UNIVERSIDAD
DE GRANADA

Facultad de Ciencias

Grado en Matemáticas

Trabajo de fin de grado

Códigos Goppa

Presentado por:
Cristian Pozo González

Curso académico 2021-2022

Códigos Goppa

Cristian Pozo González

Cristian Pozo González *Códigos Goppa*.

Trabajo de fin de Grado. Curso académico 2021-2022.

Responsable de tutorización Francisco Javier Lobillo Borrero
Departamento de Algebra

Grado en Matemáticas
Facultad de Ciencias
Universidad de Granada

Declaración de originalidad

D./Dña. Cristian Pozo González

Declaro explícitamente que el trabajo presentado como Trabajo de Fin de Grado (TFG), correspondiente al curso académico 2021-2022, es original, entendida esta, en el sentido de que no ha utilizado para la elaboración del trabajo fuentes sin citarlas debidamente.

En Granada a 29 de junio de 2022

Fdo: Cristian Pozo González

Índice general

Índice de figuras

Índice de tablas

Índice de algoritmos

Summary

Whatever the channel through which information is transmitted will be a noisy channel, that is, the information received will not be the same as the one sent. Even in person-to-person communication there are environmental factors that prevent us from hearing each and every syllable of a conversation. To solve this problem, error correction codes appear, which are able to correct as many errors as possible in the transmission from the received word.

Another big problem that arises is that of the security of the transmission of information, for this there are many secure encryption techniques, until nowadays, although with the development of quantum computers some of the most commonly used encryption algorithms are vulnerable, so it is necessary to bet on more secure algorithms, such as those based on error correction codes.

One of the encryption algorithms based on codes, is the one proposed by McEliece that uses the Goppa codes, because they are very similar to the random codes and also present an efficient decoding algorithm, so this work is the link between the error correction codes and the encryption methods.

The title of this work, encompasses a large part of code theory, so we will make a journey from linear codes to Goppa codes, passing by cyclic codes, BCH, Reed-Solomon and evaluation codes among others.

In the first chapter, we will deal with linear codes describing all their properties and giving preliminary concepts about code theory, that will serve us for all the work. Also, we will give techniques to find the generating matrix and the parity control matrix of a linear code, which will allow us to encode and decode words with ease into any linear code. At the end of this chapter we will introduce the concept of spheres that will help us determine the number of words that can be corrected by a code and we will also give dimensions for the size of the code, such as the Singelton height.

In the second chapter, we will develop the theory of cyclic codes that will serve us in the following chapters. Unlike linear codes, cyclics have an extra structure, they can be seen by an isomorphism, as ideals of $\mathbb{F}_q^n$ which will allow us to make a more exhaustive study. Finally we will finish the chapter by giving some techniques to obtain the parity control matrix, using the polynomial and the generator matrix of a cyclic code.

In the third chapter, we will continue a bit inside the cyclic codes, we will start developing the BCH codes, which unlike the cyclic ones, has a predefined minimum distance, characteristic that makes them very important since we have to determine exactly the minimum distance of a code is a NP problem and, unless $P = NP$, we cannot determine it in time. polynomial.

We will continue with the Reed-Solomon codes that can be seen as a restriction of the BCH and finally we will move on to the evaluation codes and to the GRS (a generalization of the Reed-Solomon codes), which include within them most of the linear codes.

Finally, in the last chapter, we will discuss in depth about the Goppa codes. We will start by defining ourselves and introducing its generating matrix, once we have got an efficient coding system, we will move on to decoding. Thanks to Euclid's extended algorithm, we will develop a decoding algorithm, the Sugiyama algorithm, which will allow us to obtain the polynomials locator and evaluator, that is, to decode the received word. Finally we will talk about heights at the minimum distance (asymptotic and non-asymptotic) and we will show that the Goppa codes reach the Gilbert-Varshamov heights.

Introducción

La seguridad de nuestras comunicaciones en la actualidad esta basada en dos pilares fundamentales, por un lado la veracidad y fiabilidad de los mensajes que recibimos, ámbito que abarca desde la posibilidad de corregir errores de transmisión hasta la velocidad y eficiencia en la transmisión.

En este ámbito entran los códigos de corrección de errores donde, ajustando la cantidad de redundancia en la comunicación, somos capaces de maximizar la eficiencia y la fiabilidad de los mensajes. Uno de los ejemplos más conocidos es el documento nacional de identidad, donde la letra representa el resto modulo 23 del numero o por ejemplo el código IBAN de las cuentas bancarias. Se podrían poner infinitud de ejemplos como estos, todos ellos son necesarios para corregir o al menos detectar los errores en la transmisión, entiéndase todo tipo de transmisión de información no solo informática.En ambos ejemplos antes descritos, el emisor suele ser una ser humano y el receptor una maquina, quien se encarga de contrastar la fiabilidad de la información recibida.

Por otro lado, está la integridad, confidencialidad y autenticidad de nuestras comunicaciones, que se consigue con la encriptación de nuestros mensajes, ámbito muy estudiado y con algoritmos eficientes como RSA, usados desde la seguridad de nuestras conversaciones en las redes sociales, hasta la seguridad en las transacciones bancarias.

Con el desarrollo de los ordenadores, nuestros sistemas de seguridad más eficientes hasta la actualidad (RSA entre otros) van a pasar a la historia, algoritmos cuánticos como el algoritmo de SHOR es capaz de factorizar números en tiempo $O(\log(N)^3)$.

Aunque todavía queda muchísimo desarrollo tecnológico por delante para tener ordenadores cuánticos capaces de "romper" las claves que se usan en la actualidad, deberíamos apostar por técnicas de encriptación para las cuales hasta el momento no se conozcan algoritmos cuánticos capaces de romper las claves en tiempo polinomial, naciendo así la criptografía post-cuántica, que engloba teorías como

- Criptografía post-cuántica basada en retículos.
- Criptografía post-cuántica basada en funciones hash.
- Criptografía post-cuántica basada en funciones polinómicas multivariantes.
- Criptografía post-cuántica basada en códigos correctores de errores.

Este trabajo introduciremos desde la base los códigos correctores de errores, hasta llegar a los

conocidos como códigos Goppa introducidos por V.D Goppa en 1970 , que son la base de uno de los criptosistemas post-cuánticos basados en códigos correctores de errores, introducidos por McEliece en 1978.

Dada la envergadura del título, decidimos hacer una síntesis de la teoría de códigos correctores de errores hasta introducir los Goppa y luego posteriormente hablar sobre sus propiedades y finalmente hablar sobre el algoritmo de Sugiyama, un algoritmo de descodificación eficiente.

Durante todo el trabajo trataremos con códigos lineales ya que los no lineales, aunque se usan en la practica en casos muy concretos, su naturaleza tan variada los hace bastante más difíciles de estudiar.

Dentro de los códigos lineales podemos encontrar códigos cíclicos donde conviven la estructura de espacio vectorial con la de $\mathbb{F}_q -$ algebra , de los que hablaremos largo y tendido en el Capítulo 2.

Posteriormente, dedicaremos un capítulo (Vease Capítulo 3) a hablar sobre los códigos BCH y Red-Solomon, que a parte de tener todas las propiedades de los códigos cíclicos, ademas tiene una distancia mínima predefinida.

Al final de este capítulo extenderemos las definición de los códigos Red-Solomon para obtener los códigos GRS o códigos Red-Solomon Generalizados, que saliendo un poco de los códigos cíclicos son la antesala de nuestro objetivo primordial, los códigos Goppa.

Por ultimo en Capítulo 4 llegaremos a nuestro objetivo, los códigos Goppa, que aparte de ser los precursores del criptosistema de MCEliece de ellos se conocen algoritmos eficientes de descodificación, como el algoritmo de Sugiyama (Vease Sección 4.2) y son la puerta de entrada a los códigos de geometría algebraica, de los que no hablaremos en el trabajo aunque tienen propiedades muy interesante.

1 Códigos lineales

Cuando nuestro objetivo es transmitir información por un canal ruidoso, una de las opciones es usar códigos correctores de errores.
Los codigos correctores de errores no son más que un conjunto de técnicas matemáticas que permiten transmitir mensajes a travez de un canal ruidoso sin perder la información del mismo.
Existen multitud de códigos aunque en este trabajo nos vamos a centrar en una familia de códigos que poseen una estructura matemática que nos facilita su tratamiento, los códigos lineales, dotados de estructura de espacio vectorial.

Imaginamos que queremos transmitir una palabra compuesta por N símbolos a través de un canal ruidoso. Para ello comenzamos por modelar matemáticamente el concepto de código corrector de errores, considerando un conjunto de símbolos, que pueden ser elementos de un cuerpo $\mathbb{F}_q$.

Por otro lado consideramos la palabra que queremos transmitir como una $N-$tupla de $\mathbb{F}_q^N$, es decir, un elemento del espacio vectorial de dimensión N, sobre $\mathbb{F}_q$.

1.1. Introducción

Consideramos $\mathbb{F}_q$ cuerpo finito de q elementos y sobre él construimos el espacio vectorial de todas las n-tuplas, $\mathbb{F}_q^n$.

Definición 1.1. Código Dado $\mathbb{F}_q^n$ espacio vectorial sobre $\mathbb{F}_q$, un (n, M) codigo $\mathcal{C}$ sobre $\mathbb{F}_q$ es un subconjunto de $\mathbb{F}_q^n$ tal que el cardinal de $\mathcal{C}$ es M.

Definición 1.2. Palabras A cada uno de los vectores que componen un (n, M) código $\mathcal{C}$ se les denomina palabras del código.

Si nos fijamos en la definición dada de código, nos percatamos de que basta con que sea un subconjunto de un espacio vectorial, lo que lo dota de una mayor generalidad pero hace que su tratamiento sea también mucho más complicado.

Por lo que comenzaremos por estudiar los códigos que sí poseen una estructura más definida, los códigos lineales.

1.2. Códigos Lineales

Definición 1.3. Código lineal Dado $\mathbb{F}_q^n$ espacio vectorial sobre $\mathbb{F}_q$, un (n, M) código lineal $\mathcal{C}$ sobre $\mathbb{F}_q$ es un subespacio de $\mathbb{F}_q^n$ tal que el cardinal de $\mathcal{C}$ es M.

La condición de ser subespacio, en vez de la de subconjunto, nos reduce el número de códigos construibles pero todos ellos tienen una estructura matemática que nos permite estudiarlos en profundidad ya que son espacios vectoriales.

En primer lugar, por ser $\mathcal{C}$ un espacio vectorial, podemos calcular su dimensión y pasar a referirnos a él en función de ella.

Un $[n, k]$ código lineal $\mathcal{C}$, es un código lineal sobre $\mathbb{F}_q^n$ de dimensión k.

A partir de su dimensión podemos obtener el tamaño, por lo que ambas definiciones son equivalentes.

Lema 1.1. *Un $[n, k]$ código lineal C contiene a q^k palabras.*

Demostración. Para la demostración basta recordar el resultado conocido de que cualesquiera dos espacios vectoriales de dimensión k, son isomorfos. Entonces basta usar la combinatoria para contar los elementos que contienen el espacio $\mathbb{F}_q^k$, es decir q elementos distribuidos en $k-$tuplas, q^k elementos en total. □

Definición 1.4. Matriz Generadora Dado un $[n, k]$ código lineal C, se denomina matriz generadora del código a una matriz $A \in \mathcal{M}_{k\times n}(\mathbb{F}_q)$, cuyas filas son los vectores de una base de $\mathcal{C}$.

Esta definición de matriz generadora es totalmente valida ya que $\mathcal{C}$ posee estructura de espacio vectorial y además por ser $\mathcal{C}$ de dimensión k la matriz generadora es de rango máximo.

Definición 1.5. Redundancia Se denomina redundancia de un $[n, k]$ código lineal sobre $\mathbb{F}_q^n$ al valor $r = n - k$.

La redundancia de un código lineal es directamente proporcional a la seguridad del sistema de trasmisión de información e inversamente proporcional a la eficiencia de la comunicación vista como la carga que soporta nuestro canal al enviar una palabra.

En algunos códigos podemos considerar que las primeras k coordenadas de una palabra, de $\mathcal{C}$, son las que contienen la información a transmitir, las $r = n - k$ restantes son de redundancia.

En este caso podemos construir la matriz generadora de $\mathcal{C}$ de forma única como $G = [\mathcal{I}_k|A]$ con $A \in \mathcal{M}(\mathbb{F}_q)_{k\times r}$ y $\mathcal{I}_k$ la matriz identidad de orden $k \times k$.

Definición 1.6. Dado $\mathcal{C}$ un $[n,k]$ código lineal, se denomina sistemático si y solo si tiene una matriz generadora de la forma

$$G = [\mathcal{I}_k|A]$$

con $A \in \mathcal{M}_{k\times r}(\mathbb{F}_q)$ y $\mathcal{I}_k$ la matriz identidad de orden $k \times k$.

Por último definimos una matriz que nos permitirá comprobar si una palabra cualquiera del espacio $\mathbb{F}_q^n$ pertenece al un código lineal dado.

Definición 1.7. Matriz de Paridad Dado $[n,k]$ código líneal $\mathcal{C}$ se denomina matriz de paridad de $\mathcal{C}$ a una matriz $H \in \mathcal{M}_{r\times n}(\mathbb{F}_q)$, que cumpla

$$\mathcal{C} = \{x \in \mathbb{F}_q^n \mid Hx^T = 0\}.$$

Con esta definición podemos comprobar fácilmente si una palabra pertenece o no a un código dado, pero ¿Cómo obtenemos la matriz de paridad?, en el caso de códigos sistemáticos el siguiente teorema nos proporciona un algoritmo eficiente.

Teorema 1.1. *Si $G = [\mathcal{I}_k|A]$ es una matriz generadora de un $[n,k]$ código lineal $\mathcal{C}$ sistemático, entonces $H = [-A^T|\mathcal{I}_{n-k}]$ es una la matriz de paridad de $\mathcal{C}$*

Demostración. Consideramos por un lado $G = [\mathcal{I}_k|A] \in \mathcal{M}_{k\times n}(\mathbb{F}_q)$ una matriz generadora de un $[n,k]$ código lineal $\mathcal{C}$ y H construida como en el enunciado del teorema $H = \left(\begin{array}{c|c} -A^T & \mathcal{I}_{k-n} \end{array} \right)$.

Calculamos HA^T:

$$HA^T = \left(\begin{array}{c|c} -A^T & \mathcal{I}_{k-n} \end{array} \right) \begin{pmatrix} \mathcal{I}_k \\ A^T \end{pmatrix} = -A^T + A^T = 0 \in \mathcal{M}_{(n-k)\times n}(\mathbb{F}_q).$$

Por lo que puedo asegurar que el espacio generado por A pertenece al $\ker(f)$, donde f es la aplicación lineal definida como $x \to Hx^T$, y además la dimensión de el subespacio generado por las filas de A es un subespacio de dimensión k, coincidiendo con la dimensión del $ker(f)$. □

Para el caso no sistemático, basta con resolver el sistema lineal generado por $Mx^T = 0$, donde M es la matriz generadora.

1.2.1. Códigos Duales

Durante toda la sección hemos tratado a $\mathbb{F}_q^n$ como espacio vectorial pero podemos dotar a $\mathbb{F}_q^n$ del producto escalar usual.

Definición 1.8. Dado $\mathcal{C}$ un $[n,k]$ código lineal, se denomina código dual de $\mathcal{C}$ al conjunto

$$\mathcal{C}^\perp = \{x \in \mathbb{F}_q^n \mid x \cdot c = 0 \ \forall c \in \mathcal{C}\}.$$

La pregunta ahora es cómo construir a partir de un código lineal $\mathcal{C}$ su código dual. En el Capítulo 2 encontraremos técnicas eficientes para generar el código dual de un código cíclico[1].

En el caso lineal basta con resolver las ecuaciones cartesianas generadas por $xM = 0$, siendo M la matriz generadora de un código.

Por otro lado demostramos que la matriz generadora del dual de un código lineal coincide con la matriz de control de paridad del código.

Teorema 1.2. *Dado $\mathcal{C}$ un $[n,k]$ código lineal y su código dual $\mathcal{C}^\perp$, entonces la matriz de control de paridad de $\mathcal{C}$ es la matriz generadora de $\mathcal{C}^\perp$.*

Demostración. Por definición (véase Def. 1.7), tenemos que encontrar H tal que

$$\mathcal{C} = \{x \in \mathbb{F}_q^n \mid Hx^T = 0\}.$$

Tomamos M, matriz generadora del código dual $\mathcal{C}^\perp$ y ademas sabemos por definición que para todo $x \in \mathbb{F}_q^k$

$$x \cdot M \in \mathcal{C}^\perp \Rightarrow x \cdot M \cdot c = 0 \ \forall c \in \mathcal{C} \Rightarrow Mc = 0 \ \forall c \in \mathcal{C}.$$

$$\boxed{M \text{ es la matriz de control de paridad de } \mathcal{C}.}$$

□

1.3. Distancias y pesos

Una vez dados algunos preliminares sobre códigos lineales, podemos dotarlos de una estructura de espacio métrico.

[1] Como veremos en el Capítulo 2, los códigos cíclicos son un subconjunto de los lineales, aunque tiene una estructura especial

Definición 1.9. (Distancia de Hamming) Dado $\mathcal{C}$ un código lineal sobre $\mathbb{F}_q$, se define como distancia la aplicación $d : \mathbb{F}_q^n \times \mathbb{F}_q^n \to \mathbb{R}$, que asigna a cada par de vectores el numero de componentes distintas.

Esta aplicación fue introducida por Richard Hamming (véase [5]) y se le conoce como métrica de Hamming, ya que como veremos a continuación, es efectivamente una métrica, lo que dota a los códigos lineales de estructura de espacio métrico.

Teorema 1.3. *Dado $\mathcal{C}$ un código lineal sobre $\mathbb{F}_q$, la función distancia $d : \mathcal{C} \times \mathcal{C} \to \mathbb{R}$ definida en (Def. 1.9) cumple*

1. *Es no negativa ($\forall\ x, y \in \mathcal{C} d(x, y) \leq 0$) y la igualdad se da solo cuando $x = y$*
2. *Simétrica ($d(x, y) = d(y, x)\ \forall x, y \in \mathcal{C}$)*
3. *Cumple la desigualdad triangular ($d(x, z) = d(x, y) + d(y, x)\ \forall x, y, z \in \mathcal{C}$)*

Demostración. Dados $x, y\ \in \mathcal{C}$ comenzamos por definir el conjunto

$$\Gamma_{x,y} = \{i \in 1, \ldots, n\ /\ x_i - y_i \neq 0\}$$

Puedo definir la distancia a partir de este conjunto de la siguiente forma

$$d(x, y) = \#\Gamma_{x,y}\ \forall (x, y) \in \mathcal{C} \times \mathcal{C}$$

Definir de esta forma la distancia es totalmente lícito ya que el conjunto $\Gamma_{x,y} \subseteq \{1, \ldots, n\}$, por lo que es finito $\forall x, y \in \mathcal{C}$. Esta nueva definición nos proporciona la no negatividad de la función distancia, y ademas su mínimo se alcanza en 0 cuando $\Gamma_{x,y} = \emptyset$

$$\#\Gamma_{x,y} = 0 \Leftrightarrow \Gamma_{x,y} = \emptyset \Leftrightarrow x_i = y_i\ \forall\ i \in \{1, \ldots, n\} \Leftrightarrow x = y$$

Por otro lado tengo que

$$\Gamma_{x,y} = \{i \in 1, \ldots, n\ /\ x_i - y_i \neq 0\} = \{i \in 1, \ldots, n\ /\ x_i \neq y_i\} = \{i \in 1, \ldots, n\ /\ 0 \neq y_i - x_i\} = \Gamma_{y,x}$$

Lo que me afirma la simétria de la función distancia, y por último sólo queda probar la desigualdad triangular. Dados $x, y, z \in \mathcal{C}$ basta observar que si $x_i \neq z_i$ entonces $x_i \neq y_i$ o $y_i \neq z_i$, por lo que

$$\boxed{d(x, z) =} \#\,\Gamma_{x,z} = \#\left(\{i \in 1, \ldots, n\ /\ x_i \neq y_i\} \cup \{i \in 1, \ldots, n\ /\ y_i \neq z_i\}\right) \boxed{\leq}$$

$$\#\left(\{i \in 1, \ldots, n\ /\ x_i \neq y_i\}\right) + \#\left(\{i \in 1, \ldots, n\ /\ y_i \neq z_i\}\right) = \#\Gamma_{x,y} + \#\Gamma_{y,z} = \boxed{d(x, y) + d(y, z).}$$

□

Una vez demostrado que la aplicación definida en la Def. 1.9 es una métrica, puedo dotar a los códigos lineales de estructura de espacio métrico.

Definición 1.10. (Peso de Hamming) Dado $\mathbb{F}_q^n$, entonces la aplicación $w : \mathbb{F}_q^n \to \mathbb{N}$ que a cada vector le asigna el numero de coordenadas distintas de cero, se le denomina peso de Hamming,

$$w(x) = d(x, 0).$$

La función peso de un código $\mathcal{C}$ está plenamente relacionada con la función distancia definida en Def. 1.9, como manifiesta el siguiente teorema.

Teorema 1.4. *Sea $\mathcal{C}$ un código lineal $[n, k]$ y además $x, y \in \mathcal{C}$ entonces $d(x, y) = w(x - y)$.*

Demostración. Consideramos $x, y \in \mathcal{C}$ entonces

$$w(x - y) = \#\{i \in 1, \ldots, n \ / \ x_i - y_i \neq 0\} = \#\{i \in 1, \ldots, n \ / \ x_i \neq y_i\} = d(x, y).$$

□

Una vez demostrada la intima relación entre la función distancia y la función peso podemos definir otra característica de todo código lineal, como es su distancia (peso) mínima(o).

Definición 1.11. Sea $\mathcal{C}$ un código lineal sobre $\mathbb{F}_q^n$, se define la mínima distancia del código como $\min_{\substack{x,y \in \mathcal{C} \\ x \neq y}} d(x, y)$.

Esta definición es totalmente lícita ya que el conjunto $\mathcal{C}$ es finito, sobre $\mathbb{F}_q$.

Por otro lado, de mano de la relación dada en (Teorema 1.4), podemos demostrar que la distancia mínima coincide con el peso mínimo de un código.

A partir de ahora podemos caracterizar los códigos lineales por 3 parámetros.

Notación: Dado $\mathbb{F}_q$ un cuerpo finito de orden q, un código lineal de longitud n, dimensión k y distancia mínima de Hamming d, se denota como un $[n, k, d]$ código lineal.

1.4. Tamaño del código

En la teoría de códigos, es importante conocer los parámetros de los códigos, o al menos conocer de antemano cotas para él.

Consideraremos en esta sección conceptos generales de códigos lineales y no lineales. En el caso de códigos no lineales, no tiene sentido referirnos a su dimensión por lo que sustituiremos el parámetro k por M que denota el tamaño del código, es decir el número de palabras que lo componen. Para este menester se introducen distintas cotas superiores o inferiores para los tamaños, algunas de las cuales desarrollaremos en esta sección.

Notación: Dado $\mathcal{C}$ un código (lineal o no lineal) sobre $\mathbb{F}_q$, con distancia mínima d y longitud n, se denota el tamaño de $\mathcal{C}$ a $A_q(n,d)$.En el caso de códigos lineales se denota con $B_q(n,d)$.

Para comenzar pasamos a dar una de las cotas más conocidas, la cota Singleton, introducida por Richard C. Singleton en [10].

Esta cota nos proporcionara la definición de los códigos con distancia máxima separable o MDS (por sus siglas en ingles), que a su vez contienen otra familia muy importante,los códigos Reed-Solomon.

Teorema 1.5. *Dados $d \leq n$:*

$$A_q(n,d) \leq q^{n-d+1}.$$

En el caso de códigos $[n,k,d]$ lineales sobre $\mathbb{F}_q$,si existen, cumplen que $k \leq n-d+1$.

Antes de la demostración es necesario dar algunos resultados previos necesarios.

Lema 1.2. *$B_q(n,d) \leq A_q(n,d)$ y además $B_q(n,d)$ es una potencia no negativa entera de q.*

Demostración. Para la primera parte es simplemente darse cuenta que todos los códigos lineales pueden ser vistos como códigos no lineales, es decir, la clase de códigos no lineales contiene a los lineales, por lo que $B_q(n,d) \leq A_q(n,d)$.

El segundo enunciado basta utilizar la dimensión del espacio vectorial, $k = \log_q(M)$, donde M es el número de palabras del código considerado, entonces despejando obtenemos que $M = q^k$ y como $k \in \mathbb{Z}^+$, podemos afirmar que $B_q(n,d)$ es una potencia no negativa entera de q. □

Lema 1.3. *$A_q(n,n) = B_q(n,n) = q$.*

Demostración. En el caso de los códigos lineales es bastante simple, basta considerar que si tienen tamaño q es por que el código $\mathcal{C}$ es de la siguiente forma:

$$\mathcal{C} = \{\lambda(1,\cdots,1) \mid \lambda \in \mathbb{F}_q\} \subseteq \mathbb{F}_q^n.$$

Tiene longitud n, $\#\mathcal{C} = q$ y distancia mínima n, por lo que $B_q(n,n) = q$.

En el caso no lineal, por el lema anterior sabemos que $A_q(n,n) \geq B_q(n,n)$, suponemos por reducción al absurdo que $A_q(n,n) > B_q(n,n) = q$.

Si $A_q(n,n) > q$ entonces existe un código de más de q palabras y distancia mínima n, por lo que al menos dos palabras coinciden en alguna coordenada, lo que implica que tienen distancia mínima menor estricta que n. Contradicción, por lo que

$$\boxed{A_q(n,n) = B_q(n,n) = q}.$$

□

Lema 1.4. *$A_q(n,d) \leq q \cdot A_q(n-1,d)$ y además $B_q(n,d) \leq q \cdot B_q(n-1,d)$.*

Demostración. Partimos de un código $\mathcal{C}$ sobre $\mathbb{F}_q$, de longitud n, distancia mínima d y de tamaño $M = A_q(n,d)$. Consideramos ahora el subcódigo

$$\mathcal{C}_a = \{x \in \mathcal{C} \mid x = (\cdots, a)\}$$

Es decir, las palabras que tienen a en la coordenada n, $\#\mathcal{C}_a \leq M/q$.

Por otro lado el código $\mathcal{C}_a$ es un código de orden $n-1$ y distancia mínima d, por lo que despejando obtenemos que

$$M/q \leq A_q(n-1,d) \Rightarrow M \leq q \cdot A_q(n-1,d) \Rightarrow \boxed{A_q(n,d) \leq q \cdot A_q(n-1,d)}.$$

El mismo razonamiento podemos seguir para los códigos lineales y obtener que $\boxed{B_q(n,d) \leq q \cdot B_q(n-1,d).}$

□

Demostración de Teorema 1.5. Por un lado en el caso que $d = n$, el segundo de lema nos asegura que $A_q(n,n) = q \leq q^{n-n+1} = q$.

Supongamos ahora el caso $d < n$, por el tercer lema tenemos que $A_q(n,d) \leq q \cdot A_q(n-1,d)$, por inducción podemos obtener que $A_q(n,d) \leq q^{n-d} A_q(d,d)$, por lo que aplicando el segundo lema tenemos que

$$\boxed{A_q(n,d) \leq q^{n-d+1}.}$$

Por último en el caso de códigos $[n,k,d]$ lineales tenemos que el tamaño es q^k, además $q^k \leq B_q(n,d)$ y aplicando la primera parte del teorema tenemos que

$$q^k \leq q^{n-d+1} \Rightarrow \boxed{k \leq n-d+1}$$

□

También tenemos otros tipos de cotas como son las asintóticas, entre ellas una de las más importantes introducida por Gilbert y arshamov que estudiaremos en la sección Sección 4.3.

Dejando un poco atrás las cotas para el tamaño de los códigos, también encontramos lo que se denominan cotas para la distancia mínima, donde dado un tamaño del código podemos dar cotas inferiores para la distancia mínima.

1.5. Corrección de errores de transmisión

Cuando transmitimos información, tenemos que asumir que inevitablemente se producen errores en la transmisión, para ello se usan los códigos correctores de errores.

A parte de detectar si una palabra ha sufrido modificaciones o no, es muy importante saber corregir los errores asumidos y en este punto juega un papel crucial la distancia mínima de un código.

Ejemplo: Supongamos que transmitimos la palabra a a través del un canal ruidoso y recibimos la palabra $b = a + \epsilon$, siendo ϵ el error soportado. Una forma muy fácil de calcular el error soportado conociendo ambos extremos de la comunicación es usando la distancia.

En virtud del ejemplo anterior podemos afirmar que, si el canal siempre produce el mismo error, cosa bastante improbable, basta con enviar una palabra de control para detectar el error y a partir de ahí ya podemos corregir todos los errores producidos. En esta situación no sería necesario emplear un código corrector de errores, basta con usar la distancia de Hamming como herramienta. Este supuesto es bastante exótico ya que en la realidad de los canales ruidosos por lo en general no es tan fácil resolver el problema sin usar códigos correctores.

En esta sección vamos a tratar de ver que cantidad de errores puede asumir un código, en función de sus 3 parámetros.

1.5.1. Esferas

Una de los principales objetivos de los códigos correctores de errores es llegar a conseguir la mayor eficiencia posible en la comunicación.

Es claro que un código con mucha redundancia será muy seguro y será casi imposible pasar por correcto un mensaje que ha sufrido alguna modificación pero, ¿Es asumible el coste de transmitir 100 bit para solo enviar una palabra de 10 bit?, la respuesta dependerá del propósito en cada caso, pero como norma general queremos llegar a un equilibrio.

Dado un $[n, k, d]$ código lineal $\mathcal{C}$, podemos usar la estructura de espacio métrico para determinar su capacidad de corregir errores, para lo que usaremos las esferas.

Definición 1.12. Dado $r \in \mathbb{N}$ y $\mathbb{F}_q^n$ un espacio métrico, con la distancia de Hamming, se define la aplicación esfera

$$S_r : \mathbb{F}_q^n \to \mathcal{P}(\mathbb{F}_q^n)$$
$$S_r(u) = \{x \in \mathbb{F}_q^n \ / \ \mathrm{d}(x,u) \leq r\}.$$

A partir de la definición anterior, podemos calcular el cardinal de una esfera dada sobre $\mathbb{F}_q^N$.

Teorema 1.6. *Dado $r \in \mathbb{N}$ y $\mathbb{F}_q^n$ un espacio métrico con la distancia de Hamming, entonces*

$$\#(S_r(u)) = \sum_{i=0}^{r} \binom{n}{i} (q-1)^i.$$

Demostración. Para demostrarla supongamos que $u = 0 \in \mathbb{F}_q^n$, ya que en otro caso por traslación podemos conseguirlo. Entonces por definición de distancia, tenemos que $d(x,0) \leq r$, lo implica que x tenga menos de $r+1$ coordenadas distintas de 0, por lo que consideramos

$$\#(S_r(u)) = \sum_{i=0}^{r} \#\{x \in \mathbb{F}_q^n \mid d(x,0) = i\}.$$

Por otro lado aplicando combinatoria tenemos que

$$\#\{x \in \mathbb{F}_q^n \mid d(x,0) = i\} = \binom{n}{i} (q-1)^i.$$

Ya que tenemos n posibles lugares, para colocar i coordenadas distintas de 0 y además en cada posición, podemos poner $q-1$ elementos.

$$\boxed{\#(S_r(u)) = \sum_{i=0}^{r} \binom{n}{i} (q-1)^i}.$$

□

1.5.2. Algoritmo de corrección de errores

Una vez dadas las palabras que componen una esfera, podemos pasar a estudiar una forma de corregir errores.

Uno de los métodos de corrección de errores se fundamenta en encontrar la palabra más cercana a la recibida que pertenece al código, por lo que para corregir de forma correcta necesitamos que las palabras estén lo mas separadas posibles.

En la sección anterior (véase Subsección 1.5.1) introducimos las esferas que nos serán de ayuda, ya que encontrando esferas con un solo elemento, vamos a poder corregir todos los errores que produzcan palabras dentro de la esfera (aunque no dentro del código).

Teorema 1.7. *Si d es la distancia mínima de un código lineal $\mathcal{C}$ (lineal o no lineal), entonces las esferas de radio $r = \lfloor \frac{d-1}{2} \rfloor$ tienen un único elemento en $\mathcal{C}$, es decir, su centro.*

Demostración. Dados $c_1, c_2 \in \mathcal{C}$ supongamos que $z \in S_r(c_1) \cap S_t(c_2)$, entonces aplicando la desigualdad triangular, tenemos que

$$d(c_1, c_2) \leq d(z, c_1) + d(z, c_2) \leq 2r < d,$$

lo que implica que $c_1 = c_2$, contradicción.

□

La idea principal que estudiaremos se basa en poder corregir todas las palabras que caigan dentro de una esfera con una única palabra de $\mathcal{C}$.

Corolario 1.1. *Dado $\mathcal{C}$ un código, si la palabra enviada es c y la recibida es y, con menos de r errores, entonces c es la uníca palabra de $\mathcal{C}$ que pertenece a la esfera $S_r(y)$.*

En la Figura 1.1 podemos encontrar un boceto de lo que ocurre en la transmisión.

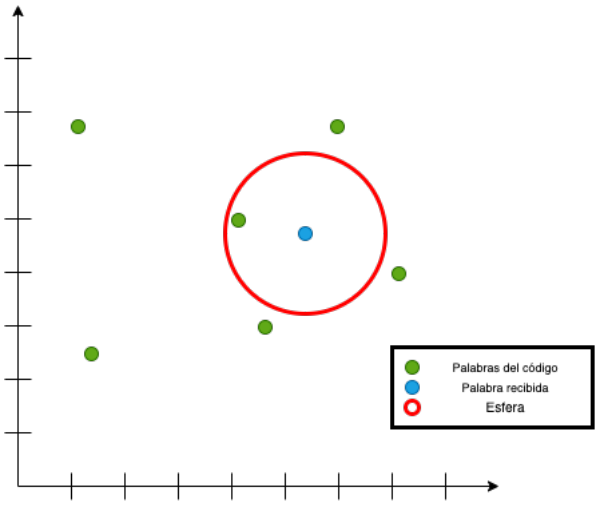

Figura 1.1: Correción de errores de transmisión

Estos teoremas ya nos dan un algoritmo, muy básico, para la corrección de las palabras recibidas.

El problema de corrección de errores, pasa a ser ahora un problema de encontrar un algoritmo eficiente que sea capaz de corregir hasta r errores, uno de los más conocidos es el implementado en Algoritmo 1.

Algoritmo 1 Algoritmo de Corrección Errores

Input: Código cíclico $[n, k, d]$, $\mathcal{C}$ y $y \in \mathbb{F}_q^n$
Output: $c \in \mathcal{C}$
1: $r = \lfloor \frac{d-1}{2} \rfloor$
2: Construimos la esfera $S_r(y)$
3: $c =$ única palabra de $\mathcal{C}$ tal que $c \in S_r(y)$.

Este algoritmo es la base de todos los algoritmos de decodificación que vamos a ver en este trabajo, en secciones posteriores introduciremos lo que se denomina descodificación por síndrome, que precisará de algunos cálculos, pero una vez realizados podemos decodificar todas las palabras que provengan de ese código.

2 Códigos cíclicos

Dentro de los código lineales podemos encontrar una subclase que son códigos cíclicos, que aunque son menos generales, presentan una estructura matemática más compleja lo que los dota de propiedades muy interesantes sobre las que versaremos en este capítulo.

Los códigos cíclicos o también llamados códigos de redundancia cíclica, como ya hemos comentado, son códigos lineales que además se pueden ver como un ideal de un espacio de polinomios.

A lo largo del capítulo le daremos forma, pero en resumen un código cíclico se puede ver como un ideal de $\mathcal{R}^n = \frac{\mathbb{F}_q[x]}{\langle x^n+1\rangle}$.

2.1. Introducción

Comenzamos por dar una definición de estos nuevos códigos que luego reformularemos más adelante a lo largo del capítulo.

Definición 2.1. Un código lineal $\mathcal{C}$ será un código cíclico si y solo si para todo vector $\mathbf{c} = c_0c_1\dots c_{n-1} \in \mathcal{C}$, el vector $\hat{\mathbf{c}} = c_{n-1}c_0\dots c_{n-2}$ pertenece a $\mathcal{C}$.

En vistas de la definición, nos damos cuenta de que podemos permutar la letras de una palabra de manera cíclica, valga la redundancia, sin salirnos del código.

$$\mathbf{c} = c_0c_1\dots c_{n-1} \in \mathcal{C} \Rightarrow \hat{\mathbf{c}} = c_{\sigma(0)}c_{\sigma(1)}\dots c_{\sigma(n-1)} \in \mathcal{C} \text{ con } \sigma(i) = i + t \pmod{n}.$$

Por otro lado podemos representar a un código lineal sobre $\mathbb{F}_q[x]$ como polinomios de grado a lo máximo $n-1$.

Para ello usaremos el siguiente teorema conocido.

Teorema 2.1. *Existe una biyección entre los polinomios de grado $n-1$ en $\mathbb{F}_q[x]$ y los vectores $\boldsymbol{c} = c_0c_1\dots c_{n-1}$ de $\mathbb{F}_q^n$.*

En virtud del Teorema 2.1, podemos identificar los códigos lineales con polinomios de la siguiente forma

$$\mathbf{c} = c_0c_1 \dots c_{n-1} \equiv c(x) = c_0 + c_1x + \cdots + c_{n-1}x^{n-1}.$$

La Def. 2.1, nos da una propiedad muy importante que los códigos cíclicos tienen a diferencia de un código lineal general, por lo que nuestra intención es trasladar esto al anillo de polinomios $\mathbb{F}_q^n[x]$.

$$h(c_{\sigma(0)}c_{\sigma(1)} \cdots c_{\sigma(n-1)}) = x^t h(c) \text{ tal que } \sigma(i) = i + t \pmod n \quad t \in \mathbb{N} \quad t < n.$$

Esta definición no coincide exactamente con la dada ya que $x^t h(c) = c_0x^t + c_1x^{t+1} + \cdots + c_{n-1}x^{t+n-1} \neq c_t + c_{t-1}x + \cdots + c_{n-t-1}x^{n-1} = h(c_{n-t} \cdots c_{n-1}c_0 \cdots c_{n-t-1})$,

donde h es el isomorfismo dado por Teorema 2.1.

Para solucionarlo basta con que $x^n = 1$, para lo que nos debemos restringir elementos al conjunto de clases residuales de resto $\mathcal{R}_n = \mathbb{F}_q[x]/\langle x^n - 1\rangle$.

Después de todo este proceso, finalmente obtenemos una correspondencia biunívoca entre los códigos cíclicos y el álgebra $\mathcal{R}_n[x] = \mathbb{F}_q[x]/\langle x^n - 1\rangle$ (conjunto de las clases de resto de $\mathbb{F}_q[x]$ módulo $x^n - 1$.)

Por otro lado, usando un resultado conocido

Teorema 2.2. *El conjunto $\mathcal{R}_n$ de clases de resto de $\mathbb{F}_q[x]$, forma una $\mathbb{F}_q -$ algebra con dimension n,sobre el cuerpo $\mathbb{F}_q$.*

Teorema 2.3. *En el álgebra $\mathcal{R}_n$ un subconjunto es la imagen por el isomorfismo dado en Teorema 2.1 de un código $\mathcal{C}$ cíclico si y solo si es un ideal de $\mathcal{R}_n$.*

Demostración. La base principal de la demostración se centra en la idea de que multiplicar por x, en $\mathcal{R}_n$, es una permutación cíclica de los coeficientes.

$$x(a_{n-1}x^{n-1} + a_{n-2}x^{n-2} \cdots a_0) = a_{n-1}(x^n - 1) + a_{n-2}x^{n-1} \cdots a_0x + a_{n-1}.$$

Asi que tomando clases modulo $x^n - 1$,

$$a_{n-1}\overline{(x^n - 1)} + a_{n-2}\overline{x^{n-1}} \cdots a_0\overline{x} + a_{n-1} \equiv a_{n-2}\overline{x^{n-1}} \cdots a_0\overline{x} + a_{n-1}.$$

Por un lado tengo que si V es un ideal de $\mathcal{R}_n$ y $v \in V$, entonces $\overline{x}v \in V$, por lo que aplicando lo anterior tengo que V es un subespacio cíclico.

Por otro lado supongamos que V es un subespacio cíclico, entonces por definición de ideal para todo $v \in V$ $\overline{x}v \in V$, por inducción, para todo $v \in V$ tenemos que $\overline{x}^j v = \overline{x^j}v \in V \ \forall j \in \mathbb{N}$.

Como V es un subespacio tenemos que cualquier combinación lineal se queda en V, por lo que

$$c_{n-1}\overline{x^{n-1}}v + c_{n-2}\overline{x^{n-1}}v + \cdots + c_0 v = \overline{(c_{n-1}x^{n-1} + c_{n-2}x^{n-2} + \cdots + c_0)}v,$$

pertenece a V, y entonces el producto de cualquier elemento de V con un elemento cualquiera de $\mathcal{R}_n$ pertenece a V, por lo que V es un ideal de $\mathcal{R}_n$. □

2.2. Estructura

Una vez dados todos los preliminares teóricos, podemos pasar a dar teoremas de estructura para ver claramente las ventajas de trabajar con códigos cíclicos.

Teorema 2.4. *Dado un $\mathcal{C} \subseteq \mathcal{R}_n$ un código cíclico no nulo sobre $\mathbb{F}_q$. Existe un polinomio $g(x) \in \mathcal{R}_n$ que cumple lo siguiente:*

1. $\mathcal{C} = \langle g(x) \rangle$,[1]
2. *$g(x)$ es el único polinomio mónico de grado mínimo en $\mathcal{C}$,*
3. $g(x) | (x^n - 1)$.

Demostración. Tomamos $g(x)$ un polinomio mónico y de grado mínimo de $\mathcal{C}$, cuya existencia esta asegurada por $\mathcal{C}$ ser no nulo.

Considero ahora $c(x) \in \mathcal{C}$, aplicamos el Algoritmo de División en $\mathbb{F}_q[x]$ $c(x) = g(x)h(x) + r(x)$, donde $r(x) = 0$ ó $\deg(r(x)) < \deg(g(x))$.

Por usando la minimalidad de $g(x)$ podemos afirmar que $\deg(r(x)) = 0 \Rightarrow r(x) = 0$.

$$\boxed{c(x) = g(x)h(x) \;\; \forall c(x) \in \mathcal{C} \Rightarrow \mathcal{C} = \langle g(x) \rangle.}$$

Además la unicidad del algoritmo de división nos asegura que

$$\boxed{g(x) \text{ es el único polinomio mónico de grado mínimo en } \mathcal{C}.}$$

Para 3) podemos usar un argumento similar $x^n - 1 = g(x)h(x) + r(x)$ donde $r(x) = 0$ ó $\deg(r(x)) < \deg(g(x))$. Como $x^n - 1 \equiv 0$ en $\mathcal{R}_n$ y además $r(x) \in \mathcal{C}$ podemos afirmar que

$$\boxed{x^n - 1 = g(x)h(x) \Rightarrow g(x) | (x^n - 1).}$$

□

[1] $\langle g(x) \rangle$ denota el ideal principal generado por $g(x)$ sobre $\mathcal{R}_n$

2.2.1. Codificación y matriz generadora

Una vez dada la estructura de un códigos cíclicos y además una forma de generar todas sus palabras a partir de generar el ideal correspondiente. Paso a dar una forma eficiente de codificar palabras, como es la matriz generadora del códigos.

Lema 2.1. *Dado $\mathcal{C}$ un código cíclico, el conjunto*

$$\{g(x), xg(x), \cdots, x^{k-1}g(x)\}$$

forma una base de $\mathcal{C}$.

Demostración. Consideramos $k = n - \deg g(x)$.

Si $c(x) \in \mathcal{C}$ con $c(x) \neq 0$ o $\deg c(x) < n$ entonces $c(x) = g(x)f(x)$. Tenemos dos casos:

1. Si $c(x) = 0$ entonces $f(x) = 0$.
2. Si $c(x) \neq 0$ entonces $\deg c(x) < n$ entonces tengo que $\deg f(x) < k$.

Por lo que en ambos casos podemos considerar

$$\mathcal{C} = \{g(x)f(x) \mid f(x) = 0 \text{ ò } \deg f(x) < k\}.$$

Por lo que fijando k , tenemos que una base de $\{f \in \mathbb{F}_q[x] \mid f \text{ es un polinomio de grado menos que } k\}$ es

$$\{1, x, x^2, \cdots, x^{k-1}\}.$$

Por lo que podemos encontrar una base de $\mathcal{C}$

$$\{g(x), xg(x), x^2g(x), \cdots, x^{k-1}g(x)\}.$$

□

Una vez encontrada una base de los códigos cíclicos, podemos calcular su matriz generadora para poder codificar palabras.

Teorema 2.5. *Dados $\mathcal{C} = \langle g(x) \rangle$ un código cíclico sobre $\mathbb{F}_q$. Entonces*

$$M = \begin{pmatrix} g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} \end{pmatrix}$$

es una matriz generadora del código.

Demostración. Conocida la base de un código $\mathcal{C}$ (Véase *Lema* 2.1) podemos colocar por filas los elementos de la base en coordenadas[2].Por lo que

$$M = \begin{pmatrix} g(x) & & & \\ & xg(x) & & \\ & & \ddots & \\ & & & x^{k-1}g(x) \end{pmatrix} = \begin{pmatrix} g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & g_0 & g_1 & g_2 & \cdots & g_{n-k} \end{pmatrix}$$

es la matriz generadora del código $\mathcal{C}$. □

Como en el caso de códigos lineales, una vez dada la matriz generadora del código, tengo un algoritmo eficiente de codificar palabras.

Algoritmo 2 Algoritmo de Codificación

Input: Código cíclico $\mathcal{C}$ y $m \in \mathbb{F}_q^k$
Output: $c \in \mathbb{F}_q^n$
1: Calcular la matriz M de generadora de $\mathcal{C}$.
2: $c = Mn$

2.2.2. Matriz de control de paridad

Por otro lado podemos ver una forma de comprobar fácilmente si un polinomio pertenece o no a un código cíclico.En el Capítulo 1 hablemos de la matriz de control de paridad de un código lineal (véase Def. 1.7), pero gracias a la estructura que nos brindan los códigos cíclicos, podemos hablar de polinomio de control de un código cíclico $\mathcal{C}$, además este polinomio nos proporcionará de forma directa la matriz de control de paridad de un código cíclico.

Definición 2.2. Dado $\mathcal{C}$ un código cíclico y $g(x)$ su polinomio generador, se denomina polinomio de control de $\mathcal{C}$ a

$$h(x) = \frac{x^n - 1}{g(x)} \in \mathcal{R}_n.$$

Basta destacar que la definición anterior es totalmente lícita ya que $g(x)$ es divisor de $x^n - 1$ por definición.

Una vez introducido el concepto de polinomio de control de un código cíclico, en el siguiente teorema resumiremos sus propiedades.

[2]Importante, todos los polinomios están vistos en $\mathcal{R}_n$.

Teorema 2.6. *Dado $\mathcal{C}$ un código cíclico y sea $h(x)$ su polinomio de control, entonces se cumple*

$$\mathcal{C} = \{p(x) \in \mathcal{R}_n \mid p(x)h(x) = 0\}.$$

Demostración. Vamos a proceder por doble inclusión para demostrar la igualdad.

1. $\boxed{\subseteq}$ Tomamos $c(x) \in \mathcal{C}$, por definición tenemos que existe $k(x)$ tal que $c(x) = k(x)g(x)$. Entonces ahora calculamos $c(x) \cdot h(x)$

$$c(x) \cdot h(x) = k(x) \cdot g(x) \cdot h(x) = k(x)g(x)\frac{x^n - 1}{g(x)} = k(x)x^n - 1 = 0.$$

2. $\boxed{\supseteq}$ Tomamos $p(x) \in \mathcal{R}_n$ tal que $p(x)h(x) = 0$. Usando el algoritmo de la división tengo que

$$p(x) = q(x)g(x) + r(x) \Leftrightarrow 0 = h(x)p(x) = h(x)q(x)g(x) + h(x)r(x) = \boxed{h(x)r(x) = 0.}$$

Por otro lado tenemos que el grado de $r(x)$ es menor que k, por lo que

$$\deg(h(x)r(x)) < k + n - k = n \wedge h(x)r(x) = 0 \Rightarrow r(x) = 0 \Rightarrow p(x) \in \mathcal{C},$$

$$\boxed{\mathcal{C} = \{p(x) \in \mathcal{R}_n \mid p(x)h(x) = 0\}.}$$

□

Las propiedades de los polinomios generadores nos recuerdan en gran medida a los códigos duales u ortogonales (véase Def. 1.8). En el caso lineal construir un código dual, se fundamenta en encontrar un subespacio ortogonal a $\mathcal{C}$ en el caso de los cíclicos su estructura nos lo facilita en gran medida.

Por ello vamos a pasar a estudiar los códigos duales de los códigos cíclicos, que como vemos en el siguiente teorema, también son códigos cíclicos.

Teorema 2.7. *Dado $\mathcal{C}$ un código cíclico tenemos que su dual , $\mathcal{C}^\perp$, es un código cíclico.*

Demostración. Por un lado tenemos que construir el código dual de $\mathcal{C}$, para poder demostrar que es cíclico.

$$\mathcal{C}^\perp = \{x \in \mathbb{F}_q^n \mid x \cdot c = 0 \ \forall c \in \mathcal{C}\}.$$

Y ahora basta ver que si $c \in \mathcal{C}$ entonces $xc \in \mathcal{C}^\perp$, para demostrar que $\mathcal{C}$ es cíclico. Tomamos $\hat{c} \in \mathcal{C}^\perp$, entonces

$$\hat{c} \cdot k = 0 \ \forall k \in \mathcal{C} \Rightarrow x \cdot \hat{c} \cdot k = 0 \Rightarrow x\hat{c} \in \mathcal{C}^\perp.$$

Afirmamos que $\mathcal{C}^{\perp}$ es cíclico. □

Una vez vista esta propiedad, basta con encontrar el polinomio generador del código dual. Aunque esta tarea podría parecer simple teniendo el polinomio de control de $\mathcal{C}$, no es así.

El concepto de código dual (véase Def. 1.8), está definido en términos de vectores de $\mathbb{F}_q^n$ y depende del producto escalar, pero el isomorfismo que construimos entre $\mathbb{F}_q^n$ y $\mathcal{R}_n$, no conserva el producto escalar usual.

Observación 2.1. En este capítulo vamos a introducir la matriz de paridad usando el polinomio generador, y conservando el isomorfismo usual entre $\mathbb{F}_q^n$ y $\mathcal{R}_n$ (véase Teorema 2.1) y el producto escalar usual, aunque se podría redefinir el producto escalar para que se conservara por el isomorfismo.

Para solucionar este aparente problema usamos el concepto de polinomio reciproco

$$f(x) = f_k x^k + f_{k-1}x^{k-1} + \cdots + f_1 x + f_0 \Leftrightarrow f_R(x) = f_0 x^k + f_1 x^{k-1} + \cdots + f_{k-1}x + f_k.$$

Definición 2.3. Dado $f(x) \in$
$mathbbF_q^n$ de grado k, se define como polinomio recíproco al polinomio

$$f_R = x^k f(x^{-1}).$$

Una de las propiedades que necesitamos para nuestro propósito sera la idempotencia de la aplicación reciproco, es decir

$$(\cdot)_R : \mathbb{F}_q[x] \to \mathbb{F}_q[x]$$
$$(f)_R = f_R.$$

Esta propiedad se resume en la siguiente proposición

Proposición 2.1. *Dado $f(x) \in \mathcal{F}_q[x]$ de grado k, entonces se cumple que*

$$(f_R)_R\,(x) = f(x).$$

Demostración.

$$(f_R)_R\,(x) = \left(x^k f(x^{-1})\right)_R\,(x) = x^k x^{-k} f(x) = f(x).$$

□

Después de todos estos preliminares podemos dar una relación entre el producto escalar de dos vectores en $\mathbb{F}_q^n$ y el producto de sus polinomios asociados en $\mathcal{R}_n$.

Teorema 2.8. *Dados a y b vectores en $\mathbb{F}_q^n$ y $\hat{a}(x)$, $\hat{b}(x)$ sus polinomios asociados, entonces a es ortogonal a b y a todas sus permutaciones cíclicas si y solo si $\hat{a}(x) \cdot \hat{b}_R(x) = 0$ en $\mathcal{R}_n$.*

Demostración. Supongamos $b^{(i)}$ una permutación cíclica de b de orden i entonces

$$a \cdot b^{(i)} = 0 \Leftrightarrow \sum_{j=0}^{n-1} a_j b_{j+i} = 0.$$

Pero por otro lado tenemos que

$$a(x)b_R(x) = 0 \Leftrightarrow a(x)(x^{n-1-\deg(b(x))})b(x) = 0.$$

$$a(x)(x^{n-1-\deg(b(x))})b(x) = sum_{i=0}^{n-1} \left(\sum_{j=0}^{n-1} a_j b_{j+i} x^{n-1-i} \right).$$

Esta última implicación se da si se cumple la primera, es decir, si y solo si $a \cdot b^{(i)} = 0 \ \forall i$.

Por lo que podemos afirmar que

$$\boxed{a \cdot b^{(i)} = 0 \ \forall i \Leftrightarrow \hat{a}(x) \cdot \hat{b}_R(x) = 0.}$$

□

Usando la relación anterior, pasamos a construir el código dual de un código cíclico.

Teorema 2.9. *Dado $\mathcal{C}$ un código cíclico y sea $h(x)$ su polinomio de control, entonces*

$$\mathcal{C}^{\perp} = \left\langle \frac{x^k h(x^{-1})}{h(0)} \right\rangle = \left\langle \frac{h_R(x)}{h(0)} \right\rangle.$$

Demostración. Comenzamos por la definición de código dual u ortogonal:

$$\mathcal{C}^{\perp} = \{x \in \mathbb{F}_q^n \mid x \cdot c = 0 \ \forall c \in \mathcal{C}\}.$$

Aplicando el Teorema 2.8, podemos dar una definición equivalente

$$\mathcal{C}^{\perp} = \{a(x) \in \mathcal{R}_n \mid a(x)b_R^{(i)}(x) = 0 \ \forall \ b(x) \in \mathcal{C} \ \wedge \ \forall i \in \{1, \cdots, n\}\}.$$

Cabe destacar que $b_R^{(i)}(x)$ denota a $x^i b_R(x)$, pero he usado notación equivalente a la de espacios vectoriales.

Por otro lado tenemos, por definición de polinomio de control, que $c(x)h(x) = 0\ \forall c(x) \in \mathcal{C}$, entonces basta considerar en la definición anterior $b(x)_R = h(x)$, aplicando las propiedades del recíproco (véase Proposición 2.1)

$$b(x) = h(x)_R \Leftrightarrow b_R^{(i)}(x) = h^{(i)}(x).$$

Por lo que considerando $h_R^{(i)}(x)$, tengo k vectores linealmente independientes contenidos en $\mathcal{C}^\perp$, y además por tener dimensión k, puedo afirmar que $\{h_R(x), xh_R(x), \cdots x^{k-1}h_R(x)\}$, es una base de $\mathcal{C}^T$.

Puedo dividir por $h(0)$ para convertir el polinomio en mónico y así se mantiene la base de $\mathcal{C}^T$ $\{\frac{h_R}{h(0)}(x), x\frac{h_R}{h(0)}(x), \cdots x^{k-1}\frac{h_R}{h(0)}(x)\}$.

Finalmente tenemos el resultado deseado

$$\boxed{\mathcal{C}^\perp = \left\langle \frac{h_R(x)}{h(0)} \right\rangle.}$$

□

Una vez completado esto podemos calcular su matriz de control de paridad, ya que basta con calcular la matriz generadora del código dual (véase Teorema 1.2).

Pasamos a calcular la matriz generadora de $\mathcal{C}^\perp$

Teorema 2.10. *Dado $\mathcal{C}$, entonces la matriz generadora de $\mathcal{C}^\perp$ es*

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

donde $h(x) = \sum_{i=0}^{k} h_i x^i$.

Demostración. Sabemos por un lado que el polinomio generador de C^T es $g(x)^T = x^k h(x^{-1})/h(0)$, por lo que por ser un código cíclico sabemos como generar su matriz.

$$H = \begin{pmatrix} g(x)^T & & & \\ & xg(x)^T & & \\ & & \ddots & \\ & & & x^{k-1}g(x)^T \end{pmatrix} = \begin{pmatrix} x^k h(x^{-1})/h(0) & & & \\ & xx^k h(x^{-1})/h(0) & & \\ & & \ddots & \\ & & & x^{k-1}x^k h(x^{-1})/h(0) \end{pmatrix}.$$

Por lo que haciendo los cálculos en $\mathcal{R}_n$, obtenemos el resultado deseado.

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

□

Por lo que aplicando todo lo anterior podemos calcular la matriz de control de paridad de $\mathcal{C}$.

Teorema 2.11. *Dado $\mathcal{C}$ un código cíclico y sea $h(x)$ su polinomio de control, entonces*

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

es su matriz de control de paridad.

Demostración. Para la demostración basta aplicar los dos teoremas anteriores Teorema 2.10 y Teorema 1.2, y basta tomar la matriz generadora de $\mathcal{C}^\perp$, como la matriz de paridad de $\mathcal{C}$.

□

2.3. Decodificación

Antes de proceder a la descodificación, debemos introducir el concepto de síndrome de un $\mathcal{C}$.

Definición 2.4. Dado $\mathcal{C}$ código lineal sobre $\mathbb{F}_q$ y sea H su matriz de control de paridad, (vease Teorema 4.1) se define como síndrome a la función

$$\begin{aligned} \text{syn} : \mathbb{F}_q^n &\to \mathbb{F}_q^n \\ \text{syn}(x) &= Hx^T. \end{aligned}$$

La función síndrome es muy útil la descodificación, ya que presenta propiedades muy interesantes y además es computable a priori lo que nos permite computarla una vez y usarla siempre para todas las transmisiones de información.

Por la propia definición de matriz de paridad, tenemos que $\text{syn}(x) = 0 \Leftrightarrow x \in \mathcal{C}$.

Proposición 2.2. *Dado $\mathcal{C}$ un código cíclico sobre $\mathbb{F}_q$ y sea syn su función síndrome, entonces*

$$syn(x_1) = syn(x_2) \Leftrightarrow x_1 - x_2 = 0.$$

Demostración. Basta tomar $x_1, x_2 \in \mathcal{C}$ que cumplan la hipótesis de partida $\text{syn}(x_1) = \text{syn}(x_2)$, entonces

$$Hx_1^T = Hx_2^T \Leftrightarrow Hx_1^T - Hx_2^T = 0 \Leftrightarrow H(x_1^T - x_2^T) = 0 \Leftrightarrow H(x_2 - x_1)^T = 0.$$

Por lo que, intercambiando los papeles tenemos que

$$\boxed{x_1 - x_2 \in \mathcal{C}.}$$

□

Después de esta proposición, puedo construir una relación de equivalencia que me facilite encontrar los síndromes de las palabras.

Definición 2.5. Dado $\mathcal{C}$ un código cíclico sobre $\mathbb{F}_q$ y sea syn su función síndrome, se define la siguiente relación de equivalencia

$$x_1 \sim_{\text{syn}} x_2 \Leftrightarrow \text{syn}(x_1) = \text{syn}(x_2).$$

Una vez dada la relación de equivalencia, tomamos como representante de cada clase al vector e_s, tal que $\text{syn}(e_s) = s$ y además el peso de e_s dentro de su clase de equivalencia es mínimo.

La idea del algoritmo de decodificación es computar todos los síndromes y tabular su clase de equivalencia, posteriormente cuando recibamos una palabra y, computamos su síndrome, elegimos la clase de equivalencia a la que pertenece e_s y la decodificamos como $y - e_s$.

El Algoritmo 3 es un tipo de decodificación por vecinos más cercano y es uno de los más sencillos de implementar aunque no de los más eficientes.

Algoritmo 3 Algoritmo de decodificación por síndrome

Input: $c \in \mathbb{F}_q^n$

1: $e_c \leftarrow \text{syn}(c)$
2: $x \leftarrow c - e_c$
3: **return** x

3 Códigos BCH, Reed-Solomon y GRS

En este capítulo, comezaremos hablando de códigos BCH, una clase de códigos cíclicos que presentan propiedades muy importantes, y además facilitan la construccion de códigos con una distancia mínima predefinida, lo que nos da el número de errores que pueden transmitir de antemano.

En la segunda parte, pasaremos a ver los códigos Reed-Solomon, que se pueden considerar como una subclase de los BCH con $n = q - 1$.

Finalmente, saliendo un poco de los códigos cíclicos, daremos algunas pinceladas sobre los códigos Reed-Solomon Generalizados o códigos GRS.

3.1. Codigos BCH

En esta sección nos vamos a centrar en los códigos BCH, un tipo de códigos cíclicos que tienen una distancia mínima predefinida. Fueron introducidos por el matemático Alexis Hocquenghem en 1959 [6] y simultáneamente por los matemáticos Raj Bose y DK Ray-Chaudhuri en 1960 [2].

Su nombre, códigos Bose–Chaudhuri–Hocquenghem y su acrónimo BCH, forma de la cual nos referiremos a ellos a lo largo del trabajo, proviene de sus apellidos (aunque de forma errónea en el caso de Ray-Chaudhuri).

3.1.1. Preliminares: Clases Ciclotomicas y raíces de $\mathbf{g}(X)$

En el estudio de los códigos cíclicos, hay múltiples formas de introducirlos, en Capítulo 2 usamos la idea de las permutaciones cíclicas de todas las palabras del código se quedan en el código y posteriormente encontramos el polinomio generador en cada caso.

Durante todo el capitulo anterior usamos el polinomio generador para dar todas las propiedades de los cíclicos pero, en el caso de los BCH, aunque podemos hacer una construcción similar, es necesario estudiar previamente los polinomios para poder elegir como generador el más idóneo en cada caso.

3.1.1.1. Clases ciclotómicas

Definición 3.1. Dados $\mathbb{F}_q$ un cuerpo finito y $\mathbb{F}_{q^t}$ una extensión suya, se define como *clase $q-$ciclotomica de s módulo q^t-1* al conjunto

$$C_s = \{s, sq, sq^2, \cdots sq^{r-1}\} \ (\text{ mód } q^t-1),$$

con r el menor entero positivo tal que $sq^r \equiv s(\text{ mód } q^t-1)$.

Teorema 3.1. *Las clases $q-$ciclotómicas módulo q^t-1, tienen las siguientes propiedades.*

1. $|C_s| < t$ *para cada s.*
2. *Todas y cada una de ellas son conjuntos disjuntos o iguales.*
3. *Establecen una partición del conjunto* $\{0, 1, 2, \ldots, q^t-2\}$.

Demostración. Para 1) supongamos que existe alguna clase que tiene tamaño mayor o igual que t, por lo que

$$\exists s \in \mathbb{N} \text{ tal que } sq^t \in C_s.$$

Contradicción ya que $sq^t \geq q^t-1$.

Para demostrar 2) supongamos que existen dos clases distintas que tiene un elemento común,entonces $\exists i < j \in \{1, \ldots r-1\}$ tal que $sq^i \equiv \hat{s}q^j(\text{ mód } q^t-1)$.

Multiplicando por q:

$$sq^{i+1} \equiv \hat{s}q^{j+1}(\text{ mód } q^t-1).$$

Por inducción podemos probar que son iguales, todos los elementos, ya que ambas tienen tamaño menor que t.

Para 3) basta con aplicar la propiedad anterior. □

3.1.1.2. Raíces de $g(x)$

Como indiquemos en la introducción, antes de hablar de los códigos BCH, necesitamos estudiar las raíces de los polinomios generadores de códigos cíclicos y además fijar que $\text{mcd}(q, n) = 1$.

Por un lado sabemos por el Teorema 2.4 sabemos que $g(x)|(x^n-1)$, por lo que todas las raíces de $g(x)$ las tenemos localizadas, siendo raíces $n-$simas de la unidad.

Por otro lado, el conjunto U_n de las raíces $n-$ésimas sobre un cuerpo es un grupo cíclico, por lo que podemos afirmar que todas las raíces de $g(X)$, en su cuerpo de descomposición $\mathbb{F}_{q^t}$, están generadas por un único elemento.

De estas dos propiedades podemos deducir que para encontrar la descomposición de $g(x)$, basta con tomar la raíz n-ésima primitiva de la unidad y tomar sus potencias. De esta propiedad radica la definición del conjunto de definición, valga la redundancia, de un código cíclico.

Definición 3.2. Dado $g(X)$ el polinomio generador de un código cíclico $\mathcal{C}$, y sean $\{\alpha^{i_1}, \alpha^{i_2}, \ldots, \alpha^{i_r}\}$ las raíces de $g(X)$ en una extensión adecuada,se define como el conjunto de definición de $\mathcal{C}$ al conjunto $T = \{i \in \{1, \ldots, n\} \ / \ g(\alpha^i) = 0\}$.

Observación 3.1. Dado código cíclico $\mathcal{C} = \langle g(X) \rangle$, su conjunto de definición (vease Def. 3.2) lo determina unívocamente ya que, como vimos antes, todas las raíces de $g(X)$ son potencias de una raíz $n-$esima de la unidad, entonces podemos construir $g(X) = (x - \alpha^{i_1}) \cdots (x - \alpha^{i_r})$.

3.1.2. Cota inferior BCH

Teorema 3.2. *Dado $\mathcal{C}$ un código cíclico generado por el polinomio $g(X)$, si $\alpha^{e_1} \cdots \alpha^{e_{n-k}}$ son las raíces de $g(X)$, en alguna extensión adecuada. Supongamos que $T = (e_1, \cdots, e_{n-k})$ tiene $\delta - 1$ elementos consecutivos, entonces la distancia mínima,d, cumple que $d \geq \delta$.*

Demostración. Supongamos que $g(\alpha^h) = 0 \ \ \forall h \in \{b, b+1, \ldots, b+\delta-2\}$. Tomamos $c(x) \in \mathcal{C}$ una palabra no cero

$$c(x) = \sum_{j=1}^{w} c_{i_j} x^{i_j}.$$

Supongamos que $w < \delta$. Entonces $c(\alpha^h) = 0$, para todo $k \in T$. Tomamos ahora

$$M = \begin{pmatrix} \alpha^{i_1 b} & \alpha^{i_2 b} & \cdots & \alpha^{i_w b} \\ \alpha^{i_1 (b+1)} & \alpha^{i_2 (b+1)} & \cdots & \alpha^{i_w (b+1)} \\ \vdots & \vdots & \vdots & \vdots \\ \alpha^{i_1 (b+w-1)} & \alpha^{i_2 (b+w-1)} & \cdots & \alpha^{i_w (b+w-1)} \end{pmatrix}.$$

Y $u = c_{i_1} c_{i_2} \cdots c_{i_w}$, entonces tenemos que $Mu^T = 0$. Además como $u \neq 0$, tenemos que M es una matriz singular, pero por otro lado la podemos ver como $\det M = \alpha^{(i_1+i_2+\cdots+i_w)b} \det V$, donde V es la matriz de Vandermonde, lo que da la contradicción buscada. Entonces no existen palabras con peso $w < \delta$, por lo que la distancia mínima es $d \geq \delta$.

□

Este teorema nos proporciona una cota para los códigos cíclicos, que a diferencia de las introducidas en Sección 1.4 son para la distancia mínima de un código.

Definición 3.3. Dado $\mathcal{C}$ un código cíclico generado por el polinomio $g(X)$,se denomina distancia de Bose, al mayor número de enteros consecutivos del conjunto $T = (e_1, \cdots, e_{n-k})$ dado en el Teorema 3.2.

A partir de la definición anterior, surgen los códigos BCH que son construidos a partir de una distancia mínima predefinida.

Definición 3.4. Dada d la distancia mínima requerida, se denomina código BCH con distancia definida d, al código $\mathcal{C}$ sobre $\mathbb{F}$ que tiene como conjunto de definición a

$$T = C_b \cup C_{b+1} \cup \cdots \cup C_{b+d-2}.$$

donde C_i son las clases $q-$ciclotómicas módulo n (véase Def. 3.1).

Esta definición nos da un método de construcción de los códigos BCH como veremos en la Subsección 3.1.3, pero ¿Nos asegura, qué el conjunto de definición tiene $d-1$ elementos consecutivos?.

Para responder a esta pregunta basta con darnos cuenta de que $s \in C_s \quad \forall s$, por lo que el conjunto T contiene $d-1$ elementos consecutivos.

La principal ventaja con respecto a otros códigos cíclicos, radica en que su construcción parte de una distancia mínima predefinida, por lo que sabemos de antemano la cantidad de error que podemos corregir (Véase Sección 1.5).

3.1.3. Construcción de los códigos BCH

Para construir un código BCH, necesitamos conocer varios parámetros como son:

1. $n :=$ Longitud.
2. $\mathbb{F}_q :=$ Cuerpo base (Tamaño del abecedario).
3. $d :=$ Distancia mínima del código.

Una vez conocidos todos estos parámetros podemos pasar a construir el código BCH.

Consideramos el conjunto de definición $T = C_b \cup C_{b+1} \cup \cdots \cup C_{b+d-2}$,

y a partir de aquí pasamos a construir el código que tenga como conjunto de definición T, para lo que construimos su polinomio generador como nos indica Observación 3.1.

En la practica, los códigos BCH no se construyen como en Algoritmo 4 , sino que, como generar el polinomio es un trabajo repetitivo y además basta hacerlo una sola vez, encontramos tablas como Tabla 3.1, donde están tabulados los polinomios generadores de los códigos BCH, de parámetros n (longitud del código), d (distancia mínima) y $\mathbb{F}_q$ (cuerpo base).

Algoritmo 4 Construcción de un Código BCH

Input: n Longitud del código.
Input: d distancia mínima requerida.
Input: $\mathbb{F}_q$ cuerpo finito sobre el que generar el código.
1: Calcular $T = C_b \cup C_{b+1} \cup \cdots \cup C_{b+d-2}$
2: $g(x) \leftarrow \prod_{i \in T}(x - \alpha^i)$
3: **return** $g(x)$

Polinomios generadores de un código BCH		
n	d	Polinomio Generador
3	2	$x^2 + x + 1$
5	2	$x^4 + x^3 + x^2 + x + 1$
	3	
	4	
7	2	$x^3 + x + 1$
	3	
	4	$x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$
	5	
	6	

Tabla 3.1: Polinomios generadores de un código BCH sobre $\mathbb{F}_2$

3.2. Codigos Reed-Solomon

Otro tipo de códigos muy importantes son los códigos Reed-Solomon, que son una subclase de los códigos BCH Def. 3.4 con $n = q - 1$.Estos códigos fueron introducidos por Irving S. Reed y Gustave Solomon en 1960.

Como hemos dicho en la introducción podemos considerar a los códigos Reed-Solomon, de aquí en adelante códigos RS, como una subclase de la BCH.

Definición 3.5. Dado un cuerpo finito $\mathbb{F}_q$ y δ la distancia mínima requerida, se denomina RS código, al código de orden $n = q - 1$ y distancia mínima δ que tiene como junto de definición:

$$T = \{b, b + 1, \cdots, b + \delta - 2\}.$$

La condición de que $n = q - 1$, implica que todas las clases $q-$ciclotomicas módulo n, tengan un solo elemento. Lo que proporciona características muy especiales a los códigos RS, con respecto a los BCH. Algunas de ellas se enumeran en el Teorema 3.3.

Teorema 3.3. *Dado un código RS, $\mathcal{C}$ sobre $\mathbb{F}_q$, longitud $n = q - 1$ y distancia predefinida δ. Entonces $\mathcal{C}$ tiene distancia mínima $d = \delta$ y dimensión $k = n - d - 1$.*

Demostración. Para la demostración de 1) utilizamos la cota Singelton (véase Teorema 1.5), fijando la dimensión k y la mínima distancia d, tenemos que

$$k \leq n - d + 1 \leq n - \delta + 1 = k \Rightarrow d = \delta \text{ y } k = n - d + 1.$$

□

3.3. Códigos de evaluación

Por otro lado vamos a introducir los códigos de evaluación, que son los precursores de los códigos de geometría algebraica, que se pueden ver como la evaluación, valga la redundancia, de un conjunto de puntos bajo un conjunto de funciones.

Definición 3.6. Dado $\mathcal{P}$ conjunto de funciones y L un conjunto de puntos, se define como código de evaluación al conjunto

$$\mathcal{C} = \{f(s) | \forall s \in L \ \forall f \in \mathcal{P}\}.$$

Podemos ver una definición alterativa de los códigos RS, que nos servirá de comienzo para la siguiente sección y además los convierte en códigos de evaluación.

Teorema 3.4. *Si α es un elemento primitivo de $\mathbb{F}_q$ y fijado k tal que $0 < k < n = q - 1$ entonces*

$$\mathcal{C} = \{(f(1), f(\alpha^2), \cdots, f(\alpha^{q-2})) \mid f \in \mathcal{P}_k\},$$

es un código RS $[n, k, n - k + 1]$, donde $\mathcal{P}_k$ es el conjunto de polinomio de grado menor o igual que K.

Demostración. Por un lado tenemos que demostrar que $\mathcal{C}$ es un código lineal sobre $\mathbb{F}_q$, basta observar que $\mathcal{P}_k$ es un subespacio lineal sobre $\mathbb{F}_q$ de $\mathbb{F}_q[x]$. Entonces la operación evaluación conserva esta condición de subespacio lineal.

Por otro lado tenemos, nos fijamos en la dimensión k, sabemos por un lado que $\mathcal{P}_k$ tiene dimensión k. Pero, para poderlo trasladar a $\mathcal{C}$, debemos demostrar que dados dos polinomios distintos, obtenemos dos evaluaciones distintas.

Supongamos por reducción a lo absurdo, dados dos polinomios distintos f, f_1 que dan las mismas evaluaciones, $(f - f_1)(\alpha^i) = 0 \ \forall \ i$.

En este caso tenemos que es un polinomio de grado mayor o igual que $k-1$ y n raíces distintas, lo cual es imposible. Por lo que podemos afirmar que $\mathcal{C}$ es k dimensional.

Y por último comprobamos que es un código $RS\ [n,k,n-k+1]$, para ello partimos de $\mathcal{D}$ un códigos $RS\ [n,k,n-k+1]$ y demuestro que $\mathcal{C}=\mathcal{D}$.

Por un lado tenemos que el conjunto de definición de $\mathcal{D}$ es $T=\{1,2,\ldots,n-k\}$. Dado $c(x)\in\mathcal{C}\Rightarrow c(x)=\sum_{j=0}^{n-1}c_jx^j\in\mathcal{C}$, entonces por la definición de C, existe un polinomio $f(x)=\sum_{l=0}^{k-1}f_lx^l\in\mathcal{P}_k$ que cumple $c_i=f(\alpha^i)$.

Por lo que para que $c(x)\in\mathcal{D}$, necesitamos que $c(\alpha^i)=0\ \forall i\in T$.

$$c(\alpha^i)=\sum_{j=0}^{n-1}c_j\alpha^{ij}=\sum_{j=0}^{n-1}\left(\sum_{l=0}^{k-1}f_l\alpha^{jl}\right)\alpha^{ij}=\sum_{l=0}^{k-1}f_l\sum_{j=0}^{n-1}c_j\alpha^{(i+l)j)}=\sum_{m=0}^{k-1}f_m\frac{\alpha^{(i+m)n}-1}{\alpha^{i+m}-1}.$$

Por ser α una raíz n-esima de la unidad $\alpha^{(i+m)n}=1$ y además $\alpha^{i+m}\neq 1\ \forall\ 1\leq i+m\leq n-1=q-2$, y como $k\leq 2$ podemos asegurar que $c(\alpha^i)=0\ \forall i\in T$ y consecuentemente $\mathcal{C}\subseteq\mathcal{D}$, con lo que hemos demostramos que $\boxed{\mathcal{C}=\mathcal{D}}$.

□

3.3.1. Códigos GRS

Una vez introducidos los códigos Reed-Solomon, podemos usar la definición Teorema 3.4 para generalizarlos a códigos lineales, pero no necesariamente cíclicos, se denominan códigos Reed-Solomon Generalizados, o códigos GRS que además son la antesala de los códigos Goppa, que definiremos en el siguiente capítulo (véase Capítulo 4).

Definición 3.7. Dada una n-tupla de elementos distintos en $\mathbb{F}_q$, $\gamma=(\gamma_0,\cdots,\gamma_{n-1})$ y por otro lado $v=(v_0,\cdots,v_{n-1})$, otra n-tupla sobre $\mathbb{F}_q$, de elementos no nula pero no necesariamente distintos y $\mathcal{P}_k$ como el conjunto de polinomios de grado menor o igual que k. Se define como código Reed-Solomon generalizados al conjunto:

$$GRS_K(\gamma,v)=\{(v_0f(\gamma_0),\cdots,v_{n-1}f(\gamma_{n-1}))|f\in\mathcal{P}_k\}.$$

Teorema 3.5. *Con la notación anterior tenemos que $GRS_k(\gamma,v)$ es un código $[n,k,n-k+1]$.*

Demostración. Por un lado tenemos que ver que es un código k dimensional, para lo que repitiendo la demostración de Teorema 3.4, lo obtenemos

$$\boxed{GRS_k(\gamma,v)\text{ es un código }k-\text{dimensional }.}$$

Ya que un polinomio no nulo $f \in \mathcal{P}_k$ tiene como mucho $k-1$ ceros, entonces la distancia mínima es al menos $n-k+1$.

Por otro lado aplicando la cota Singelton tenemos que la distancia mínima es como mucho $n-k+1$, por lo que

$$\boxed{\text{La distancia mínima de } GRS_k(\gamma, v) \text{ es } n-k+1\ .}$$

Así que

$$\boxed{GRS_k(\gamma, v) \text{ es un código } [n, k, n-k+1].}$$

□

4 Códigos Goppa

Los códigos Goppa clasicos fueron introducidos por V.D. Goppa en 1970, podemos encontrar una traducción en [4].

En la actualidad son una de las familias más usadas en criptografía postcuántica ya que poseen propiedades fundamentales, que aunque no son objetivo del trabajo, los hacen muy parecidos a los códigos lineales aleatorios pero a diferencia de estos sí que se conocen algoritmo de codificación y descodificación eficientes.

Es por estas propiedades, que MCEliece eligió estos códigos para en 1978 desarrollar su criptosistema.

4.1. Introducción

En este capítulo vamos a trabajar un poco en medio de los códigos BCH y los códigos GRS, ya que los códigos Goppa son una generalización de los BCH y a su vez una subclase de los GRS, para ser más precisos son códigos subcuerpo de los GRS.

En este trabajo no vamos a ampliar la teoría de códigos subcuerpo, aunque dando unas pequeñas pinceladas podemos ver los códigos Goppa como la restricción de los GRS a un subcuerpo más pequeño (véase [12]).

Los códigos Goppa son además un tipo de códigos que se denominan alternantes y estos a su vez son un código subcuerpo de los GRS.

Definición 4.1. Dados $L = \{\gamma_0, \cdots, \gamma_{n-1}\}$ n elementos distintos de $\mathbb{F}_{q^t}$ y un polinomio $G(X) \in \mathbb{F}_{q^t}[X]$, se define el códigos Goppa como el conjunto

$$\Gamma(G, L) = \mathcal{C} = \left\{ (c_0, \cdots, c_{n-1}) \in \mathbb{F}_q^n \;\middle|\; \sum_{i=0}^{n-1} \frac{c_i}{x - \gamma_i} \equiv 0 \text{ mód } G(x) \right\}.$$

Una vez definidos los códigos Goppa, podemos comenzar con su estudio calculando, por ejemplo, con la matriz de control de paridad para comprobar fácilmente si una palabra pertenece al código o no.

Teorema 4.1. *Dado $\mathcal{C}$ un código Goppa $\Gamma(G, L)$, donde $L = (\gamma_0, \cdots, \gamma_{n-1})$ y además $G(x) =$*

$\sum_{j=1}^{d} g_j x^j$ *entonces:*

$$H = \begin{pmatrix} G(\gamma_0)^{-1} g_w & G(\gamma_1)^{-1} g_w & \cdots & G(\gamma_{n-1})^{-1} g_w \\ G(\gamma_0)^{-1}(g_{d-1} + g_w \gamma_0) & G(\gamma_1)^{-1}(g_{d-1} + g_w \gamma_1) & \cdots & G(\gamma_{n-1})^{-1}(g_{d-1} + g_w \gamma_{n-1}) \\ \vdots & \vdots & \vdots & \vdots \\ G(\gamma_0)^{-1}(\sum_{j=1}^{w} g_j \gamma_0^{j-1}) & G(\gamma_1)^{-1}(\sum_{j=1}^{w} g_j \gamma_1^{j-1}) & \cdots & G(\gamma_{n-1})^{-1}(\sum_{j=1}^{w} g_j \gamma_{n-1}^{j-1}) \end{pmatrix},$$

es la matriz de control de paridad de $\mathcal{C}$.

Lema 4.1. *Se cumple la siguiente igualdad:*

$$\frac{1}{x - \gamma_i} \equiv -\frac{1}{G(\gamma_i)} \frac{G(x) - (\gamma_i)}{x - \gamma_i} \text{ mód } G(x).$$

Demostración. Para demostrarlo basta observar que ocurre con $-G(\gamma_i)(G(x) - \gamma_i)$ mód $G(x)$

$$-G(\gamma_i)(G(x) - \gamma_i) \text{ mód } G(x) \equiv -G(\gamma_i)G(x) - 1 \text{ mód } G(x) \equiv 1 \text{ mód } G(x).$$

Por lo tanto

$$\boxed{\frac{1}{x - \gamma_i} \equiv} - G(\gamma_i)(G(x) - \gamma_i)\frac{1}{x - \gamma_i} \equiv \boxed{-\frac{1}{G(\gamma_i)} \frac{G(x) - (\gamma_i)}{x - \gamma_i} \text{ mód } G(x).}$$

□

Demostración. Demostración de Teorema 4.1 Partimos del Lema 4.1,

$$\frac{1}{x - \gamma_i} \equiv -\frac{1}{G(\gamma_i)} \frac{G(x) - (\gamma_i)}{x - \gamma_i} \text{ mód } G(x).$$

y usamos la definición de código Goppa (véase Def. 4.1)

$$c \in \mathcal{C} \Leftrightarrow \sum_{i=0}^{n-1} \frac{c_i}{x - \gamma_i} \equiv 0 \text{ mód } G(x) \Leftrightarrow \sum_{i=0}^{n-1} c_i \frac{G(x) - G(\gamma_i)}{x - \gamma_i} G(\gamma_i)^{-1} \equiv 0 \text{ mód } G(x).$$

Consideramos $G(x) = \sum_{j=0}^{w} g_j x^j$, con $w = \deg(G(x))$, y estudiamos elemento a elemento la suma anterior

$$c_i \frac{G(x)-G(\gamma_i)}{x-\gamma_i} G(\gamma_i)^{-1} = G(\gamma_i)^{-1} \frac{\sum_{j=0}^{k=0} g_j x^j - \sum_{k=0}^{w} g_k \gamma_i^k}{x-\gamma_i} = G(\gamma_i)^{-1} \sum_{j=0}^{w} g_j \sum_{k=0}^{j-1} x^k \gamma^{j-1-k} =$$
$$= G(\gamma_i)^{-1} \sum_{j=0}^{w-1} x^k \left(\sum_{j=k+1}^{w} g_j \gamma^{j-1-k} \right).$$

Por lo que poniéndolo en forma de matriz, podemos darnos cuenta de $c \in \mathcal{C}$ sí y solo si $Hc^T = 0$, donde

$$H = \begin{pmatrix} G(\gamma_0)^{-1} g_w & G(\gamma_1)^{-1} g_w & \cdots & G(\gamma_{n-1})^{-1} g_w \\ G(\gamma_0)^{-1}(g_{d-1} + g_w \gamma_0) & G(\gamma_1)^{-1}(g_{d-1} + g_w \gamma_1) & \cdots & G(\gamma_{n-1})^{-1}(g_{d-1} + g_w \gamma_{n-1}) \\ \vdots & \vdots & \vdots & \vdots \\ G(\gamma_0)^{-1}(\sum_{j=1}^{w} g_j \gamma_0^{j-1}) & G(\gamma_1)^{-1}(\sum_{j=1}^{w} g_j \gamma_1^{j-1}) & \cdots & G(\gamma_{n-1})^{-1}(\sum_{j=1}^{w} g_j \gamma_{n-1}^{j-1}) \end{pmatrix}.$$

Afirmamos que

$$\boxed{H \text{ es la matriz de paridad de } \mathcal{C}.}$$

□

Proposición 4.1. *La matriz de paridad de $\mathcal{C}$ dada en Teorema 4.1 se puede reducir a:*

$$H = \begin{pmatrix} G(\gamma_0)^{-1} & G(\gamma_1)^{-1} & \cdots & G(\gamma_{n-1})^{-1} \\ G(\gamma_0)^{-1}\gamma_0 & G(\gamma_1)^{-1}\gamma_1 & \cdots & G(\gamma_{n-1})^{-1}\gamma_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ G(\gamma_0)^{-1}\gamma_0^{d-1} & G(\gamma_1)^{-1}\gamma_1^{d-1} & \cdots & G(\gamma_{n-1})^{-1}\gamma_{n-1}^{d-1} \end{pmatrix}.$$

Demostración. Partimos de

$$H = \begin{pmatrix} G(\gamma_0)^{-1} g_w & G(\gamma_1)^{-1} g_w & \cdots & G(\gamma_{n-1})^{-1} g_w \\ G(\gamma_0)^{-1}(g_{d-1} + g_w \gamma_0) & G(\gamma_1)^{-1}(g_{d-1} + g_w \gamma_1) & \cdots & G(\gamma_{n-1})^{-1}(g_{d-1} + g_w \gamma_{n-1}) \\ \vdots & \vdots & \vdots & \vdots \\ G(\gamma_0)^{-1}(\sum_{j=1}^{w} g_j \gamma_0^{j-1}) & G(\gamma_1)^{-1}(\sum_{j=1}^{w} g_j \gamma_1^{j-1}) & \cdots & G(\gamma_{n-1})^{-1}(\sum_{j=1}^{w} g_j \gamma_{n-1}^{j-1}) \end{pmatrix}.$$

Y podemos descomponerla como

$$H = \underbrace{\begin{pmatrix} g_w & 0 & 0 & \cdots & 0 \\ g_{w-1} & g_w & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ g_1 & g_2 & g_3 & \cdots & g_w \end{pmatrix}}_{T} \underbrace{\begin{pmatrix} G(\gamma_0)^{-1} & G(\gamma_1)^{-1} & \cdots & G(\gamma_{n-1})^{-1} \\ G(\gamma_0)^{-1}\gamma_0 & G(\gamma_1)^{-1}\gamma_1 & \cdots & G(\gamma_{n-1})^{-1}\gamma_{n-1} \\ \vdots & \vdots & \vdots & \vdots \\ G(\gamma_0)^{-1}\gamma_0^{d-1} & G(\gamma_1)^{-1}\gamma_1^{d-1} & \cdots & G(\gamma_{n-1})^{-1}\gamma_{n-1}^{d-1} \end{pmatrix}}_{H'},$$

finalmente tenemos que ver que H' también es matriz de paridad de $\mathcal{C}$

$$Hc^T = 0 \Leftrightarrow TH'c^T = 0 \Leftrightarrow H'c^T = T^{-1} \cdot 0 \Leftrightarrow H'c^T = 0.$$

En todo este proceso he supuesto que T es invertible, para asegurarlo basta observar su determinante que es $g_w^w \neq 0$.

□

Teorema 4.2. *Sea un código Goppa $\mathcal{C} = \Gamma(L, G)$ con $\deg(G(x)) = w$ y $t = ord_q(n)$, entonces $\mathcal{C}$ es un código $[n, k, d]$ donde $k \geq n - wt$ y además $d \geq w + 1$.*

Demostración. Para esta demostración, partimos de la matriz H' del teorema anterior y consideramos cada elemento de $\mathbb{F}_{q^t}$, como coordenadas sobre $\mathbb{F}_q$, es decir, como un vector columna de tamaño $t \times 1$. Entonces obtenemos $tw \times n$ matriz sobre $\mathbb{F}_q$ y además sigue siendo una matriz de control de paridad.

Por lo que ahora estudiando H'', tenemos que sus columnas son linealmente dependientes ya que el rango como mucho wt.

Entonces la dimensión de $\mathcal{C}$ es mayor que $n - wt$.

Supongamos por reducción a lo absurdo que alguna palabra c de $\mathcal{C}$, tiene peso w o menos, entonces la primera parte de la ecuación, que define los códigos Goppa, tenemos que es una función racional cuyo numerador tiene grado $w - 1$ o menos, pero este numerador tiene que ser múltiplo de $G(x)$, cosa imposible ya que $\deg(G(x)) = w$.

Por lo que todas las palabras tienen peso mayor que w, así que

$$\boxed{k \geq n - wt.}$$

Por otro lado la cota Singuelton nos asegura que $k < n - d + 1$, entonces

$$k \leq n - d + 1 \Leftrightarrow n - wt \leq n - d + 1 \Leftrightarrow d - 1 \leq w \Leftrightarrow d \geq w + 1.$$

Por lo que puedo afirmar que

$$\boxed{k \geq n - wt \text{ y además } d \geq w + 1.}$$

□

4.2. Algoritmo de Sugiyama

Una vez introducidos los códigos Goppa, y sabiendo como codificar palabras, es importante obtener un algoritmo eficiente para decodificar las palabras recibidas.

Uno de los algoritmos más eficientes, es el algoritmo de Sugiyama, introducido por Yasao Sugiyama, Masao Kasahara, Shigeichi Hirasawa y Toshihiko Namekawa en [11] donde resuelven las ecuaciones clave para la decodificación de los códigos Goppa con la ayuda del algoritmo extendido de Euclides.

Definición 4.2. Dado $\mathcal{C} = \Gamma(L, g)$, un código Goppa, y sea r la palabra recibida entonces se denomina polinomio síndrome a

$$S_r(x) = -\sum_{i=0}^{n} r_i \frac{g(x) - g(\alpha_i)}{x - \alpha_i} g^{-1}(\alpha_i).$$

Definición 4.3. Dado $\mathcal{C} = \Gamma(L, g)$, un código Goppa, y sea e el vector de errores que ha recibido la palabra recibida r, entonces se denomina polinomio localizador de errores a

$$\sigma(x) = \prod_{i \in E} (x - \alpha_i),$$

donde E es el conjunto de índices donde e es distinto de 0.

Definición 4.4. Dado $\mathcal{C} = \Gamma(L, g)$, un código Goppa, y sea e el vector de errores que ha recibido la palabra recibida r, entonces se denomina polinomio evaluador de errores a

$$\eta(x) = \sum_{i \in E} e_i \prod_{\substack{i' \in E \\ i' \neq i}} (x - \alpha_{i'}),$$

donde E es el conjunto de índices donde e es distinto de 0.

Teorema 4.3. *Dado $\mathcal{C} = \Gamma(L, g)$ y sea $S(x), \sigma(x), \eta(x)$ sus polinomios síndrome, localizador y evaluador respectivamente, entonces se cumple*

$$\sigma(x) \cdot S(x) \equiv \eta(x) \text{ mód } g(x).$$

Demostración. Para esta demostración basta con aplicar la definición de cada uno de los polinomios y tomar módulo $g(x)$

$$S(x) = -\sum_{i=0}^{n} r_i \frac{g(x) - g(\alpha_i)}{x - \alpha_i} g^{-1}(\alpha_i) \Rightarrow S(x) \equiv \sum_{i=0}^{n} \frac{r_i g(\alpha_i) g^{-1}(\alpha_i)}{x - \alpha_i} \text{ mód } g(x),$$

$$S(x) \equiv \sum_{i=0}^{n} \frac{r_i}{x - \alpha_i} \text{ mód } g(x).$$

Y al otro lado de la igualdad

$$\frac{\eta(x)}{\sigma(x)} = \frac{\sum_{i \in E} e_i \prod_{\substack{i' \in E \\ i' \neq i}} (x - \alpha_{i'})}{\prod_{i \in E}(x - \alpha_i)} \Rightarrow \frac{\eta(x)}{\sigma(x)} \equiv \frac{\sum_{i \in E} e_i \prod_{\substack{i' \in E \\ i' \neq i}} (x - \alpha_{i'})}{\prod_{i \in E}(x - \alpha_i)} \text{ mód } g(x).$$

Finalmente desarrollando el sumatorio y tomando factor común en todas la fracciones de la expresión de $S(x)$ mód $g(x)$, obtengo el resultado deseado.

$$\boxed{S(x) \equiv \frac{\eta(x)}{\sigma(x)} \text{ mód } g(x).}$$

□

4.2.1. Algoritmo extendido de Euclides

El algoritmo extendido de Euclides, es ampliamente conocido, pero en esta sección recordaremos algunos de los resultados más importantes que usaremos en el algoritmo de Sugiyama.

Por un lado recordamos las ecuaciones del algoritmo

$$r_{-1} = r_0 q_1 + r_1,$$
$$r_0 = r_1 q_2 + r_2,$$
$$r_1 = r_2 q_2 + r_3,$$

$$\vdots$$

$$r_{n-2} = r_{n-1}q_n + r_n.$$

Dicha relación se puede formular en notación matricial

$$\begin{pmatrix} r_{i-2}(x) \\ r_{i-1}(x) \end{pmatrix} = \begin{pmatrix} q_i(x) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} r_{i-1}(x) \\ r_i(x) \end{pmatrix}.$$

También podemos construir los coeficientes de Bezout, comenzando por $U_0(x) = 1, U_{-1}(x) = 0$, $V_0(x) = 0$ y $V_{-1}(x) = 1$.

$$\begin{pmatrix} U_i(x) & U_{i-1}(x) \\ V_i(x) & V_{i-1}(x) \end{pmatrix} = \begin{pmatrix} q_1(x) & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} q_2(x) & 1 \\ 1 & 0 \end{pmatrix} \cdots \begin{pmatrix} q_i(x) & 1 \\ 1 & 0 \end{pmatrix}.$$

Juntando ambas igualdades tengo que

$$\begin{pmatrix} r_{-1}(x) \\ r_0(x) \end{pmatrix} = \begin{pmatrix} U_i(x) & U_{i-1}(x) \\ V_i(x) & V_{i-1}(x) \end{pmatrix} \begin{pmatrix} r_{i-1}(x) \\ r_i \end{pmatrix}.$$

También tenemos por otro lado la igualdad $U_i(x)V_{i-1}(x) - V_i(x)U_{i-1}(x) = (-1)^{i+1} \neq 0$ por lo que puedo multiplicar por la inversa y obtener

$$\begin{pmatrix} r_{i-1}(x) \\ r_i \end{pmatrix} = (-1)^i \begin{pmatrix} V_{i-1}(x) & -U_{i-1}(x) \\ -V_i(x) & U_i(x) \end{pmatrix} \begin{pmatrix} r_{-1}(x) \\ r_0(x) \end{pmatrix}.$$

Así que el polinomio resto de la iteración i-ésima se puede obtener a partir de $r_{-1}(x)$ y $r_0(x)$, de la siguiente forma

$$r_i(x) = (-1)^i(-V_i(x)r_{-1}(x) + U_i(x)r_0(x)),$$

de donde tomando módulo $r_{-1}(x)$ obtenemos:

$$\boxed{r_i(x) = (-1)^i U_i(x)r_0(x) \text{ mód } r_{-1}(x).}$$

4.2.2. Resolución de las ecuaciones Clave

Comenzamos por darnos cuenta de que los polinomios $\sigma(x)$ y $\eta(x)$ son primos relativos.

Lema 4.2. *Dado $\mathcal{C} = \Gamma(L, g)$ un código Goppa y sea $\sigma(x)$ y $\eta(x)$ los polinomios evaluador y localizador del Teorema 4.3, entonces*

$$mcd(\sigma(x), \eta(x)) = 1.$$

Demostración. Para demostrarlo basta comprobar que no tiene la mismas raíces. Comenzamos por las raíces de $\sigma(x)$:

$$\sigma(x) = 0 \Leftrightarrow x = \alpha_i \; i \in E.$$

Por otro lado sustituimos en $\eta(x)$

$$\eta(\alpha_i) = \sum_{j \in E} e_j \prod_{\substack{j' \in E \\ j' \neq j}} (\alpha_i - \alpha_{j'}) = e_i \prod_{\substack{j' \in E \\ j' \neq i}} (\alpha_i - \alpha_{j'}) \neq 0.$$

Por lo que podemos afirma que

$$\text{mcd}(\sigma(x), \eta(x)) = 1.$$

□

En esta sección vamos a intentar resolver las ecuaciones del Teorema 4.3, es decir, dado $\mathcal{C} = \Gamma(L, g)$, con $\deg g(x) = 2t$ y un polinomio síndrome $S(x)$ de grado menor que $2t$, encontrar $\sigma_S(x)$ y $\eta_S(x)$, primos relativos que además cumplan

$$\sigma_S(x) S(x) \equiv \eta_S(x) \text{ mód } g(x).$$

Por un lado, los polinomios localizador y evaluador, por Teorema 4.3 cumplen la ecuación y además son primos relativos (véase Lema 4.2)

$$\eta_S(x) = \eta(x),$$
$$\sigma_S(x) = \sigma(x).$$

Entonces, solo resta encontrar un método eficiente para encontrar soluciones y además probar que es única, por lo que coincidiría con los polinomios evaluador y localizador.

Lema 4.3. *Dado $\mathcal{C} = \Gamma(L, g)$ y sean $S(x)$ y $\sigma(x)$ sus polinomios síndrome y localizador respectivamente, entonces*

$$\deg S(x) \geq 2 \cdot t - \deg \sigma(x) \geq t.$$

Demostración. Supongamos por reducción a lo absurdo que $\deg S(x) < 2t - \deg \sigma(x)$. Entonces por el lema anterior $\deg \eta(x) = \deg S(x) + \deg \sigma(x)$, pero a partir de la definición de $\sigma(x)$ y $\eta(x)$, tengo que $\deg \eta(x) < \deg \sigma$, lo que es imposible.

Por lo que podemos afirmar que

$$\boxed{\deg S(x) \geq 2 \cdot t - \deg \sigma(x) \geq t.}$$

□

Lema 4.4. *Sea $p(x)$ el máximo común divisor de $g(x)$ y $S(x)$. Entonces $p(x)$ divide a $\eta(x)$ y además* $\deg p(x) \leq \deg \eta(x) \leq t - 1$.

Demostración. Partimos de la siguiente igualdad

$$\sigma(x)S(x) = \eta(x) \text{ mód } g(x),$$

si $p(x)$ divide a $\sigma(x)$ y $S(x)$, por ser su máximo común divisor, podemos afirmar que divide a $\eta(x)$.

Entonces consecuentemente $\deg p(x) \leq \deg \eta(x) \leq t - 1$.

□

Una vez vistos estos lemas tecnicos podemos demostrar el teorema que une la descodificación de Códigos Goppa y el Algoritmo de Euclides, dando lugar al conocido como algoritmo de Sugiyama.

Algoritmo 5 Algoritmo de Sugiyama

Input: Código Goppa $\mathcal{C} = \Gamma(L, g)$ y $S(x)$

1: $i \leftarrow 1$
2: $r_{-1}(x) \leftarrow g(x)$
3: $r_0(x) \leftarrow S(x)$
4: $U_0(x) \leftarrow 1$
5: $U_{-1}(x) \leftarrow 0$
6: **while** $\deg r_{i-1} \geq t \ \wedge \ \deg r_i \leq t - 1$ **do**
7: $\quad r_{i+1} \leftarrow r_{i-1}$ mód r_i
8: $\quad q_i \leftarrow (r_{i-1} - r_{i+1})/r_i$
9: $\quad U_{i+1} \leftarrow U_{i-1} - q_i U_i$
10: $\quad i \leftarrow i + 1$
11: **end while**
12: **return** $i \ \wedge \ r_i \ \wedge \ U_i$

Antes de demostrar que el algoritmo devuelve la única solución de Teorema 4.3, recordamos un lema ya conocido sobre el algoritmo extendido de Euclides.

Lema 4.5. *Usando la notación del algoritmo extendido de Euclides, se cumple*

$$U_i(x)V_{i-1}(x) - U_{i-1}(x)V_i(x) = (-1)^{i+1}.$$

Teorema 4.4. *Sea $\mathcal{C} = \Gamma(L, g)$ un código Goppa, y sean $r_i(x), U_i(x)$ las salidas del Algoritmo de Sugiyama (Vease Algoritmo 5), entonces*

$$\eta(x) = (-1)^i \lambda r_i(x),$$

$$\sigma(x) = \lambda U_i(x),$$

donde λ es una constante, no cero, que hace mónico a $\sigma(x)$.

Demostración. En primer lugar pasamos a comprobar que el valor i obtenido en el Algoritmo 5, se alcanza y además el valor de i es único.

Aplicando Lema 4.3:

$$\deg S(x) \geq t \Rightarrow \deg r_0(x) \geq t.$$

Entonces existe una iteración $i-$esima que satisface que

$$\deg p(x) \leq \deg r_i \leq t - 1,$$

Y además la sucesión de grados de los restos es monótona decreciente, lo que nos afirma que el número de iteraciones k es único, por lo que el algoritmo está bien definido y además termina en un tiempo finito.

Partimos ahora del algoritmo extendido de Euclides, tenemos que en cada iteración se cumple

$$U_i(x)g(x) + V_i(x)S(x) = r_i. \tag{4.1}$$

Por otro lado tenemos la ecuación clave

$$\sigma(x)S(x) \equiv \eta(x) \text{mód } g(x).$$

Desarrollando el módulo tengo que

$$a(x)g(x) + \sigma(x)S(x) = \eta(x). \tag{4.2}$$

Multiplicando la ecuación (4.2) por V_i y la ecuación (4.1) por $\sigma(x)$ obtengo

$$U_i(x)g(x)\sigma(x) + V_i(x)S(x)\sigma(x) = r_i\sigma(x).$$

$$a(x)g(x)V_i(x) + \sigma(x)S(x)V_i(x) = \eta(x)V_i(x).$$

Restando y tomando módulo $g(x)$ obtengo que

$$r_i(x)\sigma(x) \equiv \eta(x)V_i(x) \text{ mód } g(x).$$

Como $\deg(\sigma(x)) \leq t$ y además por la elección de i $\deg(r_i(x)\sigma_i(x)) = \deg(r_i(x) + \deg\sigma(x) < t + t = 2t)$.

Usando Lema 4.3, tengo que el $\deg(\eta(x)V_i(x)) = \deg(\eta(x)) + \deg(V_i(x)) \leq 3t - t = 2t$. Tengo que $r_i(x)\sigma(x) = \eta(x)V_i(x)$.

Y finalmente, sustituyendo en las ecuaciones anteriores, puedo afirmar que

$$U_i(x)\sigma(x) = a(x)V_i(x) \Rightarrow \boxed{\sigma(x) = \lambda V_i(x)}.$$

Sustituyendo esta expresión de nuevo en las ecuaciones anteriores, obtengo que

$$\boxed{\eta(x) = (-1)^i \lambda r_i(x).}$$

Finalmente para resolver las ecuaciones clave basta ajustar λ, para hacer $\sigma(x)$ mónico. □

Por ultimo basta comprobar que la solución de la ecuación clave es única, entonces coincidiría con la obtenida en elAlgoritmo 5.

Teorema 4.5. *Sea $\Gamma(L, g)$ un código Goppa, supongamos que $\deg(g) = 2t$, entonces las soluciones de la ecuación clave (Vease Teorema 4.3),$\sigma(x)$ y $\eta(x)$, primas relativas y además $\deg(\eta(x)) < t$ y $\deg(\sigma(x)) \leq t$ son únicas, salvo constante multiplicativa.*

Demostración. Supongamos que existen dos soluciones $\eta_1(x)$,$\eta_2(x)$ y $\sigma_1(x)$,$\sigma_2(x)$ y ambas satisfacen las ecuaciones clave (véase Teorema 4.3).

Entonces

$$\sigma_i(x)S(x) \equiv \eta_i(x) \text{ mód } g(x) \quad \forall i \in 1, 2.$$

Además, para cada $i = 1, 2$, $\sigma_i(x)$ y $\eta_i(x)$ son primas relativas, por lo que $\sigma_i(x)$ y $g(x)$ también lo son.

Entonces existe $\lambda_i(x)$ que cumpla

$$\sigma_i(x)\lambda_i(x) \equiv \text{ mód } g(x).$$

Por lo que

$$S(x) \equiv \eta_i(x)\lambda_i(x) \text{ mód } g(x)\ \forall i \Rightarrow \eta_1(x)\lambda_1(x) \equiv \eta_2(x)\lambda_2(x) \text{ mód } g(x).$$

Multiplicando por $\sigma_1(x)\sigma_2(x)$ obtenemos,

$$\sigma_1(x)\eta_2(x) \equiv \sigma_2(x)\eta_1(x) \text{ mód } g(x).$$

Como $\deg(\sigma_i) \leq t$ y $\deg(\eta_i(x)) < t$ y $\deg(g(x)) = 2t$,entonces

$$\sigma_1(x)\eta_2(x) = \sigma_2(x)\eta_1(x).$$

Con lo que $\sigma_1(x)|\sigma_2(x)$, pero como $\sigma_1(x)$ y $\eta_1(x)$ son primos relativos, tengo que $\sigma_2(x)|\sigma_1(x)$.

Así que podemos afirmar que

$$\boxed{\sigma_1(x) = \sigma_2(x) \Leftrightarrow \eta_1(x) = \eta_2(x)}.$$

Observación 4.1. Las igualdades anteriores son ciertas salvo constante multiplicativa.

□

4.3. Cota de Gilbert–Varshamov

En la sección Sección 1.4 estuvimos estudiando cotas para el tamaño de un código, como es la cota Singelton (Véase Teorema 1.5). En este capítulo vamos a estudiar algunas cotas asintóticas, es decir vamos a estudiar la tendencia del tamaño del código cuando el orden y la distancia tienden a infinito.

Definición 4.5. Para una familia de códigos sobre $\mathbb{F}_q$, se define como tamaño asintótico con distancia relativa aproximándose a δ a la función

$$\alpha_q(\delta) = \lim_{n\to+\infty} \sup n^{-1} \log_q A_q(n, \delta n).$$

Antes de llegar a las cotas asintóticas comienzo por definir la cota de Gilbert para códigos lineales.

Teorema 4.6. *Sea $\mathcal{C}$ un $[n,k,d]$ código lineal, entonces*

$$B_q(n,d) \geq \frac{q^n}{\sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i}.$$

Demostración. Como $\mathcal{C}$ es lineal sobre $\mathbb{F}_q$ con $B_q(n,d)$ palabras, las esferas de radio $d-1$ sobre elementos de $\mathcal{C}$ cubren todo $\mathbb{F}_q$. Por Teorema 1.6 tenemos que una esfera de radio $d-1$ contienen $\sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i$ palabras del código. Entonces $B_q(n,d)$ esferas llenan el espacio con lo que tenemos que

$$B_q(n,d) \sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i = q^n \Leftrightarrow \boxed{B_q(n,d) \geq \frac{q^n}{\sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i}}.$$

□

Una vez definido el tamaño asintótico y la cota de Gilbert (no asintótica), nos centraremos en la cota que da nombre a esta sección, la cota de Gilbert–Varshamov.

Comenzamos por definir la función entropía de Gilbert.

Definición 4.6. (**Función entropía de Gilbert**) Dado q se define como función entropía de Gilbert a la siguiente función

$$H_q(x) = \begin{cases} 0, & \text{si } x = 0, \\ x\log_q(q-1) - x\log_q(x) - (1-x)\log_q(1-x), & \text{si } 0 < x \leq r. \end{cases}$$

Definición 4.7. Dados q y δ,se denomina **Cota Asintótica de Gilbert–Varshamov** a

$$\boxed{1 - H_q(\delta).}$$

Definición 4.8. Dado $\mathbb{F}_q$, entonces

$$V_q(n,a) = \sum_{i=0}^{a} \binom{n}{i} (q-1)^i,$$

esta definición coincide con la cantidad de elementos de una esfera de radio a sobre $\mathbb{F}_q^n$.

La siguiente cota proviene tanto de la cota de Gilbert (no asintótica), como de la de Varsamov por ello presenta el nombre de ambos, aunque nosotros solo la demostraremos desde la cota de Gilbert ya que ambas demostraciones son equivalentes.

Teorema 4.7. *Dado un código $\mathcal{C}$ y dados $0 < \delta \leq 1 - q^{-1}$ con $q > 2$, se cumple*

$$\alpha_q(\delta) \geq 1 - H_q(\delta).$$

Demostración. De la cota de Gilbert no asintótica, tenemos que $A_q(n, \delta n) \geq \frac{q^n}{V_q(n, \lfloor \delta n \rfloor - 1)}$. Entonces,

$$\alpha_q(\delta) = \limsup_{n \to +\infty} n^{-1} \log_q A_q(n, \delta n) \geq \limsup_{n \to +\infty} n^{-1} 1 - n^{-1} \log_q(V_q(n, \lfloor \delta n \rfloor)) = 1 - H_q(\delta).$$

□

4.3.1. Irreducibles en $\mathbb{F}_q[x]$

Para generar un código Goppa es necesario encontrar irreducibles en $\mathbb{F}_q[x]$, esta tarea no siempre es fácil, por lo que en esta sección vamos a intentar contar cuántos irreducibles de grado m existen en $\mathbb{F}_q[x]$.

Definición 4.9. Dado $\mathbb{F}_q[x]$, entonces se define I_k al conjunto de todos los polinomios irreducibles mónicos de grado k.

Durante toda la sección vamos a dar resultados que nos permitan contar cuantos elementos tiene I_k, pero para ello nesecitamos algunos preliminares, como definir la fórmula de inversión de Moebius, dada en el siguiente teorema.

Teorema 4.8. *Dado $\mathbb{F}_q[x]$, y*

$$f(n) = \sum_{\substack{d \\ d|n}} g(d).$$

Entonces se define

$$g(n) = \sum_{\substack{d \\ d|n}} \mu(d) f\left(\frac{n}{d}\right),$$

donde

$$\mu(d) = \begin{cases} 1 & \text{si } d = 1\,, \\ (-1)^k & \text{si } d \text{ es producto de } k \text{ primos distintos} \\ ,0 & \text{si } d \text{ contiene factores primos repetidos} \end{cases}.$$

Demostración. Supongamos que

$$n = \prod_{i=1}^{j} p_i^{e_i},$$

donde p_i son primos, entonces d divide a n si y solo si

$$d = \prod_{i=1}^{j} p_i^{k_i}$$

donde $0 \leq k_i \leq e_i$. Entonces la fórmula pasa a ser

$$f\left(\prod_{i=1}^{j} p_i^{e_i}\right) = \sum_{k_1=0}^{e_1} \sum_{k_1=0}^{e_2} \cdots \sum_{k_1=0}^{e_j} g\left(\prod_{i=1}^{j} p_i^{k_i}\right),$$

.

como la fórmula es válida para todos los valores de n, podemos usar

$$n = p_1^{e^1-1} \prod_{i=2}^{j} p_i^{e_i}.$$

En este caso obtenemos que

$$f\left(p_1^{e^1-1} \prod_{i=2}^{j} p_i^{e_i}\right) = \sum_{k_1=0}^{e_1-1} \sum_{k_1=0}^{e_2} \cdots \sum_{k_1=0}^{e_j} g\left(\prod_{i=1}^{j} p_i^{k_i}\right),$$

despejando obtenemos que

$$\sum_{k_1=e_1-1}^{e_1} (-1)^{e_1-k_1} f\left(p_1^{e^1-1} \prod_{i=2}^{j} p_i^{e_i}\right) = \sum_{k_1=0}^{e_2} \cdots \sum_{k_1=0}^{e_j} g\left(\prod_{i=1}^{j} p_i^{k_i}\right).$$

Continuamos haciendo el mismo proceso, remplazando las sumas de g por las de f.

$$\sum_{k_1=e_1-1}^{e_1} \sum_{k_2=e_2-1}^{e_2} \cdots \sum_{k_j=e_j-1}^{e_j} (-1)^{\sum_i e_i - \sum_i k_i} f\left(\prod_{i=1}^{j} p_i^{k_i}\right) = g\left(\prod_{i=1}^{j} p_i^{e_i}\right).$$

Sustituyendo $m_i = e_i - k_i$:

$$g\left(\prod_{i=1}^{j} p_i^{e_i}\right) = \sum_{m_1=0}^{1} \sum_{m_2=0}^{1} \cdots \sum_{m_j=0}^{1} (-1)^{\sum_i m_i} f\left(\prod_{i=1}^{j} p_i^{e_i-m_i}\right).$$

Finalmente podemos reescribirlo como en el enunciado

$$g\left(\prod_{i=1}^{j} p_i^{e_i}\right) = \sum_{m_1=0}^{1}\sum_{m_2=0}^{1}\cdots\sum_{m_j=0}^{1}\mu\left(d=\prod_{i=1}^{j} p_i^{m_i}\right) f\left(\prod_{i=1}^{j} p_i^{e_i-m_i}\right),$$

donde μ es la definida en el enunciado, entonces hemos probado que

$$\boxed{g(n) = \sum_{\substack{d \\ d|n}} \mu(d) f\left(\frac{n}{d}\right).}$$

□

Teorema 4.9. *Dado $\mathbb{F}_q[x]$, entonces*

$$q^k = \sum_{\substack{m \\ m|k}} m I_m$$

Demostración. Para la demostración basta usar un resultado conocido, el producto de todos los polinomios irreducible mónicos de grado menor que k sobre $\mathbb{F}_q[x]$ es el polinomio $x^{q^k} - x$, por lo que contando los grados tengo que

$$\boxed{q^k = \sum_{\substack{m \\ m|k}} m I_m}$$

□

Finalmente voy a usar ambos teoremas para encontrar una fórmula para el número de polinomios irreducibles mónicos de grado k sobre $\mathbb{F}_q[x]$.

Teorema 4.10. *Dado $\mathbb{F}_q[x]$, entonces*

$$I_k = \frac{1}{k}\sum_{\substack{k \\ k|n}} \mu(k) q^{n/k}$$

Demostración. Para la demostración partimos de Teorema 4.9

$$q^k = \sum_{\substack{m \\ m|k}} m I_m$$

Y aplicamos la fórmula de inversión de Moebius (véase Teorema 4.8), considerando $f(k) = q^k$ y $g(m) = mI_m$.

$$g(k) = \sum_{\substack{k \\ k|n}} \mu(k) f\left(\frac{n}{k}\right) \Leftrightarrow \boxed{I_k = \frac{1}{k} \sum_{\substack{k \\ k|n}} \mu(k) q^{n/k}}$$

□

4.3.2. Códigos Goppa y la cota de Gilbert–Varshamov

Desde que se introdujo esta cota inferior para la distancia (Vease Def. 4.7), no había códigos que la superaran, hasta que V.D Goppa introdujo sus códigos de geometría algebraica (Vease Def. 4.1) que la alcanzan.

Lema 4.6. *Dado* $0 < \delta \leq 1 - q^{-1}$ *donde* $q \geq 2$, *entonces*

$$\lim_{n \to \infty} n^{-1} \log_q V_q(n, \lfloor \delta n \rfloor) = H_q(\delta)$$

Demostración. Consideramos en la definición de $V_q(n, d)$ $d = \lfloor \delta n \rfloor$

$$\begin{pmatrix} n \\ \lfloor \delta n \rfloor \end{pmatrix} (q-1)^{\lfloor \delta n \rfloor} \leq V_q(n, \lfloor \delta n \rfloor) \leq (1 + \lfloor \delta n \rfloor) \begin{pmatrix} n \\ \lfloor \delta n \rfloor \end{pmatrix} (q-1)^{\lfloor \delta n \rfloor}.$$

Tomando logaritmo en esta expresión tengo que

$$A + n^{-1} \begin{pmatrix} n \\ \lfloor \delta n \rfloor \end{pmatrix} \log_q(q-1) \leq n^{-1} \log_q V_q(n, \lfloor \delta n \rfloor).$$

Y por otro lado

$$n^{-1} \log_q V_q(n, \lfloor \delta n \rfloor) \leq A + n^{-1} \lfloor \delta n \rfloor \log_q(q-1) + n^{-1} \log_q(1 + \lfloor \delta n \rfloor),$$

donde $A = n^{-1} \log_q \begin{pmatrix} n \\ \lfloor \delta n \rfloor \end{pmatrix}$, aplicando el desarrollo de la combinatoria tenemos que

$$A = \frac{n!}{\lfloor \delta n \rfloor! (n - \lfloor \delta n \rfloor)!}.$$

Entonces tomando límite

$$\lim_{n\to\infty} n^{-1}\log_q V_q(n,\lfloor \delta n\rfloor) = \lim_{n\to\infty} A + \delta\log_q(q-1).$$

Y además

$$\frac{n^{n+1/2}e^{-n+7/8}}{\lfloor\delta n\rfloor^{\lfloor\delta n\rfloor+1/2}e^{-\lfloor\delta n\rfloor+1}(n-\lfloor\delta n\rfloor)^{n-\lfloor\delta n\rfloor+1/2}e^{-n+\lfloor\delta n\rfloor+1}} \leq \begin{pmatrix} n \\ \lfloor\delta n\rfloor \end{pmatrix},$$

$$\begin{pmatrix} n \\ \lfloor\delta n\rfloor \end{pmatrix} \leq \frac{n^{n+1/2}e^{-n+1}}{\lfloor\delta n\rfloor^{\lfloor\delta n\rfloor+1/2}e^{-\lfloor\delta n\rfloor+7/8}(n-\lfloor\delta n\rfloor)^{n-\lfloor\delta n\rfloor+1/2}e^{-n+\lfloor\delta n\rfloor+7/8}}.$$

Entonces

$$Be^{-9/8} \leq \begin{pmatrix} n \\ \lfloor\delta n\rfloor \end{pmatrix} \leq Be^{-3/4},$$

donde

$$B = \frac{n^{n+1/2}}{\lfloor\delta n\rfloor^{\lfloor\delta n\rfloor+1/2}(n-\lfloor\delta n\rfloor)^{n-\lfloor\delta n\rfloor+1/2}}.$$

Sustituyendo en el límite tengo que

$$\lim_{n\to\infty} n^{-1}\log_q(Be^k) = \lim_{n\to\infty} n^{-1}(\log_q B + k\log_q e) = \lim_{n\to\infty} n^{-1}\log_q B,$$

$$\lim_{n\to\infty} A = \lim_{n\to\infty} n^{-1}B = -\delta\log_q\delta - (1-\delta)\log_q(1-\delta).$$

Entonces finalmente obtengo el resultado deseado

$$\lim_{n\to\infty} n^{-1}\log_q V_q(n,\lfloor\delta n\rfloor) = -\delta\log_q\delta - (1-\delta)\log_q(1-\delta) + \delta\log_q(q-1) = H_q(\delta).$$

$$\boxed{\lim_{n\to\infty} n^{-1}\log_q V_q(n,\lfloor\delta n\rfloor) = H_q(\delta).}$$

□

Lema 4.7. *Dado $\mathbb{F}_{q^t}[x]$, y sea I_m el número de polinomios mónicos de grado m sobre $\mathbb{F}_{q^t}[x]$. Entonces*

$$I_m \geq \frac{q^{tm}}{m}\left(1 - q^{-te/2+1}\right)$$

Demostración. Partimos de la expresión obtenida para I_m sobre $\mathbb{F}_{q^t}[x]$ (Vease Teorema 4.10)

$$I_m = \frac{1}{m}\sum_{\substack{k \\ k|n}} \mu(k) q^{n/k}$$

Y hacemos algunas simplificaciones para obtener la cota deseada

$$\frac{1}{m}\sum_{\substack{k \\ k|n}} \mu(k) q^{n/k} > \frac{1}{m}\left(q^{tm} - q^{tm/2} - q^{tm/3} - \cdots\right) > \frac{1}{m}\left(q^{tm} - \sum_{i=0}^{r/2} q^{ti}\right) > \frac{q^{tm}}{m}\left(1 - q^{te/2+1}\right)$$

Por lo que ya tengo el resultado deseado

$$\boxed{I_m > \frac{q^{tm}}{m}\left(1 - q^{te/2+1}\right)}$$

□

Teorema 4.11. *Existe una familia de Códigos Goppa sobre $\mathbb{F}_q$ que alcanzan la cota asintótica de Gilbert–Varshamov.*

Demostración. Dados t y d enteros positivos. Consideramos el código Goppa de longitud $n = q^t$ con $L = \{\gamma_0, \cdots, \gamma_{n-1}\}$ y $G = \mathbb{F}_{q^t}[X]$ que además es irreducible de grado mayor que 1 sobre $\mathbb{F}_{q^t}$ para evitar que $G(x)$ tenga raíces en L.

Supongamos que $c \in \mathcal{C}$, entonces cumple

$$\sum_{i=0}^{n-1} \frac{c_i}{x - \gamma_i} \equiv 0 \text{ mód } G(x) \Leftrightarrow \sum_{i=0}^{n-1} \frac{c_i}{x - \gamma_i} \equiv \frac{a(x)}{b(x)},$$

donde $b(x)$ es $\prod_{i=0}^{n-1}(x - \gamma_i)$ y $a(x)$ tiene al menos un factor irreducible que es $G(x)$ y además tiene como máximo $(w-1)/e$ factores irreducibles.

Por otro lado si tenemos que $c \notin \mathcal{C}$, tenemos que el polinomio generador del código podemos elegirlo entre todos los irreducible de $\mathbb{F}_{q^t}[X]$ de grado e salvo $(w-1)/e$ polinomios.

Por cada $1 < w < d$ existen $\begin{pmatrix} n \\ w \end{pmatrix} (q-1)^w$ elementos c que no pertenecen a $\mathcal{C}$, por lo que tenemos

$$\sum_{w=1}^{d-1} \frac{w-1}{e} \begin{pmatrix} n \\ w \end{pmatrix} (q-1)^w \leq \frac{q}{e} V_q(n,d)$$

polinomios que no podemos usar como G.

Nuestro objetivo es construir un código Goppa de distancia mínima d, por lo que necesito un polinomio irreducible G que cumpla las condiciones, por la igualdad anterior lo voy a poder conseguir siempre que

$$\frac{d}{e}V_q(n,d) < \frac{q^{te}}{e}\left(1 - q^{-te/2+1}\right).$$

Entonces si suponemos $\delta = d/n$ tomando logaritmos en base q.

$$\log_q(d) - \log_q(e) + \log_q(V_q(n,d)) \leq te - \log_q(e) + \log_q(1 - q^{-te/2+1}).$$

Y ahora simplificando y dividendo por n:

$$n^{-1}\left(\log_q(\delta n) + \log_q(V_q(n,d))\right) \leq \frac{te}{n} + n^{-1}\log_q(1 - q^{-te/2+1}).$$

La idea sería tomar $n \to \infty$:

$$\lim_{n\leftrightarrow\infty}\left(n^{-1}\left(\log_q(\delta n) + \log_q(V_q(n,d))\right)\right) \leq \lim_{n\leftrightarrow\infty}\left(\frac{te}{n} + n^{-1}\log_q(1 - q^{-te/2+1})\right).$$

Aplicando el lema anterior tenemos que

$$\lim_{n\leftrightarrow\infty} n^{-1}\left(\log_q(\delta n)\right) \to 0 + \lim_{n\leftrightarrow\infty}\left(\log_q(V_q(n,d))\right) \leq \lim_{n\leftrightarrow\infty}\frac{te}{n} + \lim_{n\leftrightarrow\infty} n^{-1}\log_q(1 - q^{-te/2+1}) \to 0,$$

$$H_q(\delta) \leq \lim_{n\to\infty} te/n.$$

Restando a 1 podemos deducir que

$$1 - H_q(\delta) \geq 1 - \lim_{n\to\infty} te/n,$$

como $t = \log_q n$ podemos tomar una sucesión creciente $\{e_n\}$ que mantiene la desigualdad anterior, lo que nos asegura que tenemos una familia de códigos Goppa con distancias crecientes y distancias mínimas relativas al menos δn.

Y además con la misma sucesión podemos afirmar que $H_q(\delta) = \lim_{n\to\infty} te/n$.

Finalmente por Teorema 4.2 podemos afirmar que los códigos generados en la secuencia tiene una ratio al menos de $1 - te/n$, por lo que alcanzan la cota de Gilbert–Varshamov. □

Bibliografía

[1] Berlekamp, E. R. *Algebraic coding theory (revised edition)*. World Scientific Publishing, Singapore, Singapore, May 2015.

[2] Bose, R. C., and Ray-Chaudhuri, D. K. On a class of error correcting binary group codes. *Information and Control 3* (1960), 68–79. 25

[3] Cancellieri, G. *Polynomial Theory of Error Correcting Codes*. Springer International Publishing, 2015.

[4] Goppa, V. D. *Geometry and Codes*. Mathematics and its Applications,24 (Soviet series). Springer Science+Business Media, 1988. 33

[5] Hamming, R. W. Error detecting and error correcting codes. *The Bell System Technical Journal 29*, 2 (1950), 147–160. 5

[6] Hocquenghem, A. Codes correcteurs d'erreurs. *Chiffres 2* (1959), 147–156. 25

[7] Huffman, W. C., and Pless, V. Fundamentals of error-correcting codes, 2003.

[8] Li, W. *Decoding evaluation codes and their interleaving*. PhD thesis, 2015.

[9] Peterson, W. W. *Error correcting codes*, 2nd ed. ed. MIT Press, Massachusetts, 1972.

[10] Singleton, R. Maximum distance q-nary codes. *IEEE Transactions on Information Theory 10*, 2 (1964), 116–118. 7

[11] Sugiyama, Y., Kasahara, M., Hirasawa, S., and Namekawa, T. A method for solving key equation for decoding goppa codes. *Information and Control 27*, 1 (Jan. 1975), 87–99. 37

[12] Tsfasman, M. A., and Vladut, S. G. *Algebraic-Geometric Codes*. Mathematics and its Applications. Springer, New York, NY, Nov. 2001. 33

[13] van Lint, J. H. *Introduction to Coding Theory*. Springer Berlin Heidelberg, 1982.

[14] Voss, C., and Stichtenoth, H. Asymptotically good families of subfield subcodes of geometric goppa codes. *Geometriae Dedicata 33*, 1 (Jan. 1990).

[15] y Margarita Cabrera, F. T. *Códificación de canal I: Introducción y códigos de bloque*. Universidad Oberta de Catalunya, 2012.