UNIVERSITAT JAUME I

Grado en Matemática Computacional

Trabajo Final de Grado

# Códigos Cíclicos y Cuasi-Cíclicos

*Autor:*
Raúl Adsuara Nebot

*Tutor académico:*
Fernando Hernando Carrillo

Fecha de lectura: __ de _____ de 20__
Curso académico 2021/2022

## Resumen

En este trabajo de fin de grado sobre la *codificación de la información* queremos introducir al lector los conceptos básicos de la codificación así como las *distancia mínima*, las formas de medir la calidad de un código, como optimizarlo e incluso extenderlo.

También mostrar los teoremas y lemas necesarios para poder comprender las codificaciones más elementales como son las lineales y las cíclicas.

Por otra parte, nos adentraremos en los códigos cíclicos que tienen una gran importancia debido al uso y estudio que tienen en la actualidad. Mostrando así, por que debemos buscar generadores de códigos dotados de estructuras matemáticas que puedan facilitar, agilizar o menguar la cantidad de recursos necesarios para su decodificación.

Finalmente, buscaremos introducir los códigos cuasi-cíclicos, que requieren de una estructura algebraica más potente y complejidad, y, dejar las puertas abiertas a una mayor profundización en este tipo de códigos ampliamente utilizados en la actualidad.

## Palabras clave

códigos - cíclicos - cuasi-cíclicos - estructuras-lineales - distancia-mínima

## Keywords

codes - cyclics - quasi-cyclics - linear structures - minimum distance

# Índice general

**1. Introducción** **7**

1.1. História de la codificación . . . . . 8

1.2. Contexto y motivación del proyecto . . . . . 10

**2. Introducción a los códigos** **13**

2.1. Conceptos previos . . . . . 13

2.2. Medida de la información . . . . . 18

2.3. Tipos de canales . . . . . 23

2.3.1. Canales sin ruido . . . . . 23

2.3.2. Canales con ruido . . . . . 28

**3. Estructuras Lineales** **37**

3.1. Código de Hamming . . . . . 47

3.2. Código de Golay . . . . . 49

3.3. Cotas en los parámetros de un código . . . . . 53

**4. Códigos Cíclicos** **55**

**5. Códigos Cuasi Cíclicos** **69**

**6. Conclusiones** **77**

# Capítulo 1

# Introducción

A partir de la creación de los ordenadores llegó una nueva forma de tratar la información, y con ella, millones de avances y de nuevos problemas a resolver que crearon nuevos campos de investigación. Estos nos permitieron manejar increíbles cantidades de información por segundo, enviar datos de una parte del mundo a otra en cuestión de milésimas de segundo, se abrieron las puertas a una nueva era.

No obstante, estos avances se apoyaron en uno de los pilares más importantes que utilizamos hoy en día para poder transmitir, almacenar y encriptar información, la *teoría de códigos*.

Una de las aplicaciones más utilizadas recientemente es en *Local Recovery Codes*[1], es decir, recuperación local de códigos. Este proceso consiste en la recuperación de información tras la caída de uno o varios servidores a partir de la información almacenada en sus otros servidores.

El algoritmo utilizado tras la caída de un solo servidor suele tener un tiempo de recuperación notablemente más rápido que para varios. En cambio, para pérdidas de información de más de un servidor tomará un mayor tiempo en recuperarse, no obstante, cubrirá igualmente la necesidad de restablecer la información.

Por último, cabe remarcar que el uso de códigos tiene una gran cantidad de campos de estudio con todo tipo de propiedades con muchas aplicaciones. Es por ello que cada vez es más importante profundizar en este tipo de campos.

Por otra parte expondremos la estructura consolidada en este artículo, describiendo las partes de cada punto a tratar.

Primeramente explicaremos las bases y conceptos básicos de la teoría de la información así

como la estructura de diferentes *códigos lineales*, sus parámetros, codificación y decodificación.

A continuación, hablaremos de la manera en que transformaremos los conceptos en objetos matemáticos, así como el *mensaje*, el *código* o la *cantidad de información* que emitimos en cada transmisión.

Los diferentes canales sobre los que podemos enviar dicho flujo de información, ya sea sobre un *canal con ruido*, o, para la explicación de conceptos más simples utilizaremos un idílico *canal sin ruido*.

Una vez introducidos los canales con ruido empezaremos a hablar de *distáncia de Hamming* y *distancia mínima* con tal de calcular la seguridad de un código ante las alteraciones durante la transmisión.

Finalmente, expondremos la estructura de los *códigos lineales*, haciendo hincapié en los *códigos de Hamming* y *de Golay* y terminaremos con los *códigos cíclicos* profundizando en los *cuasi cíclicos*.

## 1.1. História de la codificación

La historia de la codificación de la información nace en el año 1948 tras la publicación de Claude Elwood Shannon de *A Mathematical Theory of Communication*[3]. Este libro describe una forma de convertir los procesos de transmisión en matemáticas.

Una de las ideas en las que se basa dicha teoría es que a la hora de la transmisión de la información, el contenido semántico del mensaje debe ser totalmente independiente del tratamiento de dicha información.

Por otra parte, y citando al autor de dicho libro «*La característica fundamental de la información es que puede ser transmitida*».

Para Shannon, el estudio de la teoría de la codificación de la información implica el estudio matemático de procesos de transmisión de mensajes y la medida de la cantidad de información que contienen.

Para obtener información digital, primeramente, necesitaremos transformar la información analógica (libros, mapas, fotos...) en digital (páginas web, libros electrónicos, periódicos digitales, vídeos, audios...) utilizando diferentes recursos como pueden ser escáneres, grabación digital, cámaras de fotos...

Uno de los primeros problemas que encontramos fue cómo almacenar información que requería una gran cantidad de espacio de la manera más óptima posible.

Con tal de ponernos en contexto, con la aparición de los primeros satélites capaces de captar imágenes, no obstante la calidad de estas no era la mejor ya que para guardar una fotografía de alta calidad requería un mayor espacio.

Es decir, una imagen con una resolución de 15 metros por 15 metros de precisión no podía ser comparable a una de 82 centímetros por 821.1 (cada píxel tiene una resolución de 82x82 centímetros) tal y como se muestra en la imagen sacada del artículo[5].

(a) (b)

Cuadro 1.1: Resolución de imagen 15x15 metros y 82x82 centímetros respectivamente.

No obstante, no siempre hemos podido almacenar la información en lujosos ordenadores capaces de entender cualquier formato de información, por ello, se hizo uso de CDs, DVDs o disquetes entre otros.

Estos instrumentos capaces de almacenar una serie de datos que más tarde podríamos recuperar sigue los mismos principios de codificación a los utilizados actualmente:

1. *Estructura y parámetros*: Se define una estructura estándar para la transmisión de la información. En el caso de los CDs, en 1968 se reunieron 35 grandes fabricantes de CDs[4] para la unificación de criterios.

2. *Codificación*: Se define una estructura de codificación de los mensajes eficaz para el tipo de necesidad que tengamos. Los CDs están construidos con policarbonato, gracias a ello la información puede grabarse con un láser a alta potencia. Además, como ya hemos puntualizado, sigue las bases de la codificación ya que este se graba en formato binario sobre el disco[4], representando por 0s o 1s la información dependiendo de la cantidad de tinte que calentaba el láser en cada parte del disco.

3. *Decodificación*: Gracias a la definición de un estándar, se podrán leer los datos almacenados en sus distintos dispositivo de recuperación de información.

**Modos de codificación de un mensaje**

Más allá de los problemas que podamos encontrar en un canal, dependiendo de la necesidad que tengamos a la hora de transmitir un mensaje utilizaremos unos u otros modos de codificación.

- *Compresión de la información*: Utilizar el menor espacio posible para almacenar la información.
- *Codificación contra errores*: Aplicar la menor redundancia posible para conseguir un código corregible ante canales con ruido.
- *Cifrado*: Trata de asegura la confidencialidad del receptor.

Estos modos de codificación pueden estudiarse independientemente aun que en la práctica tienden a generar intersecciones. No obstante, debido a su increíble crecimiento se han convertido en tres grandes campos:

- *Teoría de la Compresión de Información*
- *Teoría de Códigos Correctores de errores*
- *Criptología*

## 1.2. Contexto y motivación del proyecto

En este trabajo mostraremos la teoría de la codificación desde cero considerando que el lector tiene una pequeña base matemática sobretodo en los campos del álgebra lineal y conmutativa, puesto que serán la base de los códigos que estudiaremos en los últimos capítulos.

El objetivo será introducir al lector en el mundo de la codificación, transformando los conceptos de la transmisión y codificación en objetos matemáticos, dotar los códigos de estructuras algebraicas que nos permitan utilizar como herramientas para hacer mucho más óptima su codificación, compresión y decodificación.

Una vez consolidados los conceptos más básicos comenzaremos con códigos que utilizan los álgebra lineal, aprovechándonos de sus propiedades.

A continuación definiremos una estructura un tanto más compleja puesto que encontraremos matrices cuya estructura se encontrará basada en el álgebra conmutativa, la cual nos dará una mayor cantidad de herramientas y nos permitirá definir su estructura a partir de la cantidad de errores que queramos corregir.

No obstante, este tipo de códigos se verá condicionado por la cantidad de letras del alfabeto utilizado, es por ello que encontraremos una nueva estructura que contendrá a esta anterior, los códigos cuasi-cíclicos.

Los códigos cuasi-cíclicos contendrán algunas propiedades algebraicas que nos brindan los cíclicos puesto que su matriz generatriz se basará en unión de varios de estos, por otra parte, con tal de tratar con estos deberemos recurrir a las bases de *Gröbner*.

Por otra parte, estos códigos tienen un estudio que requieren una amplia profundidad y complejidad, es por ello que en este trabajo nos limitaremos a mostrar los conceptos básicos de su codificación y una explicación de como se forma su estructura.

# Capítulo 2

# Introducción a los códigos

## 2.1. Conceptos previos

Como ya sabemos, para poder codificar la información debemos transformarla en una serie de símbolos que nos permita el tratamiento de dicha información.

Por ejemplo, en nuestro ordenador podemos codificar el número nueve por el símbolo 9. De igual manera, podemos utilizar el código binario, 1001 o cualquier otra forma de codificación conocida. El conjunto de símbolos utilizados para representar la información le llamaremos alfabeto.

Para comenzar a emplear términos matemáticos, definiremos *alfabeto* como cualquier conjunto finito de elementos (3.1.a), y, llamaremos *información digital* a cualquier sucesión de elementos de un alfabeto (3.1.b).

| A B C D E F G H I J<br>K L M N Ñ O P Q R<br>S T U V W X Y Z | AMIGO CASA<br>PISO REVISAR<br>ORDENADOR<br>ESTUDIAR AZUL |
|---|---|
| (a) | (b) |

Cuadro 2.1: Ejemplo en el lenguaje español de alfabeto e información digital.

### Transmisión de la información

*Todos los teoremas utilizados en esta sección se encuentran basados en el libro Codificación de la Información*[2]

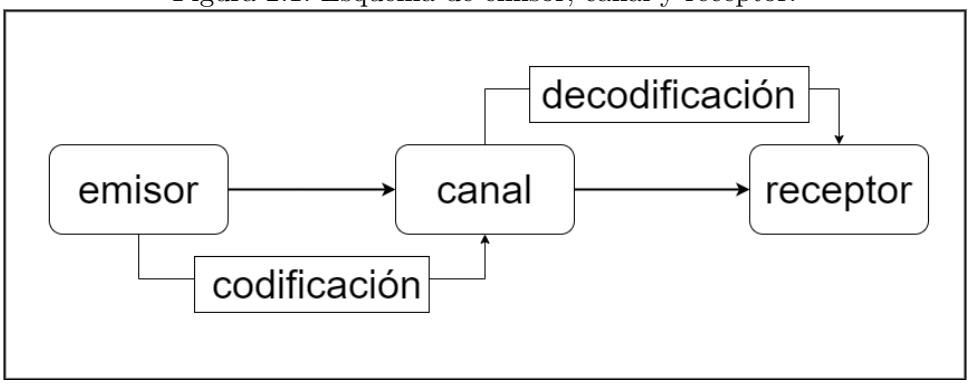
A la hora de la transmisión de información entre un emisor y un receptor deberemos de disponer de un canal a través del cual enviaremos nuestro código.

El emisor deberá codificar la información antes de enviarla por el canal (*Figura 2.1*). El receptor, por su parte, obtendrá la información codificada y para poder leer el mensaje tendrá que decodificarla.

Por otra parte, tal vez la simbología utilizada ocupe demasiado, o sea vulnerable a interferencias o ataques. Por ello, deberemos codificar la información de la manera que más nos convenga, es decir, reescribirla en otro alfabeto siguiendo unas reglas establecidas.

Con tal de saber que formato debemos utilizar, recurriremos a *códigos compresores*, *códigos correctores* o *códigos criptográficos* (o secretos) dependiendo de nuestras necesidades..

Figura 2.1: Esquema de emisor, canal y receptor.




### Codificación de alfabetos

Tal y como explicábamos en el apartado anterior, la manera en que disponemos de la información no siempre es la más óptima a la hora de codificar información. Es por ello que transformaremos la información del *alfabeto fuente* en *alfabeto código.*

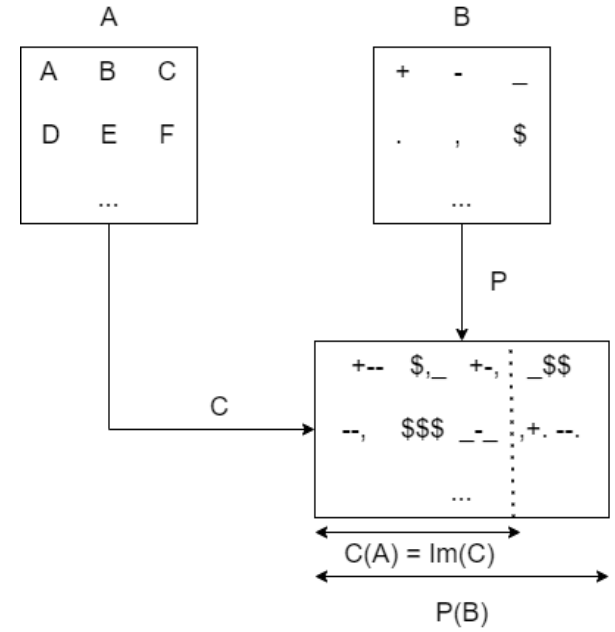
**Definición 1** *Sea $\mathcal{A}$ un alfabeto cualquiera, llamaremos palabra $\mathcal{P}(\mathcal{A})$, a cualquier secuencia finita de elementos de $\mathcal{A}$.*

**Definición 2** *Para codificar* $\mathcal{A} = \{a_1, ..., a_p\}$ *en* $\mathcal{B} = \{b_1, ..., b_p\}$ *utilizaremos la aplicación inyectiva,*

$$c : \mathcal{A} \to \mathcal{P}(\mathcal{A})$$

Además, cada $a_i \in \mathcal{A}$ se codificará como c($a_i$) tal que $C = Im(c) \subseteq P(B)$ y llamaremos a cualquier c $\in \mathcal{C}$ *palabra código* (o *palabra* para abreviar).

Figura 2.2: Esquema de codificación de letras de nuestro alfabeto fuente




**Nota 1** *Uno de los principios lógicos de la codificación es que cada palabra debe ser distinta a las demás, es por ello que su definición introduce una aplicación inyectiva en la que cada elemento de A tendrá uno solo en el subconjunto C.*

**Nota 2** *El tipo de símbolo utilizado a la hora de codificar la información nos es totalmente irrelevante desde el punto de vista matemático. La única propiedad que utilizaremos del alfabeto B es la longitud del alfabeto.*

Es decir, no nos importa si se codifica en dígitos, letras, guiones, espacios, etc. Tan solo nos interesará saber si es un código binario, ternario, decimal, hexadecimal, etc.

**Definición 3** *La longitud de una palabra indica el número de símbolos necesarios para componerla.*

**Definición 4** *La longitud de una palabra puede ser en bloque, si todas las palabras tienen la misma longitud n, o variables, si cada palabra tiene una longitud diferente.*

### Codificación del mensaje

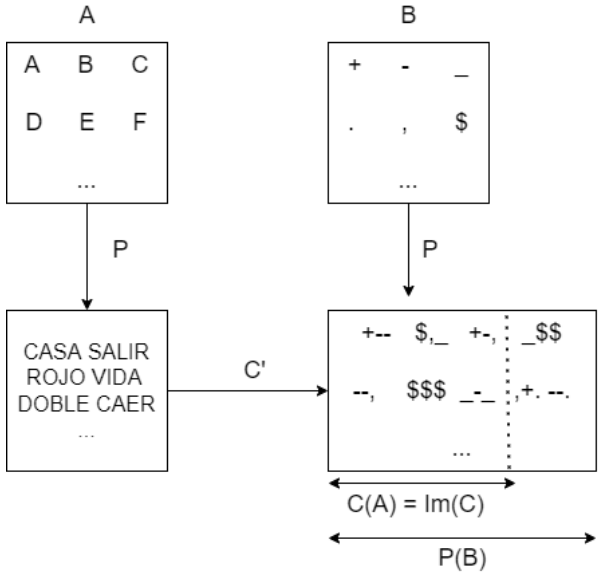
Como bien hemos explicado para codificar un alfabeto tendremos que aplicar una codificación $C$ sobre cada símbolo de nuestro alfabeto $A$. No obstante, en la práctica, nuestro objetivo no será transformar letras si no palabras.

Con este nuevo problema definiremos una nueva aplicación no necesariamente inyectiva $c'$ tal que,

$$c' : P(A) \to P(B)$$

De manera que $c'(a_{i_1}a_{i_2}...a_{i_n}) = c(a_{i_1})c(a_{i_2})...c(a_{i_n})$ (*Figura 2.3*)

Figura 2.3: Esquema de codificación de palabras en el alfabeto fuente




.

**Decodificación del mensaje**

A la hora de decodificar un mensaje (transformar la codificación recibida en el alfabeto fuente) solo la podremos realizar en caso de recepción de una sola letra o si el código es de decodificación única.

Con tal de entender el problema que encontramos cuando queremos decodificar un mensaje pondremos un ejemplo.

**Ejemplo 1** *Como sabemos, la codificación de cada palabra o letra debe ser única, en cambio a la hora de la decodificación no siempre tiene por que cumplir esta regla tal y como mostramos a continuación,*

*Supongamos que tenemos la siguiente codificación:*

| | |
|---|---|
| A | 0 |
| B | 10 |
| C | 101 |

Cuadro 2.2: Tabla de codificación de ejemplo I

.

*Por tanto, al recibir el mensaje 1010, una posible decodificación podría ser BB y otra CA.*

**Decodificación única**

Tal y como hemos hablado, hay una serie de códigos con la propiedad de que tanto su codificación de palabras como su decodificación son únicos, matemáticamente, este suceso se genera cuando $c'$ es una aplicación inyectiva.

Por otra parte, los códigos de mayor uso suelen ser de este tipo ya que lo que nos interesa es una decodificación única. Los códigos en bloque siempre tienen una decodificación única, en cambio otros como el Morse utilizan espacios para separar las letras.

## 2.2. Medida de la información

Con tal de tratar la información nos interesará medir la cantidad de veces que aparece un símbolo del alfabeto fuente en los mensajes emitidos, para ello utilizaremos distribuciones de probabilidad.

**Definición 5** *Definiremos una distribución de probabilidad sobre* $A = \{a_1, a_2, ...a_n\}$ *tal que para cada* $a_i \in A$ $p_i = prob(a_i)$ *y cumpliendo con las propiedades:*

1. $\forall p_i \in \mathbb{R} \quad 0 \leq p_i \leq 1$
2. $p_1 + p_2 + ...p_m = 1$

**Definición 6** *Llamaremos fuente de información* $F$ *al conjunto de un alfabeto fuente* $A$ *y su respectiva distribución de probabilidad* $p_1, p_2, ..., p_m$ *sobre* $A$ *tal que* $prob(a_{i_1}...a_{i_r}) = p_{i_1}...p_{i_r}$*, ya que se asume que son sucesos independientes.*

Partiendo de esta idea, si hemos recibido un mensaje compuesto de 0s y 1s, tomaríamos un alfabeto $A = \{0, 1\}$ con probabilidades $p_1$ y $p_2$. No obstante, otra posibilidad podría ser tomar el alfabeto $A^2 = \{00, 01, 10, 11\}$ cuyas probabilidades serían,

$$\begin{cases} p'_1 = p_1 p_1 \\ p'_2 = p_1 * p_2 \\ p'_3 = p_2 * p_1 \\ p'_4 = p_2 * p_2 \end{cases}$$

Y podríamos continuar así hasta generar cualquier $A^k$.

**Definición 7** *Sea la fuente de información* $F$ *asociada al alfabeto* $A = \{a_1, ..., a_m\}$ *y con una distribución de probabilidad* $p_1, ...p_m$*, definimos como extensión k-ésima de* $F$ *a la fuente de información* $F^k$ *cuyo alfabeto fuente será* $A^k$*, y por tanto, su distribución de probabilidad será* $prob(a_{i_1}, ..., a_{i_k}) = p_{i_1}...p_{i_k}$*, tal y como mostramos en el ejemplo anterior.*

### Medida de la información

Para explicar el concepto de medida de información definiremos primero una fuente de información. Esta fuente se basará en el comportamiento de una moneda ideal, es decir, $A = \{0, 1\}$ representando 0 como *cara* y 1 como *cruz* y con una probabilidades $p(0) = \frac{1}{2}$ y $p(1) = \frac{1}{2}$.

A la probabilidad de que aparezca cierta información le llamaremos *cantidad de información*, y, en nuestro caso, $I(\text{"}cara\text{"}) = 0{,}5$ y $I(\text{"}cruz\text{"}) = 0{,}5$.

**Definición 8** *Llamaremos bit a la cantidad de información que nos ofrece cada uno de los símbolos que componen una fuente de información con dos símbolos equiprobables.*

**Definición 9** *Sea el alfabeto fuente $A = \{a_1, a_2, ...a_m\}$ y las probabilidades $p_1, ..., p_m$ definiremos $I(a_i)$ como la cantidad de información del símbolo $a_i$ y debe cumplir las siguientes propiedades:*

1. *$I(a_i)$ solamente dependerá de sus parámetros $a_i$ y $p_i$.*
2. *La función $I$ se encontrará definida en el intervalo $(0, 1]$ ya que $p_i \in (0, 1]$ exceptuando el 0 puesto que $I(0) = \infty$.*
3. *Para cada par $a_i, a_j \in A$, $I(a_i, a_j) = I(a_i) + I(a_j)$ porque la fuente no tiene memoria.*
4. *$I$ será una función decreciente, cuando más improbable un suceso, mayor información aportará.*

**Nota 3** *Debido a la propiedad 1, podremos representar la cantidad de información tanto como $I(a_i)$ como por $I(p_i)$.*

**Proposición 1** *Sea la función $f : (0, 1] \to \mathbb{R}$ tal que,*

1. *$f$ es decreciente*
2. *$f(xy) = f(x) + f(y) \qquad \forall x, y \in (0, 1]$*

*Entonces $\exists k > 0$ constante tal que $f(x) = kln(x)$*

Por ejemplo, para medir la información del código ASCII que cuenta con 256 caracteres cuya codificación binaria implicará entonces 8 bits ($log_2(256) = 8$)

**Corolario 1** *$I(p_i) = -log_2(p_i) = log_2(\frac{1}{p_i})$ (bits)*

**Entropía**

Además de saber la cantidad de información que aporta cada símbolo de nuestra codificación, también querremos calcular una media más general a la que llamaremos *entropía*.

**Definición 10** *Dada una fuente de información $F$, con un alfabeto $A = \{a_1, a_2, ...a_m\}$ y cuyas probabilidades son $p_1, ..., p_m$, definiremos la entropía al resultado de la siguiente operación,*

$$H(F) = \sum_{i=1}^{m} p_i I(p_i) = \sum_{i=1}^{m} p_i log(\tfrac{1}{p_i})$$

*Es decir, la suma de la probabilidad de que aparezca cada símbolo por la cantidad de información que desprende.*

**Ejemplo 2** *Continuando con el ejemplo de la moneda, la entropía de dicha fuente de información se calcula de la siguiente manera,*

$$H(F_{moneda}) = p_{cara} I(p_{cara}) + p_{cruz} I(p_{cruz}) = 0{,}5 log(\frac{1}{0{,}5}) + 0{,}5 log(\frac{1}{0{,}5}) = 2$$

**Nota 4** *Puesto que la entropía de $F$ depende únicamente de $p_i$, la denotaremos como $H(F)$ o $H(p_1, ..., p_m)$ indistintamente.*

**Nota 5** *En caso de que cualquier $p_i = 0$, encontraremos el siguiente problema $p_i log(\frac{1}{p_i}) = 0 log(\frac{1}{0})$. Con tal de evitarlo, por convenio, acordaremos que $0 log(\frac{1}{0}) = 0$.*

**Notación 1** *Los logaritmos utilizados son todos en base 2, y, con tal de simplificar, escribiremos $log(x)$ en vez de $log_2(x)$. En caso de excepción se escribirá su debida base.*

**Lema 1** *Dadas dos distribuciones de probabilidad $p_1, ..., p_m$ y $q_1, ..., q_m$ sobre $A$,*

$$\sum_{i=1}^{m} p_i log(\frac{1}{p_i}) \leq \sum_{i=1}^{m} p_i log(\frac{1}{q_i})$$

*con igualdad sí y solo sí, $p_i = q_i$.*

**Teorema 1** *Sea $F$ una fuente de información asociada al alfabeto $A$ de $m$ símbolos diferentes y cuya distribución de probabilidad es $p_1, ..., p_m$,*

$$0 \leq H(F) \leq log(m)$$

1. $H(F) = 0 \iff \exists i : p_i = 1$
2. $H(F) = log(m) \iff p_i = \frac{1}{m} \forall i$

**Proposición 2** *Sea la fuente extendida $F^k$, entonces*

$$H(F^k) = kH(F)$$

**Extensión**

Con tal de poner un uso peculiar en el que se utiliza el concepto de medición de la información para ataques a códigos cuya codificación se hace independientemente símbolo a símbolo y donde conocemos el idioma utilizado de la información fuente, expondremos una metáfora para entender el problema y una explicación más técnica al final pondremos el siguiente ejemplo.

Supongamos que tenemos tres vasos puestos hacia abajo, en uno de ellos tenemos una bola y nuestro objetivo es encontrar en que vaso se encuentra.

En un modelo hecho con vasos totalmente opacos las probabilidades serían de $\frac{1}{3}$, es decir, ningún vaso revela más información que los otros y es por ello que puede considerarse un modelo prácticamente aleatorio, lo cual buscaremos a la hora de encriptar mensajes.

Ahora imaginemos que uno de esos vasos es transparente, es decir, este vaso nos revelaría una gran cantidad de información, lo cual es malo para el modelo y si la bola se encuentra en este vaso podríamos acertarlo sin problema. O imaginemos que cada vaso es de un color y hemos observado en anteriores juegos que el trilero tiende a poner la bola en el vaso azul, esa pequeña dosis de información nos ayudaría a la hora de decidir.

Finalmente, que pasaría si tuviésemos dos vasos transparentes, es decir, que aporten una gran cantidad de información. Básicamente seríamos de capaces de acertar siempre el resultado ya que gracias a la información que revelan algunas partes podríamos ser capaces de desvelar todo el juego.

Similarmente pasa con los códigos, un código sería más seguro cuanto menor información revele, ya que si tenemos un mensaje codificado en español(también podríamos utilizar el mismo procedimiento para otros idiomas) con la premisa de que la codificación se hace individualmente carácter por carácter.

**Algoritmo 1** *Utilizaremos el siguiente procedimiento a la hora de decodificar la información,*

1. *Buscaremos una tabla de cuales son las letras con mayor concurrencia en los textos en español.*
2. *En el código interceptado buscaremos en que porcentajes aparecen cada patrón de codificación.*
3. *Finalmente sustituiremos cada patrón por su correspondiente letra, y será con alta probabilidad un texto similar al original, exceptuando errores puntuales.*

## 2.3. Tipos de canales

En este apartado estudiaremos el comportamiento y problemas de la codificación a través del canal. Empezaremos con *canales sin ruido* con los que podremos explicar las bases de la transmisión y una vez explicados dichos conceptos, procederemos a introducir otros factores a tener en cuenta puesto que un canal sin ruido es un tanto idílico en la práctica.

### 2.3.1. Canales sin ruido

En esta subsección hablaremos de las implementaciones de diferentes códigos sin tener en cuenta el ruido que podría causar el canal.

**Códigos instantáneos**

Con tal de buscar códigos de codificación única podemos utilizar dos métodos,

1. Utilizar separadores. *No es muy recomendable puesto que siempre buscaremos el código más compacto posible.*
2. Utilizar palabras que no sean prefijos de otras. *Es una mejor opción que explicaremos a continuación.*

**Definición 11** *Un código será de tipo instantáneo si ninguna palabra es prefijo de otra.*

**Ejemplo.** Utilizando los datos del ejemplo 4.1, podemos ver como al decodificar la palabra 100, leerá los dos primeros caracteres 10 y puesto que hay dos palabras posibles (10 y 100) deberá leer el siguiente carácter para saber a que letras se refiere.

Dado este otro ejemplo,

| | |
|---|---|
| A | 0 |
| B | 10 |
| C | 11 |

Cuadro 2.3: Tabla de codificación de ejemplo I

.

Podemos observar que al recibir la misma palabra 100, en el momento en que el decodificador lea 10 sabrá inmediatamente que se refiere a la letra $B$ y al leer el 0 restante que es la letra $A$.

**Proposición** A partir de esta información fácilmente podremos deducir que todo código instantáneo tiene decodificación única.

Gracias a esta propiedad de los códigos instantáneos podremos utilizarlos para decodificar mensajes en directo, es decir, no necesitaremos tener todo el mensaje para empezar a decodificar. Este tipo de códigos son utilizado en televisión, líneas telefónicas, *streamings*, etc. Opuestamente, la mayoría de codificaciones necesitan recibir todo el mensaje para su decodificación.

**Teoremas de Kraft y McMillan**

La *condición necesaria y suficiente* para que exista un código q-ario instantáneo con $m$ palabras código de longitudes $l_1, ..., l_m$ tal que $l_i \geq 1$ es que verifique,

$$q^{-l_1} + ... + q^{-l_m} \leq 1$$

**Ejemplo.** Veamos si existe una codificación binaria instantánea de los números del 1 al 3 de longitud de cada palabra código 1.

$$2^{-1} + 2^{-1} + 2^{-1} + 2^{-1} = 2 > 1$$

Por tanto, *no existe ningún código instantáneo que satisfaga tales condiciones.*

**Desigualdad de McMillian**

Todo código q-ario de con $m$ palabras de longitud $l_1, ..., l_m$ tal que $l_i \geq 1$ con decodificación única verifica la *desigualdad de Kraft.*

Es decir, que ante un código que satisface las condiciones nombradas anteriormente (la existencia de un código de decodificación única, longitudes de palabra $l_1, ..l_m$ y que cumpla la desigualdad de Kraft), podremos garantizar que entonces existe un código instantáneo que cumplirá dichas condiciones y que será el más óptimo que podremos encontrar.

**Longitud media de un código**

El objetivo que perseguiremos a partir de este punto será hacer el código instantáneo con menor longitud de código posible, para ello utilizaremos la siguiente definición.

**Definición 12** *Llamaremos longitud media de $C$ al resutlado de la siguiente operación,*

$$l(C) = \sum_{i=1}^{m} p_i l_i$$

*por tanto, un mensaje de $n$ simbolos contendrá una media de $nl(C)$ símbolos código.*

**Definición 13** *Dada la fuente de información $F$ y un código instantáneo $q$-ario $C$, denotaremos a la menor longitud media como $l_q(F)$.*

Además, consideremos que un código es *óptimo* si $l(C) = l_q(F)$

Con tal de poder ayudarnos a acercarnos a una mejor cota para nuestro código utilizaremos una serie de proposiciones y teoremas.

**Proposición 3** *Dada la fuente de información $F$ y un código instantáneo $q$-ario $C$, entonces*

$$l(C) \geq \frac{H(F)}{log(q)}$$

**Proposición 4** *Dada la fuente de información $F$ y un código instantáneo binario $C$, este verifica,*

$$l(C) \geq H(F)$$

**Teorema 2** ***Teorema de Shannon para canales sin ruido***

*Sea $F$ cualquier fuente, entonces,*

$$\frac{H(F)}{log(q)} \leq l_q(F) \leq \frac{H(F)}{log(q)} + 1$$

**Corolario 2** *Además toda fuente $F$, verificará*

$$\lim_{k\to\infty} \frac{l_q(F^k)}{k} = \frac{H(F)}{log(q)}$$

**Ejemplo 3** *Con tal de aplicar algunos de estos conceptos definiremos* $A = \{0, 1\}$ *y,*

$$\begin{cases} prob(0) = 0{,}3 \\ prob(1) = 0{,}7 \end{cases} \qquad \begin{cases} c(00) = 100 \\ c(01) = 11 \\ c(10) = 101 \\ c(11) = 0 \end{cases}$$

*Por tanto, la longitud media de este código será de,*

$$\begin{aligned} l(100) = 3 \qquad & prob(00) = prob(0)prob(0) = 0{,}3 * 0{,}3 = 0{,}09 \\ & \to l(100)prob(00) = 3 * 0{,}09 = 0{,}27 \\ l(11) = 2 \qquad & prob(01) = prob(0)prob(1) = 0{,}3 * 0{,}7 = 0{,}21 \\ & \to l(11)prob(01) = 2 * 0{,}21 = 0{,}42 \\ l(101) = 3 \qquad & prob(10) = prob(1)prob(0) = 0{,}7 * 0{,}3 = 0{,}21 \\ & \to l(101)prob(10) = 3 * 0{,}21 = 0{,}63 \\ l(0) = 1 \qquad & prob(11) = prob(1)prob(1) = 0{,}7 * 0{,}7 = 0{,}49 \\ & \to l(0)prob(11) = 1 * 0{,}49 = 0{,}49 \end{aligned}$$

*Es decir, la longitud media de* $C$ *será* $0{,}27 + 0{,}42 + 0{,}63 + 0{,}49 = 2{,}57$.

*Luego supongamos que emite un mensaje de* $n$ *símbolos, el mensaje tendrá una longitud media de* $2{,}57\frac{n}{2} = 1{,}285n$ *símbolos.*

**Nota 6** *Para el calculo de la longitud media de un mensaje hemos multiplicado por la fracción* $\frac{n}{r}$ *(en este caso* $r = 2$*) porque al encontrarnos en* $F^2$ *tomaremos la codificación de los símbolos de 2 en 2 (*$00, 01...$*).*

**Definición 14** *Dado un código q-ario* $C$ *de la fuente* $F$*, llamaremos eficacia de un código* $\eta$*, a la capacidad de comprimir la información, y lo calcularemos de la siguiente manera,*

$$\eta(C) = \frac{H(F)}{log(q)l(C)}$$

*tal que* $\eta(C) \in [0, 1]$

**Nota 7** *Utilizando el ejemplo expuesto anteriormente donde la media de información por palabra es* $2{,}57\frac{n}{2}$*, vamos a suponer la siguiente fórmula general* $l(C)\frac{n}{r}$ *y que el código* $C$ *tiene una* $l(C) = m$*.*

*Podemos observar que a mayor extensión de la fuente, más crecerá la* $r$ $(F^r)$*, y por tanto al aumentar el denominador de la fracción, más reduciremos la longitud media del mensaje, pudiendo aproximar su eficacia altamente a 1.*

*No obstante, a mayor longitud, más difícil será poder manejar los códigos y los codificadores utilizarán un tiempo excesivo si la extensión es demasiado grande, por tanto, siempre nos interesará buscar un equilibrio entre eficacia del código y velocidad de codificación*

### Construcción de códigos óptimos

Con tal de construir códigos lo más óptimos posibles vamos a utilizar el *método de Huffman*, primeramente para códigos binarios y posteriormente daremos una explicación de los códigos óptimos no binarios.

### Códigos óptimos binarios

El planteamiento utilizado a lo largo de esta sección será el siguiente, una fuente de información $F$ con $m$ símbolos $A = \{a_1, ..., a_m\}$ asociados a sus respectivas probabilidades $p_1, ..., p_m$ y $p_1 \geq ... \geq p_m$.

Con tal de construir un código óptimo binario de $F$ utilizaremos el método de Huffman.

**Definición 15** *Definiremos fuente reducida, notado por* $F_{(1)}$ *a la fuente asociada al alfabeto* $A = \{a_1, ..., a_{m-1}\}$*, con las probabilidades* $prob(a_1) = p_1, ..., prob(a_{m-2}) = p_{m-2}, prob(a_{m-1}a_m) = p_{m-1} + p_m$ *y dado* $C_{(1)}$ *un código óptimo, entonces, si definimos* $C$ *como,*

$$\begin{gathered} c(a_1) = c_{(1)(a_1)} \\ ... \\ c(a_{m-2}) = c_{(1)(a_{m-2})} \\ c(a_{m-1}) = c_{(1)(a_{m-1})0} \\ c(a_m) = c_{(1)(a_{m-1})1} \end{gathered}$$

entonces $C$ también será un código óptimo.

**Teorema 3** *Si $C_{(1)}$ es un código óptimo para $F_{(1)}$, entonces $C$ también lo es para la fuente $F$.*

**Lema 2**

$$l(C) = l(C_{(1)}) + p_{m-1} + p_m$$

**Lema 3** *Sea $C'$ un código óptimo con longitudes de palabra $l_1, ..., l_m$ y $p_1 \geq ... \geq p_m$, entonces $l_1 \leq ... \leq l_m$.*

**Lema 4** *Existe un código óptimo tal que la codificación de $a_{m-1} y a_m$ solo se diferencian en el último bit.*

#### Códigos óptimos no binarios

Para la construcción de códigos óptimos no binarios utilizaremos una adaptación del mismo procedimiento, es decir, en cara reducción de código utilizaremos los $q$ códigos menos probables.

Puesto que en cada reducción el código perderá en $q-1$ el número de elementos de la fuente, esta solo será posible ante una cardinalidad de código $m : q - 1|m$

Con tal de poder encajar esta reducción dentro de un código de longitud $m$, en la primera reducción que hagamos, y por tanto, la menos importante a priori, solamente colapsaremos los $d$ primeros elementos tal que $q - 1|m - d$, y de esta manera las cuentas nos cuadran.

### 2.3.2. Canales con ruido

La situación a partir de este punto tendrá en cuenta un *canal* que distorsionará la información, de manera que el mensaje enviado no será igual al recibido y deberemos tratar de encontrar los errores y corregirlos para leer el mensaje. Los códigos utilizados para este fin son llamados *Códigos correctores de Errores.*

#### Tipos de errores

A partir del tipo de alteraciones del mensaje distinguiremos entre,

- *errores*, los símbolos que recibiremos serán diferentes a los enviados.

- *borrones*, símbolos ilegibles que no podremos interpretar pero sí detectarlos.

Representaremos como $\sqcup$ los errores cuya posición es conocida.

Por otra parte, dependiendo de la periodicidad de los errores encontraremos,

- errores/borrones *aleatorios*, aparecen de forma aislada y siguen un patrón aleatorio a lo largo del mensaje.
- errores/borrones *a ráfagas*, aparecen en posiciones consecutivas en ciertas partes del mensaje recibido.

### Descripción de un canal con ruido

**Notación.** Notaremos los diferentes elementos que vamos a utilizar como,

- Conjunto de los símbolos de entrada $\rightarrow A = a_1, ..., a_m$
- Conjunto de los símbolos de salida $\rightarrow S = s_1, ..., s_h$
- La probabilidad condicionada de que dado un símbolo $s_i$ de salida este se corresponda con $a_j$ tal que $j = 1, ..., hj = 1, ..., m \rightarrow prob(s_i|a_j)$

Si no encontramos borrones en el código recibido, entonces $S = A$ $(s_i = a_i : i \in \{1, ..., m\})$, por contra, si sí los hay $S = A \cup \{\sqcup\}$ $(s_i = a_i : i \in \{1, ..., m\}, s_{m+1} = \sqcup)$

El modelo de canal que seguiremos será *discreto*, puesto que emitiremos una señal por unidad de tiempo y *sin memoria*, ya que cada señal es independiente de la siguiente.

También podremos calcular probabilidad de cada símbolo del alfabeto de salida,

$$prob(s_i) = \sum_{j=1}^{m} prob(a_j) prob(s_i|a_j)$$

### Capacidad de un canal

Al encontrarnos ante un canal con ruido, somos conscientes de que cada símbolo enviado puede perder información. Una alta pérdida de información podría dar lugar a un canal de imposible comunicación.

Es por ello que nos interesará medir la *capacidad del canal*, es decir, la probabilidad de que introduciendo un símbolo $a_j$ obtengamos el símbolo $s_i$, con tal de calcular la calidad de un canal.

Por tanto, aplicando las *leyes de Bayes*,

$$prob(a_j \rightsquigarrow s_i) = prob(a_j)prob(s_i|a_j) = prob(s_i)prob(a_j|s_i)$$

Despejando $prob(a_j|s_i) \rightarrow prob(a_j|s_i) = \frac{prob(a_j)prob(s_i|a_j)}{prob(s_i)}$

Ahora trataremos de adaptar estos nuevos conceptos a los ya definidos en los canales sin ruido, acercándose más a la realidad ya que tendremos en cuenta los errores que el canal puede causar.

Por ejemplo, aplicando el concepto de entropía, obtenemos la fórmula,

$$H(A|s_i) = \sum_{j=1}^{m} prob(a_j|s_i)log(\frac{1}{prob(a_j|s_i)})$$

**Definición 16** *Definiremos la entropía de la entrada A condicionada por la salida a,*

$$H(A \mid S) = \sum_{i=1}^{m+1}\sum_{j=1}^{m} prob(a_j \rightsquigarrow s_i)log(\frac{1}{prob(a_j|s_i)}) \qquad (bits/símbolo)$$

**Definición 17** *También podremos calcular aproximadamente la cantidad de información que perdemos en cada transmisión a causa del canal restando la cantidad media de información que deberíamos recibir ($I(A)$) con la realmente recibida $I(A|S)$.*

**Ejemplo 4** *Con tal de ver con mayor claridad el uso de este último concepto supongamos que la cantidad media de información que enviamos es $H(A) = 1$ y de la información recibida es $H(A|S) = 0{,}25$.*

*Utilizando la fórmula definida anteriormente, $I(A|S) = 1 - 0{,}25 = 0{,}75$, es decir, la cantidad media de información que que recibiremos perderá un 75 % respecto de la original.*

**Definición 18** *Definiremos la capacidad del canal C a,*

$$C = \begin{array}{c} max \quad I(A|S) \\ p_1, ..., p_m \in [0,1] \\ p_1 + ... + p_m = 1 \end{array}$$

**Canales simétricos**

**Definición 19** *Llamaremos canal simétrico a un canal cuya* $p_{ij} = prob(s_i|a_j)$ *y tanto las filas como sus columnas de la matriz* $(p_{ij})$ *tienen los mismos valores pero en distinto orden.*

**Ejemplo 5** *Pondremos a continuación un ejemplo de matriz simétrica* $2x2$*,*

$$\begin{pmatrix} 1\text{-}p & p \\ p & 1\text{-}p \end{pmatrix} \tag{2.1}$$

**Proposición 5** *Dado un canal simétrico,*

1. $\exists h$ *constante,* $: H(S|a_j) = h \forall j \in \{1, ..., m\}$
2. $C = log(\#S) - h$

Y, a partir de la anterior proposición podemos deducir que la capacidad de un canal simétrico binario es

$$C = 1 - H(p) = 1 + plog(p) + (1-p)log(1-p)$$

**Corrección de Errores**

Las bases en la codificación para la corregir errores se basan en utilizar códigos en bloque con redundancia que nos ayudaran a posteriori a encontrar errores en el mensaje recibido.

**Ejemplo 6** *Con tal de entender de una manera muy sencilla el uso de la redundancia de los códigos, supongamos que tenemos la siguiente tabla,*

| | |
|---|---|
| A | 00 |
| B | 10 |
| C | 11 |

Cuadro 2.4: Tabla de codificación de ejemplo I

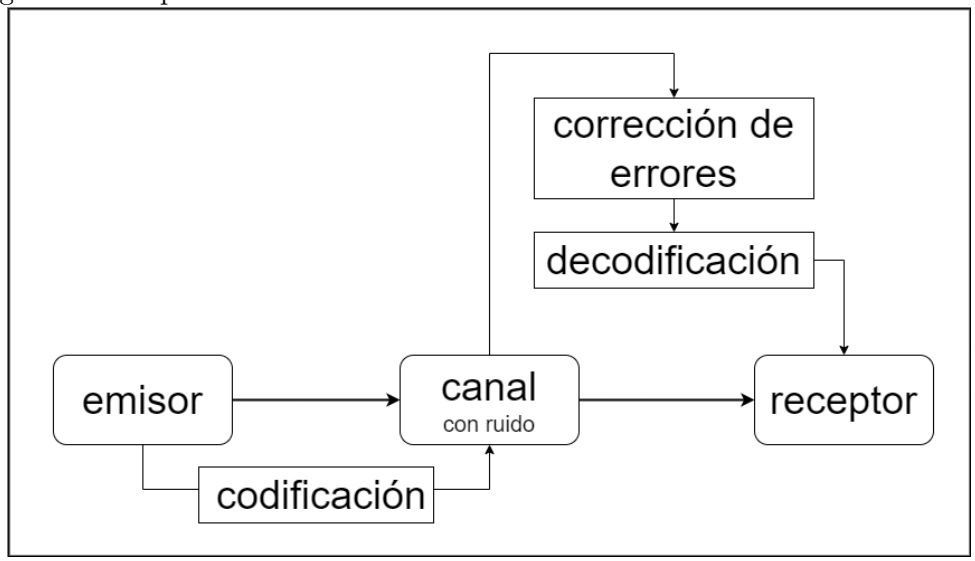
.

*Y vamos a poner una redundancia, tal que cada bit lo repetiremos 3 veces, de esta manera, si queremos enviar el mensaje 0010 (AB), deberemos enviar 000 000 111 000.*

*Ahora imaginemos que por errores del canal recibimos el mensaje, 100 000 101 001. Si sabemos que la probabilidad de error es menor del* 50 % *en este canal, entonces sabremos corregir los errores con facilidad.*

Por tanto, el nuevo esquema de la transmisión de información nos quedará de la siguiente manera 2.4.

Figura 2.4: Esquema de transmisión de la información con corrección de errores




.

**Nota 8** *Cabe recalcar que la parte más costosa de la decodificación suele ser la de corrección de errores.*

**Códigos de bloque**

Dado un alfabeto código $B$ y su subconjunto, el código de bloque $B^n$, definiremos sus elementos en forma de notación vectorial $x = (b_1, ..., b_n) : b_i \in B$.

**Nota 9** *En caso de no encontrar confusiones de notación* $x = (b_1, ..., b_n) \to x = b_1...b_n$.

La *teoría de códigos correctores de errores* busca la manera de encontrar códigos con capacidad de corregir el mayor número de errores posibles de la manera más compacta posible, y, buscando un equilibrio entre ambos.

**Distancia de Hamming (Distancia mínima)**

Partiendo el ejemplo mostrado anteriormente, este tipo de códigos correctores se llaman *de repetición*. La propiedad que se busca en estos tipos de algoritmos es que cada palabra será lo más *diferente* posible a las demás, ya que de esta manera siempre podremos tratar de aproximar cada código alterado en el código más similar a esta.

Este planteamiento produce una necesidad de medir *cuan diferente son las palabras del código* entre ellas, para ello utilizaremos la distancia de Hamming.

**Definición 20** *Dado* $x, y \in B^n, x = (x_1, ..., x_n), y = (y_1, ..., y_n)$*, llamamos distancia de Hamming entre* $x$ *e* $y$ *a,*

$$d(x, y) = \#\{i \mid 1 \leq i \leq n : x_i \neq y_i\}$$

*Además esta aplicación cumple las propiedades de las distancias,*

1. *No negativa,* $d \geq 0$ *y* $d = 0 \iff x = y$
2. *Simetría,* $d(x, y) = d(y, x)$
3. *Desigualdad triangular,* $d(x, y) + d(y, z) \geq d(x, z)$

**Definición 21** *Calcularemos la distancia mínima del código* $C$*,*

$$d(C) = d = min\{d(x, y) \mid x, y \in C : x \neq y\} \qquad 1 \leq d(C) \leq n$$

Además podremos calcular la distancia mínima relativa como,

$$\delta(d) = \frac{d(C)}{n} \qquad 0 < d(C) \leq 1$$

**Decodificación**

Tras la emisión de una palabra $c \in C$, recibiremos un vector de $n$ símbolos (algunos de estos pueden ser $\sqcup$) al que notaremos como $y$.

A la hora de decodificar seguiremos el siguiente *algoritmo*,

1. Calcularemos la *distancia mínima* de cada palabra $y$ recibida a las contenidas en el código $C$.
2. Convertiremos cada $y$ a su $c \in C$ con menos distancia y decodificaremos el mensaje.

**Nota 10** *En caso de que varias distancias mínimas coincidan el algoritmo fallará. Es por ello que a la hora de crear una buena codificación las palabras deben ser lo más distintas posibles entre ellas.*

**Proposición 6** *Dado un código en bloque con distancia mínima $d$ y longitud de bloque $n$,*

- *Podremos* ***detectar*** *$t$* ***errores****, siempre y cuando $t < d$*
- *Podremos* ***corregir*** *$t$* ***errores****, siempre y cuando $2t < d$*
- *Podremos* ***corregir*** *$s$* ***borrones****, siempre y cuando $s < d$*
- *Podremos* ***corregir*** *$t$* ***errores*** *y $s$* ***borrones****, siempre y cuando $2t + s < d$*

Por tanto, para saber cual es la capacidad de corrección de un código C aplicaremos la siguiente definición,

**Definición 22** *sea $d$ la distancia mínima de $C$, entonces dicho código corrige $[\frac{d-1}{2}]$ errores, por tanto lo llamaremos, código $[\frac{d-1}{2}] - corrector$.*

**Tasa de transmisión de información**

Dado un código $C$ con longitud $n$ en el alfabeto $B$ de $q$ símbolos y aplicando un poco de combinatoria básica, podemos deducir que existen como máximo $q^n$ combinaciones.

Por tanto, un bloque de $m$ palabras tendrá una longitud de al menos $log_q(m)$ (cantidad de palabras que podremos formar con ese bloque) y los restantes, podemos considerarlos como información de control.

**Definición 23** *Dado un código $C$ de longitud $n$ y un alfabeto de $m$ palabras con $q$ elementos, llamaremos tasa de transmisión de información de $C$ a,*

$$R(C) = \frac{log_q(m)}{n}$$

**Teorema 4** ***Teorema de Shannon***

*Con tal de poder exponer el Teorema de Shannon, supondremos un error de $p < 0{,}5$ y un código binario $C$ de longitud $n$ con $m$ palabras.*

**Definición 24** *Llamaremos probabilidad de error $prob_{err}(C)$ a la probabilidad media de que una palabra llegue de forma errónea al receptor. Para ello utilizaremos la siguiente fórmula,*

$$prob_{err}(C) = \frac{1}{m} \sum_{c \in C} prob(error\ en\ la\ transmisión\ de\ c)$$

Ahora supongamos que $C$ es capaz de corregir una $n - upla$ siempre que hayan menos de $t$ errores, entonces, la probabilidad de que el mensaje recibido sea erroneo para el método de decodificación por mínima distancia es,

$$prob(\leq t\ errores) = \sum_{i=1}^{t} \binom{n}{i} p^i (1-p)^{n-i}$$

Por tanto,

$$prob_{err}(C) \leq 1 - (\sum_{i=1}^{t} \binom{n}{i} p^i (1-p)^{n-i})$$

**Teorema 5** ***Teorema de Shannon para canales con ruido***

*Dado un canal simétrico binario de capacidad* $C > 0$, $\forall \epsilon \in \mathbb{R} : \epsilon > 0$, $\exists C_n$ *código de bloque de longitud* $n$ *tal que* $prob_{err}(C_n) < \epsilon$ *y* $R(C_n) \geq C - \epsilon$.

*En particular,* $\exists (C_n)_{n=1}^{\infty}$, *tal que,*

$$\lim_{n\to\infty} prob_{err}(C_n) = 0 \qquad y \qquad \lim_{n\to\infty} R(C_n) = C$$

Este teorema no deja de poderse interpretar como su parejo en los canales con ruido, es decir, podemos aumentar la longitud del código hasta casi la capacidad completa del canal con un error arbitrariamente bajo.

Este teorema es una de las bases más importantes sobre las que sustentar la teoría de la información. Aun que el teorema nos indica como deberán ser los códigos para optimizarlos al máximo, no nos muestra la forma en que codificaremos la información, y es a partir de aquí cuando se nos abre un amplio abanico métodos que tratarán de ajustarse al máximo a la cota de Shannon.

**Nota 11** *Con tal de tratar los alfabetos como cuerpos finitos y así poder aplicar disciplinas sobre la que tenemos mayor control teoría de números y álgebra, precisaremos restricciones como,*

- *El alfabeto fuente y el alfabeto código debe coincidir.*
- *El cardinal del alfabeto* $q$ *debe ser potencia de un número primo.*

# Capítulo 3

# Estructuras Lineales

Junto a la creación de los códigos de bloque surgió un nuevo problema, los computacionalmente altos costos de la codificación y decodificación de estos, junto a la gran capacidad de almacenamiento que estos requerían. Para la solución de dichos problemas recurrieron a estructuras algebraicas.

**Estructura de los códigos lineales**

*Todos las preposiciones, lemas, teoremas enunciados en este apartado se encuentran enunciados y demostrados en el libro*[3].

**Definición 25** *Llamaremos código lineal de longitud $n$ sobre $F_q$ al subespacio vectorial $F_q^n$.*

**Definición 26** *Dado un código lineal $C$ de longitud $n$ cuyo espacio vectorial es de dimensión $k$ y con distancia mínima $d$ definiremos parámetros fundamentales de $C$ los parámetros $n$, $k$ y $d$ y, por tanto, diremos que es de tipo $[n, k, d]$ o $[n, k]$.*

Para calcular la redundancia $r$ la calcularemos con la fórmula $r = n - k$

Sea $C$ de tipo $[n, k, d]$, definiremos la tasa de transmisión de un código lineal como,

$$R(C) = \frac{k}{n}$$

Con tal de codificar la información a través de estructuras lineales, utilizaremos una aplica-

ción $f$. Esta aplicación será lineal, inyectiva e irá del espacio $F_q^k$ al espacio $F_q^n$ tal que $q$ es el *sistema de numeración*, $k$ es la dimensión de los dígitos a codificar y $n$ la codificación incluyendo su redundancia.

$$f : F_q^k \to C \subset F_q^n$$

Como bien sabemos, para realizar la transformación de la aplicación podemos definir una matriz $kxn$.

**Definición 27** *Definimos matriz generatriz de $C$ a la matriz de la aplicación inyectiva $f : F_q^k \to C \subset F_q^n$ $kxn$ cuyas filas son una base de $C$.*

**Nota 12** *La base de un código $C$ no es única pero todas son semejantes. Para pasar de una base a su semejante existe una matriz inversible $P : G_1 = PG_2P^{-1}$ $kxn$.*

A partir de estos conceptos deducimos que ahora $C = \{aG|a \in F_q^k\}$ y $a \in F_q^k$ y $aG \in F_q^n$ de esta manera solucionaremos el problema de espacio a la hora de almacenar la codificación ya que tan solo tendremos que guardar la matriz de conversión $G$ ($nk$ elementos) en vez de $nq^k$ elementos como es el caso de los códigos en bloque no lineales.

**Ejemplo 7** *Con tal de entender bien estos conceptos introduciremos el siguiente ejemplo,*

$$G = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} \tag{3.1}$$

*Como podemos comprobar este código binario ($q = 2$) tendrá longitud 3 y dimensión 2, por tanto encontraremos $q^k = 2^2 = 4$ que serán $100, 111, 000, 011$ cuya distancia mínima $d$ será 1, por tanto, el mensaje $01$ lo codificaremos de la siguiente manera,*

$$\begin{pmatrix} 01 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 111 \end{pmatrix} \tag{3.2}$$

**Codificación sistemática**

A veces, con tal de agilizar el proceso de decodificación trataremos que la primera parte la codificación sea la palabra sin codificar y el final los símbolos de control.

Para ello utilizaremos una matriz generatriz de la forma $\left(\mathrm{I}_k, C\ \right)$ tal que $I_k$ será una submatriz identidad $kxk$ y $C$ la codificación de control obteniendo así para la palabra de $a \in F_q^k$ la codificación $\left(\mathrm{a},\ \mathrm{z}\right)$.

**Definición 28** *Dados dos códigos $C_1, C_2$ con longitud $n$ sobre $F^q$ serán equivalentes si podemos encontrar una permutación $\sigma$ de $\{1, ..., n\}$ tal que $C_2 = \{\sigma(c) \mid c \in C_1\}$*

**Nota 13** *Cuando hablamos de $\sigma(x)$ realmente estaremos hablando de $(x_{\sigma(1)}, ..., x_{\sigma(n)})$.*

De echo, pese a que hayan diferentes permutaciones entre estos dos códigos, no todos tienen porqué generar códigos distintos, de echo podemos dividirlos en varios subgrupos, $S_n$ con

$$Aut(C) = \{\sigma \in S_n \mid \sigma(C) = C\}$$

**Proposición 7** *Todo equipo es equivalente a uno sistemático.*

**Ejemplo 8** *A continuación vamos a mostrar un ejemplo de código sistemático.*

*Tenemos por ejemplo la matriz,*

$$G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} \tag{3.3}$$

*Podemos ver como las columnas 1 y 2 son linealmente dependientes, por ello vamos a hacer la permutación de la columna 2 con la 3, y la matriz queda de la siguiente manera,*

$$G' = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \tag{3.4}$$

*Una vez obtenidas las primeras k filas independientes, haremos una serie de operaciones elementales con tal de estandarizar la matriz,*

$$G' = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} \rightarrow^{f2=f2+f1} \rightarrow \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{3.5}$$

**Matriz de control**

También podemos definir el subespacio vectorial $F_q^n$ en el que nos encontramos como un sistema de ecuaciones implícitas.

**Definición 29** *Dado el vector $x \in F_q^n$ llamaremos matriz de control $H$ de $C$ a la matriz que verifica,*

$$Hx^t = 0$$

**Nota 14** *Si $C$ es un código de tipo $[n, k]$ entonces $H$ tiene tamaño $(n - kxn)$ y rango $n - k$.*

**Proposición 8** *Sean las matrices $G$ generatriz y $H$ de control, entonces $GH^T = 0$.*

**Nota 15** *Dada una matriz generatriz $G = (I_k, C) \Rightarrow H = (-C^t, I_{n-k})$, por tanto, una matriz de control será de la forma estándar sí es $(B, I_{n-k})$.*

**Definición 30** *Dado $x = (x_1, ..., x_n) \in F_q^n$ llamamos soporte de $x$ a,*

$$sop(x) = \{i \mid 1 \leq i \leq n : x_i \neq 0\}$$

*Llamaremos peso de Hamming de $x$ a*

$$w(x) = \#sop(x) = d(x, 0)$$

Podemos observar que estamos definiendo la norma de $F_q^n$ como $w$ con la distancia asociada a esta $d$.

**Ejemplo 9** *Para que quede claro una breve aplicación de este concepto.*

*Supongamos que tenemos un dos codificaciones* $c_1, c_2 \in C$ *tal que* $c_1 = (10010)$*, que* $c_2 = (11011)$ *y* $x = c_2 - c_1 = (01001)$*.*

*Aplicando la definición de soporte, buscaremos en que posiciones* $x_i \neq 0$*, en este caso será en la posición* 2 *y* 5*, por tanto,* $sop(x) = \{2, 5\}$*.*

*Ahora apliquemos la definición de peso* $w(x)$*, es decir, cuantas posiciones hemos encontrado al calcular el soporte.* $\#sop(x) = w(x) = 2$*. Por tanto el peso es* 2 *y tal y como podemos observar son la cantidad de dígitos que difieren* $c_1$ *de* $c_2$*, Tras esta observación vamos a mostrar una definición y un lema que refleja este resultado.*

**Definición 31** *Podemos definir la distancia mínima del código* $C$ *como,*

$$w(C) = min\{w(c) \mid c \in C, c \neq 0\}$$

**Lema 5** *En un código lineal, la distancia mínima es igual al peso mínimo.*

*La siguiente proposición ha sido sacada del artículo Introducción a la teoría de códigos [6].*

**Proposición 9** *Dado un* $(n, k)-$*código lineal y* $H$*, una matriz de control. La distancia mínima será de al menos* $d > r$ *sí, y solo sí dadas cualesquiera* $r$ *columnas de* $H$ *estas son independientes entre ellas* $r$ *a* $r$*.*

**Corolario 3** *Dado el código* $C$ *cuya matriz de control es* $H$*,*

$$d(C) = min\{\#\{ \text{ columnas linealmente dependientes de } H\}\}$$

**Ejemplo 10** *Dada la siguiente matriz de control* $H$ *vamos a tratar de calcular la distancia mínima del código asociado.*

$$H = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \tag{3.6}$$

*Puesto que es una matriz de dimensión 3, el número máximo posible de columnas linealmente independientes será 3.*

*Como las tres primeras columnas son la base* $< (100), (010), (001) >$*, vemos que estas 3 columnas serán independientes. Ahora, recurriendo a la proposición previamente anunciada, tenemos* $d - 1 = 3 \rightarrow d = 4$*, con 4 columnas linealmente dependientes.*

*Por tanto la distancia mínima será de al menos 4.*

**Dualidad**

Podemos interpretar la matriz de control $H$ del código $C$ como una matriz generatriz. El código que esta genera lo llamaremos *código dual* y lo denotaremos por $C^{\perp}$.

Además como $GH^t = 0 \Rightarrow HG^t = 0$, por tanto, $G$ será una matriz de control para $C^{\perp}$.

**Proposición.** Dado $C$ un código lineal, y $C^{\perp}$ su dual, entonces $C$ y $C^{\perp}$ son perpendiculares.

**Nota.** Puesto que la forma bilineal es simétrica y no degenerada,

$$(C^{\perp})^{\perp} = C$$

Por otra parte, la intersección entre un código y su dual no tiene porqué ser el conjunto vacío, tal y como podemos mostrar en el siguiente ejemplo.

**Ejemplo 11** *Dado* $(1,1) \in F_2^2 \rightarrow (1,1) \times (1,1) = (0,0) = 0 \Rightarrow (1,1) \in C$ *y* $(1,1) \in C^{\perp}$.

*Por tanto, a veces,* $C \cap C^{\perp} \neq 0$*, e incluso puede llegar a darse que* $C = C^{\perp}$*, pero, puesto que* $k = n - k \rightarrow n = 2k \Rightarrow$ *la longitud de* $C$ *deberá ser par.*

**Definición 32** *Llamaremos código autodual al código* $C$ *tal que,*

$$C = C^{\perp}$$

**Polinomios de pesos**

Con tal de calcular los pesos de cada palabra de $C$ consideraremos,

$$a_i(C) = a_i = \#\{c \in C \mid w(C) = i\}$$

**Nota 16** *Podemos deducir que* $a_0 = 1 \quad \forall i \in \{1, ..., d\}$ *y que* $\sum_{i=0}^{n} a_i = q^k$.

**Definición 33** *Llamaremos polinomio de pesos de* $C$ *a,*

$$W(X) = \sum_{i=0}^{n} a_i X^i = \sum_{c \in C} X^{w(c)}$$

*es decir,* $W(X)$ *será un polinomio donde el coeficiente será la cantidad de códigos cuyo peso viene marcado el grado de su respectivo monomio.*

No obstante, algunas veces nos interesará utilizar dicho polinomio homogeneizado,

$$W(X, Y) = \sum_{i=0}^{n} a_i X^i Y^{n-i}$$

**Ejemplo 12** *Supongamos que tenemos un código* $C$ *con los siguientes parámetros,*

$$\begin{aligned} a_0 &= 1 \\ a_1 &= 0 \\ a_2 &= 3 \\ a_3 &= 0 \\ a_4 &= 5 \end{aligned}$$

*Por tanto, nuestro polinomio de pesos será de la siguiente forma,*

$$W(X) = 1 + 3X^2 + 5X^4$$

**Definición 34** *Podremos relacionar un polinomio de pesos con el de su dual a través de la identidad de Mac Williams donde dado el código lineal* $C \subset F_q$ *de tipo* $[n, k]$ *y su respectivo dual* $C^{\perp}$*,*

$$W^{\perp}(X,Y) = q^{-k}W(Y-X, Y+(q-1)X)$$

**Decodificación de los códigos lineales**

Tal y como explicamos en la sección anterior, sabemos que dado un código lineal $C$ de tipo $[n,k,d]$ sobre $F_q$ seremos capaces de corregir, $t = [\frac{d-1}{n}]$ errores.

Recibido un vector $y \in F_q^n$ querremos decodificar la información recibida con tal de obtener $c \in C$. A partir del mensaje original codificado $c$ y el recibido $y$ podremos ver cual es el error real, $e = y - c$.

**Algoritmo de decodificación**

Con tal de decidificar de la información simplemente aplicaremos los siguientes pasos,

1. Calcularemos la distancia de $y$ a todas las palabras de $C$.
2. Decodificaremos por la distancia más corta a ser posible.

No obstante, pueden sucedernos tres casos,

- *Decodificación correcta.* Si $w(e) \leq t \rightarrow d(e,y) = w(e) \leq t \Rightarrow$ gracias a esto encontraremos su $c$ correspondiente.
- *Se producen errores.* Si $t \leq w(e) < d \Rightarrow$ detectaremos los errores pero no corregirlos.
- *Decodificación falla.* Si $w(e) \geq d$ la decodificación fallará.

**Síndrome**

Con tal de conocer la mayor información posible sobre el error recurriremos al síndrome.

**Definición 35** *Como ya sabemos, $y = c + e$, por tanto, dado una matriz de control $H$ sobre $C$, llamaremos síndrome de $y$ al vector,*

$$s(y) = Hy^t \in F_q^{n-k}$$

$y \in C \iff s(y) = 0 \Rightarrow s(y) = s(c+e) = s(c) + s(e) = Hc^t + s(e) \rightarrow$ por definición $Hc^t = 0 \rightarrow s(y) = s(e)$, por tanto, recibimos $y$ e inmediatamente conoceremos el síndrome de su respectivo error.

**Proposición 10** *El síndrome del vector recibido y es una combinación lineal de las columnas de H correspondientes a las posiciones con errores.*

**Ejemplo 13** *Con tal de entender bien el uso del síndrome vamos a ilustrarlo con el siguiente ejemplo.*

*Supongamos que tenemos la matriz de control H tal que,*

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{3.7}$$

*Hemos enviado el mensaje* $c = 0010110$ *e imaginemos que por un error recibimos el mensaje* $y = 0011110$*, con tal de conocer la posición en la que ha ocurrido el error calcularemos su síndrome,*

$$H(0011110)^t = (100)^t$$

*Como* 100 *es el número 4 en binario, sabemos que el error se encuentra en dicha posición y lo corregiremos tal que nos quedará 001**0**110.*

**Clases**

Dada la *relación de equivalencia* $u \sim v \iff u - v \in C$. Y sea $\frac{F_q^n}{C}$ el espacio vectorial cociente, los elementos de dicho espacio serán clases de equivalencia tal que $u + C = \{u + x \mid x \in C\}$.

**Nota 17** *$u, v$ estarán en la misma clase $\iff$ u-v $\in C \iff s(u) = s(v)$, por tanto al recibir y y conocer su síndrome $s(y)$, también conoceremos su clase.*

**Definición 36** *Llamaremos líder de una clase al elemento de peso único y mínimo de esta, sí existe.*

**Proposición 11** *Cada clase de* $\frac{F_n^q}{C}$ *posee como máximo un elemento de peso* $\leq t$

**Algoritrmo del líder**

Una ver recibido $y$, lo decodificaremos a través de este algoritmo utilizando el siguiente proceso,

1. Calcularemos $s(y)$ y la buscaremos en la columna de síndromes.
2. *Clase sin líder*, la decodificación falla.
3. *Clase con líder*, $e \to e$ será el error cometido $\to$ la palabra decodificada es $y - e$

**Códigos construidos a partir de otros**

A continuación vamos a exponer un par de ejemplos de códigos construidos a partir de otros, de esta manera podremos mejorar sus parámetros para utilizarlos en distintas aplicaciones,

- **Códigos extendidos**

  Los códigos extendidos $\bar{C}$ se forman a partir de otro código $C$, es decir,

  $$\bar{C} = \{(c_1, ..., c_n, c_{n+1}) \mid (c_1, ..., c_n) \in C : c_1 + ... + c_n + c_{n+1} = 0$$

  La matriz de control entonces será de la forma,

  $$\bar{H} = \left(\begin{array}{c|c} & 0 \\ H & \vdots \\ & 0 \\ \hline 1 \dots 1 & 1 \end{array}\right)$$

  Por tanto, los parámetros de $\bar{C}$ de reajustarán de la siguiente manera,
  $\bar{n} = n + 1$
  $\bar{k} = k$
  $\bar{d} = d$ (par) ò $d + 1$ (impar)

- **Códigos recortados**

  Para la implementación del *códigos recortados* de $C$, $C^*$, se obtiene opuestamente a la codificación anterior, es decir, *'recortando'* la j-ésima posición y utilizando el mismo proceso que en los códigos extendidos pero a la inversa podremos implementarlo.

Por otra parte los parámetros se recalcularán de la siguiente manera,
$n_j^* = n - 1$
$k_j^* = k$ (par) ò $k - 1$ (impar)
$d_j^* \geq d$

## 3.1. Código de Hamming

Uno de los códigos más conocidos, al menos en la teoría, ya que hoy en día no son tan utilizados en la práctica son los *códigos de Hamming*. Estos presentan una elegante estructura algebraica y sirven de introducción para el estudio de otros códigos lineales.

**Códigos de Hamming binarios**

Comenzaremos a explicar las bases de este algoritmo sobre un código binario debido a su mayor simplicidad.

*Imaginemos que nos piden construir sobre* $F_2$ *un algoritmo capaz de corregir un error y la mayor cantidad de palabras posibles en relación a su longitud.*

Tal y como hemos estudiado, para conseguir un código que solucione 1 error, necesitaremos una distancia entre cada palabra de al menos 3 bits, es decir, cualesquiera 2 columnas de la matriz de control $H$ han de ser linealmente independientes (utilizando el corolario nombrado en el capítulo anterior).

**Definición 37** *Dado* $r \geq 2$*, llamaremos código de Hamming binario cuya redundancia es* $r$*,* $\mathcal{H}(r)$*, al código cuya matriz de control tiene por columnas vectores no nulos de* $F_2^r$*.*

Por tanto, para calcular los parámetros de $\mathcal{H}(r)$, lo haremos de la siguiente manera,

*Longitud* $\rightarrow n = \#F_2^r - 1 = 2^r - 1$ *Dimensión* $\rightarrow k = n - r = 2^r - r - 1$

**Ejemplo 14** $\mathcal{H}(2)$ *es de tipo* $[3, 1]$*, tiene como matriz generatriz* $G = (111)$ *y matriz de control,*

$$H = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix} \tag{3.8}$$

**Proposición 12** *La distancia mínima de un código de Hamming binario es* 3*.*

**Polinomios de peso de $\mathcal{H}_2(r)$**

**Proposición 13** *Dado $\mathcal{H}_2(r)$, de tipo $[2^r-1, 2^r-r-1]$ y $C=\mathcal{H}_2(r)^{\perp}$ su dual,*

$$W_c(X) = 1 + (2^r-1)X^{2^{r-1}}$$

**Decodificación de Hamming de códigos binarios**

La decodificación del código de Hamming binaria será capaz de corregir un error, ya que su distancia mínima es 3.

Gracias a las estructuras de control utilizadas, no tendremos necesidad de utilizar una tabla con todos los síndromes y líderes anotados, nos bastará con el cálculo de su síndrome.

Por tanto, recibido un vector $y$, sí $y \in C$, entonces, el mensaje recibido será correcto. Sí $y \notin C$ y contiene un error, calculamos su síndrome $s(y)$ que coincidirá con una de sus columnas de control. Este resultado binario lo transformaremos en uno decimal y corregiremos la posición indicada.

En caso de tener más de un error la transmisión habrá sido errónea.

**Decodificación de Hamming de códigos no binarios**

Para la decodificación de cualquier código $F_q$, puesto que cada vector de $F_q^r$ contiene $q-1$ múltiplos diferentes de 0, podemos definir diferentes clases $\frac{(q^r-1)}{(q-1)}$, partiendo los $(q^r-1)$ vectores de $F_q^r$.

El tipo de esta matriz será $[\frac{q^r-1)}{(q-1)}, \frac{q^r-1)}{(q-1)} - r]$ cuya redundancia de este código $q-ario$ será $r$.

**Ejemplo 15** *Imaginemos que tenemos un código de Hamming $\mathcal{H}_3(3)$ cuya matriz de control es,*

$$H = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 2 & 2 & 2 \\ 1 & 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 & 0 & 1 & 2 \end{pmatrix} \tag{3.9}$$

*Enviado el mensaje $c = 0120120120000$ y recibido el mensaje $x = 0120120122000$ obtenemos el síndrome, $s(y) = (221)^t = 2(112)^t$. Puesto que en la matriz de control la columna $(112)^t$ se*

*encuentra en la posición* 10*, sabemos que el error ha sucedido en dicha posición y el error su valor es* 2.

## 3.2. Código de Golay

Los códigos de Golay están formados por 4 grandes familias, dos son *binarias* $\mathcal{G}_{24}$ y $\mathcal{G}_{23}$ y las otras dos *ternarias* $\mathcal{G}_{12}$ y $\mathcal{G}_{11}$[7].

**Códigos autoduales**

**Definición 38** *Dada una matriz A kxk sobre el cuerpo* $F_q$*, diremos que A es una matriz conferencia si* $AA^t = (q-1)I_k$.

para explicar el uso de este nuevo concepto supongamos que tenemos un código $C$ generado a partir de la matriz generatriz de la forma $G = (I_k, A)$.

Por tanto $C$ será un código $[2k, k]$ tal que $GG^t = 0$ y de aquí concluimos que dada la igualdad de dimensiones, $C$ y $C^\perp \to C$ será *autodual*.

De la misma manera, dado un código autodual, podemos encontrar su matriz conferencia transformando la matriz generatriz a la forma estándar.

**Proposición.** Dado una matriz conferencia $A$ y una matriz generatriz $G = (I_k, A)$ que genera el código autodual $C$.

Si $A$ es simétrica se verifica que $\forall a, b \in F_q^k : (a, b) \in C$, $(-b, a) \in C$

*Tal y como hemos avanzado en la introducción a esta sección, tan solo nos vamos a interesar por los cuerpos* $F_2$ *y* $F_3$.

**Proposición 14** *Dado un código autodual C sobre* $F_2$*, todas sus palabras tendrán un peso múltiplo de 2.*

**Nota 18** *La proposición anterior también puede extrapolarse al caso* $F_3$ *donde todos sus pesos serán múltiplos de 3.*

**Códigos de Golay binarios**

Llamaremos *código de Golay binario* $\mathcal{G}_{24}$, al código cuya matriz generatriz es la siguiente,

$$G_{24} = \left( \begin{array}{c|cccccccccccc} & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ I_{12} & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \end{array} \right)$$

Llamaremos $A_{12}$ a la submatriz de la derecha de $\mathcal{G}_{24}$ y $A_{11}$ a la matriz obtenida al eliminar la primera fila y columna de la matriz $A_{12}$.

Tal y como podemos observar, nos encontramos ante la matriz circulante $A_{11}$, ya que cada fila se desplaza un elemento a la izquierda conforma avanza a la columna siguiente.

**Nota 19** *Tanto $A_{11}$ como $A_{12}$ son matrices simétricas.*

*Con tal de poder introducir el siguiente lema, precisaremos previamente de una nueva operación definida de la siguiente manera,*

$$u * v = (u_1 v_1, ..., u_n v_n) \qquad u, v \in F_q^n$$

**Lema 6** *Para cada vector $u, v \in F_2^n$,*

$$w(u + v) = w(u) + w(v) - 2w(u * v)$$

**Lema 7** *Todas las palabras de $\mathcal{G}_{24}$ tendrán peso múltiplo de 24.*

**Lema 8** *Dado $f_1, ..., f_{12}$ los vectores fila de $\mathcal{G}_{24}$, para cada $i, j : 1 \leq i, j \leq 12$,*

1. $w(f_i * f_j) = 4 \qquad \forall i, j : i \neq j \ y \ i, j \neq 1$
2. *Sea* $j \neq 1 \Rightarrow w(f_j * f_j) = 8 \ y \ w(f_1 * f_j) = 6$
3. $w(f_1 * f_1) = 12$

**Proposición 15** *$\mathcal{G}_{24}$ tiene distancia mínima 8.*

Puesto que la distancia mínima de $\mathcal{G}_{24}$ es 8, el código será capaz de corregir 3 errores (También ocurrirá para una distancia de 7).

**Definición 39** *Llamaremos código de Golay binario $\mathcal{G}_{23}$, al código binario cuya matriz generatriz se obtiene a partir de la matriz $\mathcal{G}_{24}$ eliminando una columna cualquiera.*

De esta manera el nuevo código será de tipo, $[23, 12, 7]$, seguirá corrigiendo 3 errores y disminuiremos en uno la longitud y la distancia mínima del código.

**Nota 20** *Algunos autores llaman código de Golay binario al generado a partir de su matriz $\mathcal{G}_{23}$ y código de Golay binario extendido a $\mathcal{G}_{24}$.*

### Códigos de Golay ternarios

Llamaremos *códigos de Golay ternarios* a los códigos construidos a partir de la matriz generatriz,

$$G_{12} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 1 & 1 & 2 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 2 & 0 & 2 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 2 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 2 & 1 & 1 & 2 & 0 \end{pmatrix}$$

Este código generado es un código autodual de tipo $[12, 6, 6]$ con palabras múltiplo de 3. Además, tal y como hemos mostrado en los códigos binarios, obtendremos el código de tipo $[11, 6, 5]$ al que llamaremos *código de Golay ternario $\mathcal{G}_{11}$*.

**Decodificación de los códigos de Golay**

**Nota 21** *Para simplificar, notaremos,*

$$G_{24} = G$$
$$H_{24} = H$$
$$I_{12} = I$$
$$A_{12} = A$$

Tal y como ya sabemos, el vector $y = c + e$ tal que $w(e) \leq 3$.

Ahora vamos a estructurar $e$ tal que $e = (e_I, e_D) : e_I, e_D \in F_2^{12}$. Puesto que $G$ también puede actuar como a matriz de control, sacaremos los siguientes síndromes,

$$s_H(y) = s_H = Hy^t = He^t = (A, I), (e_I, e_D)^t = Ae_i^t + e_D^t = (e_I A + e_D)^t$$
$$s_G(y) = s_g = Gy^t = Ge^t = (I, A), (e_I, e_D)^t = e_i^t + Ae_D^t = (e_I + e_D A)^t$$

**Proposición 16** *Dado un error $e = (e_I, e_D)$ de peso $w(e) \leq 3$ ,*

$$w(s_G) \leq 3 \qquad sie_D = 0$$
$$w(s_G) \geq 5 \qquad sie_D \neq 0$$

$$w(s_H) \leq 3 \qquad sie_I = 0$$
$$w(s_H) \geq 5 \qquad sie_I \neq 0$$

- *Si* $w(s_G) \leq 3 \rightarrow e_D = 0 \Rightarrow e_I = s_G^t$ y $e = (s_G^t, 0)$
- *Si* $w(s_H) \leq 3 \rightarrow e_I = 0 \Rightarrow e_D = s_H^t$ y $e = (0, s_H^t)$

*Si* $e_I \neq 0$ y $e_D \neq 0 \Rightarrow w(s_G) \geq 5$ y $w(s_H) \geq 5 \Rightarrow w(e) \leq 3$, por tanto,

- *Si* $w(e_I) = 1 \rightarrow \exists! i : e_I = u_i \Rightarrow e_I + u_i = 0$ y $y + (u_i, 0)$ no tiene ningún error en las primeras 12 coordenadas, entonces,

$$w(s_H(y+(u_j,0))) \leq 3 \qquad sie_I = u_j$$
$$w(s_H(y+(u_j,0))) \geq 5 \qquad sie_I \neq u_j$$

Y calcularemos los 12 síndromes de $s_H(y+(u_i,0)) \qquad i \in \{1,...,12\}$

- *Si encontramos uno cuyo peso* $\leq 3 \Rightarrow e = (u_i, s_H(y+(u_i,0))^t)$
  *Si ninguno de es de peso* $\leq 3 \Rightarrow w(e_I) \neq 1$ y realizamos el mismo proceso con $s_G(y+(u_i,0))$
- *Si encontramos uno cuyo peso* $\leq 3 \Rightarrow e = (s_H(y+(0,u_i))^t, u_i)$
- *Si todos los síndromes tienen peso* $\geq 5 \Rightarrow$ *se habrán cometido 5 o más errores.*
  *Tal y como podemos observar, calcularemos un total de* $2+12+12=26$ síndromes.

## 3.3. Cotas en los parámetros de un código

Las cotas de los parámetros de un código serán utilizados para medir que valores $k$ y $d$ serán los más indicados en un código de longitud $n$.

No obstante, utilizaremos las cotas con mayor importancia y que mayor uso podremos dar a la hora de enfocar nuestro trabajo respecto a los *códigos cíclicos* y sobretodo los *cuasi-cíclicos.*

**Teorema 6** ***La Cota de Singleton***

*Dado un código $C$ de tipo $[n,k,d]$ sobre $F_q$, entonces $k+d \leq n+1$.*

**Nota 22** *Este teorema entrará dentro de los referidos a las cotas superiores.*

*Para el enunciado del siguiente teorema necesitaremos un cálculo previo.*

*Dado un $r = 1,...,n$, $\exists\binom{n}{r}(q-1)^r$ vectores en $F_q^n$ de peso $r$. Por tanto, la cantidad de elementos de $F_q^n$ en una bola de radio $t \in \mathcal{N}$ centrada en $0$ es $V_q(n,t)$ tal que,*

$$V_q(n,t) = 1 + \binom{n}{1}(q-1) + ... + \binom{n}{t}(q-1)^t$$

A continuación, ya podemos enunciar el teorema de existencia de códigos con ciertos parámetros perdefinidos.

**Teorema 7** ***La Cota de Gilbert-Varshmov***

*Llamaremos cota de Gilbart-Varshmov a la inecuación,*

$$q^{n-k+1} > V_q(n, d-1)$$

*Esto implicará un código lineal de tipo* $[n, k]$ *sobre* $F_q$ *con distancia* $\geq d$.

*Este último teorema es de gran importancia para este trabajo, ya que los códigos cuasi-cíclicos sobrepasaron esta cota tal y como explicaremos más adelante.*

# Capítulo 4

# Códigos Cíclicos

Los códigos cíclicos son unos de los más utilizados a días de hoy, es por ello que son una de las de códigos familias más estudiadas e importantes en el campo de códigos correctores de errores.

Con tal de definir los *códigos cíclicos*, mostraremos su característica principal.

Dada la matriz de control de $\mathcal{H}_2(3)$,

$$H' = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{4.1}$$

Aplicamos la siguiente permutación,

$$\sigma = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 2 & 4 & 3 & 7 & 5 & 6 \end{bmatrix} \downarrow \tag{4.2}$$

y la matriz nos queda de la siguiente manera,

$$H = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \tag{4.3}$$

Dada por ejemplo la palabra $c = 1001011$, podemos encontrar $2^3 - 1 = 7$ permutaciones posibles,

1001011
0010111
0101110
1011100
0111001
1110010
1100101

podemos comprobar con la matriz $H$ que se encuentra dentro del código $C$.

Esta propiedad la albergan los códigos denominados *Códigos cíclicos*, donde encontramos algunos equivalentes como los códigos de Hamming binarios, los no binarios y los de Golay.

**Notación 2** *Por temas de conveniencia, a partir de ahora el vector $x = (x_1, ..., x_n)$ lo denotaremos como $x = (x_0, ..., x_{n-1})$.*

**Definición 40** *Llamaremos a un código cíclico si dado una palabra código $(c_0, ..., c_{n-1}) \in C \Rightarrow (c_{n-1}, c_0, ..., c_{n-2}) \in C$.*

**Nota 23** *Esto implicará que $\exists S \subset Aut(C)$ de orden $n$ tal que $S$ es cíclico.*

Ahora vamos a definir un espacio vectorial $F_q[X]^{n-1} =$, es decir que contiene todos los polinomios de $F_q$ cuyo grado será $\leq n$.

Por otra parte, definimos el anillo cociente $A = \frac{F_q[X]}{<X^n - 1}$

.

Gracias al isomorfismo $F_q^n \cong F_q[X]^{n-1} \cong A$.

**Nota 24** *Recordemos que dados dos conjuntos ordenados $P$ y $Q$, y una función biyectiva $f : (P, \geq) \to (Q, \geq')$, existirá un isomorfismo entre ellos si dado $p_1, p_2 \in P$ y $q_1$, si $p_1 \geq p_2 \Rightarrow f(p_1) \geq' f(p_2)$.*

Entonces dado un vector $(a_0, ..., a_{n-1})$ definiremos el vector $F_q[X]^{n-1} = a_0 + a_1X + ... + a_{n-1}X^{n-1}$ y por la propia definición de anillo cociente $A = a_0 + a_1X + ... + a_{n-1}X^{n-1} + < X^n - 1 >$ tal que $F_q \subset A$.

Puesto que queremos tener todos los factores de $< X^n - 1 >$ irreducibles restingiremos los valores de $n$ tal que $mcd(q, n) = 1$ (como se puede observar, sobre $q = 2^r$ todos sus códigos serán de longitud $n$ impar).

Por otra parte, sus raices formarán un grupo cíclico de orden $n$.

**Teorema 8** *$C$ será cíclico $\iff$ dado $c \in C \Rightarrow cA \in Ao/yAc \in A$, es decir, $C$ es un ideal.*

**Teorema 9** *Sea $C$ un código cíclico de longitud $n \Rightarrow \exists! g(X) \in F_q[X] : g(X) \mid X^n - 1$ polinomio mónico tal que $C =< g(X) >$.*

Lo que este teorema quiere decir es que para cada elemento de $C$, puede asociarse con un polinomios (de grado $\leq n$) múltiplo de $g(X)$.

Además, si queremos un total de $m$ factores irreducibles de $X^n - 1$ sobre $F_q$, encontraremos $2^m$ códigos de longitud $n$.

**Ejemplo 16** *Si tenemos un código cíclico binario cuya longitud es 15, $X^{15} - 1$ tiene 5 factores irreducibles, por tanto, tendremos un total de $2^5 = 32$ códigos.*

**Matriz Generatriz**

**Proposición 17** *Sea $C$ un código cíclico de longitud $n$ en $F_q$, el polinomio generados $g(X)$ de grado $n - k$, entonces, una base del código $C$ será $\{g(X).g(X)X, ..., g(X)X^{k-1}\}$ tal que $C$ tendrá dimensión $k$.*

**Corolario 4** *Sea el código cíclico $C$ de longitud $n$ y polinomio generador $g(X) = g_0 + g_1X + ... + g_{n-k}X^{n-k}$, la matriz generatriz de será de la forma,*

$$G = \begin{pmatrix} g_0 & g_1 & g_2 & \dots & \dots & g_{n-k} & 0 & \dots & \dots & \dots & 0 \\ 0 & g_0 & g_1 & \dots & \dots & g_{n-k-1} & g_{n-k} & 0 & \dots & \dots & 0 \\ 0 & 0 & g_0 & g_1 & \dots & \dots & \dots & g_{n-k} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & g_2 & \dots & \dots & \dots & 0 & g_0 & g_1 & \dots & g_{n-k} \end{pmatrix}$$

Ahora, si queremos crear una matriz generatriz de la forma estandar $(I_k, R)$ deberemos seguir el siguiente proceso,

1. $X^j = g(X)a_j(X) + r_j(X) \qquad deg(r_j(X)) < n-k \qquad j \in \{n-k, ...n-1\}$
2. Dado $X^j - r_j(X) \in C$,

$$g_j(X) = X^k(X^j - r_j(X)) \equiv X^{j+k-n} - X^k r_j(X)\ (mod X^n - 1) \qquad j \in \{n-k, ...n-1\}$$

Y obtendremos los valores $X^{j+k-n}$, es decir, $1, X, ..., X^{k-1}$ cuyo grado $k$ de $X^k r_j(X)$ se encontrará entre $k$ y $n$.

**Codificación de los códigos cíclicos**

Dado un código cíclico $C$ de tipo $[n, k]$, podremos codificar la información a través de las matrices generatrices explicadas anteriormente, ya sea estandar o no, o utilizando la notación polinómica, es decir, dado el polinomio generador de $C$, $g(X) : deg(X) = n - k$ y transformaremos el mensaje a codificar en el polinomio $a(X)$, tal que,

$$g(X)a(X) \in C$$

Por otra parte, si deseamos una codificación sistemática para las últimas $k$ posiciones utilizaremos la división euclidea,

$$X^{n-k}a(X) = g(X)q(X) + r(X) \qquad deg(r(X)) < deg(g(X)) = n - k$$

Y, utilizando esta ecuación y despejando $g(X)q(X)$ ya que eso significa que el elemento generatriz con otro del conjunto generará un código que será,

$$X^{n-k}a(X) - r(X) \in C$$

**Matriz de control**

**Definición 41** *Llamaremos al polinomio $h(X)$ polinomio de control si dado un código cíclico $C$ de longitud $n$ sobre $F_q$ cuyo polinomio generador es $g(X)$ de grado $n-k$, entonces $h(X)$ verifica que,*

$$h(X) = \frac{X^n - 1}{g(X)} = h_0 + h_1 X + ... + h_k X^k$$

**Proposición 18** *La matriz de control* $(n-k)xn$ *de* $C$ *definida anteriormente será de la forma,*

$$H = \begin{pmatrix} 0 & 0 & \dots & \dots & 0 & h_k & h_{k-1} & \dots & \dots & h_1 & h_0 \\ 0 & 0 & \dots & \dots & h_k & h_{k-1} & \dots & \dots & h_1 & h_0 & 0 \\ 0 & \dots & 0 & h_k & h_{k-1} & \dots & \dots & h_1 & h_0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ h_k & h_{k-1} & \dots & \dots & h_1 & h_0 & 0 & 0 & \dots & \dots & 0 \end{pmatrix}$$

**Nota 25** *Tal y como sabemos, el polinomio de control* $h(X)$ *genera un código dual* $C$*, esto implicará que el dual de un código cíclico es mismamente cíclico.*

**Ceros en el código**

Puesto que para la generación del código nos hemos hecho servir de polinomios, vamos a utilizar sus raíces con tal de generar ceros en el código y saber rápidamente si una palabra se encuentra en el código.

Esta idea es utilizada en códigos de Hamming y Golay tratados como cíclicos y la idea principal de los códigos BCH.

Dado $X^n - 1 = f_1(X)...f_m(X)$ la descomposición en *factores irreducibles*, $\alpha_i$ una raíz de $f_i(X)$ y $C_i$ el código cíclico generado por $f_i(X)$,

$$C_i =< f_i(X) >= \{c(X) \mid c(\alpha_i) = 0\}$$

Y de forma genérica para todo $C$ generado por $g(X) = f_{i_1}...f_{i_r}$ lo podremos expresar,

$$C =< g(X) >= \{c(X) \mid c(\alpha_{i_1}) = ... = c(\alpha_{i_r})\}$$

De esta manera ya no tenemos por que crear un polinomio y buscar los ceros asociados si no que partiendo del conjunto $\{\alpha_1, ..., \alpha_r\}$, de diversas extensiones finitas $F_{q^{t_1}}, ..., F_{q^{t_r}} F_{q^t} : t = mcm(\{t_1, ..., t_r\})$ crearemos un código $C$,

$$C = \{c(X) \in A \mid c(\alpha_1) = ...c(\alpha_r) = 0\}$$

Y este código será cíclico ya que dado un $f_i(X)$ de raíz $\alpha_i$, verificaremos que $C =< g(X) >= mcm(f_1, ..., f_r)$ donde dado cada $n_i$ asociado a $\alpha_i$ de la extensión $F^*_{q^{t_i}}$ y siendo $n = \{n_1, ..., n_r\}$, entonces $g(X) \mid X^n - 1$ y el $mcd(n, q) = 1$.

Ahora para saber si una palabra está en el código definiremos la matriz $H'$

$$H' = \begin{pmatrix} 1 & \alpha_1 & \dots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \dots & \alpha_2^{n-1} \\ \vdots & \vdots & . & \vdots \\ 1 & \alpha_r & \dots & \alpha_r^{n-1} \end{pmatrix} \tag{4.4}$$

Entonces para comprobar si $f(X) = f_0 + f_1X + ... + f_{n-1}X^{n-1} \in C$, deberemos ver si $H'f(X) = (f(\alpha_1), ..., f(\alpha_r)) = 0$.

No obtante, cabe observar que la matriz $H'$ no es fiel a la definiciónd e matriz de control ya que no tiene coeficientes en $F_q$ ni dimensión $(n-k)xn$. Para transformarla en una matriz de control deberemos cambiar cada $\alpha_i^j$ por el vector columna de sus coordenadas y eliminando las filas linealmente independientes.

**Hamming y Golay como codigos cíclicos**

**Proposición 19** *Dado el código de Hamming q-ario $\mathcal{H}_q(r)$, si $mcd(q-1, r) = 1 \Rightarrow$ esequivalenteauncódigocícli*

**Nota 26** *A partir de esta definición podemos deducir que el código de Hamming binario serán equivalentes a uno cíclico.*

**Ejemplo 17** *Dado el código cíclico $\mathcal{H}_\in(\ni)$ de longitud $2^3 - 1 = 7$, generado por el polinomio irreducible $F_2$ del elemento primitivo $F_{2^3}$, tal polinomio es $1 + X + X^3$.*

**Proposición 20** *$\mathcal{G}_{23}$ es el código cíclico binario de longitud 23 generado por el polinomio $1 + X^2 + X^4 + X^5 + X^6 + X^{10} + X^{11}$ y $\mathcal{G}_{11}$ el código cíclico ternario de longitud 11 generado por el polinomio $-1 - X - X^2 - X^3 + X^5$.*

**Decodificación de códigos cíclicos**

Con tal de hacer más eficiente la decodificación la información recibida nos aprovecharemos de las propiedades de este tipo de códigos. Ya hemos visto que uno de los grandes obstáculos a la hora de decodificar es tener que almacenar una tabla de síndromes y líderes de gran extensión.

No obstante, gracias a la estructura de los códigos cíclicos podremos ahorrar mucho espacio puesto que la tabla de síndromes será $n$ veces menor a la de tabla necesaria, aun que calcularemos $n$ síndromes de más.

Al ser $C$ un código cíclico nos dedicaremos a corregir solamente los errores que se producirán en una posición fija, normalmente fijaremos la coordenada $n-1$.

De esta manera, recibido el mensaje $y = (y_0, ..., y_{n-1})$ calcularemos tan solo la *tabla reducida* e iremos permutando el vector $y$ tal que,

$$y^{(1)} = (y_{n-1}, y_0, ..., y_{n-2})$$

Esto implicará que dado $y = e + 1 \Rightarrow y^{(1)=c^{(1)}+e^{(1)}}$.

Por tanto, tendremos la tabla reducida, iremos permutando el código recibido, calculando su síndrome $s(y)$ y donde encontremos el síndrome cambiaremos la posición enunciada por el elemento líder.

**Ejemplo 18** *Imaginemos que tenemos el código $\mathcal{H}_2(3)$ binario de Hamming cuya matriz de control es,*

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{4.5}$$

*Con la forma de decodificación clásica sería seria necesario una tabla completa,*

| Síndrome | Líder |
|---|---|
| 000 | 0000000 |
| 001 | 1000000 |
| 010 | 0100000 |
| 011 | 0010000 |
| 100 | 0001000 |
| 101 | 0000100 |
| 110 | 0000010 |
| 111 | 0000001 |

Cuadro 4.1: Tabla de síndromes completa

.

*No obstante gracias a los códigos cíclicos podremos utilizar tan solo la tabla reducida,*

| Síndrome | Líder |
|---|---|
| 111 | 0000001 |

Cuadro 4.2: Tabla de síndromes reducida

.

*Ahora supongamos que hemos enviado la palabra* $c = 1001100$ *y recibimos el vector* $y = 1011100$*, y ahora surgirá el problema de calcular 7 síndromes, uno para cada permutación,*

$$
\begin{aligned}
s(y) = s(1011100) = 011 &\rightarrow \textit{la última coordenada } 0 \textit{ es correcta.}\\
s(y^{(1)}) = s(0101110) = 100 &\rightarrow \textit{la última coordenada } 0 \textit{ es correcta.}\\
s(y^{(2)}) = s(0010111) = 101 &\rightarrow \textit{la última coordenada } 1 \textit{ es correcta.}\\
s(y^{(3)}) = s(1001011) = 110 &\rightarrow \textit{la última coordenada } 1 \textit{ es correcta.}\\
s(y^{(4)}) = s(1100101) = 111 &\rightarrow \textit{la última coordenada } 1 \textit{ es } \boldsymbol{incorrecta}\\
&\rightarrow \textit{porque } 111 \textit{ aparece en la tabla reducida.}\\
s(y^{(5)}) = s(1110010) = 001 &\rightarrow \textit{la última coordenada } 0 \textit{ es correcta.}\\
s(y^{(6)}) = s(0111001) = 010 &\rightarrow \textit{la última coordenada } 1 \textit{ es correcta.}
\end{aligned}
$$

*Por tanto la coordenada última de* $y^{(4)}$*, es decir,* $n - 4$ *es* $0$ *no* $1$*, por tanto el mensaje corregido será* $1001100$*.*

### Errores a Ráfagas

**Definición 42** *Llamaremos ráfaga al vector $x \in F_q^n$ tal que todas las coordenadas no nulas son consecutivas.*

**Nota 27** *La longitud de la ráfaga se medirá como $w(x)$.*

**Proposición 21** *Un código cíclico $C$ de parámetros $[n, k]$ no contiene ninguna ráfaga de longitud $l \leq n - k$ por tanto detectará hasta $l \leq n - k$ errores en ráfaga.*

**Proposición 22** *Dado un código cíclico de parámetros $[n, k]$, los errores de un vector recibido $y$ tienen una ráfaga de longitud $n - k$ como máximo, entonces, $\exists j : e^{(j)}(X) = s[y^{(j)}](X)$.*

### Intercalado

Con tal de combatir los errores de ráfagas utilizaremos la técnica del *intercalado*, es decir, dadas las palabras $c^1, ..., c^m \in C$ construiremos las palabras por intercalado,

$$(c_0^1, ..., c_0^m; c_1^1, ..., c_1^m; ...; c_{n-1}^1, ..., c_{n-1}^m) = C^{(m)}$$

**Proposición 23** *El polinomio generador de $C^{(m)}$ es $g(X^m)$, parámetros $[mn, mk]$ y detectará ráfagas de longitud $m(n - k)$.*

### Códigos BCH

Los códigos $BCH$ son llamados así debido a sus descubridores Bose, Chaudhuri y Hocquenghem. Esta familia de códigos es de gran importancia, en parte por estar basada en el comportamiento de los códigos cíclicos, cuya estructura consigue una alta optimización de este y por otra parte porque podremos definir el código partiendo de la cantidad de errores que queremos corregir.

Primeramente determinaremos su matriz que podríamos llamar de control (ya que pese no ser fiel a su definición, esta realiza la misma función) a partir de los elementos que realizan los ceros $\{\alpha_1, ..., \alpha_r\}$ del polinomio generador,

$$H' = \begin{pmatrix} 1 & \alpha_1 & \dots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \dots & \alpha_2^{n-1} \\ \vdots & \vdots & . & \vdots \\ 1 & \alpha_r & \dots & \alpha_r^{n-1} \end{pmatrix} \tag{4.6}$$

Por otra parte, la distancia mínima $d$ será $\geq d$ si para cualquier $d-1$ columnas de $H'$, estas son linealmente independientes.

Una de las formas más simples de determinar la distancia mínima es utilizando *potencias consecutivas de una raíz primitiva* $n-$ésima de la unidad $\alpha_i = \alpha^i \ : \ i = 1, .., r < n$. De esta manera, $d(C) \geq r + 1$.

**Definición y parámetros**

Con tal de mostrar la definición de este tipo de códigos, supondremos que nos encontramos en un cuerpo finito $F_q$ y $n, b, \delta \in \mathbb{N} \mid 2 \leq \delta \leq n$.

Además dado $q$, llamaremos $m$ al orden multiplicativa $q$ modulo $n$ ($q^m \equiv 1 mod n$). Una vez explicado el contexto, procederemos con la siguiente definición.

**Definición 43** *Nos encontraremos ante un código BCH de longitud $n$ sobre $F_q$ y distancia mínima prevista $\delta$ si su polinomio generador tiene por raíces $\alpha^b, \alpha^{b+1}, ...\alpha^{b+\delta-2}$.*

Por otra parte, dependiendo de la definición de sus parámetros, nos encontraremos ante diferentes clases de códigos,

- *BCH estrictamente*, si $b = 1$.
- *BCH primitivo*, si $n = q^m - 1$.
- *Reed-Solomon*, si $n = q^m - 1$ y además $m = 1 \Rightarrow n = q - 1$.

**Proposición 24** *Dado un código BCH $C$ de distancia prevista $\delta$, entonces su distancia mínima $d \geq \delta$.*

**Decodificación de loc códigos BCH**

Otra de las grandes ventajas de este código es su efectividad a la hora de decodificar el código recibido.

Ahora, dado un código BCH $C$ sobre $F_q$, longitud $\delta = 2t + 1$, es decir, con capacidad de corrección de $t$ errores y determinando su matriz de control como,

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \dots & \alpha^{n-1} \\ \vdots & \vdots & \vdots & . & \vdots \\ 1 & \alpha^{\delta-1} & \alpha^{2(\delta-1)} & \dots & \alpha^{(n-1)(\delta-1)} \end{pmatrix} \tag{4.7}$$

Supongamos que tras el envio de la palabra $c \in C$ recibimos el vector $y = c + e$ con $w(e) = r \leq t$, $0leqi_1 < ... < i_r \leq n - 1$ las posiciones de los distintos errores y $e_{i_1}, ..., e_{i_r}$ las coordenadas del error en dicha posición.

Primeramente calcularemos el síndrome $s = s(y) = Hy^t = s(e) = (s_0, ..., s_{\delta-2})^t$, que pasado a la forma polinómica nos quedará,

$$s(X) = s_0 + ... + s_{\delta-2}X^{\delta-2}$$

Para cada $h = 0, ..., \delta - 2$,

$$S_h = y(\alpha^{h+1}) = \sum_{j=1}^{r} e_{i_j}(\alpha^{h+1})^{i_j} = \sum_{j=1}^{r} e_{i_j}(\alpha^{i_j})^{h+1}$$

Llamaremos *localizadores del error* a $\eta_j = \alpha^{i_j}$ para todo $j = 1, ..., r$ y *valores del error* $\epsilon_j = e_{i_j}$, y a partir de estos $2r$ valores, podremos conoces el error.

A partir de ahora vamos a suponer que $s(x) \neq 0$, ya que sí $s(X) = 0 \Rightarrow y \in C$ y no necesitaremos decodificación.

**Definición 44** *Llamamos polinomio localizador de errores al polinomio,*

$$L(X) = (1 - \eta_1 X)...(1 - \eta_r X)$$

*y llamaremos polinomio evaluador de errores al polinomio,*

$$E(X) = \sum_{j=1}^{r} \epsilon_j \prod_{i \neq j} (1 - \eta_i X)$$

*Cuyos polinomios son de grado $r$ y $r-1$ respectivamente.*

**Proposición 25** *Utilizando las condiciones definidas anteriormente,*

- *Si $p_1, ..., p_r$ son las raíces de $L(X) \Rightarrow p_1^{-1}, ..., p_r^{-1}$ son localizadores de errores.*
- *Dados $\eta_1, ..., \eta_r$ los valores del error son,*

$$\epsilon_j = \frac{-\eta_j E(\eta_j^{-1})}{L'(\eta_j^{-1})} \qquad L'(X) \text{ derivada de } L(X)$$

.

**Teorema 10** ***Ecuación clave.*** *Los polinomios $L(X), E(X), s(X)$ se relacionan a través de la ecuación,*

$$E(X) \equiv L(X)s(X) mod X^{\delta - 1}$$

Para terminar la decodificación deberemos conocer los valores de $E(X)$ y $L(X)$, para ello utilizaremos el *método euclideo* que explicaremos a continuación, pese que también existe el *método de Berlekamp-Massey* menos utilizado.

**Método Euclídeo**

Con tal de poder explicar el algoritmo a utilizar debemos mostrar una serie de resultados previos.

**Lema 9** $mcd(L(X), E(X)) = 1$

**Proposición 26** *Sea $\tilde{L}(X)$ y $\tilde{E}(X)$, existe un polinomio $\lambda(X)$ tal que $\tilde{L}(X) = \lambda(X)L(X)$ y $\tilde{E}(X)\lambda(X)E(X)$, y además,*

- $deg((X)) \leq t$ *y* $deg((X)) < t$
- $\tilde{E}(X) \equiv s(X)\tilde{L}(X) mod X^{\delta-1}$

**Proposición 27** *Si $r \leq t$ entonces se cumple la proposición anterior.*

**Lema 10** $mcd(\tilde{L}(X), \tilde{E}(X)) = 1$

Tendremos, por tanto $L(X)$ y $E(X)$ definido como,

$$L(X) = \lambda u_j(X)\ E(X) = (-1)^{j+1}\lambda f_j(X)$$

Y determinaremos $\lambda \in F_q$ como el auténtico polinomio localizador que verificará $L(0) = 1 =_j (0)$, probando así el siguiente resultado.

**Teorema 11** *Si $r \leq t$, los polinomios,*

$$L(X) = \frac{u_j(X)}{u_j(0)}\ , \quad E(X) = \frac{(-1)^{j+1} f_j(X)}{u_j(0)}$$

*son los auténticos polinomios localizadores y evaluadores respectivamente.*

Por tanto el algoritmo final para decodificar el mensaje recibido será el siguiente,

1. $s(X) = s_1 + ... + s_{\delta-1}X^{\delta-2}$ con $s_i = u(\alpha^i)$.
2. Aplicaremos el algoritmos de Euclides modificando $f_0(X) = X^{\delta-1}$ y $f_1(X) = s(X)$ hasta obtener un índice $j$ tal que $f_j(X) < t$.
3. Localizar y evaluar los errores a partir de los polinomios,

$$L(X) = \frac{u_j(X)}{u_j(0)}\ , \quad E(X) = \frac{(-1)^{j+1} f_j(X)}{u_j(0)}$$

4. Buscar las raíces de $L(X)$ y encontradas sus raíces $\alpha^{h_1}, ..., \alpha^{h_r}$ y cuyo inverso es el localizados del error $\eta = \alpha^{n-h_1}, ..., \eta_r = \alpha^{n-h_r}$ cuyo error asociado será,

$$\epsilon = \frac{-\alpha^{n-h_j} E(\alpha^{h_j})}{L'(\alpha^{h_j})}$$

5. Corregir el mensaje para la coordenada $i \in \{0, ..., n-1\}$,

$$\begin{array}{ll} y_i & si \;\; i \neq n - h_1, ..., n - h_r \\ y_i - \epsilon_j & si \;\; i = n - h_j \end{array}$$

# Capítulo 5

# Códigos Cuasi Cíclicos

En esta sección vamos a centrarnos en el estudio de los códigos cuasi-cíclicos. Estos son de gran importancia ya que la limitación de un código cíclico de extensión $q$ no puede tener una longitud mayor a $q$, en cambio, los cuasi-cíclicos se basa en estructuras modulares con tal de poder anexar varios códigos cíclicos y poder aumentar la longitud que deseemos.

Otra de las curiosidades de este código es que fue capaz de alcanzar la *cota de Gilbert-Varshamov* tal y como se nombra en el artículo [8].

La idea principal de estos códigos será basarnos en automorfismos de grupos cíclicos con tal de representarlos como módulo sobre el anillo de polinomios $F_q[X]$. Además cuanta mayor propiedades algebraicas dotemos la estructura, dispondremos de más herramientas para definir y optimizar nuestros códigos.

**Introducción**

La estructura básica de los códigos cuasi-cíclicos de longitud $lm$ consistirá en $l$ índices que formarán el espacio vectorial $(G_1, ..., G_l)$ que corresponderán a matrices cíclicas $mxm$.

Además los valores a escoger para los distintos objetivos de nuestro código han sigo ampliamente estudiados.

Por ejemplo, *Tilborg* estudió y encontró un método que hacia uso de una amplias búsquedas computacionales para construir códigos $1 - generadores$ cuasi-cíclicos de alfabeto binario, $m =$ 7,8 y longitud 120 tal y como se explica en el artículo [9].

A este estudio le siguió el de *Gulliver* y *Bhargav* quienes evitaron las exhaustiva búsquedas

e utilizaron métodos matemáticos, concretamente se basaron en técnicas heurísticas de optimización combinatoria junto a algoritmos de selección para encontrar nuevos $1-generadores$ en las cotas más bajas, y posteriormente continuaron su búsqueda en códigos cuasi-cíclicos $2-generadores$.

A estos le siguieron una gran cantidad de autores encontrando los mejores códigos generadores para cada caso, tal y como se explica en el *paper* [10], por ello, notaremos gran tanto interés en este tipo de códigos.

**Nota 28** *Un código cíclico puede considerarse como un caso particular de los códigos cuasi-cíclico cuando la longitud* $l = 1$.

En este trabajo, tal y como explicamos en la motivación del proyecto, mostraremos una base de los códigos cuasi-cíclicos, profundizando sobretodo en su estructura.

**Estructura básica**

**Definición 45** *Llamaremos código cuasi-cíclico* $C$ *de longitud* $lm$*, índice* $l$ *y definido bajo la menor potencia del operador cíclico sobre la que* $C$ *es invariante a los códigos generados por la matriz generatriz de forma que cada elemento de* $c \in C$ *será de la forma* $c = (c_1(x), ..., c_l(x))$ *cuyo grado de los polinomios no podrá ser mayor a* $m$ *y para cada permutación cíclica cumplirá que,* $xc = (xc_1(x)\ mod(x^m - 1), ..., xc_l(x)\ mod(x^m - 1)) \in C$.

Sea $R = F[X]$ un espacio finito tal que $R$ será un submódulo de $R^l$.

Por otra parte, definiremos como la preimagen del código $C$ en $F[X]^l$, tal que los submódulos de $\tilde{K}F[X] : \tilde{K} =< (X^m - 1)e_i : i = 1, ..., l >$ donde definiremos $e_i$ como el vector de la base cuya coordenadas $i$ será 1 y 0 las restantes.

Como $\tilde{C}$ es un submódulo del ideal principal del dominio $F[X]$ y que contiene $\tilde{K}$, tiene como matriz generadora el conjunto de la forma, $\{r_i(X^m - 1)e_j :, i = 1, ..., t, j = 1, ....l\}$ y con $r_i = (r_{i_1}, ..., r_{i_l})$. Las filas de la matriz que generan $\tilde{C}$ serán de la forma,

$$M = \begin{pmatrix} r_{11} & \dots & r_{1l} \\ r_{21} & \dots & r_{2l} \\ \vdots & & \vdots \\ r_{t1} & \dots & r_{tl} \\ X^m - 1 & \dots & 0 \\ \vdots & & \vdots \\ 0 & \dots & X^m - 1 \end{pmatrix} \tag{5.1}$$

A través de operaciones elementales podremos triangular la anterior matriz a la que notaremos como,

$$M = \begin{pmatrix} g_{11} & g_{12} & \dots & g_{1l} \\ 0 & g_{22} & \dots & g_{2l} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & g_{ll} \end{pmatrix} \tag{5.2}$$

de manera que $g_{ii}$ dividirá $X^m - 1 \quad \forall i$.

**Nota 29** *Cada elemento* $\tilde{c} \in \tilde{C} : \tilde{c} \neq 0$ *de la forma* $\tilde{c} = (0, ..., 0, c_r, ..., c_l) : r \geq 1$ *y* $c_r \neq 0$.

Por tanto, $c_r$ será una combinación lineal de $g_i$ tal que,

$$c_r = \sum_{i=1}^{l} a_i g_i$$

Y, como es obvio, el componente $c_r$ será divisible entre $g_{rr}$ ya que hemos ajustado los valores en la conversión de la matriz .

Por ello, consideraremos como una base Gröbner respecto a la base de $\tilde{G}$ refiriéndose a la posición sobre el termino en $F[X]^l$ tal que $e_1 > ... > e_l$ y el orden natural de los monomios $x^i$.

Por tanto,

$$\delta(g_{ij} < g_{jj}) \qquad : \qquad i < j$$

tal que $\delta$ corresponde al grado y $\delta(0) = -1$.

Consideraremos así a $\tilde{G}$ como la base reducida de Gröbner de $\tilde{C}$.

Si $\tilde{G}$ es una base de Gröbner reducida su componente de la diagonal $g_{ii} = X^m - 1 \Rightarrow (0, ..., 0, g_{i,i+1}, ..., g_{il}) \in \tilde{C}$, es decir, será $F[X] - combinación\ lineal$ de $\{g_{i+1}, ..., g_l\}$

Y sea $\delta(g_{ij}) < \delta(g_{jj})\ :\ j > i \Rightarrow g_{ij} = 0\ \forall j > i \Rightarrow g_i = (X^m - 1)e_i$.

**Definición 46** *Llamaremos término lider* $Lt(0, ..., 0, v_r, ..., v_l)\ :\ r \geq 1, v_r \neq 0$ *de un elemento* $v \in F[X]^l$ *al monomio* $X^{\delta_{v_r} e_r}$.

Además, cada $v$ solo se podrá definir como una sola forma normal, tal que, $Nf_{\tilde{G}}(v) = (0, ..., 0, v'_s, ..., v'_l)$ respecto a $\tilde{G}$ obtenida a partir de la división sucesiva de sus componentes por $g_{ii}$ y que satisface,

$$v = (0, ..., 0, v'_s, ..., v'_l) + \sum_{i=1}^{l} b_i g_i \qquad s \geq r, \delta v'_j < \delta g_{jj}\ :\ s \leq j \leq l$$

.

$v \in \tilde{C} \iff Nf_{\tilde{G}}(v) = (0, ..., 0)$

**Teorema 12** *Cada submódulo de* $\tilde{C}$ *de* $F[X]^l$ *que contiene* $\tilde{K}$*, tiene una base de Gröbner de la forma,*

$$\tilde{G} = \{g_i = (g_{i1}, g_{i2}, ..., g_{il})\} \qquad i \in \{1, ..., l\}$$

*tal que,*

1. $g_{ij} = 0\ \forall j < i$
2. $\delta(g_{ki}) < \delta(g_{ii})\ k < i$
3. *Dado el componente diferente a cero más a la izquierda de un elemento de* $\tilde{C}$ *en el lugar* $i$*, entonces será divisible por* $g_{ii}$*, concretamente su divisor será* $X^m - 1$.
4. *Sea* $g_{ii} = X^m - 1$*, entonces,* $g_i = (X^m - 1)e_i$

5. *La $F-$dimension de $\frac{F[X]^l}{\tilde{C}}$ es $\sum_{i=1}^{l}\delta g_{ii}$*

**Nota 30** *Cualquier $\tilde{G}$ triangular en una base de Gröbner del submódulo de $F[X]^l$ que genera.*

Si el submódulo contiene $\tilde{K} \iff \exists \tilde{A} \in Mat_l(F[X])$ tal que,

$$\tilde{A}\tilde{G} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1l} \\ a_{21} & a_{22} & \dots & a_{2l} \\ \vdots & \vdots & \ddots & \vdots \\ a_{l1} & a_{l2} & \dots & a_{ll} \end{pmatrix} \begin{pmatrix} g_{11} & g_{12} & \dots & g_{1l} \\ g_{21} & g_{22} & \dots & g_{2l} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & g_{ll} \end{pmatrix} = (X^m - 1)I \tag{5.3}$$

La matriz $I$ corresponde con la matriz identidad.

**Teorema 13** *$\tilde{G}$ es una base de Gröbner del submódulo $F[X]^l$ que contiene $\tilde{K} \iff \exists a_{ij} : 1 \leq i,j \leq l$ que verifica,*

$$\begin{aligned} a_{ij} &= 0 && sij < i \\ a_{ij} &= \frac{X^m - 1}{g_{ii}} && sij = i \\ a_{ij} &= \frac{-1}{g_{jj}}(\sum_{k=i}^{j-1} a_{ik}g_{kj}) && sij > i \end{aligned}$$

*Es más, $g_{ii}$ y $a_{ij}$ también cumplen que dado $m = \delta(g_{ii}) = \delta(a_{ii})\ \forall i$, la base de Gröbner es reducida $\iff \delta(g_{ii}) > \delta(g_{ji}) \forall j < i \iff \delta(a_{ii}) > \delta(a_{ij}) \forall j > i$*

**Nota 31** *Si suponemos que tenemos un conjunto triangularizado $\tilde{G}$. En la práctica, para verificar que es generado por un submódulo $\tilde{K}$, realizaremos los siguientes pasos.*

1. *Verificaremos que cada componente diagonal es divisor de $X^m - 1$.*

2. *Si se cumple la condición anterior, entonces el generador $g_i$ estará multiplicado por $a_{ii} = \frac{(X^m-1)}{g_{ii}}$ y restado $(Xm-1)e_i$.*

3. *El vector resultante debe ser reducido a 0 restando múltiples de $g_{i+1}, ..., g_l$.*

*Este proceso tomará $\frac{(l-1)(l-i+1)}{2}$ operaciones de multiplicación y resta de polinomios y un total de $\frac{(l-1)(l(l+1))}{6}$ operaciones requeridas.*

Dada la aplicación homomorfica[11] sobre $C \subseteq R^l$,

$$\begin{array}{rccl} \phi: & F[X]^l & \to R^l & \\ & (c_1, ..., c_l) & \longrightarrow & (c_1 + < x^m - 1 >, ..., c_l + < X^m - 1 >) \end{array} \tag{5.4}$$

Entonces, su preimagen $\phi^{-1}(C) = \tilde{C} \subset F_q[X]^l$ submódulo de $F_1[X]$ que contiene $\tilde{K} = \{(X^m - 1)e_j \ : \ 0 \leq j \leq l-1\}$, entonces tendrá un conjunto generador al que llamaremos $GB - generador$ de la forma,

$$\{u_1, ..., u_p, (X^m - 1)e_0, ..., (X^m - 1)e_{l-1}\}$$

y $u_b = (u_{b_0}(x), ..., u_{b_{l-1}}(x)) \in F_q[X]^l \ \forall b \in \{1, ..., p\}$.

**Corolario 5** *La dimensión de un código $C$ de conjunto GB-generador $\{\phi(g_i) \ : \ i \in \{1, ..., l\}\}$ verifica,*

$$lm - \sum_{i=1}^{l} \delta g_{ii} = \sum_{i=1}^{l} (m - \delta g_{ii})$$

**Notación 3** *A partir de ahora, denotaremos $X^m - 1 = \prod_{n=1}^{s} f_n^{\epsilon}$ tal que $m = (charF)^t m'$ tal que $mcd(m', charF) = 1$ y $\epsilon = (charF)^t$ para toda descomposición de $X^m - 1$ en factores irreducibles $f_n \in F$.*

**Lema 11** *Sea $mcd(m, charF) = 1 \Rightarrow$ Cada componente primario de $C$ se descompone directamente en una suma de submódulos cíclicos generados por elementos de su $RGB - conjunto$ generador.*

**Teorema 14** *Sea $C$ el código generado por $(f_1, f_2, ..., f_l)$, entonces, sus componentes de la diagonal de la base Gröbner de la preimagen $\tilde{C}$ son,*

$$f_{11} = mcd(f_1, X^m - 1)$$

$$f_{ii} = \frac{(X^m - 1)mcd(f_1, ..., f_i, X^m - 1)}{mcd(f_1, ..., f_{i-1}, X^m - 1)} \qquad i = 2, ..., l$$

**Corolario 6** *La dimensión del código generado por $(f_1, ..., f_l)$ es,*

$$m - \delta(mcd(f_1, ..., f_l, X^m - 1))$$

Gracias a toda la estructura que contendrá el código, tendremos una gran cantidad de herramientas que podremos utilizar a la hora decodificar la información.

No obstante, debido a su alta complejidad y extensión, nos limitaremos a mostrar tan solo su estructura.

*Con tal de poner en práctica las ideas de la estructura de un código cíclico explicaremos uno de los ejemplos dados por el artículo*[10].

**Ejemplo 19** *Dado un código binario $C_1$ cuyo índice es $l = 3$ y longitud $n = lm = 21 \Rightarrow m = 7$ que genera los elementos,*

$$v_1 = (X^5 + X^4 + 1, X^4 + X^3 + X + 1, X^4 + X^3X^2)$$
$$v_2 = (X^4 + X^3 + X^2 + 1, X, X^4 + X^3 + X + 1)$$

*Sea $f_1 = X + 1, f_2 = X^3 + X + 1, f_3 = X^3 + X^2 + 1 \Rightarrow X^7 + 1 = f_1 f_2 f_3$, entonces la base reducida de Gröbner de $\tilde{C}_1 =< v_1, v_2, (X^7 + 1)e_1, (X^7 + 1)e_2, (X^7 + 1)e_3 >$ es de la forma,*

$$\begin{pmatrix} f_2 & f_1^2 & x^2 \\ 0 & f_3 & f_1 f_3 \\ 0 & 0 & X^7 + 1 \end{pmatrix} \quad (5.5)$$

*A partir de sus componentes diagonales podemos calcular la dimension de* $C_1$*,*

$$\sum_{i=1}^{l}(m-\delta(g_{ii}))=4+4+0=8$$

*El código binario* $C_2$ *de índice* $l=3$ *y longitud* $n=84, m=28$ *cuyos conjuntos RGB-generadores vienen dados por,*

$$\begin{pmatrix} f_1^2f_2f_3^3 & f_1^2(X^2+X+1) & 1 \\ 0 & f_1^4f_2f_3 & f_2X \\ 0 & 0 & f_1^3 \end{pmatrix} \tag{5.6}$$

*cuya dimensión será,*

$$\sum_{i=1}^{l}(m-\delta(g_{ii}))=14+18+19=51$$

# Capítulo 6

# Conclusiones

Tras haber hecho un recorrido a través de todos los conceptos básicos de la teoría de la codificación, explicando algunos de los códigos más elementales como son los lineales y los cíclicos, y llegando a una introducción sobre las estructuras de los códigos cuasi-cíclicos, podemos encontrar una gran cantidad de puertas abiertas para continuar con nuestra formación en esta rama.

Por una parte, hemos visto un ejemplo de como se transforman los conceptos de una materia que queremos estudiar en objetos matemáticos, definiendo así desde qué es un alfabeto, la distancia mínima en un código, etc, hasta demostrar lemas y teoremas que fijen unas propiedades que podremos utilizar como herramientas para ahorrar espacio y agilizar la codificación entre otros.

Por otra parte, tras haber asimilado los conceptos previos nos adentramos en estudios con un uso práctico así como los códigos lineales, una base de una gran rama de estudio.

No obstante, los códigos cíclicos tienen un mayor interés puesto que se acerca a los códigos utilizados a la hora de codificar en la realidad.

Finalmente, todos los códigos tienen sus ventajas y desventajas, y dependiendo de nuestras necesidades recurriremos a unos u otros, ya sea por motivos de compresión, necesidad de alta corrección de errores, velocidad a la hora de decodificar.

# Bibliografía


[1] Sivakanth Gopi. Local codes for distributed storage (Princeton University)

[2] Munuera Gómez, Juan y Tena Ayuso, Juan. Codificación de la información. (Secretariado de Publicaciones e Intercambio Científico, Universidad de Valladoid, [1997])

[3] Shannon, Claude Elwood. A Mathematical Theory of Communication (Bell System Technical Journal)

[4] Sistemas de Procesamiento de Datos I. Documentación del Trabajo Práctico - CD-ROM (Universidad Abierta Interamericana)

[5] Rafael Gonzalez y Richard Woods. Tratamiento digital de imágenes (Addison-Wesley Iberoamericana S.A. Primera edición) [1996]

[6] M.A. García, L. Martínez, T. Ramírez. Introducción a la Teoría de Códigos (Facultad de Ciencia y Tecnología. UPV/EHU)

[7] Juan Jacobo Simón Pinero. Códigos correctores de errores (Universidad de Murcia)

[8] T. Kasami. A Gilbert-Varshamov bound for quasi-cyclic codes of rate 1/2 (IEEE Trans. Information Theory, IT-20:679) [1974]

[9] H.C.A van Tilborg. Quasi-cyclic codes with rate 1=m (IEEE Trans. Inform. Theory)[1978]

[10] Kristine Lally, Patrick Fitzpatrick. Algebraic structure of quasicyclic codes (Department of Mathematics, National University of Ireland, Cork, Ireland) [2000]

[11] Cem Günerian Ling y Buket Özkaya. Concise Encyclopedia of Coding Theory. [2007]