beta : \beta \in \mathbb{F}_{q^m}\}$.

Capítulo 2

Factorización de polinomios sobre cuerpos finitos

A lo largo de las secciones anteriores ha quedado patente la importancia de disponer de algoritmos eficaces para la factorización de polinomios. Tanto la construcción de extensiones de cuerpos como la obtención de polinomios primitivos depende de esta tarea. Factorizar polinomios ciclotómicos resulta de vital importancia a la hora de generar códigos cíclicos. Pero además, fuera del alcance de este trabajo está el cifrado en flujo y la generación de secuencias pseudoaleatorias, para las cuales la factorización de polinomios sobre cuerpos finitos juega un papel fundamental. Además, ser capaces de realizar estas factorizaciones resulta útil para factorizar polinomios sobre otras estructuras algebraicas, como sobre el anillo de números enteros.

Es por todo ello que este es el capítulo central del texto. Las técnicas aquí expuestas fueron desarrolladas principalmente por **Elwyn Ralph Berlekamp y Hans Zassenhaus** a lo largo de las década de 1960. [Berlekamp, 1967] [Zassenhaus, 1969]

2.1. El Algoritmo de Berlekamp

Es bien sabido que el problema de factorización de un entero en sus factores primos nos es computacionalmente inasumible según vamos involucrando números grandes. Ni aún suponiendo que dicho entero N sea de la forma $N = pq$ con p, q números primos conocemos ningún algoritmo polinomial en el tamaño de la entrada que lo factorice. La seguridad del criptosistema de clave pública RSA depende de esta imposibilidad. Sin embargo, al cambiar el dominio de factorización única $\mathbb{Z}$ por $\mathbb{F}_q[x]$, podemos usar nuestro conocimiento detallado sobre la estructura de los cuerpos finitos en combinación con técnicas del Álgebra Lineal para obtener algoritmos decentes en esta tarea. El primero de ellos es el algoritmo de Berlekamp.

Reduciendo el problema: Antes de enunciar el resultado en el cual se basa el método, hemos de comprender que podemos centrar nuestra atención en factorizar polinomios sin factores irreducibles repetidos. En concreto, sea $f \in \mathbb{F}_q[x]$ mónico no trivial, escrito de forma única como $f = f_1^{e_1} ... f_k^{e_k}$ como producto de factores irreducibles en $F_q[x]$, y supongamos que conocemos un método eficiente, B, que en efecto factorice polinomios sin factores no triviales repetidos. Sea entonces $d = mcd(f, f')$, computado sin dificultad mediante el algoritmo de Euclides. Se presentan tres alternativas:

1. $d = 1$, en cuyo caso sabemos que f no tiene factores repetidos y es apto para ser sometido a B.

2. $d = f$, lo que implica $f \mid f'$ y al ser $grado(f') < grado(f)$ necesariamente $f' = 0$ lo cual es solo posible si $f = g^p$ donde $g \in \mathbb{F}_q[x]$ y $p = car(\mathbb{F}_q)$. g se obtiene de forma trivial gracias al automorfismo de Frobenius, y el proceso volvería a empezar aplicado a g. Como en particular $grado(g) < grado(f)$, este bucle no se puede repetir indefinidamente.

3. $0 < grado(d) < grado(f)$. Entonces $\frac{f(x)}{d(x)}$ no tiene factores repetidos, luego le aplicamos el método B. Por otro lado, factorizaríamos $d(x)$ por este proceso recursivo, y dado que $grado(d) < grado(f)$, nos detendríamos eventualmente. La factorización de f es trivialmente reconstruida a partir de las de f/d y d.

Nos centraremos pues en **factorizar polinomios de la forma $f = f_1...f_k$ con f_i irreducible y mónico para $i = 1, ..., k$ y distintos dos a dos**. El siguiente es un teorema de demostración sencilla que nos ofrece una factorización parcial:

Teorema 2.1. *Sea $f \in \mathbb{F}_q[x]$ mónico y $h \in \mathbb{F}_q[x]$ tal que $h \equiv h^q \pmod f$. Entonces:*

$$f(x) = \prod_{c \in \mathbb{F}_q} mcd(f(x), h(x) - c). \tag{2.1}$$

Demostración. Primero veamos que $\prod_{c\in\mathbb{F}_q} mcd(f(x), h(x) - c) \mid f(x)$. Fijando $c \in \mathbb{F}_q$, se tiene que $mcd(f(x), h(x) - c) \mid f(x)$. Además, si $c, d \in \mathbb{F}_q$ son distintos, y $g \in \mathbb{F}_q[x]$ divide tanto a $h(x) - c$ como a $h(x) - d$ podemos escribir:

$$\left.\begin{array}{l} h(x) - c = g(x)g_1(x) \\ h(x) - d = g(x)g_2(x) \end{array}\right\}$$

para ciertos g_1, g_2. Combinando ambas expresiones, tenemos $0 \neq d - c = g(g_1 - g_2)$, y obtenemos que necesariamente g es de grado 0. Es decir, $mcd(h(x) - c, h(x) - d) = 1$ para todo $c \neq d \in \mathbb{F}_q$. Por lo tanto, también todos los factores del productorio son relativamente primos dos a dos, luego el producto de todos ellos divide a f.

Probemos ahora que $f(x) \mid \prod_{c\in\mathbb{F}_q} mcd(f(x), h(x) - c)$. Sabemos por (1.1) que el polinomio $g = x^q - x$ puede factorizarse como $g(x) = \prod_{c\in\mathbb{F}_q}(x - c)$. Por tanto, $h(x)^q - h(x) = g(h(x)) = \prod_{c\in\mathbb{F}_q}(h(x) - c)$. Por hipótesis, $f(x) \mid h(x)^q - h(x)$, luego $f \mid \prod_{c\in\mathbb{F}_q}(h(x) - c)$. Si en cada factor $h(x) - c$ del producto eliminamos los factores que no dividen a f (es decir, tomamos el máximo común divisor) obtenemos finalmente que $f(x) \mid \prod_{c\in\mathbb{F}_q} mcd(f(x), h(x) - c)$. □

Observación 2.2. *No cualquier polinomio h en las condiciones del teorema anterior nos resultará útil. Por ejemplo, si $f(x) \mid h(x) - a$ para algún $a \in \mathbb{F}_q$, la factorizacion (2.1) es*

trivial. Por el contrario, si $1 \leq grado(h) < grado(f)$*, entonces* $grado(mcd(f(x), h(x) - c)) < grado(f)$ *para todo* $c \in \mathbb{F}_q$*, y habremos avanzado en nuestro objetivo. Estos polinomios caen dentro de la siguiente definición:*

Definición 2.3. *Decimos que un polinomio* $h \in \mathbb{F}_q[x]$ *con* $h^q \equiv h \pmod{f}$ ***reduce*** f *si la factorización (2.1) es no trivial.*

La estrategia a seguir consistirá en buscar polinomios que reduzcan f. Nuestra esperanza está fundamentada:

Proposición 2.4. *Sea* $f = f_1 ... f_k$ *sin factores irreducibles repetidos. Entonces existen exactamente* q^k *polinomios* $h \in \mathbb{F}_q[x]$ *con* $grado(h) < grado(f)$ *satisfaciendo:*

$$h^q \equiv h \pmod{f} \tag{2.2}$$

Demostración. Elegimos k elementos de $\mathbb{F}_q$ libremente, digamos $c_1, ..., c_k$. Entonces, dado que $mcd(f_i, f_j) = 1$ para todo $i \neq j = 1, ..., k$, el Teorema Chino de los Restos nos garantiza la existencia de un polinomio $h \in \mathbb{F}_q[x]$ único módulo f (único si lo tomamos con $grado(h) < grado(f)$) tal que:

$$\begin{cases} h \equiv c_1 \pmod{f_1} \\ h \equiv c_2 \pmod{f_2} \\ \vdots \\ h \equiv c_k \pmod{f_k} \end{cases} \tag{2.3}$$

Así pues, para todo $i = 1, ...k$: $h^q \equiv c_i^q = c_i \equiv h \pmod{f_i}$. De nuevo por $mcd(f_i, f_j) = 1$ para todo $i \neq j$, se tiene $h^q \equiv h \pmod{f}$. También notamos que una elección distinta de los $c_1, ..., c_k$ proporciona una solución distinta, luego obtenemos, en total q^k polinomios distintos satisfaciendo la condición. Pero no hay más. Si h verifica $h^q \equiv h \pmod{f}$, entonces:

$$f_1 ... f_k = f \mid h^q - h = \prod_{c \in \mathbb{F}_q} (h(x) - c),$$

luego para cada $i \in \{1, ...k\}$, por ser f_i irreducible, existe $c_i \in \mathbb{F}_q$ tal que $f_i(x) \mid h(x) - c_i$, esto es, $h \equiv c_i \pmod{f_i}$. Es decir, existen $c_1, ..., c_k \in \mathbb{F}_q$ tales que se satisfacen las ecuaciones (2.3). Por tanto, si $grado(h) < grado(f)$, por unicidad h ha de coincidir con la solución construida mediante el Teorema Chino de los Restos para la elección $c_1, ..., c_k$. □

Como los factores $f_1, ..., f_k$ son desconocidos, no podemos usar la demostración de esta proposición como método para resolver la ecuación (2.2). La idea central en el algoritmo de Berlekamp consiste en crear una matriz B que tras ser aplicada al vector de coeficientes de un polinomio g con $grado(g) < n = grado(f)$, el vector resultante sea el correspondiente a los coeficientes de g^q reducido módulo f. De esta manera, podremos ver la ecuación anterior como un sistema de ecuaciones lineales. Dicha matriz puede ser construida fácilmente. Calculamos para cada $0 \leq j \leq n-1$ el resto de dividir x^{jq} entre f, que puede ser escrito como:

$$x^{jq} \equiv \sum_{i=0}^{n-1} b_{ij} x^i \pmod{f}.$$

Basta tomar como matriz $B = (b_{ij})_{i,j=0,...,n-1}$. Así, si

$$g(x) = \sum_{j=0}^{n-1} a_j x^j = (1, x, ..., x^{n-1}) \begin{pmatrix} a_0 \\ a_1 \\ ... \\ a_{n-1} \end{pmatrix},$$

entonces, por el automorfismo de Frobenius y 1.7:

$$g(x)^q = \sum_{j=0}^{n-1} a_j^q x^{jq} = \sum_{j=0}^{n-1} a_j x^{jq} \equiv \sum_{j=0}^{n-1} a_j \sum_{i=0}^{n-1} b_{ij} x^i = \sum_{i=0}^{n-1} \left(\sum_{j=0}^{n-1} b_{ij} a_j \right) x^i =$$

$$= (1, x, ..., x^{n-1}) B \begin{pmatrix} a_0 \\ a_1 \\ ... \\ a_{n-1} \end{pmatrix} \pmod{f}.$$

Por tanto, buscar soluciones de (2.2) consiste en buscar soluciones $v \in \mathbb{F}_q^n$ de $Bv = v$ o, equivalentemente, de $(B - I)v = 0$. Por la proposición anterior, sabemos que existen exactamente q^k soluciones, donde k es el número de factores irreducibles de $f = f_1...f_k$. Así por tanto, $k = dim(Ker(B-I))$. Pero hemos de notar que k nos es, a priori, desconocido: no sabemos cuantos factores irreducibles forman f. Ahora, tras la construcción de B, podemos (mediante un proceso de escalonamiento con operaciones elementales por filas), calcular $r = rango(B - I)$ y obtener $k = n - r$. De esta manera, nos habremos acercado a la resolución del sistema lineal a la vez que obtenemos una información muy valiosa: el conocimiento de k nos proporciona un **criterio de parada para el algoritmo:** una vez hayamos factorizado f en k factores no triviales, habremos terminado.

La resolución del sistema lineal $(B - I)v = 0$ tiene, por construcción, una solución trivial: $v = (1, 0, ..., 0)^t$. Obviamente su polinomio asociado, así como el de cualquiera de sus múltiplos, no reduce f. Si $k = 1$ no habría problema, pues f ya sería irreducible y, en otro caso, encontraríamos mediante la resolución del sistema una base $\{1, v_2, ..., v_k\}$ de $ker(B - I)$. Para cualquier $i = 2, ..., k$, es claro que $h_i(x) = (1, x, ..., x^{n-1})v_i$ tiene grado mayor que cero y menor que n, luego sí reduce f. En particular h_2 reduce f. Posteriormente calcularíamos $mcd(f(x), h_2(x) - c)$ para cada $c \in \mathbb{F}_q$ y, en virtud del Teorema 2.1 y de la Observación 2.2 obtendríamos una factorización de f en al menos dos factores no triviales.

Parecería natural ahora aplicar el mismo proceso a cada uno de estos factores hasta obtener k en total. Sin embargo, este procedimiento supondría, en el peor de los casos, tener que resolver $k - 1$ sistemas lineales, lo cual puede llegar a ser computacionalmente inasumible según el número de factores crece. Por fortuna, esta limitación puede ser solventada si nos limitamos a aprovechar los polinomios $\{h_2, ..., h_k\}$ que reducían f en primera instancia para reducir a su vez sus factores en su factorización parcial. De esta manera, habremos necesitado resolver tan solo un sistema lineal. En concreto:

Proposición 2.5. *Sean* $f = f_1 ... f_k \in \mathbb{F}_q[x]$ *con* $k \geq 2$ *y* $h_2, ..., h_k \in \mathbb{F}_q[x]$ *construidos mediante el procedimiento descrito. Sean* $s, t \in \{1, 2, ..., k\}$, $s \neq t$. *Entonces existen* $j \in \{2, ..., k\}$ *y* $c' \in \mathbb{F}_q$ *tales que* $f_s(x) \mid h_j(x) - c'$ *pero* $f_t(x) \nmid h_j(x) - c'$.

Demostración. Podemos utilizar el mismo razonamiento que en la segunda parte de la demostración de 2.4 para afirmar que $\forall l \in \{1, ..., k\}$ existen $c_{ls}, c_{lt} \in \mathbb{F}_q$ tales que

$$h_l \equiv c_{ls} \pmod{f_s}$$

$$h_l \equiv c_{lt} \pmod{f_t}.$$

La idea consiste en tomar $j \in \{1, ..., k\}$ tal que $c_{js} \neq c_{jt}$. Veamos que dicho j existe. Para ello, supongamos que $c_{ls} = c_{lt} =: c_l \ \forall l \in \{1, ..., k\}$. Por un lado, hemos visto en 2.4 que gracias al Teorema Chino de los Restos existe una solución de (2.2) tal que

$$h \equiv 0 \pmod{f_s}$$

$$h \equiv 1 \pmod{f_t}.$$

Sin embargo, toda solución h de (2.2) es combinación lineal de la base $\{1, h_2, ..., h_k\}$ de $Ker(B - I)$. Así por tanto existen $a_1, ..., a_k \in \mathbb{F}_q$ tales que $h = \sum_{l=1}^{k} a_l h_l$. Tomando módulos resulta:

$$h \equiv \sum_{l=1}^{k} a_l c_{ls} = \sum_{l=1}^{k} a_l c_l =: c \pmod{f_s}$$

$$h \equiv \sum_{l=1}^{k} a_l c_{lt} = \sum_{l=1}^{k} a_l c_l = c \pmod{f_t}.$$

Ahora bien, ambos sistemas de congruencias son incompatibles (implicarían $c \equiv 0 \pmod{f_s}$ y $c \equiv 1 \pmod{f_t}$, que implica $0 = c = 1$).

Sea entonces $j \in \{1, ..., k\}$ tal que $c_{js} \neq c_{jt}$. Entonces $h_j \equiv c_{js} \pmod{f_s}$, esto es, $f_s(x) \mid h_j(x) - c_{js}$. Pero $f_t(x) \mid h_j(x) - c_{jt}$ con $c_{jt} \neq c_{js}$, y entonces $f_t(x) \nmid h_j(x) - c_{js}$.

□

Consecuentemente, si f_r y f_s son factores irreducibles de f, y hemos conseguido factorizar parcialmente f como $f = g_1...g_l$ $(l \leq k)$ mediante el polinomio reductor h_2, entonces o bien ya hemos separado f_r y f_s, o bien ambos se encuentran agrupados, por ejemplo en g_1. En tal caso, esta proposición nos asegura que encontraremos en $\{h_3, ..., h_k\}$ un polinomio que reduzca g_1 logrando separar f_r y f_s en la factorización.

Este razonamiento asegura que realizando la factorización (2.1) sucesivamente con los polinomios reductores $\{h_2, ..., h_k\}$, eventualmente se logra separar f en sus k factores irreducibles. Esto concluye la justificación teórica del Algoritmo de Berlekamp. En el Apéndice A hemos incluido una implementación del mismo, puesto que será necesario para resolver el ejemplo de cifrado de McEliece con el que cerramos el trabajo.

2.2. Factorización en cuerpos finitos grandes

Limitaciones del Algoritmo de Berlekamp: hay un paso en el algoritmo de Berlekamp que resulta problemático en ciertos contextos, y que impide que este método sea útil en todas las circunstancias. Cada vez que se encuentra un polinomio h que reduzca f (o alguno de sus factores), nos vemos en la obligación de calcular q máximos comunes divisores. Esto no es ningún problema a la hora de factorizar polinomios en cuerpos *pequeños*, como por ejemplo el tan recurrido $\mathbb{F}_2$. Sin embargo, si el tamaño del cuerpo es muy superior al grado del polinomio, este paso se convierte en el mayor problema.

No obstante, existen varios métodos para sortear este escollo. Nosotros expondremos el algoritmo de Zassenhaus. Con él, podremos aprovecharnos del trabajo ya hecho: reduciremos el problema de caracterizar los elementos $c \in \mathbb{F}_q$ tales que $mcd(f(x), h(x) - c) \neq 1$ al de **calcular las raíces de un polinomio**, labor ya tratada en el capítulo anterior.

De nuevo, trabajaremos con $f = f_1...f_k \in \mathbb{F}_q[x]$ libre de cuadrados. El valor de k y los polinomios reductores h_i se calculan mediante la primera parte del algoritmo de Berlekamp. Fijamos h que reduzca f. Podemos reescribir el Teorema 2.1 eliminando los

factores triviales:

$$f(x) = \prod_{c \in C} mcd(f(x), h(x) - c) \tag{2.4}$$

donde $C = \{c \in \mathbb{F}_q : mcd(f(x), h(x) - c) \neq 1\}$. En particular $f(x) \mid \prod_{c \in C}(h(x) - c) = G(h(x))$ donde definimos G como el polinomio que tiene justamente por raíces los elementos de C, esto es $G(y) = \prod_{c \in C}(y - c) \in \mathbb{F}_q[y]$.

El algoritmo de Zassenhaus mejora el de Berlekamp mediante:

1. El cálculo de G.

2. La obtención de las raíces de G.

Para llevar a cabo el segundo paso ya hemos desarrollado un método basado en la construcción de un múltiplo afín para G. Pero de nada nos sirve esto si no sabemos calcular G eficientemente. Gracias al siguiente teorema, es tarea sencilla:

Teorema 2.6. *Se tiene que:*

1. *El polinomio $G \in \mathbb{F}_q[y]$ así definido es el único polinomio mónico de menor grado $g \in \mathbb{F}_q[y]$ tal que $f(x) \mid g(h(x))$.*

2. *$m := \deg(G) \leq k$, y m es el menor natural tal que la familia $\{1 + (f), h + (f), \ldots, h^m + (f)\}$ es ligada en $\mathbb{F}_q[x]/(f)$.*

Demostración. 1. Sea $I = \{g \in \mathbb{F}_q[y] : f \mid g(h(x))\}$. Como es claro que I es un ideal del dominio de ideales principales $F_q[y]$, se tiene que $I = (g_0)$ para algún $g_0 \in \mathbb{F}_q[y]$ mónico. Como $G \in I$, entonces $g_0 \mid G$. Denotando por C_0 al conjunto de raíces de g_0, se tiene que $C_0 \subseteq C$ y $g_0(y) = \prod_{c \in C_0}(y - c)$. Por definición de I:

$$f(x) \mid g_0(h(x)) = \prod_{c \in C_0} (h(x) - c),$$

y de la misma manera que en el Teorema 2.1:

$$f(x) \mid \prod_{c \in C_0} mcd(f(x), h(x) - c) \mid \prod_{c \in C} mcd(f(x), h(x) - c) = f(x),$$

lo cual solo es posible si $C = C_0$. En definitiva, $G = g_0$, G es el único generador mónico de I y cumple la propiedad pedida en **1.**

2. Sea $m = grado(G)$. Por definición de G, $m = card(C)$. Pero es claro por (2.4) y la definición de C que $card(C) \leq k$. Además la familia $\{1, h, h^2, ..., h^m\}$ es ligada en $\mathbb{F}_q[x]/(f)$. En efecto, si $G(y) = \sum_{i=0}^{m} a_i y^i$, con $a_i \in \mathbb{F}_q$ para $i = 0, ..., m$ y $a_m = 1$ entonces $f(x) \mid \sum_{i=0}^{m} a_i h(x)^i$ o equivalentemente $\sum_{i=0}^{m} a_i h(x)^i \equiv 0 \pmod{f}$. Pero la familia $\{1, h, h^2, ..., h^l\}$ con $l < m$ ha de ser libre. Para verlo, supongamos que existen $b_i \in \mathbb{F}_q$ para $i = 0, ..., l$ no todos nulos son tales que $\sum_{i=0}^{l} b_i h(x)^i \equiv 0 \pmod{f}$. Entonces el polinomio $\hat{G}(y) = \sum_{i=0}^{l} b_i y^i$ es no nulo y cumple $f(x) \mid \hat{G}(h(x))$. Su único asociado mónico tendría que ser G por **1.**, pero es de grado menor que G, alcanzando así una contradicción.

□

El método para encontrar G es claro: se considera primero h^2, se reduce módulo f para obtener $r_2 \equiv h^2 \pmod{f}$ con $grado(r_2) < n$ y se comprueba (con técnicas matriciales) si $\{1, h, r_2\}$ es linealmente dependiente. De no serlo, procedemos de igual manera para las potencias sucesivas de h hasta encontrar m el primer exponente tal que $\{1, h, r_2, ..., r_m\}$ es linealmente dependiente. Por **2.** del teorema anterior sabemos que dicho m coincide con el grado de G, y se encuentra antes de llegar a la potencia k-ésima. La relación de dependencia define al polinomio G de la manera vista en la demostración del teorema.

Como nota adicional, mencionamos que se han desarrollado métodos más eficientes en la práctica que hacen uso de métodos probabilísticos, como el algoritmo de Cantor-Zassenhaus [Cantor and Zassenhaus, 1981].

Capítulo 3

Aplicaciones

Bien es cierto que el interés por los Cuerpos Finitos surgió de manera independiente a la aparición de las importantísimas aplicaciones que estos han resultado tener en las últimas décadas. En cuanto aplicaciones teóricas, el estudio de los caracteres de un Cuerpo Finito conduce a una demostración de la Ley de Reciprocidad Cuadrática, uno de los principales resultados de Teoría de Números. Además, los Cuerpos Finitos pueden usarse para estudiar Geometrías Finitas, construir cuadrados latinos, u obtener resultados en combinatoria.

Sin embargo, es en su aplicación a las tecnologías de la información y comunicación donde los cuerpos finitos han resultado ser una herramienta imprescindible. La cantidad de datos que se transmiten cada segundo por multitud de medios de telecomunicación es inmensa. Los **Códigos Correctores de Errores y la Criptografía** pretenden garantizar la integridad, privacidad y autenticidad de dichos datos. Es más, veremos como el esquema de McElicee propone una unión entre ambas disciplinas, consiguiendo un criptosistema que (aún a día de hoy) se estima robusto frente a la presencia de un ordenador cuántico. Seremos capaces de comprender en detalle todo ello gracias a la teoría desarrollada sobre cuerpos finitos en las secciones precedentes.

3.1. Códigos Correctores

3.1.1. Motivación y definiciones. Códigos lineales.

El **objetivo de la teoría algebraica de códigos es proteger la información frente a ruido externo**. El envío de vídeo y audio depende de la transmisión de una gran cantidad de bits que son susceptibles de ser modificados en el medio. Los sistemas de almacenamiento (como los discos duros o CD) pueden verse dañados, existiendo el riesgo de que se pierda parte de la información. Es por ello que conviene *codificar* las cadenas de bits mediante la introducción de una pequeña redundancia que permita, en caso de pérdida de datos o de la existencia de ruido, recuperar la totalidad del mensaje. Mediante los Códigos Detectores y Correctores de Errores se pretende realizar esta tarea de forma óptima.

Nuestro interés en la teoría algebraica de códigos surge cuando consideramos transmitir palabras cuyas letras son elementos de un cuerpo finito. Es decir, el espacio de las posibles palabras emitidas será ${\mathbb{F}_q}^k$, donde k es la longitud de las palabras. Un código no es más que la imagen de una aplicación inyectiva $f : {\mathbb{F}_q}^k \longrightarrow {\mathbb{F}_q}^n$ donde $k < n$. Denotaremos $C = Imf$. Nos centraremos en códigos lineales, es decir, códigos donde dicha aplicación es lineal y diremos que k **es la dimensión del código y** n **su longitud**. Si queremos transmitir una palabra $a = a_1...a_k$ a través de un medio ruidoso, es decir, un medio por el que a puede verse alterada (sin intermediación de ningún adversario), no transmitiremos directamente a, sino su codificación $c = f(a)$. Notar que como $dim(C) < dim(\mathbb{F}_q^n)$, c contiene información redundante, la cual esperaremos que proteja la información contenida en a. El receptor, debido a que el medio ruidoso introduce un error e, recibirá en su lugar $\hat{c} = c + e \in \mathbb{F}_q^n$. Notar que no necesariamente $\hat{c} \in C$. De hecho, si el receptor comprueba que $\hat{c} \notin C$, es decir, que $\hat{c}$ no es una palabra código, necesariamente ha de ser $e \neq 0$ y se habrá **detectado** la existencia de un error en la transmisión, si bien a priori no sabremos cual. Para que el receptor pueda evaluar la cercanía o lejanía de la palabra recibida a

las palabras código y tratar así de **corregir** el error cometido es necesario dotar a $\mathbb{F}_q^k$ de estructura de espacio métrico:

Definición 3.1. *La aplicación* $wt : \mathbb{F}_q^n \longrightarrow \mathbb{R}$ *que tal que* $wt(a)$ *es el número de componentes no nulas de* a *se llama* ***función peso****, y* $wt(a)$ *es el peso de* a*. La aplicación*

$$d : F_q^n \times F_q^n \longrightarrow \mathbb{R}$$

dada por $d(a,b) = wt(a-b)$ *es una distancia, conocida como la* ***distancia de Hamming****. Es fácil comprobar, gracias a que* $d(a,b) = d(a-b,0)$*, que*

$$\min_{a \neq b \in C} d(a,b) = \min_{0 \neq c \in C} d(c,0) = \min_{0 \neq c \in C} wt(c).$$

Dicho mínimo se denota $d(C)$ *y se llama* ***distancia mínima del código lineal*** C*.*

Gracias a la distancia mínima de un código podemos establecer la bondad de un código para detectar y corregir errores. Supongamos que se recibe una palabra $\hat{c} = c + e$ a partir de $c \in C \subseteq \mathbb{F}_q^n$. Si suponemos que $t := wt(e) < d(C)$, esto es, se han cometido menos errores que la distancia mínima, se tendría que $d(\hat{c}, c) < d(C)$, luego $\hat{c}$ no puede ser una palabra código por definición de distancia mínima, y habríamos detectado la presencia de un error en la transmisión. Pero ¿cómo descodificamos? Es decir ¿cómo recuperamos c a partir de $\hat{c}$? El criterio que se sigue es el del **vecino más cercano**. Es decir, se descodifica calculando la palabra código c_1 más cercana a $\hat{c}$. Si $wt(e) \geq \frac{d(C)}{2}$, la decodificación podría ser incorrecta. En cambio, si $wt(e) < \frac{d(C)}{2}$, necesariamente ha de ser $c = c_1$, puesto que $d(c, c_1) \leq d(c, \hat{c}) + d(\hat{c}, c_1) \leq 2d(\hat{c}, c) = 2wt(e) < d(C)$ y ambas c y c_1 son palabras código. Es por ello que se dice que C **detecta** t **errores** si $t + 1 \leq d(C)$, y **C corrige** t **errores** si $2t + 1 \leq d(C)$.

Definición 3.2. *Dado un subespacio* C *de* $\mathbb{F}_q^n$ *de dimensión* k *y con* $d(C) = d$*, se dice que* C *es un* $[n,k,d]_q$*-código lineal.*

Existen fundamentalmente dos formas de describir un código: como la imagen de una **matriz generatriz** G de tamaño $k \times n$ y rango k, es decir $C = \{aG : a \in \mathbb{F}_q^k\}$, o bien

como el núcleo de una **matriz de control** H de tamaño $(n-k)\times n$ y rango máximo, $n-k$, es decir, $C=\{c\in\mathbb{F}_q^n : Hc^t=0\}$. A partir de una matriz de control de un código se puede conocer la distancia mínima de éste:

Lema 3.3. *Sea C un $[n,k,d]$-código lineal y H una matriz de control suya. Entonces: $d(C)\geq s+1$ si y solo si cualesquiera s columnas de H son linealmente independientes, o equivalentemente:*

$$d(C)=min\{l : \exists\ l\ columnas\ de\ H\ ligadas\}.$$

Demostración. Si existen s columnas de H ligadas, entonces

$$0=\sum_{j=1}^{s} c_{i_j}h_{i_j}$$

con c_{i_j} no todos nulos y h_{i_j} las columnas i_j-ésimas de H. Entonces la palabra código c con c_{i_j} en la posición i_j para cada $j=1,...,s$ y 0 en el resto, verifica que $wt(c)\leq s$ y además $Hc^t=0$, luego $d(C)\leq s$. Por tanto $d(C)\geq s+1$ implica que no existen tales s columnas, luego cualesquiera s columnas de H son linealmente independientes. Recíprocamente, si ningunas s columnas son ligadas, la existencia de una palabra código c no nula con $wt(c)\leq s$ sería contradictoria, pues obtendríamos una combinación lineal como la anterior, lo cual sería contradictorio. El enunciado equivalente no es más que una reformulación de este hecho. □

Damos ahora los dos ejemplos de códigos más sencillos.

Ejemplo 3.4. *Fijamos q y n y consideramos la siguiente matriz de tamaño $(n-1)\times n$ y rango $n-1$:*

$$H=\begin{pmatrix} -1 & 1 & 0 & \cdots & 0\\ -1 & 0 & 1 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ -1 & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Notamos que $C = KerH = \{c = (a, a, ..., a) : a \in \mathbb{F}_q\}$. *Bien por definición o usando el lema anterior se comprueba que su distancia mínima es* n, *y es claro ver que una matriz generatriz del mismo código es* $G = (1, ..., 1)$. *En este caso, se codifican mensajes de longitud* 1, $a \in \mathbb{F}_q$ *por la palabra código* $aaa...a$ *de longitud* n. *Es por ello que* C *se llama* ***código de repetición****.*

Ejemplo 3.5. *Fijamos ahora* $q = 2$ *y* k, *y consideramos la matriz de tamaño* $1 \times k+1$ *y rango uno* $H = (1, ..., 1)$. *Su núcleo* C *son las palabras* $a_1...a_{k+1}$ *tales que* $\sum_{i=1}^{k+1} a_i = 0$, *es decir las palabras con un número par de letras no nulas. La matriz de dimensión* $k \times (k+1)$*:*

$$G = \begin{pmatrix} 1 & 0 & \cdots & 0 & 1 \\ 0 & 1 & \cdots & 0 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 1 \end{pmatrix}$$

es una matriz generatriz para el mismo código. Como $(a_1, ..., a_k)G = (a_1, ..., a_k, \sum_{i=1}^{k} a_i)$ *Es decir, se codifica* $a_1...a_k$ *a través de un símbolo de control* $a_{k+1} = \sum_{i=1}^{k} a_i)$ *que certifica la paridad de la palabra código enviada. Es por ello que* C *se dice* ***código de control de paridad.*** *Su distancia mínima es* $d = 2$.

Algoritmo del líder. Las siguientes definiciones facilitan el proceso de descodificación de códigos lineales:

Definición 3.6. *Dado un código* C *de longitud* n *y dimensión* k *y* H *una matriz de control suya, se llama* ***síndrome*** *de* $x \in \mathbb{F}_q^n$ *a* $S(x) = Hx^t$. *Es fácil comprobar que* $S(c) = 0$ *si y solo si* $c \in C$, *y* $S(x) = S(y)$ *si y solo si* $x - y \in C$, *esto es,* $x + C = y + C$. *Es por ello que si* $y = c + e'$ *con* $c \in C$, *entonces* $S(y) = S(e')$. *Esto implica que se reduce el espacio de posibles errores entre los que buscar para descodificar* y*: ahora basta encontrar el error* e *que minimice* $wt(e)$ *entre* $\{e : S(e) = S(y)\} = \{e : e + C = y + C\} = \{y + \hat{c} : \hat{c} \in C\}$, *es decir, al representante de la clase* $y + C$ *con menor peso. Dicho representante se llama*

líder *de la clase. El algoritmo del líder para descodificar y consiste en calcular el líder e de la clase* $y + C$ *y descodificar por* $\hat{c} = y - e$*. La unicidad del líder y la corrección en la descodificación vienen garantizadas dentro de la capacidad correctora del código.*

Este proceso puede llevarse a cabo con relativa eficiencia construyendo lo que se denomina una **tabla de líderes**. Sin embargo, para códigos grandes la descodificación resulta muy difícil. Por este motivo es necesario introducir familias de códigos con una estructura algebraica más rica que exprime los resultados de las secciones anteriores al introducir polinomios sobre cuerpos finitos:

Definición 3.7. *Un* $[n, k, d]_q$*-código lineal* C *se dice* ***cíclico*** *si es invariante por ciclos, es decir, si* $(a_0, ..., a_{n-1}) \in C$ *implica* $(a_{n-1}, a_0, ..., a_{n-2}) \in C$*. Hacemos una identificación de cada palabra en* $\mathbb{F}_q^n$ *con un polinomio del espacio vectorial cociente* $V = \mathbb{F}_q[x]/(x^n - 1)$ *a través del isomorfismo* $\Psi \to V$ *dado por* $\Psi((a_0, ..., a_{n-1})) = a_0 + ... + a_{n-1}x^{n-1} + (x^n - 1)$*. Con esta identificación, rotar una palabra equivale a multiplicar su polinomio asociado por* x*. Es por ello que un código* C *es cíclico si y solo si* $x\Psi(c) \in \Psi(C)$ *para todo* $c \in C$*. Es decir:* ***los códigos cíclicos se identifican con los ideales de*** $\mathbb{F}_q[x]/(x^n - 1)$*, y en adelante identificaremos* $C = \Psi(C)$ *y trataremos indiferentemente sus elementos como polinomios o vectores según convenga.*

Como $x^n - 1$ no es irreducible, $\mathbb{F}_q[x]/(x^n - 1)$ no es un dominio de ideales principales. Sin embargo, sí existe una relación unívoca entre los ideales $I \trianglelefteq \mathbb{F}_q[x]/(x^n - 1)$ y los ideales $J \trianglelefteq \mathbb{F}_q[x]$ tales que $(x^n - 1) \subseteq J$ dada por $I = J/(x^n - 1)$. Como $\mathbb{F}_q[x]$ sí que es un dominio de ideales principales, y $(x^n - 1) \subseteq (g(x))$ si y solo si $g(x) \mid x^n - 1$, se tiene que **los códigos cíclicos son aquellos generados por los divisores de** $x^n - 1$**.** Un polinomio mónico $g \mid x^n - 1$ con $C = (g(x))$ (en el espacio cociente) se dice **generador del código cíclico C**. Es decir: encontrar códigos cíclicos consiste en factorizar el polinomio $x^n - 1$. Es este uno de los motivos por lo que hemos dedicado especial atención a esta labor. Sea $h(x)$ el polinomio mónico tal que $g(x)h(x) = x^n - 1$ ($gh \equiv 0 \pmod{x^n - 1}$, lo que para

nosotros será simplemente $gh = 0$). Dicho polinomio es el **polinomio de control de código.**

A partir de un polinomio generador de un código se puede obtener su dimensión y una matriz generatriz de manera inmediata. Sea C un código cíclico de longitud n y $g(x) = \sum_{i=0}^{n-k} x^i$ un polinomio generador suyo. Entonces la familia $g(x), xg(x), ..., x^{k-1}g(x)$ es libre en V, pero si añadimos $x^k g(x)$ no, puesto que $h(x)$ describe una combinación lineal de $\{g(x), xg(x), ..., x^k g(x)\}$ que se anula. Por esto, y como g genera C como ideal, dicha familia ha de generar C como espacio vectorial, de donde se concluye que C es de dimensión k. Además la matriz de tamaño $k \times n$:

$$G = \begin{pmatrix} g_0 & g_1 & \cdots & g_{n-k} & 0 & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & \cdots & g_{n-k} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & g_0 & g_1 & \cdots & g_{n-k} \end{pmatrix}$$

es una matriz generatriz de C. Notar que una palabra $a(x)$ (vista como polinomio de grado menor o igual a k) puede codificarse multiplicando $a(x)$ por $g(x)$ o, vista como vector, multiplicando por la matriz G. Por otro lado, si $h(x) = \sum_{i=0}^{k} h_i x^i$, entonces:

$$H = \begin{pmatrix} 0 & 0 & \cdots & 0 & h_k & h_{k-1} & \cdots & h_0 \\ 0 & 0 & \cdots & h_k & h_{k-1} & \cdots & h_0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots & \vdots & \vdots \\ h_k & h_{k-1} & \cdots & h_0 & 0 & 0 & \cdots & \cdots \end{pmatrix}$$

es una matriz de tamaño $(n-k) \times n$ de rango $n-k$ y tal que $HG^t = 0$, es decir, H es una matriz de control de C. Verificar si $y(x)$ pertenece a C puede hacerse multiplicando por la matriz de control o verificando si $y(x)h(x) \equiv 0 \pmod{x^n - 1}$.

El conocimiento de la estructura multiplicativa de los cuerpos finitos estudiada en las primeras secciones nos permite definir la siguiente familia de códigos cíclicos, los conocidos **códigos BCH**. Una de las ventajas que tienen es poder predefinir una distancia de

diseño $d \leq n$ de tal manera que la distancia del código construido sea al menos d. En el Teorema 1.12 hemos visto que si $mcd(n, q) = 1$, entonces $x^n - 1$ no tiene raíces múltiples. Impondremos en adelante esta condición. Escogemos una n-raíz primitiva de la unidad ξ y δ potencias consecutivas de este elemento: $\xi^b, \xi^{b+1}, ..., \xi^{b+\delta-1}$ (al conjunto de exponentes $\{b, ..., b+\delta-1\}$ se le llamará **conjunto definidor** del código). Tomamos el único polinomio mónico de grado mínimo $g(x)$ divisor de x^n-1 que las tiene a todas como raíces (el mínimo común múltiplo de los polinomios mínimos de dichas potencias). El código C que tiene a g como polinomio generador es el **Código BCH sobre $\mathbb{F}_q$ de longitud n y distancia de diseño δ.** Ha de verificarse que su distancia mínima d es en efecto $d \geq \delta$. El siguiente lema da una forma alternativa de verificar si una palabra pertenece a un código cíclico a través de una *pseudomatriz de control* (que no toma coeficientes en $\mathbb{F}_q$ sino en alguna extensión suya):

Lema 3.8. *Sea C un $[n, k, d]_q$-código cíclico con polinomio generador g, y sean $\alpha_1, ..., \alpha_{n-k}$ sus raíces. Entonces C está contenido en el núcleo de la matriz:*

$$\tilde{H} = \begin{pmatrix} 1 & \alpha_1 & \alpha_1^2 & \cdots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \alpha_2^2 & \cdots & \alpha_2^{n-1} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \alpha_{n-k} & \alpha_{n-k}^2 & \cdots & \alpha_{n-k}^{n-1} \end{pmatrix}$$

Demostración. $y(x) = y_0 + y_1 x + ... + y_{n-1}x^{n-1} \in C$ si y solo si $g \mid y$. Como g no tiene raíces múltiples, esto es equivalente a que todas las raíces de g lo sean también de y, lo cual se da si y solo si $\tilde{H}y^t = 0$. □

Observación 3.9. *Un código BCH estará contenido en el núcleo de una matriz similar anterior considerando las raíces $\alpha^b, ..., \alpha^{b+\delta-1}$. Con este hecho, el Lema 3.3 y usando la fórmula del determinante de una matriz de Vandermonde se puede probar el teorema:*

Teorema 3.10. *La distancia mínima d de un código BCH con distancia de diseño δ es $d \geq \delta$.*

Si se toma $b = 1$ se obtienen los códigos **BCH en sentido estricto** (narrow-sense BCH codes). Los códigos de **Reed-Solomon** son los códigos BCH de longitud $n = q - 1$.

Existen algoritmos específicos para la descodificación de códigos BCH. En la siguiente sección ahondamos en los códigos Goppa por su importancia en el criptosistema de McElicee. Estos son una generalización de los códigos BCH en sentido estricto, luego el algoritmo que veremos de descodificación de códigos Goppa permite descodificar estos códigos en particular.

3.1.2. Códigos Goppa

Definición 3.11. *Sea $m \in \mathbb{N}$, y $g \in \mathbb{F}_{q^m}[x]$ de grado t con $1 \leq t \leq n$. Sea también un conjunto $L = \{\gamma_0, ..., \gamma_{n-1}\} \subseteq \mathbb{F}_{q^m}$ de elementos distintos tales que ninguno de ellos es raíz de g. Entonces para todo $i = 0, ..., n-1$ está bien definido:*

$$S_i(x) = g(\gamma_i)^{-1} \frac{g(x) - g(\gamma_i)}{x - \gamma_i} \in \mathbb{F}_{q^m}[x].$$

*El **código Goppa sobre** $\mathbb{F}_q$ con **soporte** L y **polinomio Goppa** g es:*

$$\Gamma(L, g) = \{(c_0, ..., c_{n-1}) \in \mathbb{F}_q^n : \sum_{i=0}^{n-1} c_i S_i(x) = 0\} \tag{3.1}$$

Es claro que la condición anterior define un código lineal.

Ejemplo 3.12. *Si tomamos $g(x) = x^{d-1}$ y $L = \{1, \alpha^{-1}, ..., \alpha^{-(n-1)}\}$ donde α es una n-raíz primitiva de la unidad sobre $\mathbb{F}_q$, el código $\Gamma(L, g)$ coincide con el código BCH en sentido estricto con conjunto definidor $\{\alpha, ..., \alpha^{d-1}\}$.*

Para verlo se usa la siguiente fórmula, que se deduce fácilmente para cualquier anillo de polinomios sobre un cuerpo, $a \neq 0$:

$$\frac{x^k - a^k}{x - a} = \sum_{j=0}^{k-1} a^{k-1-j} x^j. \tag{3.2}$$

De esta manera, los polinomios S_i de la Definición 3.11 se escriben en este caso, tomando $a = \alpha^{-i}$ y $k = d-1$:

$$S_i(x) = \alpha^{i(d-1)} \sum_{j=0}^{d-2} \alpha^{-i(d-2-j)} x^j = \sum_{j=0}^{d-2} \alpha^{i(j+1)} x^j = \sum_{j=1}^{d-1} \alpha^{ij} x^{j-1}$$

Por tanto

$$\sum_{i=0}^{n-1} c_i S_i(x) = \sum_{i=0}^{n-1} \sum_{j=1}^{d-1} c_i \alpha^{ij} x^{j-1} = \sum_{j=1}^{d-1} (\sum_{i=0}^{n-1} c_i \alpha^{ij}) x^{j-1} = 0$$

si y solo si $\sum_{i=0}^{n-1} c_i \alpha^{ij} = 0$ para todo $j = 1, ..., d-1$, lo cual es equivalente a que $(c_0, ..., c_{n-1})$ pertenezca al código BCH en sentido estricto con los parámetros indicados (comparar con el Lema 3.8 y la Nota 3.9).

En lo que resta de capítulo los polinomios se consideran en $F_{q^m}[x]$. Buscamos ahora una pseudo-matriz de control similar a la dada en 3.8 para códigos Goppa en general, que nos ayude a determinar la dimensión y distancia mínima de estos. Denotando al polinomio Goppa $g(x) = \sum_{j=0}^{t} g_j x^j$ con soporte $L = \{\gamma_0, ..., \gamma_{n-1}\}$, y usando la fórmula (3.2) los polinomios de 3.11 se escriben:

$$S_i(x) = g(\gamma_i)^{-1} \frac{\sum_{j=0}^{t} g_j x^j - \sum_{j=0}^{t} g_j \gamma^j}{x - \gamma_i} = g(\gamma_i)^{-1} \sum_{j=0}^{t} g_j \frac{x^j - \gamma_i^j}{x - \gamma_i} =$$

$$= g(\gamma_i)^{-1} \sum_{j=0}^{t} g_j \sum_{s=0}^{j-1} \gamma_i^{j-1-s} x^s = g(\gamma_i)^{-1} \sum_{s=0}^{t-1} (\sum_{j=s+1}^{t} g_j \gamma_i^{j-1-s}) x^s.$$

Entonces $\sum_{i=0}^{n-1} c_i S_i(x) = 0$ si y solo si:

$$0 = \sum_{i=0}^{n-1} c_i \left(g(\gamma_i^{-1}) \sum_{j=s+1}^{t} g_j \gamma_i^{j-1-s} \right) = \sum_{j=s+1}^{t} g_j (\gamma_0^{j-1-s}, ..., \gamma_{n-1}^{j-1-s}) \left(\frac{c_0}{g(\gamma_0)}, ..., \frac{c_{n-1}}{g(\gamma_{n-1})} \right)^t =$$

$$= (g_{s+1}, ..., g_t, 0, ..., 0) \begin{pmatrix} \gamma_0^0 & \gamma_1^0 & \cdots & \gamma_{n-1}^0 \\ \gamma_0^1 & \gamma_1^1 & \cdots & \gamma_{n-1}^1 \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_0^{t-1} & \gamma_1^{t-1} & \cdots & \gamma_{n-1}^{t-1} \end{pmatrix} \begin{pmatrix} \frac{1}{g(\gamma_0)} & 0 & \cdots & 0 \\ 0 & \frac{1}{g(\gamma_1)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{g(\gamma_{n-1})} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{pmatrix}$$

para todo $s = 0, ..., t-1$. Es decir, el código será la intersección de $\mathbb{F}_q$ con el núcleo de:

$$\tilde{H}' = \begin{pmatrix} g_t & 0 & 0 & \cdots & 0 \\ g_{t-1} & g_t & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ g_1 & g_2 & g_3 & \cdots & g_t \end{pmatrix} \begin{pmatrix} \gamma_0^0 & \gamma_1^0 & \cdots & \gamma_{n-1}^0 \\ \gamma_0^1 & \gamma_1^1 & \cdots & \gamma_{n-1}^1 \\ \vdots & \vdots & \ddots & \vdots \\ \gamma_0^{t-1} & \gamma_1^{t-1} & \cdots & \gamma_{n-1}^{t-1} \end{pmatrix} \begin{pmatrix} \frac{1}{g(\gamma_0)} & 0 & \cdots & 0 \\ 0 & \frac{1}{g(\gamma_1)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{g(\gamma_{n-1})} \end{pmatrix}.$$

Sin embargo, dado que $g_t \neq 0$, la primera de las matrices en la factorización anterior es inversible. Por tanto, multiplicando las dos últimas, obtenemos que el núcleo de $\tilde{H}'$ coincide a su vez con el de la matriz:

$$\tilde{H} = \begin{pmatrix} \frac{1}{g(\gamma_0)} & \frac{1}{g(\gamma_1)} & \cdots & \frac{1}{g(\gamma_{n-1})} \\ \frac{\gamma_0}{g(\gamma_0)} & \frac{\gamma_1}{g(\gamma_1)} & \cdots & \frac{\gamma_{n-1}}{g(\gamma_{n-1})} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\gamma_0^{t-1}}{g(\gamma_0)} & \frac{\gamma_1^{t-1}}{g(\gamma_1)} & \cdots & \frac{\gamma_{n-1}^{t-1}}{g(\gamma_{n-1})} \end{pmatrix} \tag{3.3}$$

$\tilde{H}$ es una matriz de tamaño $t \times n$ con valores en $\mathbb{F}_{q^m}$, no en $\mathbb{F}_q$. No obstante, sabemos que cada elemento de $\mathbb{F}_{q^m}$ puede ser expresado como un vector de $\mathbb{F}_q^m$. Por ello, **a partir de $\tilde{H}$ se puede obtener una matriz H de tamaño $mt \times n$ tomando elementos en $\mathbb{F}_q$ con núcleo precisamente** $\Gamma(L, g)$. Dado que $r = rango(H) \leq mt$, $k = dim(KerH) = n - r \geq mt$. Se tiene entonces:

Teorema 3.13. *La dimensión de $\Gamma(L, g)$ es al menos $n - mt$ donde n es la longitud del código y t el grado de g. Además, su distancia mínima d es mayor que $t + 1$.*

La segunda parte del teorema puede probarse de nuevo calculando determinantes en las submatrices de $\tilde{H}$ y usando el Lema 3.3. Nosotros probaremos una mejor cota para la distancia mínima cuando nos restringimos a códigos Goppa construidos sobre $\mathbb{F}_2$, esto es, **códigos Goppa binarios**. En la práctica son los más utilizados, por ejemplo en el esquema de McElicee, que motiva la introducción de estos códigos en el trabajo.

Teorema 3.14. *La distancia mínima d de un código Goppa binario $\Gamma(L, g)$ con $L = \{\gamma_0, ..., \gamma_{n-1}\}$ construido sobre un polinomio g sin raíces múltiples de grado t cumple la cota $d \geq 2t + 1$.*

Demostración. Notemos que si $0 \neq c = (c_0, ..., c_{n-1})$ pertenece al código, y denotamos $w = wt(c) > 0$, entonces $\exists i_1, ..., i_w \in \{0, ..., n-1\}$ distintos tales que $c_{i_k} = 1$ para cada $k = 1, ..., w$ y $c_i = 0$ en el resto de posiciones. Esto permite simplificar la condición de (3.1) a: $\sum_{k=1}^{w} S_{i_k}(x) = 0$. Multiplicando esta expresión por el polinomio $f(x) = \prod_{j=1}^{w}(x - \gamma_{i_j})$ y reduciendo módulo g se obtiene:

$$0 = \sum_{k=1}^{w} g(\gamma_{i_k})^{-1} \frac{g(x) - g(\gamma_{i_k})}{x - \gamma_{i_k}} \prod_{j=1}^{w}(x - \gamma_{i_j}) = \sum_{k=1}^{w} g(\gamma_{i_k})^{-1}(g(x) - g(\gamma_{i_k})) \prod_{k \neq j=1}^{w} (x - \gamma_{i_j}) \equiv$$

$$\equiv -\sum_{k=1}^{w} \prod_{k \neq j=1}^{w} (x - \gamma_{i_j}) \pmod{g}.$$

Ahora bien, esta última expresión es igual a $-f'(x)$, luego se sigue que $g \mid f'$. Es una observación sencilla notar que, en característica dos, la derivada de ningún polinomio puede tener términos de potencia impar. Gracias al automorfismo de Frobenius, esto implica que f' es un cuadrado. Como $g \mid f'$ y g es libre de cuadrados por no tener raíces múltiples, se ha de dar que $g^2 \mid f'$. Como f tampoco tiene raíces múltiples por definición de L, se tiene $f' \neq 0$, y entonces $2t = grado(g^2) \leq grado(f') \leq w - 1$. La última desigualdad es por ser $grado(f) \geq 1$. Por tanto $w \geq 2t + 1$ para todo $0 \neq c \in \Gamma(L, g)$, lo que significa finalmente que $d \geq 2t + 1$. □

Observación 3.15. *Dado que los **polinomios irreducibles** no tienen raíces múltiples, estos se vuelven candidatos idóneos para la construcción de códigos Goppa. Dado $g \in \mathbb{F}_{2^m}[x]$ irreducible de grado t, se puede tomar $L = \mathbb{F}_{2^m}$ como soporte del código. Se obtiene así por tanto un código Goppa de longitud $n = 2^m$, dimensión $k \geq 2^m - mt$ y con distancia mínima $d \geq 2t + 1$, es decir, **capaz de corregir** t **y detectar** $2t$ **errores**. Además, existen algoritmos eficientes para su descodificación, como veremos a continuación.*

Decodificación de códigos Goppa. Supongamos que recibimos una palabra $y \in \mathbb{F}_q^n$ con un error de peso $r \leq t/2$ (es decir, menor que la capacidad correctora del código). Denotemos por $i_1, ..., i_r \in \{0, ..., n-1\}$ las posiciones de los errores, y por e_j el error cometido en la posición i_j. Denotamos como μ_j al elemento $\gamma_{i_j} \in L$ para cada $j = 1, ..., r$. A priori, tanto r como el valor de estos elementos nos son desconocidos. Nuestro objetivo es precisamente conocer dichos valores. Hacemos uso de las siguientes definiciones:

Definición 3.16. *A* $f(x) = \sum_{j=0}^{t-1} S_j x^j$*, donde* $S = (S_0, ..., S_{t-1})^t = \tilde{H} y^t$ *(con la matriz* $\tilde{H}$ *la matriz dada en (3.3)) se le llama* ***polinomio de síndromes****.*

El polinomio $s(x) = \prod_{j=1}^{r}(1 - \mu_j x)$ *recibe el nombre de* ***polinomio localizador de errores****.*

Por último, $u(x) = \sum_{j=1}^{r} e_j g(\mu_j)^{-1} \prod_{j \neq l=1}^{r}(1 - \mu_l x)$ *es el* ***polinomio evaluador de errores****.*

Observación 3.17. *Notar que si somos capaces de calcular el polinomio localizador de errores, su grado indicaría el número de errores cometidos, y calculando sus raíces y posteriormente sus inversas obtendríamos el índice de las posiciones de error. Si conociéramos además el polinomio evaluador de errores, para obtener el valor del error en la posición asociada a* μ_j*, calcularíamos:*

$$u(\mu_j^{-1}) = e_j g(\mu_j)^{-1} \prod_{j \neq l=1}^{r} (1 - \mu_l \mu_j^{-1}),$$

de donde, despejando:

$$e_j = u(\mu_j^{-1}) g(\mu_j) \prod_{j \neq l=1}^{r} (1 - \mu_l \mu_j^{-1}) \tag{3.4}$$

Sin embargo, f es el único de entre los polinomios anteriores que puede calcularse a priori. La fundamentación del método se encuentra en el siguiente resultado.

Proposición 3.18. ***Ecuación clave:*** *En* $\mathbb{F}_{q^m}[x]$ *se tiene:*

$$u(x) = f(x)s(x) \pmod{x^t}$$

Demostración. En esta prueba tomaremos elementos en el anillo de **series formales de potencias** $\mathbb{F}_{q^m}[[x]]$, es decir, usaremos expresiones de la forma $\sum_{i=0}^{\infty} a_i x^i \in \mathbb{F}_{q^m}[[x]]$ con $a_i \in \mathbb{F}_{q^m}$. En general, **las series formales de potencias sobre un anillo son inversibles si y solo si su término independiente es inversible**. Como todo polinomio puede verse como una serie formal de potencias, y $s(0) = 1$, se tiene que existe $\frac{1}{s(x)} \in \mathbb{F}_{q^m}[[x]]$. Entonces tiene sentido considerar el siguiente cociente:

$$\frac{u(x)}{s(x)} = \frac{\sum_{j=1}^{r} e_i g(\mu)^{-1} \prod_{j \neq l=1}^{r} (1 - \mu_l x)}{\prod_{l=1}^{r} (1 - \mu_l x)} = \sum_{j=1}^{r} e_j g(\mu_j)^{-1} \frac{1}{1 - \mu_j x}.$$

Ahora bien, sabido que en $\mathbb{F}_{q^m}[[y]]$ se tiene $(1-y)^{-1} = \sum_{i=0}^{\infty} y^i$, se obtiene que:

$$\frac{u(x)}{s(x)} = \sum_{j=1}^{r} e_j g(\mu_j)^{-1} \sum_{i=0}^{\infty} \mu_j^i x^i = \sum_{i=0}^{\infty} (\sum_{j=1}^{r} \frac{e_j \mu_j^i}{g(\mu_j)}) x^i.$$

Esta expresión puede dividirse como suma de un polinomio de grado t más una serie formal de potencias con menor exponente igual a t, es decir:

$$\frac{u(x)}{s(x)} = \sum_{i=0}^{t-1} (\sum_{j=1}^{r} \frac{e_j \mu_j^i}{g(\mu_j)}) x^i + x^t F(x),$$

con $F \in \mathbb{F}_{q^m}[[x]]$.

Como razonamos al exponer el algoritmo del líder, $\tilde{H} y^t = \tilde{H} e^t$ donde e denota el vector de errores. Por tanto, atendiendo a la expresión de $\tilde{H}$ en (3.3) y a la definición de μ_j para $j = 1, ..., r$, se tiene para cada $i = 0, ..., t-1$ que $S_i = \sum_{j=1}^{r} e_j \frac{\mu_j^i}{g(\mu_j)}$. Por tanto:

$$\frac{u(x)}{s(x)} = f(x) + x^t F(x) \Rightarrow u(x) = f(x)s(x) + x^t s(x) F(x).$$

Dado que el grado de u es finito, necesariamente ha de ser $F \in \mathbb{F}_{q^m}[x]$, luego $x^t \mid u(x) - f(x)s(x)$ en $\mathbb{F}_{q^m}[x]$, y se tiene la congruencia del enunciado.

□

La descodificación pasa por **encontrar una solución apropiada a dicha ecuación polinomial**. En [Sugiyama et al., 1975] se muestra como puede utilizarse el **algoritmo**

extendido de Euclides para, partiendo de $(x^t, f(x))$, encontrar sucesivos polinomios $(u_h(x), s_h(x))$ con $grado(u_{h+1}) < grado(u_h)$ verificando, para algún $F_h \in \mathbb{F}_{q^m}[x]$:

$$u_h(x) = x^t F_h(x) + f(x)s_h(x)$$

es decir, tales que $u_h(x) \equiv f(x)s_h(x) \pmod{x^t}$, la ecuación clave. De la definición de u y de la hipótesis $r \leq t/2$ se tiene que el polinomio evaluador de errores es de grado a lo sumo $t/2 - 1$. En el artículo mencionado se demuestra como tomando el primer índice h tal que se da $grado(u_h) < t/2$, los polinomios localizador y corrector de errores vienen dados respectivamente por:

$$s(x) = \frac{s_h(x)}{s_h(0)} \quad \text{y} \quad u(x) = \frac{u_h(x)}{s_h(0)} \tag{3.5}$$

Una vez calculados estos polinomios, la descodificación pasa por calcular las raíces de $s(x)$, cuyas inversas nos proporcionan las **posiciones de error**. Los valores del error vienen dados por la fórmula (3.4) de la Observación 3.17. Finalmente se restan los errores cometidos a la palabra recibida.

3.2. Criptografía de clave pública

Existen diferentes objetivos criptográficos:

- Proteger el secreto en las comunicaciones, para evitar que cualquier receptor ilegítimo que acceda al mensaje sea capaz de acceder a su contenido.
- Garantizar la integridad del mensaje frente a modificaciones provocadas por adversarios.
- Proveer garantía de que el emisor de un mensaje es quien dice ser.

Para alcanzar estos objetivos se codifica la información utilizando lo que se denominan esquemas de cifrado (o criptosistemas), de cifrado-autenticado, o esquemas de firma digital. La diferencia fundamental con la codificación tratada en la sección anterior consiste en que la criptografía pretende ofrecer **seguridad frente a la acción de adversarios**, es decir, usuarios ilegítimos, no de alteraciones provocadas por ruido externo.

Existen cifrados clásicos que se han ido desarrollando desde la antigüedad para proteger el secreto: el método de la escítala, el cifrado de César o, más recientemente, el uso de la máquina ENIGMA utilizada por los alemanes durante la Segunda Guerra Mundial. Sin embargo, según se han ido sofisticando los métodos de cifrado, también se sofistican los métodos para romperlos y acceder a los secretos. De esto se encarga el Criptoanálisis y, por supuesto, las agencias de inteligencia de los Estados. Es por ello que ambas disciplinas, Criptografía y Criptoanálisis han de estar en constante evolución.

La **criptografía de clave simétrica** se basa en utilizar una misma clave, compartida por emisor y receptor, para cifrar y descifrar. El algoritmo más utilizado hoy en día con este enfoque es el *AES*: el NIST lo adoptó como estándar en 2001. Sin embargo surge un problema fundamental que cobra mucha importancia a la hora de comunicarse por internet: la necesidad de compartir la clave. ¿Cómo puede nuestro ordenador ponerse de acuerdo con el servidor de un banco sobre que clave utilizar para cifrar, por ejemplo

mediante el AES, la información relativa a nuestras transacciones bancarias garantizando el secreto de la clave misma? La respuesta se encuentra en la **criptografía de clave pública**, en la que el receptor publica una clave que usará el emisor para cifrar y mantiene en secreto una clave privada que se usa para descifrar. El sistema más conocido en esta línea es el *RSA*, debido a **Rivest, Shamir y Adleman**, basado en el **problema de la factorización de enteros en factores primos** [Rivest et al., 1978]. Veamos como los cuerpos finitos pueden ser utilizados en este tipo de criptografía.

3.2.1. Funciones de una vía: el problema del logaritmo discreto

Una **función de una vía** es una función biyectiva f para la cual es sencillo (computacionalmente) calcular imágenes de elementos e inasumible calcular, para una imagen dada, su preimagen. Esta clase de funciones resulta ser muy útil en criptografía. En el contexto de los cuerpos finitos existe un problema que proporciona una función de este tipo: **el problema del logaritmo discreto**.

Fijemos un cuerpo finito $\mathbb{F}_q$ y un elemento primitivo $g \in \mathbb{F}_q$. Recordamos que de esta manera cualquier elemento de $\mathbb{F}_q^*$ puede ser escrito de manera única como g^a para $1 \leq a \leq q-1$. Esto permite definir el logaritmo discreto con base g en $\mathbb{F}_q$ como:

$$\log_g : y = g^a \in \mathbb{F}_q^* \longrightarrow a \in \{1, ..., q-1\}.$$

Esta función es la inversa de la exponenciación con base g. Existen algoritmos muy eficientes para el cálculo de potencias en cuerpos finitos y anillos de restos: la técnica de los cuadrados sucesivos, o la exponenciación modular rápida (ver por ejemplo [Lidl and Niederreiter, 1994]). Sin embargo, se trata de una función de una vía para nuestro conocimiento actual. **No se conocen algoritmos eficientes que sean capaces de calcular logaritmos discretos en el caso general en tiempo polinomial**, es decir, en un tiempo asumible. Sin embargo, **sí que existen algoritmos muy eficaces para ciertas elecciones de** q, lo que condiciona la elección de los parámetros, como veremos después de ejemplificar como se aplica este problema a la criptografía:

El protocolo de Diffie-Hellman. Este protocolo, propuesto por Whitfield Diffie y Martin E. Hellman en [Diffie and Hellman, 1976] supuso el incicio de la criptografía de clave pública. Se trata de un esquema para que dos individuos A y B acuerden una clave privada en común para cifrar mediante un criptosistema de clave simétrica. Se procede de la siguiente manera:

- A y B acuerdan un número primo q y un elemento primitivo $g \in \mathbb{F}_q$. **El par** (q, g) **se hace público**, luego A y B pueden compartir esta información sin problema.
- A elige aleatoriamente $a \in \{1, ..., q-1\}$ y mantiene este valor **en secreto**. Asimismo B escoge y mantiene en secreto $b \in \{1, ..., q-1\}$. Ambos calculan g^a y g^b respectivamente de manera eficiente y se intercambian dichos valores. Es en este intercambio donde **un adversario puede acceder al valor de** g^a **y** g^b**.** Sin embargo, bajo una elección adecuada de q, **el problema del logaritmo discreto impide que se pueda conocer ilegítimamente** a **y** b**.**
- A conoce g,a, y g^b. Calcula fácilmente $g^{ab} = (g^b)^a$. De igual manera B obtiene $g^{ab} = (g^a)^b$. A**hora ambos comparten** g^{ab}, y pueden utilizarlo como **clave privada**.
- Un adversario que pretenda calcular la clave utilizada a partir de (q, g, g^a, g^b) **no tiene mejor opción que enfrentarse al problema del logaritmo discreto**: la seguridad de este protocolo se basa en la dificultad del problema.

El protocolo de Diffie y Hellman puede modificarse para obtener un cifrado de clave pública propiamente dicho:

Criptosistema de ElGamal [Elgamal, 1985]. Supongamos que A quiere proponer una clave pública para recibir mensajes $m \in \mathbb{F}_q$ con q primo por parte de cualquier emisor B.

- De igual manera que en el protocolo anterior, A elige g un elemento primitivo y $a \in \{1, ..., q-1\}$. Publica (q, a, g^a) como clave pública, y mantiene a como clave privada.

- B quiere mandarle $m \in \mathbb{F}_q$ a A, manteniendo m en secreto frente a usuarios ilegítimos. B escoge en secreto b, y calcula y envía el par $(g^b, c = mg^{ab})$.

- A puede recuperar m como $m = cg^{-ab}$ pero, de igual manera que en el protocolo de Diffie y Hellman, un adversario que trate de obtener m ha de enfrentarse al problema del logaritmo discreto.

Existen también esquemas de firma digital basados en este problema.

La hipótesis en la que se basa la seguridad de estos cifrados es la suposición de que calcular logaritmos en $\mathbb{F}_q$ es muy difícil. Dicha suposición se sostiene en que los algoritmos utilizados para esta labor no son eficaces en general. Ejemplos de estos algorimos son la **rho de Pollard** [Pollard, 1978] o el **método del paso enano-paso gigante** (para más detalles ver por ejemplo [Sabater et al., 2004]). Sin embargo, una mala elección de q puede estropear estas hipótesis, debido al **algoritmo de Silver-Pohlig-Hellman** [Pohlig, 1978]. En concreto, se ha de evitar que $q-1$ se trate de un entero **liso** (smooth integer), esto es, tal que $q-1 = ord(\mathbb{F}_q^*)$ no pueda factorizarse como producto de números primos pequeños. La clave de este algoritmo se encuentra en el Teorema Chino de los Restos.

Algoritmo de Silver-Pohlig-Hellman.

1. Supongamos $q-1 = \prod_{i=1}^{n} p_i^{e_i}$ con p_i primos distintos para $i = 1, ..., n$, y sea $\alpha \in \mathbb{F}_q^*$. Denotemos $x = \log_g(\alpha)$. El objetivo es, para cada $i = 1, ..., n$, calcular $x \pmod{p_i^{e_i}}$. De esta manera, el Teorema Chino de los Restos nos permite determinar de forma única $x \pmod{\prod_{i=1}^{n} p_i^{e_i}}$, y como $1 \leq r \leq q-1$, se tendría determinado x.

2. Fijemos $i \in \{1, ..., n\}$. La expresión en base p_i de $x \pmod{p_i^{e_i}}$ se escribe como:

$$x \equiv \sum_{j=0}^{e_i-1} x_j p_i^j \pmod{p_i^{e_i}}$$

con $x_0, ..., x_{e_i-1} \in \{0, ..., p_i - 1\}$ incógnitas.

3. Se empieza calculando x_0. Para ello, se ha de notar que como g es una $(q-1)$-ésima raíz primitiva de la unidad, entonces $\gamma_i = g^{\frac{q-1}{p_i}}$ es una p_i-ésima raíz primitiva de la unidad, luego en particular $\{\gamma_i^0, \gamma_i^1, ..., \gamma_i^{p_i-1}\}$ son todos valores distintos. Además, usando $\alpha = g^x$ y $\gamma_i^{p_i} = 1$:

$$\alpha^{\frac{q-1}{p_i}} = g^{x\frac{q-1}{p_i}} = \gamma_i^x = \gamma_i^{\sum_{j=0}^{e_i-1} x_j p_i^j} = \gamma_i^{x_0},$$

luego $\alpha^{\frac{q-1}{p_i}} \in \{\gamma_i^0, \gamma_i^1, ..., \gamma_i^{p_i-1}\}$. Por tanto, buscando $\alpha^{\frac{q-1}{p_i}}$ entre $\{\gamma_i^0, \gamma_i^1, ..., \gamma_i^{p_i-1}\}$ se determina unívocamente x_0.

4. El proceso puede extenderse para calcular x_1 de la siguiente manera: se toma $\alpha_1 = \alpha g^{-x_0} = g^{\sum_{j=1}^{e_i-1} x_j p_i^j}$, y se tiene que:

$$\alpha_1^{\frac{q-1}{p_i^2}} = \gamma_i^{\sum_{j=1}^{e_i-1} p_i^j x^{j-1}} = \gamma_i^{x_1},$$

y de igual manera se busca $\alpha_1^{\frac{q-1}{p_i^2}}$ entre $\{\gamma_i^0, \gamma_i^1, ..., \gamma_i^{p_i-1}\}$ para determinar x_1, y análogamente para $x_2, ..., x_{e_i-1}$. Usando el Teorema Chino de los Restos se recupera el valor de x a partir de sus restos módulo $p_1^{e_1}, ..., p_n^{e_n}$.

En el algoritmo anterior se hace uso de un bucle de $\sum_{i=1}^n e_i$ ciclos. Es ahí donde se concentra el mayor esfuerzo computacional. Si el mayor de los factores primos de $q-1$, digamos p_k, es pequeño, entonces $\sum_{i=1}^n e_i \simeq \log_2(q)$. En [Pohlig, 1978] se muestra como bajo esta condición el número de operaciones requeridas es del orden de $(log_2 q)^2$ y, en definitiva, la exponenciación en $\mathbb{F}_q$ no se trataría de una buena función de una vía. Sin embargo, si el mayor de los factores primos de $q-1$, p_k, es del orden de $q-1$ (por ejemplo, $q - 1 = 2p_k$), entonces el uso de este algoritmo no supone ninguna ventaja frente a otros métodos conocidos.

Observación 3.19. *Se puede notar que el algoritmo presentado es especialmente ineficaz frente números primos de Mersenne (aquellos primos de la forma $2^p - 1$ con p primo). Sin embargo, también existen métodos específicos eficaces para calcular algoritmos discretos en cuerpos finitos de característica dos y en general de característica pequeña, lo que impone otra restricción sobre la elección del parámetro q.*

3.2.2. Códigos en criptografía postcuántica: el esquema de McEliece

La seguridad del criptosistema RSA está fundamentada en el problema de la factorización de un número $N = pq$ en sus factores primos p y q. Asimismo, hemos visto como pueden elaborarse esquemas de cifrado basados en el problema del logaritmo discreto. Ambos problemas son duros para cualquier algoritmo conocido implementado en un ordenador clásico. Sin embargo, el conocido método debido a Peter Shor (ver [Shor, 1997] para el problema del logaritmo discreto y [Shor, 1994] para la factorización de enteros) es capaz de resolver dichos problemas en tiempo polinomial en un **ordenador cuántico**. En el año 2001, IBM fue capaz de implementar el algoritmo de Shor en un hardware cuántico logrando factorizar 15 en sus factores 3 y 5 ([Vandersypen et al., 2001]). A día de hoy, este tipo de tecnología no está lo suficientemente desarrollada como para hacer peligrar la seguridad de la criptografía de clave pública que proporciona RSA o ElGamal. Sin embargo, la amenaza está latente: la privacidad de toda la comunicación por internet quedaría en entredicho el día que alguien disponga de ordenadores cuánticos competentes. Es por ello que se vuelve obligatorio elaborar cifrados cuya cimentación no se vea comprometida por la aparición de este tipo de hardware: cifrados que conforman la **criptografía postcuántica**. Los códigos vistos en la sección anterior (en concreto los códigos Goppa) pueden utilizarse para elaborar cifrados decentes que, además, son resistentes frente a los algoritmos cuánticos.

El criptosistema de McEliece. Robert J. McEliece propuso en 1978 el primer criptosistema de clave pública basado en códigos [McEliece, 1978]. Se fundamenta en el siguiente hecho fundamental: **mientras que se conocen** (como hemos visto) **métodos de descodificación eficientes para códigos Goppa, no sucede lo mismo para códigos lineales en general.** En concreto, se usarán códigos Goppa binarios generados por un polinomio irreducible (recordar la Observación 3.15). Se consideran $n = 2^m$ y $t < n$ como parámetros del criptosistema. Se podrán cifrar mensajes de cierta longitud $k \geq n - mt$.

1. El primer paso consiste en **generar aleatoriamente como clave privada un código Goppa binario**. Se ha de escoger de forma aleatoria un polinomio irreducible g de grado t sobre $\mathbb{F}_{2^m}$. Esto se logra generando aleatoriamente polinomios $f \in \mathbb{F}_{2^m}[x]$ y verificando su irreducibilidad por algún método eficiente, por ejemplo usando los métodos de factorización del Capítulo 2. Se puede estimar la probabilidad de éxito gracias al Corolario 1.23: la proporción de polinomios irreducibles sobre el total de polinomios de grado t en $\mathbb{F}_{2^m}[x]$ es:

$$\frac{N_{2^m}(t)}{2^{mt}} = \frac{1}{t}\sum_{d|t} \mu\left(\frac{t}{d}\right) 2^{m(d-t)},$$

   que se puede estimar por $1/t$ para valores de t grandes.

2. Una vez escogido g, se obtiene una matriz de control H del código $\Gamma = \Gamma(L, g)$ (de dimensión $k \geq 2^m - tm$) a partir de la expresión dada en (3.3). A partir de ella se obtiene una matriz generatriz G del código que un receptor A ha de mantener en secreto.

3. El siguiente paso consiste en generar la clave pública *camuflando* G (generatriz de un código Goppa) de tal manera que se asemeje a una matriz generatriz de un código lineal cualquiera. Para ello se escogen aleatoriamente una matriz $(n \times n)$ de permutación P y una matriz $(k \times k)$ regular S y se obtiene la matriz $(k \times n)$ $G' =$

SGP. Se trata de una matriz generatriz de un $[n, k, d]_2$-código lineal ($Im(SG) = Im(G)$, y la distancia mínima de $Im(SG)$ es igual a la de $Im(SGP)$ por ser P de permutación). Esta transformación hace que no haya un algoritmo eficiente para descodificar palabras codificadas por G'.

4. Se usa (G') como clave pública y se mantiene (S, G, P) como clave privada.

5. Supongamos que un emisor B quiere ahora enviarle a A un texto en claro $v \in \mathbb{F}_2^k$. Para ello B codifica v a través de G' introduciendo un ruido aleatorio $z \in \mathbb{F}_2^n$ con $wt(z) \leq t/2$. El mensaje cifrado es: $y = vG' + z$.

6. A recibe y y calcula $yP^{-1} = vSG + zP^{-1}$. Como $(vS)G \in \Gamma$ y $wt(zP^{-1}) = wt(z) \leq t/2$, puede utilizarse el algoritmo de descodificación de códigos Goppa para recuperar $c = vS$. (Podría haberse tomado $wt(z) \leq t$, gracias al Teorema 3.14, pero entonces habría que usar un algoritmo de decodificación distinto al propuesto en 3.1.2).

7. Se recupera el mensaje original como $v = cS^{-1}$.

La seguridad de la propuesta anterior se basa en dos consideraciones:

- Recuperar G a partir de G' es una tarea inviable para parámetros suficientemente grandes: **existen multitud de posibilidades** distintas para la elección de S y P.
- Recuperar m a partir de y sin conocer G depende de la **tarea de descodificar el código lineal** dado por G'. ¡Pero no se conoce algoritmo de descodificación eficaz para códigos lineales en general!

Con base en este esquema se ha desarrollado la propuesta para el concurso de estandarización del NIST de criptografía postcuántica *Classic McEliece* [Bernstein et al.,].

3.3. Un ejemplo de cifrado de McEliece

Para terminar el trabajo, realicemos como ejemplo un cifrado de McEliece tomando parámetros muy sencillos. Nos sirvirá como excusa para ilustrar varios de los contenidos tratados durante el trabajo. En concreto, tomemos $m = 4$, $n = 2^4 = 16$ y $t = 2$. Nos ponemos en el lugar de un receptor A.

1. Generación de un polinomio irreducible de grado 2 sobre $\mathbf{F}_{16}$. Denotamos en adelante $F = \mathbb{F}_{16}$ y $K = \mathbb{F}_2$. Trabajaremos con la base $\{1, a, a^2, a^3\}$ donde $a \in F$ verifica $a^4 + a + 1 = 0$. Tomamos coeficientes aleatorios $b_0, b_1 \in F$ iterativamente hasta lograr que $g(x) = x^2 + b_1 x + b_0$ sea un polinomio irreducible. Por ejemplo, en un primer intento obtenemos:

$$g(x) = x^2 + (a^3 + a^2 + 1)x + a^3 + a^2$$

Para verificar si se trata de un polinomio irreducible, le hemos aplicado a g el algoritmo de Berlekamp para contar el número de sus factores irreducibles. Se obtiene que $g(x) = (x + a^3 + a^2)(x + 1)$, luego no nos sirve.

Se intenta de nuevo, esta vez con:

$$g(x) = x^2 + (a^3 + a^2)x + a^2 + a + 1.$$

Con el algoritmo de Berlekamp se constata que esta vez g resulta ser, en efecto, irreducible en $F[x]$.

2. Construcción del código $\Gamma(F, g)$. Hemos de considerar el soporte del código (todo el cuerpo F) con un orden fijo para la elaboración de las matrices y poder detectar las posiciones de error correctamente. Nosotros consideramos:

$$L = (0, a, a^2, a^3, a+1, a^2+a, a^3+a^2, a^3+a+1, a^2+1,$$

$$a^3+a, a^2+a+1, a^3+a^2+a, a^3+a^2+a+1, a^3+a^2+1, a^3+1, 1).$$

Para construir el código, recurrimos a la expresión (3.3). Obtenemos la pseudomatriz de control:

$$\tilde{H} = \begin{pmatrix} a^2+a & a^3+a^2+a+1 & 1 & \cdots & \cdots & a^3+a & a^3+a^2 \\ 0 & a^3+a^2+1 & a^2 & \cdots & \cdots & a^2+1 & a^3+a^2 \end{pmatrix} \tag{3.6}$$

Para obtener la matriz de control del código H, hemos de sustituir cada elemento de $\tilde{H}$ por su vector de coeficientes respecto de $\{1, a, a^2, a^3\}$. La matriz generatriz G del código se obtiene resolviendo el sistema $Hx = 0$, es decir, tomando por columnas una base del núcleo de H. Se obtienen entonces:

$$H = \begin{pmatrix} 0 & 1 & 1 & \cdots & \cdots & 0 & 0 \\ 1 & 1 & 0 & \cdots & \cdots & 1 & 0 \\ 1 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 0 & 1 & 0 & \cdots & \cdots & 1 & 1 \\ 0 & 1 & 0 & \cdots & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 1 & 1 & \cdots & \cdots & 1 & 1 \\ 0 & 1 & 0 & \cdots & \cdots & 0 & 1 \end{pmatrix}, \; G = \begin{pmatrix} 1 & 0 & 0 & \cdots & \cdots & 1 & 1 \\ 0 & 1 & 0 & \cdots & \cdots & 1 & 0 \\ 0 & 0 & 1 & \cdots & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 0 & 1 \\ 0 & 0 & 0 & \cdots & \cdots & 0 & 1 \\ 0 & 0 & 0 & \cdots & \cdots & 1 & 1 \\ 0 & 0 & 0 & \cdots & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 1 & 1 \end{pmatrix}$$

Se tiene que $k = 8$, es decir, hemos obtenido un código Goppa Binario de dimensión 8, y se podrán cifrar mensajes de dicha longitud. La clave privada del cifrado es precisamente G.

3. Generación de la clave pública. Se ha de generar aleatoriamente una **matriz de permutación** P de tamaño $(n \times n)$. Este paso no tiene mayor complicación, puesto que basta generar una permutación aleatoria de $(1, ..., n)$, $(\sigma(1), ..., \sigma(n))$, y tomar la matriz $P = (e_{\sigma(1)}, ..., e_{\sigma n})$, donde e_i denota el vector i-ésimo de la base canónica. Nosotros hemos obtenido $(11, 10, 3, 7, 8, 1, 13, 2, 14, 5, 16, 12, 9, 6, 4, 15)$ y generado la matriz P correspondiente. Para generar la **matriz inversible** S de tamaño $(k \times k)$ hemos generado matrices aleatoriamente hasta obtener una de rango máximo. Hemos obtenido:

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

Y se obtiene finalmente:

$$G' = SGP = \begin{pmatrix} 0 & 0 & 1 & \cdots & \cdots & 0 & 1 \\ 1 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 1 & 1 & 1 & \cdots & \cdots & 0 & 1 \\ 1 & 0 & 1 & \cdots & \cdots & 1 & 0 \\ 0 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 0 & 1 & 0 & \cdots & \cdots & 1 & 0 \\ 0 & 1 & 0 & \cdots & \cdots & 0 & 1 \\ 1 & 1 & 1 & \cdots & \cdots & 1 & 0 \end{pmatrix}.$$

4. G' se hace pública, mientras que la tupla (S, G, P) se conserva como clave privada.

5. Cifrado. Nos ponemos ahora en el lugar de un emisor B que quiere mandarle a A la cadena 01001100, por ejemplo. Se toma $v = (0, 1, 0, 0, 1, 1, 0, 0) \in \mathbb{F}_2^k$. Generamos un ruido aleatorio z de peso $wt(z) = t/2 = 1$, obteniendo $z = (0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$. El mensaje cifrado es:

$$y = vG' + z = (1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0),$$

y se procedería a enviarle y a A.

6. Decodificación. Volvemos a jugar el papel de A suponiendo que recibimos y. Primero, deshacemos la permutación inducida por P obteniendo $y_0 = yP^{-1} = vSG + zP^{-1} = (0,1,0,1,1,1,1,0,0,0,1,1,1,0,1,0)$. Aplicando el **algoritmo de decodificación de Códigos Goppa** podemos recuperar el valor de vSG. Los síndromes resultan ser:

$$(S_0, S_1) = y_0\tilde{H}^t = (a^3 + a^2 + a, a^3),$$

y por tanto el polinomio de síndromes resulta:

$$f(x) = a^3x + a^3 + a^2 + a.$$

Haciendo la división euclídea de $x^t = x^2$ entre f obtenemos:

$$x^2 = f(x)((a^3 + a^2 + a + 1)x + a^2 + a) + a = f(x)s_0(x) - u_0(x).$$

Se verifica entonces la ecuación clave:

$$f(x)s_0(x) \equiv u_0(x) \pmod{x^2}$$

y además $grado(u_0) = 0 \leq 1 = t/2$. Por 3.5, el polinomio localizador de errores viene dado por:

$$s(x) = \frac{s_0(x)}{s_0(0)} = \frac{(a^3 + a^2 + a + 1)x + a^2 + a}{a^2 + a} = (a^3 + a + 1)x + 1.$$

Ya hemos discutido cómo se podrían calcular las raíces de un polinomio sobre un cuerpo finito en general. En este caso simplemente se tiene la única raíz de s es $\mu_1^{-1} = -(a^3 + a + 1)^{-1} = (a^3 + a + 1)^{-1}$ y por lo tanto el número localizador de errores es $\mu_1 = a^3 + a + 1$, que ocupa la séptima posición en el orden fijado para el soporte. Es importante notar que **al trabajar con códigos Goppa binarios, no necesitamos realmente calcular el polinomio evaluador de errores**: solo pueden tomar el valor 1. Por tanto, hemos obtenido que:

$$zP^{-1} = (0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0)$$

y entonces:

$$vSG = y_0 - zP^{-1} = (1,1,0,1,0,1,1,0,0,1,1,0,0,0,0,0).$$

7. Recuperación de v. Dado que A conoce (S, G) y la matriz SG es inyectiva, basta resolver en v el sistema

$$v(SG) = (1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0)$$

para recuperar v. Se obtiene finalmente $v = (0, 1, 0, 0, 1, 1, 0, 0)$, **¡valor que coincide con el mensaje de partida!**

En las siguientes figuras se proporcionan las salidas del código utilizado para resolver el ejemplo anterior:

```
Probamos con el polinomio generado aleatoriamente g = x^2 + (a^3 + a^2 + 1)*x + a^3 + a^2

        Algoritmo de Berlekamp para  factorizar el polinomio  x^2 + (a^3 + a^2 + 1)*x + a^3 + a^2 sobre sobre GF( 16 )
CASO 1:   x^2 + (a^3 + a^2 + 1)*x + a^3 + a^2 no tiene factores repetidos. Se aplica Berlekamp para libre de cuadrados.


----------------Berlekamp libre de cuadrados para el polinomio  x^2 + (a^3 + a^2 + 1)*x + a^3 + a^2  sobre GF( 16 )-----------
Matriz de Berlekamp Bf:
[1 0]
[0 1]
Número de factores irreducibles:  2
Polinomios reductores:  [x]
---------------------- Factorización obtenida: [x + a^3 + a^2, x + 1] ----------------------

  x^2 + (a^3 + a^2 + 1)*x + a^3 + a^2  se factoriza como el producto de los factores  [x + a^3 + a^2, x + 1]

CONCLUSION: g= x^2 + (a^3 + a^2 + 1)*x + a^3 + a^2 no es irreducible

&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&
```

Figura 3.1: Ejecución del Algoritmo de Berlekamp para el primer polinomio generado.

```
Probamos con el polinomio generado aleatoriamente g = x^2 + (a^3 + a^2)*x + a^2 + a + 1

        Algoritmo de Berlekamp para  factorizar el polinomio  x^2 + (a^3 + a^2)*x + a^2 + a + 1 sobre sobre GF( 16 )
CASO 1:   x^2 + (a^3 + a^2)*x + a^2 + a + 1 no tiene factores repetidos. Se aplica Berlekamp para libre de cuadrados.


----------------Berlekamp libre de cuadrados para el polinomio  x^2 + (a^3 + a^2)*x + a^2 + a + 1  sobre GF( 16 )--------
Matriz de Berlekamp Bf:
[        1         0]
[a^3 + a^2         1]
Número de factores irreducibles:  1
Polinomios reductores:  []
---------------------- Factorización obtenida: [x^2 + (a^3 + a^2)*x + a^2 + a + 1] ----------------------

  x^2 + (a^3 + a^2)*x + a^2 + a + 1  se factoriza como el producto de los factores  [x^2 + (a^3 + a^2)*x + a^2 + a + 1]


&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&

El polinomio irreducible generado es:  x^2 + (a^3 + a^2)*x + a^2 + a + 1
```

Figura 3.2: Ejecución del Algoritmo de Berlekamp para el segundo polinomio generado.

```
La matriz de control es H =
[0 1 1 1 1 1 0 0 0 1 0 1 1 0 0 0]
[1 1 0 0 1 1 1 1 1 1 1 1 1 0 1 0]
[1 1 0 0 0 0 1 1 0 0 1 1 0 1 0 1]
[0 1 0 0 1 0 0 1 1 0 1 1 1 1 1 1]
[0 1 0 0 0 0 0 0 0 1 0 1 1 1 1 0]
[0 0 0 0 1 1 1 0 0 0 0 0 1 1 0 0]
[0 1 1 0 1 0 1 0 1 1 1 1 0 0 1 1]
[0 1 0 1 1 1 1 1 0 1 1 0 0 0 0 1]

La matriz generatriz es G =
[1 0 0 0 0 0 1 0 1 0 0 0 0 1 1 1]
[0 1 0 0 0 0 1 0 0 0 0 1 0 1 1 0]
[0 0 1 0 0 0 1 0 1 1 0 1 1 0 1 0]
[0 0 0 1 0 0 0 0 1 0 0 0 1 1 0 1]
[0 0 0 0 1 0 1 0 1 1 0 1 1 1 0 1]
[0 0 0 0 0 1 1 0 0 1 0 0 0 0 1 1]
[0 0 0 0 0 0 1 0 1 0 0 1 1 1 1 0]
[0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1]
```

Figura 3.3: Matrices de control y generatriz

```
El mensaje original es v = (0, 1, 0, 0, 1, 1, 0, 0)
El mensaje codificado es uG_tilde =  (1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0)
El ruido aleatorio introducido es z =  (0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
El mensaje cifrado es y =  (1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0)
```

Figura 3.4: Mensajes en claro y cifrado

```
Partimos de y = (1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0)
 y de (S,G,P)
Sindromes:  (a^3 + a^2 + a, a^3)
Polinomio de síndromes:  a^3*x + a^3 + a^2 + a
s_0= (a^3 + a^2 + a + 1)*x + a^2 + a
u_0= a
Polinomio localizador de errores s(x) =  (a^3 + a + 1)*x + 1
Polinomio evaluador de errores u(x) =  a^3 + a^2 + a
El número localizador de errores es mu = a^3 + a + 1
La decodificación devuelve y_0-zP-1 =  (1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0)
DESCRIFRADO:  (0, 1, 0, 0, 1, 1, 0, 0)
ORIGINAL:  (0, 1, 0, 0, 1, 1, 0, 0)
ÉXITO!!
```

Figura 3.5: Descodificación

Conclusiones

El objetivo de este trabajo era profundizar en el conocimiento de los cuerpos finitos y en sus aplicaciones más relevantes. Hemos atendido especialmente a su **estructura multiplicativa**, que resulta ser **cíclica**, y visto como este hecho permite, por ejemplo, construir los **códigos BCH** y formular el **problema del logaritmo discreto**, de gran importancia en criptografía.

Los polinomios han jugado un papel fundamental en esta tarea. Los polinomios irreducibles, y en concreto los **polinomios primitivos**, resultan ser cruciales para obtener extensiones adecuadas de cuerpos. Además, los Códigos Goppa tienen muy buenas propiedades si se utilizan para su construcción polinomios irreducibles. Para generar dichos polinomios, hemos visto que existen dos enfoques: o bien tratar de conseguir una lista exhaustiva de polinomios irreducibles para cada cuerpo y grado necesario, o bien generar polinomios aleatoriamente y verificar su irreducibilidad. Para la primera tarea es conveniente obtener **factorizaciones parciales** como la obtenida en el Teorema 1.33. Para justificar la utilidad del segundo enfoque hemos dado una **fórmula para el número exacto** de polinomios irreducibles de un grado dado en el Corolario 1.23 mediante la fórmula de inversión de Möbius. Además, en ambos casos **es imprescindible disponer de algoritmos eficientes de factorización**.

Queremos destacar la utilidad del automorfismo de Frobenius a la hora de trabajar con cuerpos finitos, pues permite introducir técnicas de **linealización**. Los **polinomios linealizados** sirven tanto como **herramienta para el cálculo de raíces** como para

estudiar la **función traza**; por otro lado, el **algoritmo de Berlekamp reduce el problema de la factorización al cálculo de las soluciones de un sistema lineal**.

Hemos constatado como **los cuerpos finitos tienen presencia en aplicaciones criptográficas actuales**, además de **potencial para mantenerse relevantes gracias a los Códigos Correctores**. El problema del logaritmo discreto proporciona una **función de una vía**, pero hay que tener precauciones: **el algoritmo de Pollih-Silver-Hellman** pone de manifiesto como **una mala elección del tamaño del cuerpo utilizado puede poner en riesgo la seguridad de los protocolos**. Con el **criptosistema de McEliece** hemos ejemplificado como pueden utilizarse la codificación como función de una vía, gracias a la dificultad de descodificar códigos arbitrarios y a la **existencia de un algoritmo específico para Códigos Goppa**. Es por este motivo que hemos introducido estos códigos, caracterizado sus matrices de control y generatriz y logrado **muy buenas cotas para su capacidad correctora de errores**. Por último **hemos elaborado un ejemplo sencillo** de este cifrado en el que cristalizan los contenidos más importantes del trabajo. Para resolverlo, hemos tenido que implementar el algoritmo de Berlekamp, generar un polinomio irreducible, contruir un código Goppa desde cero y seguir el proceso de decodificación.

En definitiva, **los cuerpos finitos conforman una estructura de interés propio que, además, resulta ser uno de los pilares fundamentales en la seguridad de nuestras comunicaciones**.

Bibliografía

[Berlekamp, 1967] Berlekamp, E. R. (1967). Factoring polynomials over finite fields. *Bell System Technical Journal*, 46(8):1853–1859.

[Bernstein et al.,] Bernstein, D. J., Chou, T., Cid, C., Gilcher, J., Lange, T., Maram, V., von Maurich, I., Misoczki, R., Niederhagen, R., Persichetti, E., Peters, C., Sendrier, N., Szefer, J., Tjhai, C. J., Tomlinson, M., and Wang, W. Mceliece cryptosystem homepage. https://classic.mceliece.org.

[Cantor and Zassenhaus, 1981] Cantor, D. G. and Zassenhaus, H. (1981). A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, 36(154):587–592.

[Diffie and Hellman, 1976] Diffie, W. and Hellman, M. (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654.

[Elgamal, 1985] Elgamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472.

[Lidl and Niederreiter, 1994] Lidl, R. and Niederreiter, H. (1994). *Introduction to Finite Fields and their Applications*. Cambridge University Press, 2 edition.

[McEliece, 1978] McEliece, R. J. (1978). A public-key cryptosystem based on algebraic coding theory. *Coding Thv*, 4244:114–116.

[Mullen, 2013] Mullen, G.L., . P. D. (2013). *Handbook of Finite Fields (1st ed.)*. Chapman and Hall/CRC, 1 edition.

[Pohlig, 1978] Pohlig, S. C. (1978). An improved algorithm for computing logarithms over gp(p) and its cryptographic significance.

[Pollard, 1978] Pollard, J. M. (1978). Monte carlo methods for index computation ($\mathrm{mod}\, p$). *Mathematics of Computation*, 32(143):918–924.

[Rivest et al., 1978] Rivest, R. L., Shamir, A., and Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126.

[Sabater et al., 2004] Sabater, A. F., de la Guía Martínez, D., Encinas, L. H., Vitini, F. M., and Masqué, J. M. (2004). Técnicas criptográficas de protección de datos.

[Shor, 1994] Shor, P. W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134.

[Shor, 1997] Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509.

[Sugiyama et al., 1975] Sugiyama, Y., Kasahara, M., Hirasawa, S., and Namekawa, T. (1975). A method for solving key equation for decoding goppa codes. *Information and Control*, 27(1):87–99.

[Vandersypen et al., 2001] Vandersypen, L. M. K., Steffen, M., Breyta, G., Yannoni, C. S., Sherwood, M. H., and Chuang, I. L. (2001). Experimental realization of shor's quantum factoring algorithm using nuclear magnetic resonance. *Nature*, 414(6866):883–887.

[Zassenhaus, 1969] Zassenhaus, H. (1969). On hensel factorization, i. *Journal of Number Theory*, 1:291–311.

Apéndice A

Algoritmos

Todos los algoritmos están programados en SageMath.

Algoritmo 1: Berlekamp para factorizar polinomios sin factores irreducibles repetidos.

```
def BerlekampLibre(f,q,F):
# q potencia de primo; F cuerpo de q elementos; f en F[x] libre de cuadrados
# Devuelve: vector con los factores irreducibles de f
    a=F.gen()
    factores = [f]
    num_factores = 1              # Inicializamos la factorización
    P.<x> = PolynomialRing(F)
    print('\n-----------------Berlekamp libre de cuadrados para el polinomio ',
    f,' sobre GF(',q,')-------------')
    n=f.degree()
    # Construcción de Bf
    Bf=matrix(F, n, n)
    for i in range(n):
        vec=[]
        pol=(x^(i*q))%f        # Calculamos las congruencias para cada potencia
        vec=pol.list()
```

```
    while len(vec) < n:
        vec.append(F(0))  # Agregar coeficientes cero si es necesario
    Bf[i]=vec             # Agremamos por filas los coeficientes
print('Matriz de Berlekamp Bf: ')
print(Bf)
# Obtenemos los polinomios reductores:
Mat=Bf-identity_matrix(n) # B-I
nucleo=Mat.kernel()       # Se resuelve el sistema lineal (B-I)v=0
base=nucleo.basis()
k=len(base)               # nº de factores irreducibles de f
print('Número de factores irreducibles: ', k)
reductores=list(vector([0]*(k-1)))
coeficientes_reductores=list(base)  # coefs. de los polinomios h que reducen f
for j in range(0,k-1):
     # Construcción de los polinomios h que reducen f
    coeficientes=coeficientes_reductores[j+1]
    reductor=sum(coeficientes[i] * x^i for i in range(len(coeficientes)))
    reductores[j]=reductor
print('Polinomios reductores: ',reductores)
# Calculamos los mcd de cada nuevo factor obtenido con cada polinomio
# h - cada elemento en el cuerpo
indice_reductor=0;
while num_factores<k:
    h=reductores[indice_reductor]  # Selecciona el nuevo reductor
    factores_rotos=[];             # Almacena los índices de los factores
    #rotos por h
    for i in range(len(factores)): # Recorremos entre los factores ya obtenidos
        factor=factores[i]         # Seleccionamos el factor a 'romper'
        factor_roto=False
```

```
            nuevos_factores=[]
            for c in F:                # Recorremos los elementos de GF(q)
                f_c=gcd(f,h-c)
                if f_c!=1:             # f_c es un nuevo factor no trivial...
                    nuevos_factores.append(f_c)
                    num_factores+=1
                    factor_roto=True # h ha roto el factor
            if factor_roto:
                factores_rotos.append(i)  # Se añade a la lista de factores rotos
        factores_rotos.sort(reverse=True) # Para poder realizar
        #la eliminación de los factores
        for indice in factores_rotos:
            del factores[indice]          # Eliminamos los factores rotos...
        factores.extend(nuevos_factores)  # y añadimos los nuevos.
        indice_reductor+=1
    print('---------------------- Factorización obtenida:', factores,
    '----------------------')
    print(' ')
    return(factores)   # Samuel J. García
```

Algoritmo 2: Berlekamp generalizado.

```
def Berlekamp(f,q,F):
# q potencia de primo; F cuerpo finito de q elementos; f polinomio sobre F[x]
# Devuelve: vector con los factores irreducibles de f
    print('\n\t Algoritmo de Berlekamp para  factorizar el polinomio ',f,
          'sobre sobre GF(',q,')')
    a=F.gen()
    P.<x>=PolynomialRing(F)
    p=F.characteristic()
    n=f.degree()
```

```
df=f.derivative()
d=gcd(f,df)
if d==1:
    print('CASO 1:  ',f, 'no tiene factores repetidos',
    'Se aplica Berlekamp para libre de cuadrados. \n')
    if n==1:
        print(' ',f,' es irreducible')
        factores=[f]
    else:
        factores=BerlekampLibre(f,q,F)
elif d==f:
    coeficientes_en_f=f.list()
    coeficientes_en_g = [coeficientes_en_f[i*p] for i in range((f.degree())/p + 1)]
    g=sum(coeficientes_en_g[i] * x^i for i in range(len(coeficientes_en_g)))
    print('CASO 2:  ',f,'es de la forma (',g,')^',p)
    if g.degree()==1:
        print('pol = ',g, ' es irreducible')
        factores=[g]*p
    else:
        factores=Berlekamp(g,p,F)*p
else:
    print('CASO 3: ',f,' repite los factores en d = : ',d,'\n')
    cociente=f//d
    factores_cociente=BerlekampLibre(cociente,q,F)
    factores_d=Berlekamp(d,q)
    factores=factores_cociente+factores_d
print(' ',f,' se factoriza como el producto de los factores ', factores,'\n')
return(factores) # Samuel J. García
```