LA MATEMÁTICA DE LA TEORÍA DE CÓDIGOS

JOHN JAIRO LÓPEZ SANTANDER
HAMILTON MAURICIO RUIZ

FACULTAD DE CIENCIAS EXACTAS Y NATURALES
DEPARTAMENTO DE MATEMÁTICAS Y ESTADÍSTICA
UNIVERSIDAD DE NARIÑO
SAN JUAN DE PASTO
2013

LA MATEMÁTICA DE LA TEORÍA DE CÓDIGOS

JOHN JAIRO LÓPEZ SANTANDER
HAMILTON MAURICIO RUIZ

Trabajo presentado como requisito parcial para optar al título de
Licenciado en Matemáticas

Asesores
Fernando Andrés Benavides Agredo
Magister en Matemáticas
John Hermes Castillo Gómez
Doctor en Matemáticas

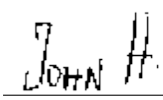
FACULTAD DE CIENCIAS EXACTAS Y NATURALES
DEPARTAMENTO DE MATEMÁTICAS Y ESTADÍSTICA
UNIVERSIDAD DE NARIÑO
SAN JUAN DE PASTO
2013

Nota de Responsabilidad

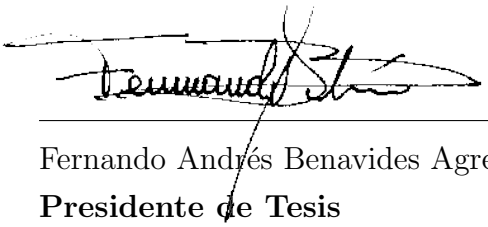
Todas las ideas y conclusiones aportadas en el siguiente trabajo son responsabilidad exclusiva de los autores.

Artículo 1^{ro} del Acuerdo No. 324 de octubre 11 de 1966 emanado por el Honorable Consejo Directivo de la Universidad de Nariño.

Nota de Aceptación



John Hermes Castillo Gómez
Presidente de Tesis



Fernando Andrés Benavides Agredo
Presidente de Tesis

Juan Miguel Velásquez Soto
Jurado

Oscar Fernando Soto Agreda
Jurado

San Juan de Pasto, Noviembre 5 de 2013

Agradecimientos

Al término de este trabajo de grado es ineludible expresar los sentimientos de agradecimiento por aquellos que contribuyeron de manera incondicional en la realización de este aporte a nuestra formación matemática.

En primer lugar agradecemos a Dios por brindarnos su inteligencia y sabiduría en la realización de cada uno de los aspectos que se desarrollaron en la planificación y elaboración de este trabajo de grado.

A nuestras familias por su confianza y apoyo desinteresado en el transcurso de estos arduos, apasionantes y satisfactorios caminos de estudio.

A nuestra alma mater por la formación académica y personal infundida en los valores y en la ética que resaltan lo grande de nuestra universidad.

Al grupo de profesores del departamento de Matemáticas por su instrucción, sus conocimientos y por la formación matemática recibida.

Al Dr. Juan Miguel Velásquez y Mg. Oscar Fernando Soto por la diligencia y sugerencias en la revisión de este trabajo.

Finalmente, a nuestros asesores Mg. Fernando A. Benavides A. y Dr. John H. Castillo G., los más sinceros agradecimientos por su dedicación, orientación y sugerencias en el desarrollo de este trabajo de grado.

Hamilton Mauricio Ruiz
John Jairo López Santander

Universidad de Nariño
Septiembre 30 de 2013.

Este trabajo está dedicado a:
Mi madre Luz Marina Ruiz como reconocimiento a su labor y valiosos consejos.
Mauricio

Este trabajo está dedicado a:
Mis padres Leonor Santander, Leonel López y a mis hermanos.
John

Resumen

En el presente trabajo se realiza el análisis de una teoría donde han surgido nuevas aplicaciones para las matemáticas, denominada Teoría de Códigos. El fin de dicha teoría es la búsqueda de métodos confiables en la transmisión de información, en el sentido en que si un mensaje enviado se altera durante la transmisión, el receptor sea capaz de recuperar el mensaje original. A pesar de que son varías las ramas de las matemáticas que han aportado a esta teoría, el presente estudio se centra en los aportes hechos tanto por el Álgebra Lineal como por el Álgebra Abstracta. En este estudio se analizan de manera detallada los conceptos fundamentales de la Teoría de Códigos y se presentan algunos de los resultados más relevantes, con el fin de identificar los conceptos matemáticos subyacentes. También se presentan ejemplos para aclarar los conceptos estudiados, algunos de ellos mediante el uso de funciones del programa GAP.

Abstract

In the present work we make the analysis of a theory where there are new applications for mathematics, called Coding Theory. The purpose of this theory is the search for reliable methods with the transmission on of information, in the sense that if a sent message is altered during transmission, the receiver can recover the original message. Even though there are several branches of mathematics that have contributed to this theory, this study focuses on the contributions made by both the Linear Algebra and Abstract Algebra. In this study we analyze in detail the fundamental concepts of Coding Theory and we present some of the most relevant results with their respective proofs, with the purpose of identifying the underlying mathematical concepts. It also presents examples to clarify the studied concepts, some of them by using GAP program's functions.

Índice general

Introducción

La transmisión de información a través de un canal de comunicación, que puede ser una línea telefónica, comunicación vía satélite, lectores de códigos de barras, un dispositivo de almacenamiento como los CDs, entre otros, usualmente no es perfecta, en el sentido en que los mensajes recibidos pueden ser diferentes a los mensajes enviados. Esto es debido a errores, los cuales pueden ser de tipo humano, interferencia, deficiencia del equipo utilizado, entre otros. En este caso se dice que la comunicación se hace a través de un canal con ruido. En 1948, Claude Shannon publicó el artículo "*A Mathematical Theory of Communication*"[18], el cual dio origen a la Teoría de la Información, de donde surge la Teoría de Códigos Correctores de Errores, cuyo objetivo principal es construir códigos que permitan enviar la mayor información posible, detectar los errores producidos en la transmisión e incluso corregirlos. Para transmitir la información se utiliza un sistema de símbolos y reglas que cambian la forma del mensaje original, a lo que se denomina un código. El proceso de codificación consiste entonces en asignar a un mensaje que se desea transmitir, una cadena de símbolos, transformándolo al lenguaje del código. Estos mensajes codificados se envían a través de un canal de comunicación y asumiendo que se produzcan errores en la transmisión, el código debe permitir que el receptor sea capaz no sólo de detectarlos, sino de corregirlos y recuperar el mensaje original. El proceso de recuperar el mensaje se conoce como decodificación. Son varias las ramas de la matemática involucradas en la construcción de estos códigos, entre ellas están Teoría de Números, Teoría de Grupos, Teoría de Anillos, Teoría de Campos Finitos, Álgebra Lineal, entre otras. Un resultado de gran importancia es el Teorema de Shannon, ver [12, pag. 22, Teorema 7], que garantiza la existencia de buenos códigos, en el sentido que permiten transmitir información con una probabilidad de error tan pequeña como se desee. Sin embargo, este resultado no muestra un procedimiento que permita construir tales códigos. En la actualidad existen múltiples construcciones que verifican lo que el Teorema de Shannon predice.

Entre los códigos más importantes se encuentran los Códigos Lineales, los cuales son subespacios vectoriales de $\mathbb{F}_q^n$, donde $\mathbb{F}_q$ es el campo de Galois de orden q y de ahí que su construcción se base en los conceptos del Álgebra Lineal. Estos códigos son de gran importancia puesto que se pueden describir totalmente por medio de una matriz generadora, cuyas filas forman una base para el código. Otra de las ventajas es que facilitan los procesos de codificación y decodificación de los mensajes. Entre estos se tienen los códigos de Reed-Muller y los códigos de Golay que han sido utilizados por la NASA en la transmisión de fotografías. Otros códigos lineales de importancia son los códigos de Hamming que constituyen uno de los primeros códigos lineales inventados. Además, a Richard Hamming se le adjudica uno de los conceptos más importantes de la Teoría de Códigos, como lo es la noción de distancia de Hamming, que está relacionada con la capacidad de detección y corrección de errores, ver [20, pag. 9, Definición 2.3.1].

El presente trabajo se desarrolló como requisito parcial para optar al título de Licenciado en Matemáticas de la Universidad de Nariño, bajo la asesoría de los docentes Mg. Fernando Andrés Benavides y Dr. John Hermes Castillo. El objetivo principal es presentar de manera clara las nociones fundamentales de la Teoría de Códigos, en particular la definición formal de código, los parámetros de un código, los procesos de codificación y decodificación, así como presentar de manera breve pero precisa los códigos lineales en general, y en particular, algunos de los más relevantes de esta clase, en cuanto que facilitan los procesos de codificación y decodificación, como por ejemplo los códigos cíclicos, ver Capítulo 5. Por último, se presenta una construcción de un código no lineal obtenida a partir de la noción de conjunto B_h. En general, se da un enfoque algebraico al estudio de los códigos, sin embargo, también se utiliza un medio computacional, procurando describir los procedimientos matemáticos implícitos en la construcción de códigos de manera que se facilite su comprensión. En este sentido, se utilizan algunas funciones del Sistema de Álgebra Computacional Discreta GAP (System for Computational Discrete Algebra) incluidas en el paquete GUAVA, las cuales permiten realizar cálculos correspondientes a la Teoría de Códigos y así ejemplificar algunos de los conceptos tratados.

El trabajo se divide en cinco capítulos. El Capítulo 1 presenta conceptos del Álgebra Lineal y de Campos Finitos necesarios para el desarrollo de los temas tratados en este trabajo. En el Capítulo 2 se muestra de manera detallada como se da la transmisión de información mediante códigos correctores de errores, también se presenta la definición formal de código de bloque, los parámetros de un código, su capacidad de detección y corrección, entre otros. En el Capítulo 3 se aborda una clase muy interesante de códigos, denominados Códigos Lineales. Se describirá sus propiedades, los procesos de codificación y decodificación y algunas otras nociones relacionados con ellos. En el Capítulo 4 se estudia el problema fundamental de la Teoría de Códigos, describiendo en que consiste y presentando algunos de los resultados más conocidos. Finalmente, en el Capítulo 5 se presentan los Códigos Cíclicos, una clase muy importante de códigos lineales y una descripción breve de la relación entre la Teoría de Códigos y la Teoría de Números Aditiva, en específico la construcción de códigos a partir de la noción de conjunto B_h.

Capítulo 1

Preliminares

En la primera sección de este capítulo se introducen algunos conceptos básicos del Álgebra Lineal, particularmente de la Teoría de Espacios Vectoriales, necesarios para una mayor comprensión de los conceptos de la Teoría de Códigos. Las demostraciones tratadas en esta sección pueden consultarse en [8] y [16].

1.1. Álgebra Lineal

Definición 1.1. (Vector). Un *vector* de n componentes se define como un conjunto ordenado de n números escritos de la siguiente manera

$$(x_1, x_2, \ldots, x_n).$$

El conjunto de números elegido para formar el vector depende del contexto.
Las tres operaciones elementales de filas en una matriz se enuncian a continuación.

Definición 1.2. (Matriz). Una *matriz* A de $m \times n$ es un arreglo rectangular de mn números dispuestos en m renglones y n columnas de la siguiente manera

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2j} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots & & \vdots \\ a_{i1} & a_{i2} & & a_{ij} & \cdots & a_{in} \\ \vdots & \vdots & & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mj} & \cdots & a_{mn} \end{pmatrix}.$$

Cuando $m = n$, A es una matriz cuadrada de orden n.

Definición 1.3. (Operaciones elementales de fila). Sea A una matriz de tamaño $m \times n$. Los tres tipos de operaciones que pueden realizarse con las filas de una matriz A se denominan *operaciones elementales de fila* y son las siguientes:

1. $E_{rs}(c)$: operación elemental que consiste en sustituir la fila r por la resultante de sustraer a la fila r, c veces la fila s.
2. P_{rs}: operación elemental que consiste en intercambiar las filas r y s.
3. $M_r(c)$: operación elemental que consiste en sustituir la fila r por el resultado de multiplicar la fila r por una constante $c \neq 0$.

Definición 1.4. (Matriz de permutación). Llamaremos *matriz de permutación* a todo producto finito de matrices elementales de tipo P_{rs}, donde P_{rs} es una matriz elemental obtenida al intercambiar las filas r y s de la matriz identidad de orden n.

Por operaciones elementales de fila cualquier matriz puede transformarse en una *matriz escalonada* cuyas características se enuncian en la siguiente definición.

Definición 1.5. (Matriz escalonada). Una matriz U de orden $m \times n$, es escalonada si satisface las siguientes condiciones, en las que llamaremos *pivote* a la primera componente distinta de cero de cada fila no nula.

- Primero vienen las filas distintas de cero y luego las filas de ceros.
- Debajo de cada pivote, en la columna correspondiente, todas las componentes son cero.
- El pivote de cada fila está a la derecha del pivote de la fila anterior.

Cuando en una matriz escalonada U todos los pivotes son iguales a 1 y las demás componentes de la columna en la que se encuentran son iguales a cero, entonces se dice que U está en la *forma escalonada reducida por filas*, en cualquier otro caso, se dice que la matriz escalonada U está en la *forma escalonada por filas*.

Definición 1.6. (Rango). Sean A una matriz de orden $m \times n$ y U una matriz escalonada obtenida a partir de A por operaciones elementales de fila. El número de filas no nulas de U es llamado el *rango* de A y se denota por $\mathrm{r}(A)$.

La suma de vectores y la multiplicación de un vector por escalar permiten definir la estructura de los espacios vectoriales.

Definición 1.7. (Suma de vectores). Dos vectores de $\mathbb{R}^n$ se suman componente a componente y se obtiene como resultado un vector de $\mathbb{R}^n$, determinado de manera única, así

$$\text{si } \mathbf{x} = (x_1, x_2, \ldots, x_n) \text{ y } \mathbf{y} = (y_1, y_2, \ldots, y_n) \text{ entonces } \mathbf{x} + \mathbf{y} = (x_1 + y_1, x_2 + y_2, \ldots, x_n + y_n).$$

Definición 1.8. (Multiplicación por escalar). Sean $(x_1, x_2 \ldots, x_n)$ un vector de $\mathbb{R}^n$ y λ un escalar. El producto entre λ y $(x_1, x_2 \ldots, x_n)$ es el vector

$$\lambda(x_1, x_2, \ldots, x_n) = (\lambda x_1, \lambda x_2, \ldots, \lambda x_n).$$

Definición 1.9. (Espacio vectorial). Un *espacio vectorial* V sobre el campo K es un conjunto de objetos, denominados *vectores*, junto con dos operaciones binarias llamadas *suma* y *multiplicación por escalar* y que satisfacen los diez axiomas enumerados a continuación.

1. Si $\mathbf{x} \in V$ y $\mathbf{y} \in V$, entonces $\mathbf{x} + \mathbf{y} \in V$ (**cerradura bajo la suma**).
2. Para todo $\mathbf{x}, \mathbf{y}$ y $\mathbf{z}$ en V, $(\mathbf{x}+\mathbf{y})+\mathbf{z} = \mathbf{x}+(\mathbf{y}+\mathbf{z})$ (**ley asociativa de la suma de vectores**).
3. Existe un vector $\mathbf{0} \in V$ tal que para todo $\mathbf{x} \in V$, $\mathbf{x} + \mathbf{0} = \mathbf{0} + \mathbf{x} = \mathbf{x}$ (el $\mathbf{0}$ se llama **vector cero** o **idéntico aditivo**).
4. Si $\mathbf{x} \in V$, existe un vector $-\mathbf{x} \in V$ tal que $\mathbf{x}+(-\mathbf{x}) = \mathbf{0}$ ($-\mathbf{x}$ se llama **inverso aditivo** de $\mathbf{x}$).
5. Si $\mathbf{x}$ y $\mathbf{y}$ están en V, entonces $\mathbf{x} + \mathbf{y} = \mathbf{y} + \mathbf{x}$ (**ley conmutativa de la suma de vectores**).
6. Si $\mathbf{x} \in V$ y $\alpha \in K$, entonces $\alpha\mathbf{x} \in V$ (**cerradura bajo la multiplicación por un escalar**).
7. Si $\mathbf{x}$ y $\mathbf{y}$ están en V y $\alpha \in K$, entonces $\alpha(\mathbf{x} + \mathbf{y}) = \alpha\mathbf{x} + \alpha\mathbf{y}$ (**primera ley distributiva**).
8. Si $\mathbf{x} \in V$ y α y β en K, entonces $(\alpha + \beta)\mathbf{x} = \alpha\mathbf{x} + \beta\mathbf{x}$ (**segunda ley distributiva**).
9. Si $\mathbf{x} \in V$ y α y β en K, entonces $\alpha(\beta\mathbf{x}) = (\alpha\beta)\mathbf{x}$ (**ley asociativa de la multiplicación por escalares**).
10. Para cada vector $\mathbf{x} \in V$, $1\mathbf{x} = \mathbf{x}$, donde 1 es la identidad multiplicativa en K.

Definición 1.10. (Subespacio). Un subconjunto no vacío H de un espacio vectorial V es un *subespacio* de V si cumple las dos reglas de cerradura.

Aprovechando la estructura de un espacio vectorial V es posible definir cuando un subconjunto de V, es un subespacio vectorial.

Proposición 1.1. *Un subconjunto no vacío H de un espacio vectorial V sobre el campo K, es un subespacio si y sólo si la siguiente condición se satisface:*

$$\textit{si } \mathbf{x}, \mathbf{y} \in H \textit{ y } \lambda, \mu \in K, \textit{ entonces } \lambda\mathbf{x} + \mu\mathbf{y} \in H.$$

Definición 1.11. (Combinación lineal). Sea V un espacio vectorial sobre un campo K y $T = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ un subconjunto finito de vectores de V. Una *combinación lineal* de los vectores $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r$ es un vector de la forma

$$t_1\mathbf{v}_1 + t_2\mathbf{v}_2 + \cdots + t_r\mathbf{v}_r,$$

donde $t_1, t_2, \ldots t_r$ son escalares.

Proposición 1.2. *Sea V un espacio vectorial sobre un campo K y $T = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ un subconjunto de V. El conjunto formado por todas las posibles combinaciones lineales de $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r$ es un subespacio de V.*

Definición 1.12. (Subespacio generado). Sea V un espacio vectorial sobre un campo K y sea $T = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ un subconjunto de V. El subespacio H que consta de todas las combinaciones lineales de $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r$, es llamado el *subespacio generado* por T y se denota por $\operatorname{gen} T = \operatorname{gen}\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ o simplemente por $H = \langle T \rangle$.

Para un subespacio H de V, un subconjunto S de V, es un *conjunto generador* de H si $H = \langle S \rangle$. Además, si S es un subespacio de V, entonces $\langle S \rangle = S$.
Algunos subespacios importantes asociados a matrices de componentes de un campo (en este caso $\mathbb{R}$), serán utilizados en el Capítulo 3. Éstos se enuncian a continuación.

Definición 1.13. (Espacio fila). Si A es una matriz de orden $m \times n$ y de componentes reales, el subespacio de $\mathbb{R}^n$, generado por las filas de A es llamado el *espacio fila* de A y se denota por $R(A^\top)$. Es decir,

$$R(A^\top) = \operatorname{gen}\{A_1^\top, A_2^\top, \ldots, A_m^\top\}.$$

Definición 1.14. (Espacio nulo). Si $A \in \mathbb{R}_{m\times n}$. El conjunto solución del sistema homogéneo $A\mathbf{x} = 0$ es llamado el *espacio nulo* de A y se denota por $N(A)$, es decir

$$N(A) = \{\mathbf{z} \in \mathbb{R}^n : \mathbf{z} \text{ es solución del sistema } A\mathbf{x} = 0\}.$$

Veamos ahora los conceptos de dependencia lineal y base necesarios para determinar el mínimo número de elementos que generan un espacio vectorial dado.

Definición 1.15. Sea V un espacio vectorial sobre un campo K y $S = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ un subconjunto de V. Decimos que S es *linealmente dependiente* si existen escalares $c_1, c_2, \ldots, c_r$ no todos iguales a cero, tales que

$$c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_r\mathbf{v}_r = 0.$$

En caso contrario, decimos que S es *linealmente independiente*.

Definición 1.16. (Base). Sea V un espacio vectorial sobre un campo K. Una *base* para el espacio vectorial V es un subconjunto B de vectores de V que satisface dos condiciones:

1. B es linealmente independiente.
2. B genera al espacio V, es decir $V = \langle B \rangle$.

Proposición 1.3. *Si $B = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_r\}$ es una base para un espacio vectorial V sobre un campo K, cada vector de V puede representarse de manera única como una combinación lineal de los elementos de la base B.*

Definición 1.17. (Dimensión). Sea V un espacio vectorial sobre un campo K.

1. Si $V \neq \{0\}$, al número de elementos de cualquier base para V, lo llamaremos *dimensión* de V y lo denotaremos por $\dim(V)$.
2. Si $V = \{0\}$, diremos que la dimensión de V es cero.

La base y dimensión para los espacios vectoriales definidos en 1.13 y 1.14 se enuncia en las siguientes proposiciones.

Proposición 1.4. *(Base y dimensión para el espacio fila). Si $A \in \mathbb{R}_{m\times n}$ y U es una matriz escalonada obtenida a partir de A mediante operaciones elementales de fila, entonces*

1. $R(A^\top) = R(U^\top)$.
2. *Una base para $R(A^\top)$ está formada por los vectores que aparecen como las filas no nulas de la matriz escalonada U.*
3. $\dim(R(A^\top)) = r(A)$.

Proposición 1.5. *(Base y dimensión para el espacio nulo). Si $A \in \mathbb{R}_{m\times n}$ y U es una matriz escalonada obtenida a partir de A por operaciones elementales de fila, entonces*

1. $N(A) = N(U)$.
2. *Una base para $N(A)$ puede obtenerse como sigue: por cada variable libre en el sistema reducido $U\mathbf{x} = 0$, se construye un vector de la base, dando a esa variable el valor de 1, a las restantes variables libres el valor de cero y calculando en estos valores las variables básicas.*
3. $\dim(N(A)) = n - r(A)$.

Observación 1.1. De las proposiciones 1.4 y 1.5, para una matriz $A \in \mathbb{R}_{m\times n}$ se tiene

$$\dim(R(A^\top)) + \dim(N(A)) = n.$$

Otra de las nociones del Álgebra Lineal utilizadas en la Teoría de Códigos es la de complemento ortogonal. Antes de tratar esta noción veamos unas definiciones previas.

Definición 1.18. (Producto escalar). Sean $\mathbf{x} = (x_1, \ldots, x_n)$ y $\mathbf{y} = (y_1, \ldots, y_n)$ vectores de $\mathbb{R}^n$. El *producto escalar* de $\mathbf{x}$ por $\mathbf{y}$ se define como

$$\mathbf{x} \cdot \mathbf{y} = x_1 y_1 + \cdots + x_n y_n.$$

Definición 1.19. Sean $\mathbf{x}, \mathbf{y}$ vectores en $\mathbb{R}^n$. Decimos que $\mathbf{x}$ es *ortogonal* a $\mathbf{y}$ si $\mathbf{x} \cdot \mathbf{y} = 0$.

Definición 1.20. (Subespacios ortogonales). Si U y W son subespacios de $\mathbb{R}^n$, decimos que V es ortogonal a W, si cada vector de V es ortogonal a todos los vectores de W.

Observación 1.2. Para probar que dos subespacios V y W de $\mathbb{R}^n$ son ortogonales, es suficiente mostrar que cada vector de una base de V es ortogonal a todos los vectores de una base para W.

Proposición 1.6.
Si U y W son subespacios ortogonales de $\mathbb{R}^n$, entonces el único vector que tienen en común es el vector cero, es decir, $U \cap W = \{\mathbf{0}\}$.

Definición 1.21. (Complemento ortogonal). Sea H un subespacio de $\mathbb{R}^n$. El conjunto formado por todos los vectores de $\mathbb{R}^n$ que son ortogonales a todos los vectores de H es llamado *complemento ortogonal* de H y se denota por $H^\perp$. Es decir,

$$H^\perp = \{\mathbf{x} \in \mathbb{R}^n : \text{ para todo } \mathbf{h} \in H,\ \mathbf{x} \cdot \mathbf{h} = 0\}.$$

Además, para todo subespacio H de $\mathbb{R}^n$, su complemento ortogonal también es un subespacio de $\mathbb{R}^n$. Algunas propiedades importantes de los subespacios ortogonales se resumen en los siguientes teoremas.

Teorema 1.1. *Para toda matriz $A \in \mathbb{R}_{m\times n}$, se satisface*

1. $(R(A^\top))^\perp = N(A)$.
2. $(N(A))^\perp = R(A^\top)$.

Teorema 1.2. *Para todo subespacio H de $\mathbb{R}^n$ se tiene*

1. $(H^\perp)^\perp = H$.
2. $\dim(H) + \dim(H^\perp) = n$.

Por último, se cita la desigualdad de Cauchy-Schwarz, la cual será utilizada en el Capítulo 4.

Teorema 1.3. *(Desigualdad de Cauchy-Schwarz).*

$$|\mathbf{x}\cdot\mathbf{y}| \leq \| \mathbf{x} \| \| \mathbf{y} \| \text{ para todo } \mathbf{x},\mathbf{y} \in \mathbb{R}^n,$$

donde $|\mathbf{x}\cdot\mathbf{y}|$ *es el valor absoluto del producto escalar de* $\mathbf{x}$ *por* $\mathbf{y}$*, y* $\| \mathbf{x} \| = \sqrt{\mathbf{x}\cdot\mathbf{x}} = \sqrt{x_1^2 + \cdots + x_n^2}$ *es la longitud del vector* $\mathbf{x}$*.*

En el Capítulo 3 volveremos a retomar estos conceptos pero bajo la suposición de que el campo utilizado en las definiciones y resultados vistos en esta sección va hacer un campo finito, noción tratada más en detalle en la siguiente sección.

1.2. Campos Finitos

Debido a la importancia de los campos finitos en la Teoría de Códigos se presentan algunas propiedades básicas utilizadas en este trabajo. Las demostraciones de los teoremas tratados en esta sección serán omitidas, sin embargo, el lector interesado en las pruebas de los mismos puede consultar [15].

Definición 1.22. (Campo finito). Para un primo p, sean $\mathbb{F}_p$ el conjunto $\{0, 1, \ldots, p-1\}$ de enteros y $\varphi : \mathbb{Z}/\langle p\rangle \to \mathbb{F}_p$ la función definida por $\varphi([a]) = a$, para $a = 0, 1, \ldots, p-1$. Entonces $\mathbb{F}_p$, dotado con la estructura de campo inducida por φ, es un campo finito, llamado el *campo de Galois* de orden p.

Teorema 1.4. *Si* F *es un campo finito, entonces* F *tiene característica prima. Además, si* $\text{car}(F) = p$*, entonces* F *tiene* p^n *elementos, para algún entero positivo* n*.*

Un campo de división F de $f(x) \in K[x]$ es el campo más pequeño que contiene todas las raíces de f y a K.

Lema 1.1. *Si* F *es un campo finito con* q *elementos y* K *es un subcampo de* F*, entonces el polinomio* $x^q - x \in K[x]$ *se factoriza en* $F[x]$ *como*

$$x^q - x = \prod_{a\in F} (x - a)$$

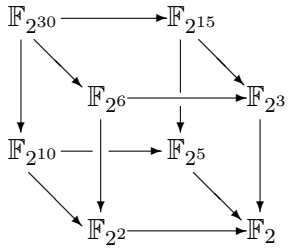
y F *es el campo de división de* $x^q - x$ *sobre* K*, por lo tanto para todo* $\alpha \in F$ *se tiene que* $\alpha^q = \alpha$*.*

Teorema 1.5. *(Existencia y unicidad de campos finitos) Para cada primo* p *y cada entero positivo* n*, existe un campo finito con* p^n *elementos. Cualquier campo finito con* p^n *elementos es isomorfo al campo de división de* $x^q - x$ *sobre* $\mathbb{F}_p$*.*

El campo finito del teorema anterior, se denota por $\mathbb{F}_q$, donde $q = p^n$ potencia del primo p y $\text{car}(\mathbb{F}_p) = p$.

Teorema 1.6. *(Criterios de subcampos) Sea $\mathbb{F}_q$ el campo finito con $q = p^n$ elementos. Cada subcampo de $\mathbb{F}_q$ tiene orden p^m, donde m es un divisor positivo de n. Recíprocamente, si m es un divisor positivo de n, entonces existe exactamente un subcampo de $\mathbb{F}_q$ con p^m.*

Ejemplo 1.1. Los subcampos del campo finito $\mathbb{F}_{2^{30}}$ se determinan encontrando los divisores positivos de 30. Las relaciones de contenencia entre éstos subcampos se muestra en el siguiente diagrama.



Para un campo finito $\mathbb{F}_q$, se denota por $\mathbb{F}_q^*$ el grupo multiplicativo de elementos no cero de $\mathbb{F}_q$. El siguiente resultado es una propiedad de este grupo.

Teorema 1.7. *Para cada campo finito $\mathbb{F}_q$, el grupo multiplicativo $\mathbb{F}_q^*$ de elementos no cero de $\mathbb{F}_q$ es un grupo cíclico.*

Definición 1.23. (Elemento primitivo). Un generador del grupo cíclico $\mathbb{F}_q^*$ se denomina un *elemento primitivo* de $\mathbb{F}_q$.

El número de elementos primitivos que contiene $\mathbb{F}_q^*$ esta dado por $\phi(q-1)$, donde ϕ es la función de Euler.

Veamos ahora dos formas de representar los elementos de un campo finito. La primera consiste en considerar el anillo factor $R = \mathbb{F}_q[x]/\langle p(x)\rangle$, con $p(x)$ un polinomio irreducible en $\mathbb{F}_q$ de grado d mayor a 1. Dado que $p(x)$ es irreducible sobre $\mathbb{F}_q$, entonces R es un campo finito con q^d elementos. Es decir,

$$R = \frac{\mathbb{F}_q[x]}{\langle p(x)\rangle} = \{r(x) + p(x) : grad(r(x)) < d\},$$

puede caracterizarse como $\mathbb{F}_{q^d}$, el conjunto de todos los polinomios sobre $\mathbb{F}_q$ de grado menor que d con la adición y multiplicación módulo $p(x)$. Además, si α es una raíz de $p(x)$ en un campo de división, entonces $\mathbb{F}_{q^d}$ puede ser visto también como el conjunto de todos los polinomios en α de grado menor que d, donde la adición y la multiplicación se realizan módulo $p(\alpha)$.

Ejemplo 1.2. El polinomio $p(x) = x^3 + x + 1$ es irreducible sobre $\mathbb{F}_2$, ya que no se puede expresar como el producto de factores lineales dado que $p(0) \neq 0$ y $p(1) \neq 0$. Como $q = 2$ y $d = grad(p(x)) = 3$ entonces

$$\mathbb{F}_2[x]/\langle x^3 + x + 1\rangle = \mathbb{F}_8.$$

Si α es una raíz de $p(x)$ en un campo de división, entonces los elementos de $\mathbb{F}_8$ se pueden representar como los ocho polinomios binarios en α, de grado menor o igual a 2, es decir

$$\mathbb{F}_8 = \{0, 1, \alpha, \alpha + 1, \alpha^2, \alpha^2 + 1, \alpha^2 + \alpha, \alpha^2 + \alpha + 1\}.$$

En esta representación, la adición de $\mathbb{F}_8$ no presenta mayor dificultad. No obstante, en la multiplicación surgen polinomios de grado mayor o igual que 3, los cuales deben ser reducidos módulo $p(\alpha)$ teniendo en cuenta que $p(\alpha) = 0$ y por lo tanto

$$\alpha^3 = \alpha + 1.$$

La segunda representación se basa en el hecho de que $\mathbb{F}_q^*$ es cíclico. Si β es un elemento primitivo de $\mathbb{F}_q$, entonces

$$\mathbb{F}_q = \{0, 1, \beta, \ldots, \beta^{q-2}\}.$$

En esta representación la multiplicación de dos elementos de $\mathbb{F}_q$ se realiza de la siguiente manera,

$$\beta^i \beta^j = \beta^{(i+j) \text{ módulo } (q-1)}.$$

Sin embargo, la suma no es muy clara como lo es con la primera representación.

Ejemplo 1.3. Sea β una raíz del polinomio $p(x) = x^3 + x + 1$, el cual es irreducible sobre $\mathbb{F}_2$. Como $\beta^3 = \beta + 1$ no es difícil probar que β es un elemento primitivo de $\mathbb{F}_8^*$. Por tanto, todo elemento de $\mathbb{F}_8^*$ puede expresarse en la forma β^k, para algún $k = 0, 1, \ldots, 6$. Las dos representaciones de los elementos de $\mathbb{F}_8$ se muestra en la siguiente tabla. Luego, si se desea realizar la multiplicación o suma de dos elementos de $\mathbb{F}_8$ se elige la representación de la primera o segunda columna respectivamente, según sea conveniente.

Potencia de β	Polinomio de grado menor que 3
0	0
1	1
β	β
β^2	β^2
β^3	$\beta + 1$
β^4	$\beta^2 + \beta$
β^5	$\beta^2 + \beta + 1$
β^6	$\beta^2 + 1$

Tabla 1.1: Tabla potencia-polinomio.

Una forma alternativa de representar los elementos de un campo finito consiste en asignar a cada polinomio en $\mathbb{F}_q$ una cadena que consta solo de sus coeficientes. Por ejemplo, para el elemento $\beta^5 = \beta^2 + \beta + 1$ de $\mathbb{F}_8$ del ejemplo anterior, podemos escribir simplemente $\beta^5 = 110$. En la Sección

5.1 se hablará más en detalle de esta representación.
El polinomio irreducible utilizado en la construcción de los campos de los ejemplos anteriores recibe un nombre especial. Veamos antes una definición.

Definición 1.24. (Polinomio minimal). Un *polinomio minimal* de un elemento $\alpha \in \mathbb{F}_{q^m}$ con respecto a $\mathbb{F}_q$ es un polinomio mónico no cero f de menor grado en $\mathbb{F}_q[x]$ tal que $f(\alpha) = 0$.

Dada la importancia de un elemento primitivo y su polinomio minimal en el desarrollo de la aritmética de los campos finitos, se presenta la siguiente definición.

Definición 1.25. (Polinomio primitivo). Un polinomio $f \in \mathbb{F}_q[x]$ de grado $m \geq 1$, se denomina un *polinomio primitivo* sobre $\mathbb{F}_q$, si es el polinomio minimal sobre $\mathbb{F}_q$ de un elemento primitivo de $\mathbb{F}_{q^m}$.

Además, para el cálculo de un polinomio minimal se cuenta con el siguiente resultado.

Teorema 1.8. *Sea $\alpha \in \mathbb{F}_{q^n}$. El polinomio minimal para α sobre $\mathbb{F}_q$ es*

$$irr(\alpha, \mathbb{F}_q) = (x - \alpha)(x - \alpha^q) \cdots (x - \alpha^{q^{d-1}}),$$

donde d es el menor entero positivo para el cual $\alpha^{q^d} = \alpha$.

La siguiente noción será utilizada en la Sección 5.1.

Definición 1.26. Las raíces de $x^n - 1$ en un campo de división $\mathbb{F}_{q^s}$ son llamadas las *raíces n-ésimas de la unidad* sobre $\mathbb{F}_q$.

Para $(n, q) = 1$, el subgrupo que consta de todas las raíces n-ésimas de la unidad es un subgrupo cíclico de orden n, bajo la multiplicación. Un elemento generador de este subgrupo se denomina una *raíz n-ésima primitiva de la unidad* sobre $\mathbb{F}_q$.

Teorema 1.9. *Sea β un elemento primitivo del campo de división $\mathbb{F}_{q^s}$ de $x^n - 1$. Entonces las $\phi(n)$ raíces n-ésimas primitivas de la unidad sobre $\mathbb{F}_q$ son precisamente los elementos*

$$\left\{ \beta^k : \, k = \frac{q^s - 1}{n} u, \, u < n, \, (u, n) = 1 \right\}.$$

En particular, $\beta^{(q^s-1)/n}$ es una raíz n-ésima primitiva de la unidad.

A continuación se describe un método para la factorización de $x^n - 1$ sobre $\mathbb{F}_q$. Para $(n, q) = 1$ el polinomio $x^n - 1$ sobre $\mathbb{F}_q$ tiene n distintas raíces, por lo tanto $x^n - 1$ es el producto de polinomios minimales de estas raíces, todos ellos distintos. El cálculo de los polinomios minimales se puede realizar haciendo uso del Teorema 1.8.
Para empezar, sea β un elemento primitivo de $\mathbb{F}_{q^s}$, donde s es el orden de q módulo n. Por el Teorema 1.9 una raíz n-ésima primitiva de la unidad esta dada por

$$w = \beta^{(q^s-1)/n}.$$

Por lo tanto, las raíces de $x^n - 1$ están dadas por

$$1, w, w^2, \dots w^{n-1} \tag{1.2.1}$$

Calculando los polinomios minimales para estas raíces y tomando los distintos polinomios obtenidos se tiene para $i = 0, \dots, n-1$, los *conjugados* de w^i, los cuales son

$$w^i, w^{iq}, w^{iq^2}, \dots w^{iq^{d-1}}, \tag{1.2.2}$$

donde d es el menor entero positivo para el cual $w^{iq^d} = w^i$. Como

$$w^{iq^d} = w^i \iff w^{iq^d - i} = 1 \iff n \mid iq^d - i \iff iq^d \equiv i \,(mód\, n),$$

entonces la condición $iq^d \equiv i \,(mód\, n)$, puede utilizarse para determinar todos los conjugados.
Por lo tanto, el polinomio minimal para las raíces de (1.2.2) es el producto

$$m_i(x) = (x - w^i)(x - w^{iq})(x - w^{iq^2}) \cdots (x - w^{iq^{d-1}}),$$

donde d es el menor entero positivo para el cual $iq^d \equiv i \,(mód\, n)$. El conjunto de exponentes dado en (1.2.2)

$$C_i = \{i, iq, \dots, iq^{d-1}\},$$

se conoce como la i-ésima *clase lateral ciclotómica* para q módulo n.

Ejemplo 1.4. Sea $h_{15} = x^{15} - 1$ sobre $\mathbb{F}_2$. Aquí $n = 15$ y $q = 2$. Un cálculo sencillo muestra que $s = 4$, por lo tanto el campo de división para h_{15} es $\mathbb{F}_{2^4} = \mathbb{F}_{16}$. Consideremos el polinomio primitivo $x^4 + x + 1$ para $\mathbb{F}_{16}$ sobre $\mathbb{F}_2$. Para cualquier β raíz de este polinomio se tiene que β es un elemento primitivo de $\mathbb{F}_{16}$ y de ahí que una raíz quinceava primitiva de la unidad es

$$w = \beta^{(q^s - 1)/n} = \beta.$$

Por lo tanto, las raíces de $x^{15} - 1$ son

$$1, \beta, \beta^2, \dots, \beta^{14}.$$

Los polinomios minimales para estas raíces son

$$\begin{aligned}
&m_0(x) = x + 1 \\
&m_1(x) = m_2(x) = m_4(x) = m_8(x) = x^4 + x + 1 \\
&m_3(x) = m_6(x) = m_9(x) = m_{12}(x) = x^4 + x^3 + x^2 + x + 1 \\
&m_5(x) = m_{10}(x) = x^2 + x + 1 \\
&m_7(x) = m_{11}(x) = m_{13}(x) = m_{14}(x) = x^4 + x^3 + 1
\end{aligned}$$

Por lo tanto la factorización de $x^{15} - 1$ sobre $\mathbb{F}_2$ es

$$x^{15} - 1 = (x + 1)(x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1)(x^2 + x + 1)(x^4 + x^3 + 1).$$

Otras nociones a tener en cuenta son.

Definición 1.27. (Ideal). Sea R un anillo. Un subconjunto no vacío I de R es un *ideal* si

- $a + b$ y $a - b$ pertenecen a I, para todo $a, b \in I$.
- $r \cdot a$ pertenece a I, para todo $r \in R$ y todo $a \in I$.

Definición 1.28. (Ideal principal). Un ideal I de un anillo R se denomina un *ideal principal* si existe un elemento $g \in I$ tal que $I = \langle g \rangle$, donde

$$\langle g \rangle = \{gr : r \in R\}.$$

El elemento g es llamado un generador de I, además, se dice que I es generado por g. Un anillo R se denomina un *anillo de ideales principales* si todo ideal de R es principal.

Teorema 1.10. *Los anillos* $\mathbb{Z}$, $\mathbb{F}_q[x]$ *y* $\mathbb{F}_q[x]/(x^n - 1)$ *son anillos de ideales principales.*

Capítulo 2

Códigos

Los medios de información tales como sistemas de comunicación y dispositivos de almacenamiento no son absolutamente confiables, en el sentido en que los mensajes transmitidos sean recibidos correctamente debido a que ruido u otro tipo de interferencia lo impide. Existe una variedad de mecanismos por los cuales se genera ruido, siendo las formas más comunes: ruido externo o interferencias (producido por el medio de transmisión), ruido interno o inherente (se presenta en los equipos electrónicos, son producidos únicamente por el receptor), ruido térmico (asociado al movimiento rápido y aleatorio de los electrones en un conductor producido por la agitación térmica), ruido atmosférico (se escucha en los receptores de comunicación, aún cuando no hay señal presente, la principal fuente son las tormentas eléctricas) y ruido creado por el hombre (o ruido industrial, aparece en ese afán del hombre por los procesos productivos). El ruido siempre está presente en estos sistemas y el efecto que produce en los mensajes es introducir errores que modifican los mensajes transmitidos. Ante este problema surge la Teoría de Códigos Correctores de Errores, cuyo principal objetivo es detectar y corregir los errores producidos por ruido. La transmisión de datos desde el punto de vista de la Teoría de Códigos supone las siguientes reglas:

- Emisor y receptor conocen la codificación de los mensajes a transmitir.
- Si ocurren errores, el receptor es capaz de detectarlos.
- Por un canal de comunicación, sólo palabras del nuevo lenguaje son enviadas.
- Los mensajes recibidos solo deben contener símbolos que se manejan en el canal, es decir, si se utilizan números en la codificación, no es posible que llegue un mensaje con algún símbolo distinto al aceptado por el emisor y receptor.

Ahora se analizará como es el proceso de comunicación bajo la suposición de que no se presentan errores en la transmisión. En primer lugar, se llamará *mensaje fuente* al conjunto de mensajes que se desean enviar, estos pueden estar en el lenguaje tradicional. Estos mensajes se

codifican, es decir, se les asigna una cadena de símbolos, los cuales son tomados de un conjunto llamado *alfabeto*. Esta asignación se denomina *codificación de la fuente*. Suponiendo que no se presentan errores en la transmisión, los mensajes codificados se envían al destino deseado a través de un *canal de comunicación*. Estos mensajes llegan al siguiente bloque del esquema de comunicación, denominado *decodificación de la fuente*. La función del decodificador es convertir los mensajes recibidos a mensajes fuente, ya que estos fueron codificados inicialmente. Una vez hecho esto, los mensajes son recibidos por el receptor tal y como se encontraban originalmente.
El esquema de comunicación descrito anteriormente se muestra en la Figura 2.1. Ver [20, p. 1].

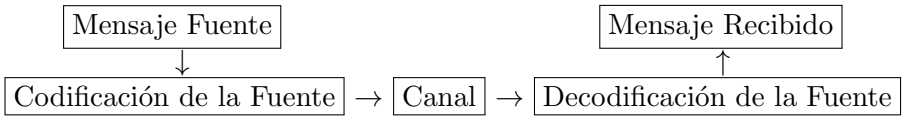


Figura 2.1: Esquema de comunicación.

Sin embargo, este esquema de comunicación tiene una falencia bastante importante originada por la suposición de que en la transmisión de los mensajes no hay errores. Es decir, si algún error por ruido se ha producido en la transmisión es muy probable que el receptor no se de cuenta y reciba un mensaje equivocado que lo lleve a una mala interpretación de la situación ó que note el error pero no pueda corregirlo. Para una mayor comprensión de lo dicho anteriormente, se analizará el siguiente ejemplo.
Supongamos que se desea enviar las coordenadas de movimiento de un objeto. Las palabras a enviar son: norte, sur, oriente y occidente, las cuales constituyen los mensajes fuente en el esquema de comunicación. Ahora se realiza la codificación de los mensajes fuente mediante asignaciones de cadenas de longitud 2, cuyos símbolos son tomados del conjunto $\{0, 1\}$ de la siguiente forma:

$$\text{norte} \to 00, \qquad \text{sur} \to 01, \qquad \text{oriente} \to 10, \qquad \text{occidente} \to 11.$$

Supongamos que el mensaje norte, el cual está codificado como 00 se envía a través de un canal de comunicación con ruido que genera un error en la transmisión de 00. Las posibles cadenas de longitud 2 que se pueden recibir teniendo en cuenta que ha ocurrido un error en el mensaje 00 son: 01 y 10. En cualquier caso, el decodificador de la fuente asignará a esta cadena bien sea sur u oriente, dependiendo de cual mensaje se reciba y así la comunicación falla. En la Figura 2.2 se ilustra el caso para el mensaje recibido 10.

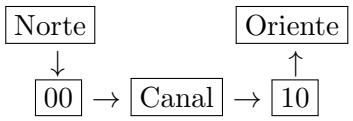


Figura 2.2: Ejemplo del proceso de comunicación.

Esta situación sugiere la modificación del esquema de comunicación. Para tal fin, se introducen dos nuevos bloques. El primero de ellos se denomina *codificación del canal*. Su función es realizar una segunda codificación a los mensajes codificados por la fuente, añadiendo siempre unos símbolos de "*redundancia*" que permitan detectar algún número determinado de errores. El segundo bloque es la *decodificación del canal*, sus funciones son detectar los errores, corregirlos, si es posible y una vez realizado esto, quitar los símbolos de redundancia añadidos por el codificador del canal. De aquí en adelante, el proceso continua con el decodificador de la fuente el cual ya se describió anteriormente. El nuevo *esquema de comunicación* se muestra a continuación. Ver [20, p. 2].

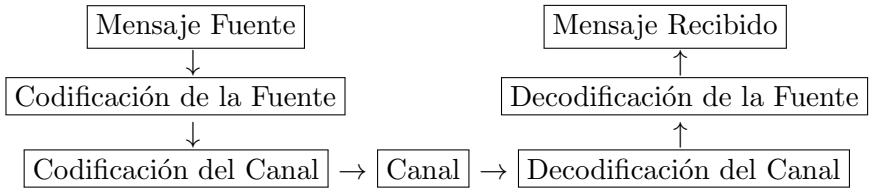


Figura 2.3: Esquema de comunicación modificado.

Para el ejemplo planteado anteriormente, la codificación del canal consiste en aumentar un símbolo de redundancia a cada cadena de longitud 2 de la siguiente forma:

$$00 \to 000, \qquad 01 \to 011, \qquad 10 \to 101, \qquad 11 \to 110.$$

Ahora si se envía el mensaje norte codificado por el canal como 000 y ocurre un error en la transmisión entonces las posibles cadenas recibidas son 001, 010 y 100. En cualquier caso, como ninguna de estas cadenas hace parte de las cadenas codificadas, entonces el decodificador del canal se da cuenta de que se han producido errores, es decir, detecta este error. Sin embargo, el decodificador no es capaz de corregir el error, porque por lo menos existen dos cadenas codificadas que difieren en exactamente una posición con cualquier cadena recibida 001, 010 y 100. Por ejemplo, si se recibe la cadena 010, entonces el decodificador no es capaz de identificar la cadena enviada, ya que esta podría haber sido 000, 011 ó 110, debido a que estas cadenas difieren en exactamente una posición con la cadena recibida. Ante esta situación se aumentan más símbolos de redundancia de la siguiente manera:

$$00 \to 00000, \qquad 01 \to 01111, \qquad 10 \to 10110, \qquad 11 \to 11001.$$

Supongamos que se envía 00000 y ocurre un error, entonces las posibles cadenas recibidas son 00001, 00010, 00100, 01000 y 10000. Sin pérdida de generalidad, supongamos que la cadena recibida es 01000. El decodificador de la fuente, por un lado detecta que hay errores porque la cadena recibida 01000 no hace parte de los mensajes codificados y por otro lado decodifica la cadena recibida a 00000, ya que entre la cadena recibida y las cadenas del mensaje distintas de 00000 hay por lo menos 2 errores, debido a la suposición inicial de que solamente un error ocurrió en la transmisión.

Una vez realizado este proceso, el decodificador del canal quita los símbolos de redundancia y envía al decodificador de la fuente la cadena 00 la cual es decodificada al mensaje norte. (ver Figura 2.4).

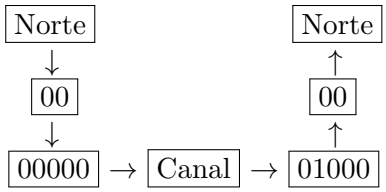


Figura 2.4: Ejemplo modificado.

Cabe resaltar que tanto el emisor como el receptor conocen la codificación de los mensajes, por lo tanto si en la transmisión de algún mensaje codificado ocurre el máximo número de errores admitido, el receptor es capaz de detectarlo, aunque no siempre corregirlo. La razón de esta cuestión es que a diferencia de la Criptografía, la Teoría de Códigos no esta interesada en mantener oculto los mensajes, su intención es detectar y corregir los errores producidos en la transmisión, además los mensajes que se transmiten sobre los canales de comunicación sólo tienen interés para el emisor y el receptor, en ellos no se maneja información confidencial como cuentas bancarias, claves secretas, etc. El caso en el cual se desee mantener oculta la información le concierne a la Criptografía.
Las anteriores cuestiones sugieren las siguientes preguntas.

- ¿Como se realizan los procesos de codificación y decodificación de los mensajes?
- ¿Como se detecta y corrige los errores producidos en la transmisión?

Estas y otras cuestiones se estudiarán a continuación.

En la primera sección de este capítulo se introducen las nociones fundamentales de los códigos de bloque que serán utilizadas a lo largo del presente trabajo. Posteriormente, en la Sección 2.2 se presenta la noción de Distancia de Hamming, la regla de decodificación usando la distancia de Hamming y otros conceptos claves. Seguidamente, en la Sección 2.3 se tratará los códigos equivalentes. Finalmente, en la Sección 2.4 se hablará de los códigos perfectos.

2.1. Códigos de Bloque

Esta sección centra su atención en los códigos de bloque y algunas características concernientes a ellos. Se empezará definiendo en términos formales lo que es un código de bloque.

Definición 2.1. (Códigos de bloque). Sean $A = \{a_1, a_2, \ldots, a_q\}$ un conjunto finito, llamado *alfabeto* y A^n el conjunto de todas las n-uplas sobre A. Cualquier subconjunto no vacío C de A^n es

llamado un *código de bloque q-ario*. Cada elemento en C se llama *palabra código* y si C contiene M elementos, entonces se dice que el código C tiene longitud n y tamaño M o simplemente que C es un (n, M)-código. Un subconjunto de un código C, es un *subcódigo* de C. Si C no es de bloque, se dice que C es de *longitud variable*.

Un ejemplo de código de longitud variable es el *código Morse*. El código C sobre el alfabeto $A = \{0, 1\}$ definido por $C = \{(0), (1, 0), (1, 1), (1, 0, 0), (1, 0, 1), (1, 1, 1)\}$ es también de esta clase. Sin embargo, este trabajo no se ocupará de estos códigos, por lo tanto, de ahora en adelante cada vez que se hable de código se entenderá código de bloque.
Normalmente el conjunto que conforma el alfabeto es tomado del campo de Galois de orden q, es decir $A = \mathbb{F}_q$, donde q es una potencia de un número primo. Cuando un código C tiene como alfabeto a $\mathbb{F}_2$, $\mathbb{F}_3$ ó $\mathbb{F}_4$ entonces C es un *código binario*, *ternario* o *cuaternario* respectivamente. Los códigos binarios son los más conocidos.

Ejemplo 2.1. Los siguientes son códigos de bloque.

- $C_1 = \{(0, 0, 0), (0, 1, 1), (1, 1, 0), (1, 1, 1)\}$ sobre $\mathbb{F}_2$ es un $(3, 4)$-código binario.
- $C_2 = \{(2, 0, 1, 0), (1, 0, 2, 0), (1, 1, 2, 2), (2, 2, 1, 1), (2, 1, 2, 1)\}$ sobre $\mathbb{F}_3$ es un $(4, 5)$-código ternario.
- $C_3 = \{(1, \alpha, \alpha), (\alpha + 1, 0, 1), (1, 0, \alpha), (\alpha, \alpha + 1, 1)\}$ sobre $\mathbb{F}_4$ es un $(3, 4)$-código cuaternario.

Notación 2.1. Cuando no haya confusión, se denotará un vector de la forma $(v_1, v_2, \ldots, v_n)$ simplemente como $v_1 v_2 \ldots v_n$.

Una de las características que define que tan bueno es un código es su tasa de información.

Definición 2.2. (Tasa de información). La *tasa de información* de un (n, M)-código q-ario C, se define por

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

Esta tasa se interpreta como el número de coordenadas que guardan información del mensaje original sobre el total de las coordenadas trasmitidas. De ahí que uno de los objetivos de la Teoría de Códigos es buscar códigos con alta tasa de información, digamos $R > \frac{2}{3}$ ó $R > \frac{3}{4}$.

Ejemplo 2.2. Para los códigos del Ejemplo 2.1 se tiene

$$R(C_1) = \frac{\log_2 4}{3} = \frac{2}{3},$$

$$R(C_2) = \frac{\log_3 5}{4} \approx 0{,}3662,$$

$$R(C_3) = \frac{\log_4 4}{3} = \frac{1}{3}.$$

Note que $|C_1| = |C_3|$, sin embargo C_1 tiene una mejor tasa de información.

Definición 2.3. (Canal). Un *canal discreto sin memoria* consta de un alfabeto de entrada $A = \{a_1, a_2, \ldots, a_q\}$, un alfabeto de salida $B = \{b_1, b_2, \ldots, b_t\}$ con $A \subseteq B$ y un conjunto de *probabilidades del canal* o *probabilidades de transición* $P(b_j \text{ recibido} \mid a_i \text{ enviado})$, que satisfacen:

$$\sum_{j=1}^{t} P(b_j \text{ recibido} \mid a_i \text{ enviado}) = 1,$$

para todo $i = 1, 2, ..., t$.
Aquí $P(b_j \text{ recibido} \mid a_i \text{ enviado})$ es la probabilidad condicional de que b_j sea recibido dado que a_i es enviado.
Además, si $\mathbf{c} = c_1 c_2 \cdots c_n$ y $\mathbf{d} = d_1 d_2 \cdots d_n$ son palabras de longitud n sobre A y B respectivamente, la probabilidad $P(\mathbf{d} \text{ recibido} \mid \mathbf{c} \text{ enviado})$ es

$$P(\mathbf{d} \text{ recibido} \mid \mathbf{c} \text{ enviado}) = \prod_{i=1}^{n} P(d_i \text{ recibido} \mid c_i \text{ enviado}).$$

La razón por la cual en la definición anterior aparece el término sin memoria, es debido a que el resultado de cualquier transmisión de una coordenada es independiente de los resultados de las transmisiones anteriores.

Definición 2.4. (Canal simétrico). Un *canal simétrico q-ario* es un canal sin memoria con alfabetos de entrada y salida iguales, de tamaño q y que cumple las siguientes condiciones.

1. Cada símbolo transmitido tiene la misma probabilidad $p < 1/2$ de recibirse en error.
2. Si un símbolo es recibido en error, entonces cada uno de los $q-1$ posibles errores es igualmente probable.

Observación 2.1. Dado un canal simétrico q-ario y $\mathbf{x} = x_1 x_2 \cdots x_n$, $\mathbf{c} = c_1 c_2 \cdots c_n$ palabras de longitud n, por la segunda parte de la definición anterior se tiene

$$P(x_i \mid c_i) = \begin{cases} 1 - p & \text{si } x_i = c_i, \\ \frac{p}{q-1} & \text{si } x_i \neq c_i. \end{cases}$$

Por la segunda parte de la Definición 2.3, para un canal simétrico q-ario se tiene

$$\begin{aligned} P(\mathbf{x} \text{ recibido } \mid \mathbf{c} \text{ enviado}) &= \prod_{i=1}^{n} P(x_i \text{ recibido } \mid c_i \text{ enviado}) \\ &= \left(\frac{p}{q-1}\right)^{e} (1-p)^{n-e}, \end{aligned}$$

donde e es el número de coordenadas en las cuales $\mathbf{x}$ y $\mathbf{y}$ difieren.

Observación 2.2. Uno de los más conocidos canales simétricos sin memoria es el *canal simétrico binario* (binary symmetric channel) BSC, el cual tiene como alfabeto a $\{0, 1\}$ y probabilidades de transición

$$P(1 \mid 0) = P(0 \mid 1) = p,$$
$$P(1 \mid 1) = P(0 \mid 0) = 1 - p.$$

Por lo tanto la probabilidad de recibir una coordenada en error es p, donde $p < 1/2$. Esta probabilidad es llamada la *probabilidad de cruce* del BSC. Ver Figura 2.5 (ver [20, p.7]).

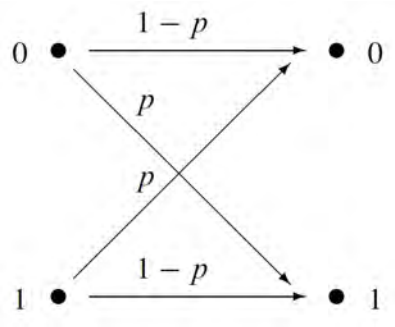


Figura 2.5: Probabilidad para el BSC.

Asumiendo que palabras de un código C son enviadas a través de un canal simétrico q-ario y que la menor cantidad de errores se han producido en la transmisión podemos encontrar la palabra probablemente enviada calculando las probabilidades del canal.

Ejemplo 2.3. Supongamos que palabras del código $C = \{000, 111\}$ son enviadas sobre un BSC con probabilidad de cruce $p = 0{,}05$. Supongamos que la palabra 110 es recibida y deseamos determinar cual fue la palabra código probablemente enviada. Para este fin se calcula las probabilidades de transición.

$$\begin{aligned} P(110 \mid 000) &= P(1 \mid 0)^2 \cdot P(0 \mid 0) \\ &= (0{,}05)^2 \cdot 0{,}95 \\ &= 0{,}002375, \end{aligned}$$

$$\begin{aligned} P(110 \mid 111) &= P(1 \mid 1)^2 \cdot P(0 \mid 1) \\ &= (0{,}95)^2 \cdot 0{,}05 \\ &= 0{,}045125. \end{aligned}$$

Como la segunda probabilidad es mayor que la primera, se concluye que 111 es la palabra código probablemente enviada.

Esta forma de decodificación recibe el nombre de *decodificación por máxima verosimilitud* (maximum likelihood decoding) MLD. Es decir, si una palabra $\mathbf{x}$ es recibida, la decodificación por máxima verosimilitud concluirá que $\mathbf{c}_x \in C$ es la palabra probablemente enviada si $\mathbf{c}_x$ maximiza las probabilidades del canal; i.e.,

$$P(\mathbf{x} \text{ recibido } \mid \mathbf{c}_x \text{ enviado}) = \max_{\mathbf{c}\in C} P(\mathbf{x} \text{ recibido } \mid \mathbf{c} \text{ enviado}).$$

Existen dos tipos de MLD, la *decodificación por máxima verosimilitud completa* CMLD y la *decodificación por máxima verosimilitud incompleta* IMLD. La regla completa elige una palabra arbitraria entre dos o más palabras con la misma probabilidad máxima con la palabra recibida mientras que la regla incompleta en la misma situación solicita la retransmisión del mensaje enviado. Otra regla de decodificación se verá en la próxima sección.

2.2. Distancia y Peso de Hamming

La noción de distancia de Hamming se denomina así en honor al matemático Richard Hamming, Profesor de la Universidad de Nebraska, Estados Unidos, quien introdujó el término para crear códigos capaces de corregir errores.

Definición 2.5. (Distancia de Hamming). Sean $\mathbf{x}, \mathbf{y}$ palabras de longitud n sobre un alfabeto A, la *distancia de Hamming* entre $\mathbf{x}$ y $\mathbf{y}$, denotada por $d(\mathbf{x}, \mathbf{y})$, se define como el número de posiciones en las cuales $\mathbf{x}$ y $\mathbf{y}$ difieren, es decir: si $\mathbf{x} = x_1x_2\cdots x_n$ y $\mathbf{y} = y_1y_2\cdots y_n$, entonces

$$d(\mathbf{x}, \mathbf{y}) = \#\{i : 1 \leq i \leq n,\, x_i \neq y_i\}. \tag{2.2.1}$$

Alternativamente se puede definir la distancia de Hamming por la siguiente expresión

$$d(\mathbf{x}, \mathbf{y}) = d(x_1, y_1) + d(x_2, y_2) + \cdots + d(x_n, y_n), \tag{2.2.2}$$

donde x_i y y_i se consideran palabras de longitud 1 y

$$d(x_i, y_i) = \begin{cases} 0 & \text{si } x_i = y_i, \\ 1 & \text{si } x_i \neq y_i. \end{cases}$$

Notación 2.2. Cuando el contexto sea claro y no haya confusión, se denotará una palabra de un código simplemente por una letra sin negrilla.

Ejemplo 2.4. Sea $A = \{0, 1, 2\}$, $x = 10112$ y $y = 20110$, entonces

$$\begin{aligned} d(x, y) &= d(1, 2) + d(0, 0) + d(1, 1) + d(1, 1) + d(2, 0) \\ &= 1 + 0 + 0 + 0 + 1 \\ &= 2. \end{aligned}$$

El siguiente resultado proporciona algunas propiedades referentes a la distancia de Hamming.

Teorema 2.1. *Sean x, y, z palabras de longitud n sobre A. Entonces*

1. $0 \leq d(x,y) \leq n$ y $d(x,y) = 0$ *si y sólo si* $x = y$.
2. $d(x,y) = d(y,x)$.
3. $d(x,z) \leq d(x,y) + d(y,z)$, *(desigualdad triangular).*

Demostración. Los items 1. y 2. son claros a partir de la definición de distancia de Hamming. Para 3. por la ecuación (2.2.2) es suficiente probar el resultado para $n = 1$. Se tienen dos casos:

- Si $x = z$, entonces por la parte 1 se tiene $d(x,y) \geq 0$ y $d(y,z) \geq 0$ y por lo tanto
$$d(x,y) + d(y,z) \geq 0 = d(x,z).$$
- Si $x \neq z$ entonces $y \neq x$ ó $y \neq z$. Si $y \neq x$, entonces $d(x,y) = 1$ y por lo tanto $d(x,z) \leq d(x,y) + d(y,z)$. Similarmente si $y \neq z$.

■

El concepto de distancia de Hamming permite definir una regla de decodificación bastante sencilla. Sea C un código, supongamos que palabras código son enviadas sobre un canal de comunicación. Si x es una palabra recibida, la *decodificación por distancia mínima* decodifica x a c_x, si $d(x,c_x)$ es mínima entre todas las palabras código, es decir

$$d(x,c_x) = \text{mín}\{d(x,c), c \in C\}. \tag{2.2.3}$$

Así como en el caso de la MLD, se distinguen 2 tipos de decodificación, *la decodificación por distancia mínima completa* y la *decodificación por distancia mínima incompleta*. Si una palabra x es recibida y 2 ó más palabras código satisfacen la ecuación (2.2.3), entonces la decodificación por distancia mínima completa elige una arbitrariamente, mientras que la incompleta pide retransmisión.

Ejemplo 2.5. Para el código $C = \{01101, 00011, 10110, 11000\}$. Usemos la decodificación por distancia mínima incompleta para decodificar las siguientes palabras recibidas.
a) 01111.
b) 11011.
a) Calculamos las distancias

$$d(01111, 01101) = 1,$$
$$d(01111, 00011) = 2,$$
$$d(01111, 10110) = 3,$$

$$d(01111, 11000) = 4.$$

Se decodifica 01111 a 01101.
b) Calculamos las distancias

$$d(11011, 01101) = 4,$$

$$d(11011, 00011) = 2,$$

$$d(11011, 10110) = 3,$$

$$d(11011, 11000) = 2.$$

Como hay dos distancias mínimas iguales a 2, entonces se pide retransmisión.
El siguiente teorema relaciona las dos reglas de decodificación tratadas anteriormente cuando se utiliza un canal simétrico q-ario con probabilidad de cruce $p < 1/2$.

Teorema 2.2. *Para un canal simétrico q-ario con probabilidad de cruce $p < 1/2$, la decodificación por distancia mínima es equivalente a la decodificación por máxima verosimilitud.*

Demostración. Sea C un (n, M)-código y $\mathbf{x}$ la palabra recibida durante la transmisión de un elemento de C. Para todo $\mathbf{c} \in C$ y todo $0 \leq i \leq n$,

$$d(\mathbf{x}, \mathbf{c}) = i \iff P(\mathbf{x} \text{ recibido} \mid \mathbf{c} \text{ enviado}) = \left(\frac{p}{q-1}\right)^i (1-p)^{n-i}.$$

Como $p < 1/2$ entonces

$$\left(\frac{p}{q-1}\right)^0 (1-p)^n > \left(\frac{p}{q-1}\right)^1 (1-p)^{n-1} > \left(\frac{p}{q-1}\right)^2 (1-p)^{n-2} > \cdots > \left(\frac{p}{q-1}\right)^n (1-p)^0.$$

Por definición, la decodificación por máxima verosimilitud decodifica $\mathbf{x}$ a $\mathbf{c} \in C$ si $P(\mathbf{x}$ recibido $\mid \mathbf{c}$ enviado) es máxima, i.e., tal que $d(\mathbf{x}, \mathbf{c})$ sea mínima (o solicita retransmisión si la decodificación incompleta es usada y $\mathbf{c}$ no es única). ■

Bajo las hipótesis del teorema anterior, la decodificación por distancia mínima es de mayor utilidad en el sentido en que los cálculos que se realizan son más sencillos que los cálculos hechos con la decodificación por máxima verosimilitud.

Definición 2.6. (Distancia mínima de un código). La *distancia mínima* de un código C, denotada por $d(C)$, se define por

$$d(C) = \text{mín}\{d(c, d) \,:\, c, d \in C,\ c \neq d\}.$$

Notación 2.3. Diremos que un código C de longitud n, tamaño M y distancia mínima d es un (n, M, d)-código.

Calcular la distancia de un código consiste en determinar la menor distancia de Hamming entre todos los pares distintos de elementos de C, sin embargo, si el tamaño del código es relativamente grande, esta comparación puede llevar bastante tiempo. GAP permite determinar la distancia de un código C de gran tamaño con el uso del comando `MinimumDistance(C)`. Para esto consideremos el siguiente ejemplo.

Ejemplo 2.6. El código $C = \{000000, 000111, 001110, 011100, 100011, 110001, 111000, 111111\}$, en GAP se escribe como se muestra en la primera linea del siguiente código fuente. Con el comando `AsSSortedList(C)` aparecen de forma explícita los elementos de C y con el comando `MinimumDistance(C)` se obtiene su distancia mínima.

```
> C:=ElementsCode(["111000", "011100", "001110", "000111", "100011", "110001",
  "000000", "111111" ],"example code", GF(2));
[ a (6,8,1..6)2..3 example code over GF(2)
> AsSSortedList(C);
[ [ 0 0 0 0 0 0 ], [ 0 0 0 1 1 1 ], [ 0 0 1 1 1 0 ], [ 0 1 1 1 0 0 ],
  [ 1 0 0 0 1 1 ], [ 1 1 0 0 0 1 ], [ 1 1 1 0 0 0 ], [ 1 1 1 1 1 1 ] ]
> MinimumDistance( C );
[ 2
```

Por simple verificación, un par de palabras códigos que determina la distancia del código son: 000111 y 001110.
La tercera linea del anterior código fuente significa que C es un $(6, 8)$-código con distancia mínima mayor o igual que 1 y menor o igual que 6. Los números 2 y 3 se refieren al radio de cubrimiento del código. Esta noción será tratada en la Sección 2.4.

Veamos ahora como la distancia de un código esta relacionada con la capacidad de detección y corrección de errores. Primero se introduce las siguiente definición.

Definición 2.7. (Código detector de t-errores). Un código C *detecta* t*-errores* si cada vez que se envía una palabra código y ocurren entre 1 y t errores durante la transmisión, la palabra resultante no es una palabra código. Un código C detecta exactamente t-errores, si este detecta t errores, pero no detecta $t + 1$ errores (es decir, existe al menos una palabra código la cual cambiando $t + 1$ coordenadas origina una nueva palabra código).

Ejemplo 2.7. El código ternario $C = \{000000, 000111, 111222\}$ es un detector de 2-errores ya que al enviar cualquier palabra código y cambiar cualquier 2 ó menos coordenadas de cada palabra, la palabra resultante no es una palabra código. Por ejemplo,

$$000000 \to 000111 \text{ necesita cambiar 3 coordenadas,}$$
$$000000 \to 111222 \text{ necesita cambiar 6 coordenadas,}$$
$$000111 \to 111222 \text{ necesita cambiar 6 coordenadas.}$$

De hecho, C detecta exactamente 2-errores, ya que al cambiar las 3 últimas coordenadas de la palabra código 000000 por unos, se obtiene la palabra código 000111.

La técnica usada en el ejemplo anterior para códigos de mayor tamaño puede llevar bastante tiempo. Un resultado que relaciona la distancia mínima y que simplifica en algo los cálculos se enuncia a continuación.

Teorema 2.3. *Un código C detecta exactamente t-errores si y sólo si $d(C) = t + 1$.*

Demostración. Supongamos que $d(C) = t + 1$. Si $c \in C$ y x es una palabra tal que

$$1 \leq d(c, x) \leq t < d(C),$$

entonces $x \notin C$, ya que de lo contrario la distancia entre c y x es menor que la distancia mínima y esto es una contradicción, por lo tanto C es un código detector de t-errores.
Como la distancia de C es $d(C) = t + 1$, entonces existen palabras código $c_1, c_2 \in C$, tales que $d(C) = d(c_1, c_2) = t + 1$. Cambiando esas $t + 1$ coordenadas de c_1 de tal forma que coincidan con c_2 se obtiene la palabra c_2, una palabra código. Luego, C no detecta $(t + 1)$-errores y por lo tanto C detecta exactamente t-errores.
Recíprocamente, cambiando t ó menos coordenadas de cualquier palabra $c \in C$, se forma una palabra x tal que $d(c, x) \leq t$. Por definición de detector de t-errores, la palabra $x \notin C$. Luego $d(C) \geq t + 1$.
Como C es un detector de exactamente t-errores entonces este no detecta $(t+1)$-errores, por lo tanto existen $c_1, c_2 \in C$ tales que $d(c_1, c_2) = t + 1$. Pero $d(C) \leq d(c_1, c_2) = t + 1$ y así $d(C) = t + 1$. ■

Si t-errores ocurren en la transmisión de una palabra código, entonces se dice que un error de tamaño t ha ocurrido. El resultado anterior permite determinar el número de errores que un código es capaz de detectar. Sin embargo, nada se ha dicho hasta ahora sobre la corrección de esos errores. Veamos primero una definición y después un resultado que resuelve esta inquietud.

Definición 2.8. (Código corrector de t-errores). Asumiendo que los empates son considerados como errores, un código C *corrige t-errores* si la decodificación por distancia mínima corrige todos los errores de tamaño t ó menos en cualquier palabra código. Un código C corrige exactamente t-errores, si éste corrige t-errores, pero no corrige $(t+1)$-errores. Dicho de otra forma, todos los errores de tamaño t son corregidos, pero al menos un error de tamaño $t+1$ es decodificado incorrectamente.

Ejemplo 2.8. Considere el código binario de repetición $C = \{000, 111\}$. Usando la decodificación por distancia mínima se tiene

- Si 000 es enviada y un error ocurre en la transmisión, entonces las posibles palabras recibidas son 001, 010 y 100. Aquí, primero hay una detección del error, ya que ninguna de estas palabras hace parte del código. En cualquier caso, la palabra recibida, será decodificada a 000, ya que la

decodificación por distancia mínima decodifica la de menor distancia entre la palabra recibida y las palabras del código. Similarmente se tiene para la palabra código enviada 111, bajo la consideración de que se ha cometido un error en la transmisión.

- Si se envía 000 y ocurren 2 errores en la transmisión entonces las posibles palabras recibidas son 011,110 y 101. Aunque el código es capaz de detectar los errores (ya que su distancia mínima es 3) no los corrige correctamente, ya que para cualquier palabra recibida 011,110 y 101, la decodificación por distancia mínima la decodificará a 111. Por lo tanto C es un código que corrige exactamente un error.

Un resultado que permite determinar el número de errores que es capaz de corregir un código en función de su distancia se presenta a continuación.

Teorema 2.4. *Un código C corrige exactamente t-errores si y sólo si $d(C) = 2t+1$ ó $d(C) = 2t+2$.*

Demostración. Supongamos que $d(C) = 2t + 1$ ó $d(C) = 2t + 2$. Sean $\mathbf{c} \in C$ una palabra enviada y $\mathbf{x}$ la palabra recibida. Si t ó menos errores ocurren en la transmisión entonces

$$d(\mathbf{x}, \mathbf{c}) \leq t.$$

Por contradicción, supongamos que existe $\mathbf{y} \in C$ con $\mathbf{c} \neq \mathbf{y}$ tal que

$$d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{c}) \leq t.$$

Por la desigualdad triangular se tiene

$$\begin{aligned} d(\mathbf{y}, \mathbf{c}) &\leq d(\mathbf{y}, \mathbf{x}) + d(\mathbf{x}, \mathbf{c}) \\ &\leq t + t \\ &\leq 2t \\ &< d(C). \end{aligned}$$

Lo cual es una contradicción. Luego, C es un código corrector de t-errores.
Supongamos que la longitud de C es n. Si $d(C) = d$ entonces existen palabras código $\mathbf{c}, \mathbf{c}' \in C$ tales que $d(\mathbf{c}, \mathbf{c}') = d$. Sin pérdida de generalidad, supongamos que $\mathbf{c}$ y $\mathbf{c}'$ difieren en las primeras d coordenadas. Si se envía $\mathbf{c}$ y se recibe la palabra $\mathbf{x}$ cuyas coordenadas están distribuidas de la siguiente manera

$$\mathbf{x} = \underbrace{x_1 \cdots\cdots x_{t+1}}_{\text{coinciden con c'}} \underbrace{x_{t+2} \cdots\cdots x_d}_{\text{coinciden con c}} \underbrace{x_{d+1} \cdots\cdots x_n}_{\text{coinciden con c y c'}}$$

se tiene

$$d(\mathbf{x}, \mathbf{c}') = d - (t + 1) \leq t + 1 = d(\mathbf{x}, \mathbf{c}).$$

Si $d = 2t+1$ entonces la decodificación por distancia mínima decodifica $\mathbf{x}$ incorrectamente como $\mathbf{c}'$.
Si $d = 2t+2$ entonces hay empate y por lo tanto error. En cualquier caso, C no es un corrector de $(t+1)$-errores.
Recíprocamente, para cualquier $\mathbf{c}_1, \mathbf{c}_2 \in C$ no es posible que $d(\mathbf{c}_1, \mathbf{c}_2) = w \leq 2t$, ya que de lo contrario al enviar $\mathbf{c}_1$ seria posible que la palabra recibida $\mathbf{x}$ con t errores con $\mathbf{c}_1$ y $w - t$ errores con $\mathbf{c}_2$ ubicados en las w coordenadas anteriormente citadas, la decodificación por distancia mínima decodifique $\mathbf{x}$ a $\mathbf{c}_2$ ó reporte empate ya que $w - t \leq t$. Luego $d(C) > 2t$.
Si $d(C) \geq 2t + 3 = 2(t+1) + 1$, entonces por el argumento anterior se tiene que C es un corrector de $(t+1)$-errores, lo cual contradice la hipótesis. Luego $d(C) = 2t + 1$ ó $d(C) = 2t + 2$. ■

Corolario 2.1. *$d(C) = d$ si y sólo si C corrige exactamente $\left\lfloor \frac{d-1}{2} \right\rfloor$-errores.*

Demostración. Supongamos que $d(C) = d$, entonces $d = 2t + 1$ ó $d = 2t + 2$.
Si $d = 2t + 1$ entonces

$$\left\lfloor \frac{d-1}{2} \right\rfloor = \left\lfloor \frac{2t+1-1}{2} \right\rfloor = \left\lfloor \frac{2t}{2} \right\rfloor = \lfloor t \rfloor = t.$$

Si $d = 2t + 2$ entonces

$$\left\lfloor \frac{d-1}{2} \right\rfloor = \left\lfloor \frac{2t+2-1}{2} \right\rfloor = \left\lfloor \frac{2t+1}{2} \right\rfloor = \left\lfloor t + \frac{1}{2} \right\rfloor = t.$$

Por el Teorema 2.4, C corrige exactamente t-errores, pero por lo anterior $t = \left\lfloor \frac{d-1}{2} \right\rfloor$. Recíprocamente, sea $t = \left\lfloor \frac{d-1}{2} \right\rfloor$, entonces por el Teorema 2.4 se tiene que $d(C) = 2t + 1$ ó $d(C) = 2t + 2$.
Si d es par, entonces $d = 2k$ para algún entero positivo k. Luego

$$\begin{aligned} 2t &= 2\left\lfloor \frac{d-1}{2} \right\rfloor = 2\left\lfloor \frac{2k-1}{2} \right\rfloor \\ &= 2\left\lfloor k - \frac{1}{2} \right\rfloor = 2k - 2 \\ &= d - 2. \end{aligned}$$

Es decir, $d = 2t + 2 = d(C)$.
Si d es impar, entonces $d = 2k + 1$ para algún entero positivo k. Entonces

$$\begin{aligned} 2t &= 2\left\lfloor \frac{d-1}{2} \right\rfloor = 2\left\lfloor \frac{2k+1-1}{2} \right\rfloor \\ &= 2\lfloor k \rfloor = 2k \\ &= d - 1. \end{aligned}$$

Es decir, $d = 2t + 1 = d(C)$. ■

Otro de los conceptos básicos en la Teoría de Códigos es el peso.

Definición 2.9. (Peso). Sea x una palabra en $\mathbb{F}_q^n$. El *peso* de x, denotado por $wt(x)$, se define como el número de coordenadas no nulas en x. Es decir

$$wt(x) = d(x, \mathbf{0}), \tag{2.2.4}$$

donde $\mathbf{0}$ es la palabra cero.

Una definición alternativa para el peso es la siguiente: si $\mathbf{x} = x_1x_2\cdots x_n$ es una palabra de longitud n sobre $\mathbb{F}_q$ entonces

$$wt(x) = wt(x_1) + wt(x_2) + \cdots + wt(x_n). \tag{2.2.5}$$

Definición 2.10. (Intersección de dos palabras). Si $\mathbf{x} = x_1x_2\cdots x_n$, $\mathbf{y} = y_1y_2\cdots y_n \in \mathbb{F}_q^n$, se define la *intersección de* $\mathbf{x}$ y $\mathbf{y}$ por

$$\mathbf{x} \cap \mathbf{y} = (x_1y_1, x_2y_2, \cdots, x_ny_n).$$

Lema 2.1.

1. *Para todo* $x, y \in \mathbb{F}_q^n$,
$$d(x, y) = wt(x - y).$$

2. *Para todo* $x, y \in \mathbb{F}_2^n$,
$$d(x, y) = wt(x) + wt(y) - 2wt(x \cap y).$$

Demostración.

1. Por la ecuación (2.2.5) podemos asumir que $x, y \in \mathbb{F}_q$, entonces $d(x, y) = 1$ ó $d(x, y) = 0$. Si $d(x, y) = 1$ entonces $x - y = a \in \mathbb{F}_q^*$ y así $w(x - y) = 1$. Si $d(x, y) = 0$ entonces $x = y$ lo cual es cierto si y sólo si $x - y = 0$ o equivalentemente $w(x - y) = 0$. Luego $d(x, y) = w(x - y)$.

2. Por la ecuación (2.2.5) es suficiente probar 2 para $x, y \in \mathbb{F}_2$. Esto se verifica en la Tabla 2.1 (ver [20, p.47]).

x	y	$x \cap y$	$w(x) + w(y) - 2w(x \cap y)$	$d(x, y)$
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	0	0

Tabla 2.1: Tabla de comparación para $x, y \in \mathbb{F}_2$.

■

Como $a = -a$ para todo $a \in \mathbb{F}_q$, cuando q es par, entonces el siguiente resultado es consecuencia inmediata de la parte 1 del Lema 2.1.

Corolario 2.2. *Para todo* $x, y \in \mathbb{F}_2^n$,

$$d(x, y) = w(x + y).$$

Otras propiedades del concepto de peso se resumen en el siguiente lema.

Lema 2.2.

1. *Si* $x, y \in \mathbb{F}_q^n$ *entonces*

$$wt(x) + wt(y) \geq wt(x + y) \geq wt(x) - wt(y).$$

2. *Si* $x, y \in \mathbb{F}_2^n$ *entonces* $wt(x \cap y) = x \cdot y (mód\ 2)$.

3. *Si* $x \in \mathbb{F}_2^n$ *entonces* $wt(x) = x \cdot x (mód\ 2)$.

4. *Si* $x \in \mathbb{F}_3^n$ *entonces* $wt(x) = x \cdot x (mód\ 3)$.

Demostración.

1. Por la ecuación (2.2.5) podemos suponer que $n = 1$. Si $x = y = 0$, la demostración es trivial. Si $x = y = 1$, entonces $wt(x) + wt(y) = 2$ y $wt(x) - wt(y) = 0$. Además, $x + y = 0$, si q es par ó $x + y = 2$ si q no es par. En cualquier caso se tiene

$$wt(x) + wt(y) \geq wt(x + y) \geq wt(x) - wt(y).$$

2. Supongamos que $n = 1$. Si $x = y = 0$ la demostración es directa. Si $x = y = 1$ entonces $wt(x \cap y) = 1$ y $x \cdot y = 1$, como se quería probar.

3. Es un caso particular de la parte 2, cuando $x = y$ y del hecho de que $wt(x \cap x) = x$.

4. Supongamos que $n = 1$. Si $x = y = 0$ la demostración es directa. Si $x = 1$ ó $x = 2$ se tiene $wt(x) = 1$ y en ambos caso $x \cdot x = 1 (mód\ 3)$.

■

Definición 2.11. (Peso de un código). Sea C un código. El *peso mínimo* (de Hamming) de C, denotado por $wt(C)$ es el mínimo de los pesos de todas las palabras no nulas de C. Es decir,

$$wt(C) = \text{mín}\{wt(c) \,:\, c \in C,\ c \neq \mathbf{0}\}.$$

Ejemplo 2.9. Para los códigos $C_1 = \{000, 101, 010, 111\}$ y $C_2 = \{000000, 000111, 001110, 011100\}$ se tiene que $wt(C_1) = 1 = d(C_1)$ y $wt(C_2) = 3 \neq d(C_2) = 2$.
Note que la distancia mínima de un código no siempre tiene que coincidir con el peso del código, como se puede observar en el código C_2.

Definición 2.12. (Código de peso constante). Un código C se llama *código de peso constante*, si el peso de todas las palabras de C es igual, es decir, $wt(c) = k$, para algún $k \in \mathbb{N}$ y para todo $c \in C$.

Notación 2.4. Un (n, M, d, w)-código es un código de longitud n, tamaño M, distancia mínima d y peso constante w.

Los códigos de peso constante se tratarán en el Capítulo 5.

Definición 2.13. (Enumeradores de peso). Si C es un (n, M)-código, se denota con A_k el número de palabras de peso k, es decir

$$A_k = \#\{c \in C \mid wt(c) = k\}.$$

Los números $A_0, \ldots, A_n$, se conocen como la *distribución de pesos* de C y la suma formal

$$W_C(s) = \sum_{k=0}^{n} A_k s^k,$$

es el *enumerador de peso* de C.

Ejemplo 2.10. Consideremos el código C del Ejemplo 2.6. Para éste se tiene $A_0 = A_6 = 1$, $A_1 = A_2 = A_4 = A_5 = 0, A_3 = 6$ y $W_C(s) = 1 + 6s^3 + s^6$.

2.3. Equivalencia de Códigos

Un (n, M)-código q-ario permite codificar M mensajes fuente. Sin embargo, ciertos tipos de códigos con los mismos parámetros facilitan este proceso de codificación. Para conocer más al respecto, se introducen algunos conceptos básicos.

Definición 2.14. (Código sistemático). Un (n, q^k)-código q-ario es llamado *sistemático* si hay k posiciones $i_1, i_2, \cdots, i_k$ tal que al restringir todas las palabras código en estas posiciones, se obtienen todas las q^k posibles palabras q-ario de longitud k. El conjunto $\{i_1, i_2, \cdots, i_k\}$ es llamado un *conjunto de información* y los símbolos de las palabras código en estas posiciones son llamados *símbolos de información*.

Ejemplo 2.11. El código binario

$$C = \{0000, 0110, 1001, 1010\},$$

es sistemático sobre la primera y la tercera posición. Por lo tanto, si la fuente es $\{00, 01, 10, 11\}$ se puede codificar como sigue

$$00 \longrightarrow \underline{0}0\underline{0}0,\ \ 01 \longrightarrow \underline{0}1\underline{1}0,\ \ 10 \longrightarrow \underline{1}0\underline{0}1,\ \ 11 \longrightarrow \underline{1}0\underline{1}0.$$

El tipo de codificación mostrado en el anterior ejemplo se denomina *codificación sistemática*. De manera similar, el proceso de codificación sistemática facilita el proceso de decodificación de los mensajes recibidos ya que a partir de ellos se puede leer directamente los mensajes fuente en el caso en que no hayan ocurrido errores en la transmisión.

Definición 2.15. (Códigos equivalentes). Dos (n, M)-códigos q-arios C_1 y C_2 son *equivalentes*, lo cual se denota por $C_1 \simeq C_2$, si existe una permutación σ de las n coordenadas y permutaciones $\pi_1, \pi_2, \ldots, \pi_n$ del alfabeto tal que

$$c_1c_2\cdots c_n \in C_1 \text{ si y sólo si } \pi_1(c_{\sigma(1)})\pi_2(c_{\sigma(2)})\cdots\pi_n(c_{\sigma(n)}) \in C_2.$$

En palabras, dos códigos son equivalentes si uno puede ser transformado en el otro por permutación de las coordenadas (mediante σ) y por permutación de los símbolos del código en cada coordenada de cada palabra código (mediante $\pi_1, \pi_2, \ldots, \pi_n$).

Ejemplo 2.12. Los códigos $C = \{00100, 00011, 11111, 11000\}$ y $C' = \{00000, 01101, 11011, 10110\}$ son equivalentes ya que al considerar $\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 4 & 3 & 2 & 5 \end{pmatrix}$, $\pi = (\pi_1, \pi_2, \pi_3, \pi_4, \pi_5)$ con $\pi_1 = \ldots = \pi_4$ la permutación identidad y $\pi_3 = (01)$ se tiene

$$C = \left\{ \begin{array}{c} 00100 \\ 00011 \\ 11111 \\ 11000 \end{array} \right\} \overset{\sigma}{\rightarrow} \left\{ \begin{array}{c} 00100 \\ 01001 \\ 11111 \\ 10010 \end{array} \right\} \overset{\pi}{\rightarrow} \left\{ \begin{array}{c} 00000 \\ 01101 \\ 11011 \\ 10110 \end{array} \right\} = C'.$$

Existe otra definición de equivalencia, pero primero veamos una definición preliminar.

Definición 2.16. (Transformación monomial). Sea σ una permutación de tamaño n. Para $i = 1, \ldots, n$, sea $\pi_i : \mathbb{F}_q \longrightarrow \mathbb{F}_q$ la multiplicación por un escalar no cero α_i en $\mathbb{F}_q$, es decir,

$$\pi_i(s) = \alpha_i s.$$

Entonces la función $\mu : \mathbb{F}_q^n \longrightarrow \mathbb{F}_q^n$ definida por

$$\mu(c_1c_2\cdots c_n) = \pi_1(c_{\sigma(1)})\pi_2(c_{\sigma(2)})\cdots\pi_n(c_{\sigma(n)}),$$

se llama *transformación monomial de grado n*. En palabras, una transformación monomial actúa sobre las n coordenadas en una permutación de éstas, seguida por la multiplicación de cada coordenada por un escalar no cero.

Cabe notar que en el caso $q = 2$ no hay operaciones de multiplicación no triviales. Por lo tanto, para este caso solo se consideran las permutaciones de las coordenadas.

Definición 2.17. (Múltiplo escalar equivalentes). Dos (n, M)-códigos C_1 y C_2 sobre $\mathbb{F}_q$ son *múltiplo escalar equivalentes* si existe una transformación monomial μ de grado n para la cual $\mu(C_1) = C_2$, donde $\mu(C_1) = \{\mu c : c \in C_1\}$. Entonces, C_1 y C_2 son múltiplo escalar equivalentes si estos son equivalentes en el sentido de la definición anterior, donde cada permutación π_i es una multiplicación por un escalar no cero.

Ejemplo 2.13. Consideremos el código ternario $C = \{000, 111, 022, 021\}$. Permutando la primera y la segunda posición, seguida por la multiplicación de la tercera posición por 2 se obtiene el código $C' = \{000, 102, 201, 202\}$, el cual es equivalente a C.

Observación 2.3. La transformación μ de la Definición 2.16 admite una representación matricial de la siguiente manera. Asignando a la permutación σ una matriz de permutación de orden n y a la multiplicación por el escalar no cero una matriz diagonal D de orden n. El producto $D \cdot P = M$, resulta en una matriz cuadrada de orden M, la cual tiene exactamente una entrada no cero en cada fila y en cada columna. Esta matriz recibe el nombre de *matriz monomial* y por construcción realiza la misma función que μ.

Para el ejemplo 2.13 se tiene la siguiente matriz monomial

$$D \cdot P = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 2 \end{pmatrix} = M.$$

Observación 2.4. Dos códigos que son múltiplo escalar equivalentes son también equivalentes. En efecto, como $\mathbb{F}_q$ es un campo, entonces si C_1 y C_2 son códigos que son múltiplo escalar equivalentes sobre $\mathbb{F}_q$ entonces son equivalentes ya que la aplicación π_i definida en 2.16 es una biyección de $\mathbb{F}_q$. Sin embargo, el recíproco de esta afirmación no es verdadero, como lo muestra el siguiente ejemplo.

Ejemplo 2.14. El código $C = \{04213, 12034, 23401, 30142, 41320\}$ sobre $\mathbb{F}_5$ es equivalente al código de repetición $C' = \{00000, 11111, 22222, 33333, 44444\}$ sobre $\mathbb{F}_5$. En efecto, aplicando las permutaciones $\pi_2 = (40321)$, $\pi_3 = (20134)$, $\pi_4 = (10243)$ y $\pi_5 = (30412)$ se tiene

$$C = \begin{Bmatrix} 04213 \\ 12034 \\ 23401 \\ 30142 \\ 41320 \end{Bmatrix} \xrightarrow{\pi_2} \begin{Bmatrix} 00213 \\ 11034 \\ 22401 \\ 33142 \\ 44320 \end{Bmatrix} \xrightarrow{\pi_3} \begin{Bmatrix} 00013 \\ 11134 \\ 22201 \\ 33342 \\ 44420 \end{Bmatrix} \xrightarrow{\pi_4} \begin{Bmatrix} 00003 \\ 11114 \\ 22221 \\ 33332 \\ 44440 \end{Bmatrix} \xrightarrow{\pi_5} \begin{Bmatrix} 00000 \\ 11111 \\ 22222 \\ 33333 \\ 44444 \end{Bmatrix} = C'.$$

Sin embargo C y C' no son múltiplo escalar equivalentes, ya que a cada permutación no se le puede asignar la multiplicación por un escalar no nulo.

Lema 2.3. *Si $0 \in A$ entonces cualquier (n, M)-código sobre el alfabeto A es equivalente (pero no necesariamente múltiplo escalar equivalente) a un código C' que contiene la palabra código cero.*

Demostración. En efecto, si $\mathbf{0} \in C$, entonces permutando todas las coordenadas de las palabras de C se obtiene un código equivalente C' con $\mathbf{0} \in C'$. Si $\mathbf{0} \notin C$ sea $\mathbf{x} \in C$ con $x_1, \ldots, x_k$ las coordenadas no nulas de $\mathbf{x}$. Considerando la permutación π del alfabeto que conserve los ceros y transforme las restantes k coordenadas no nulas en cero, se obtiene que $\pi(\mathbf{x}) = \mathbf{0}$. Aplicando π al resto de palabras de C se obtiene el código C', el cual es equivalente a C y tiene la palabra código $\mathbf{0}$. ■

Teorema 2.5. *Si C_1 y C_2 son equivalentes entonces $d(C_1) = d(C_2)$. Además, bajo la decodificación por distancia mínima y asumiendo un canal simétrico q-ario con $p < 1/2$, la probabilidad de decodificar un error es la misma para códigos equivalentes.*

Demostración. Teniendo en cuenta que la permutación de las coordenadas no afecta la distancia del código, podemos suponer que σ es la permutación identidad. Sean palabras $x, y \in C$ y $d(x, y) = d$. Supongamos que las coordenadas en las que x y y difieren son $i_1, \ldots, i_d$. Probemos que la imagen de x y y denotadas por x', y' respectivamente, difieren en exactamente esas posiciones y por lo tanto $d(x', y') = d$. Sean π_i para $1 \leq i \leq n$ las permutaciones del alfabeto. Para cada i_j con $j \in \{1, \ldots, d\}$ se tiene $\pi_i(x_{i_j}) \neq \pi_i(y_{i_j})$ y $\pi_i(x_{i_j}) = \pi_i(y_{i_j})$ si $j \notin \{1, \ldots, d\}$. Luego $d(x', y') = d$. Como x, y se eligieron arbitrariamente, entonces $d(C_1) = d(C_2)$. La segunda proposición es directa dado que la distancia de dos códigos equivalentes es igual y por lo tanto corrigen la misma cantidad de errores. ■

2.4. Códigos Perfectos

Una familia interesante de códigos estrechamente relacionados con la distancia mínima son los códigos perfectos. Veamos primero algunas definiciones previas.

Definición 2.18. (Esfera-Volumen). Sean x una palabra en A^n, donde $|A| = q$ y $r \geq 0$. La *esfera de radio r centrada en x* es el conjunto

$$S_q(x, r) = \{y \in A^n \,:\, d(x, y) \leq r\}.$$

El *volumen* $V_q(n, r)$ de la esfera $S_q(x, r)$ es el número de elementos de $S_q(x, r)$. Este volumen es independiente del centro.

Lema 2.4. *El volumen $V_q(n,r)$ de la esfera $S_q(x,r)$ esta dado por*

$$V_q(n,r) = \sum_{k=0}^{r} \binom{n}{k}(q-1)^k.$$

Demostración. Sea x un elemento fijo en A^n. Determinemos el número de elementos $y \in A^n$ que tienen distancia exactamente k con x, donde $k \leq n$. Las k posiciones en las cuales x y y difieren pueden ser obtenidas de $\binom{n}{k}$ maneras diferentes y en cada una de estas posiciones las coordenadas de y pueden elegirse de $q-1$ maneras distintas de la correspondiente coordenada en x. Por lo tanto el número de vectores y con distancia exactamente k con x es $\binom{n}{k}(q-1)^k$ y de ahí que el número total de vectores en $S_q(x,r)$ es

$$V_q(n,r) = \sum_{k=0}^{r} \binom{n}{k}(q-1)^k.$$

■

Ejemplo 2.15. Considere $x = 111 \in \mathbb{F}_3^3$ y $r = 2$, entonces la esfera de radio 2 centrada en x es

$$S_3(x,2) = \{111, 100, 010, 001, 110, 101, 011, 211, 121, 112, 221, 212, 122, 210, 201, 120, 102, 021, 012\}$$

y el volumen de $S_3(x,2)$ es

$$V_3(3,2) = \binom{3}{0}(3-1)^0 + \binom{3}{1}(3-1)^1 + \binom{3}{2}(3-1)^2 = 19.$$

Definición 2.19. (Radio de empaquetamiento y de cubrimiento). Sea $C \subset A^n$ un código. El *radio de empaquetamiento* de C es el mayor entero r para el cual las esferas $S_q(x,r)$ sobre cada palabra código x son disjuntas. El *radio de cubrimiento* de C es el menor entero s para el cual las esferas $S_q(c,s)$ sobre cada palabra código c cubren a A^n, es decir, para la cual la unión de las esferas $S_q(c,s)$ es A^n. Se denota el radio de empaquetamiento de C por $pr(C)$ y el radio de cubrimiento por $cr(C)$.

Ejemplo 2.16. Considere el código binario $C = \{1101, 0011\}$ entonces $pr(C) = 1$ dado que

$$S_2(1101,1) = \{1101, 0101, 1001, 1111, 1100\} \text{ y } S_2(0011,1) = \{1011, 0111, 0001, 0010, 0011\},$$

son disjuntos y $0001 \in S_2(0011,2) \cap S_2(1101,2)$. Por lo dicho anteriormente se tiene también que $cr(C) = 2$.

Veamos la relación que existe entre un código corrector de t-errores y las esferas de radio t sobre cada palabra código.

Teorema 2.6. *Considerando que los empates son reportados como errores, un código C corrige t-errores si y sólo si las esferas $S_q(c,t)$ sobre cada palabra código son disjuntas.*

Demostración. Supongamos que C es un corrector de t-errores y x es una palabra recibida con t errores como máximo. Por contradicción, supongamos que $x \in S_q(c_1, t) \cap S_q(c_2, t)$, con $c_1, c_2 \in C$ y $c_1 \neq c_2$. Sin pérdida de generalidad, supongamos que se envía c_2 y $d(x, c_1) < d(x, c_2)$. Luego x se decodifica a c_1, lo cual contradice el hecho de que C corrige t-errores. Incluso, si $d(x, c_1) = d(x, c_2)$, dado que los empates fueron asumidos como errores.
Recíprocamente, supongamos que x es una palabra recibida y que han ocurrido como máximo t errores en la transmisión. Entonces x pertenece a una única esfera $S_q(c, t)$, para alguna $c \in C$ y así la decodificación por distancia mínima decodificará x a c, y por lo tanto C corrige t-errores. ■

El siguiente resultado es consecuencia inmediata del teorema anterior.

Corolario 2.3. *Considerando que los empates son siempre reportados como errores, un código C corrige exactamente t-errores si y sólo si $pr(C) = t$.*

Demostración. Por contradicción, supongamos que $pr(C) > t$. Luego las esferas $S_q(c, pr(C))$ sobre cada $c \in C$ son disjuntas. Por el Teorema 2.6, C corrige $pr(C)$-errores, lo cual contradice el hecho de que C corrige exactamente t-errores. Luego $pr(C) = t$.
Recíprocamente, por el Teorema 2.6, C corrige t-errores. Supongamos que una palabra x es recibida y han ocurrido $(t+1)$-errores en la transmisión. Entonces x pertenece por lo menos a dos esferas $S_q(c, t+1)$ y $S_q(c', t+1)$, con $c, c' \in C$, $c \neq c'$, ya que $pr(C) = t$. Luego $d(x, c) = t + 1 = d(x, c')$ y como los empates son reportan como errores, entonces la decodificación por distancia mínima no puede decodificar la palabra recibida x. ■

El siguiente concepto es de gran interés en la Teoría de Códigos.

Definición 2.20. (Códigos perfectos). Un código C se dice *perfecto* si $pr(C) = cr(C)$. En palabras, un código $C \subset A^n$ es perfecto, si existe un número r para el cual las esferas $S_q(c, r)$ centradas sobre cada palabra código c son disjuntas y cubren a A^n, es decir

$$A^n = \bigcup_{c \in C} S_q(c, r).$$

Ejemplo 2.17. Uno de los códigos más famosos, es el *código q-ario de Hamming* $H_q(r)$, el cual es un $(\frac{q^r-1}{q-1}, q^n, 3)$-código. Para $q = 2$ y $r = 3$ se tiene el código binario de Hamming $H_2(3)$ el cual es un $(7, 16, 3)$-código.

$$H_2(3) = \left\{ \begin{array}{cccc} 0000000 & 0001101 & 1111111 & 1110010 \\ 1101000 & 1000110 & 0010111 & 0111001 \\ 0110100 & 0100011 & 1001011 & 1011100 \\ 0011010 & 1010001 & 1100101 & 0101110 \end{array} \right\}.$$

Como $H_2(3)$ corrige un error entonces $pr(H_2(3)) = 1$ y así las esferas $S_2(c, 1)$ sobre las palabras de $H_2(3)$ son disjuntas. Además

$$|H_2(3)| = 16, \quad |S_2(c, 1)| = 1 + \binom{7}{1} = 8 \quad \text{y} \quad \left|A^7\right| = 2^7 = 128.$$

Como $16 \cdot 8 = 128$, entonces las esferas $S_2(c, 1)$ cubren a A^7. Por lo tanto $H_2(3)$ es perfecto.

Teorema 2.7 (La condición de empaquetamiento de esferas)**.** *Sea C un (n, M, d)-código q-ario. C es un código perfecto si y sólo si $d = 2t + 1$ y*

$$M \cdot V_q(n, t) = q^n, \tag{2.4.1}$$

es decir,

$$M = q^n \Big/ \sum_{k=0}^{t} \binom{n}{k} (q-1)^k.$$

Demostración. Sean $x, y \in C$ tales que $d(C) = d(x, y)$. Por contradicción, supongamos que $d(x, y) = 2t + 2$. Eligiendo $t + 1$ coordenadas en las cuales x y y difieren formamos la palabra x' cuyas $t + 1$ coordenadas citadas anteriormente coinciden con x y las demás coinciden con y. Entonces $d(x, x') = t+1 = d(x', y)$. Como C es perfecto, entonces existe $c \in C$ tal que $x' \in S_q(c, t)$. Luego $d(x', c) \leq t$. Por la desigualdad triangular se tiene

$$\begin{aligned} d(x, c) &\leq d(x, x') + d(x', c) \\ &\leq t + 1 + t \\ &\leq 2t + 1. \end{aligned}$$

Pero $d(x, c) \geq 2t + 2$ ya que $x, c \in C$ y así $2t + 2 \leq 2t + 1$, lo cual es una contradicción. Luego $d(C) = 2t + 1$. Por otra parte, como $cr(C) = pr(C) = t$, entonces el conjunto de las M esferas de radio t sobre las palabras en C forman una partición de las q^n cadenas de longitud n y por lo tanto $M \cdot V_q(n, t) = q^n$.
Recíprocamente, si $d(C) = 2t + 1$ entonces C corrige exactamente t-errores y por el Corolario 2.3 se tiene $pr(C) = t$. Luego las esferas de radio t sobre cada palabra son disjuntas y por la condición de empaquetamiento de esferas, éstas cubren a A^n. Luego $pr(C) = cr(C) = t$. ■

Cabe resaltar que la existencia de tales números n, M y t que verifican la ecuación (2.4.1), no implica la existencia de códigos perfectos con esos parámetros. Sin embargo, para el caso binario, un resultado cuya prueba se encuentra en [17, p.164] presenta una condición que verifican los códigos binarios perfectos.

Teorema 2.8. *Si C es un $(n, M, 2t + 1)$-código binario perfecto entonces los números*

$$\lambda_s = \binom{n-s}{t+1-s} \Big/ \binom{2t+1-s}{t+1-s}, \tag{2.4.2}$$

son enteros, para todo $1 \leq s \leq t$.

Ejemplo 2.18. Los valores de $n = 90$, $M = 2^{78}$ y $d = 5$ verifican la ecuación (2.4.1) para $q = 2$, dado que

$$2^{78} \cdot \sum_{k=0}^{2} \binom{n}{k} = 2^{78} \cdot (1 + 90 + 4005) = 2^{78} \cdot (4096) = 2^{78} \cdot 2^{12} = 2^{90}.$$

Sin embargo, si existiera un código perfecto con estos parámetros se verificaría la ecuación (2.4.2), lo cual no ocurre como se muestra a continuación

$$\lambda_1 = \binom{89}{2} \Big/ \binom{4}{2} = \frac{1958}{3} \notin \mathbb{Z} \quad \text{y} \quad \lambda_2 = \binom{88}{1} \Big/ \binom{3}{1} = \frac{88}{3} \notin \mathbb{Z}.$$

Luego, no existe un $(90, 2^{78}, 5)$-código binario perfecto.

El problema de determinar todos los códigos perfectos no ha sido resuelto aún. Van Lint en 1967 mostró que las únicas soluciones para la condición de empaquetamiento de esferas para

$$n \leq 1000,\ d \leq 2001,\ (t \leq 1000),\ q \leq 100,$$

son

1. $(n, M, d) = (n, q^n, 1)$.
2. $(n, M, d) = (n, 1, 2n + 1)$.
3. $(n, M, d) = (2m + 1, 2, 2m + 1)$.
4. $(n, M, d) = (\frac{q^r-1}{q-1}, q^{n-r}, 3)$, $r \geq 2$.
5. $(n, M, d) = (23, 2^{11}, 7)$.
6. $(n, M, d) = (90, 2^{78}, 5)$.
7. $(n, M, d) = (11, 3^6, 5)$.

La primera solución corresponde a A^n, donde $|A| = q$. La segunda, al código de una sola palabra cuya distancia mínima es indefinida y se adopta convenir $d = 2n+1$. La tercera solución corresponde al código binario de repetición. Éstos son conocidos como los *códigos perfectos triviales*.
La cuarta solución corresponde a los códigos de Hamming, mientras que los de la quinta y séptima solución corresponden a los códigos de Golay. Por el ejemplo 2.18 no existen códigos perfectos con los parámetros de la solución seis.
Lo anterior muestra que no existen muchos códigos perfectos. Algunos problemas interesantes en este campo son
1. Salvo equivalencia, determinar todos los códigos perfectos con los mismos parámetros que los códigos perfectos de Hamming y de Golay.
2. Encontrar códigos perfectos sobre alfabetos cuyo tamaño no sea igual a la potencia de un número

primo.
3. Encontrar códigos perfectos para parámetros distintos a los encontrados por Van Lint en 1967.

Se finaliza esta sección con la siguiente definición.

Definición 2.21. (Código quasi-perfecto). Un código C se dice *quasi-perfecto* si $cr(C) = pr(C)+1$. En palabras, un código $C \subset A^n$ es quasi-perfecto si existe un número r para el cual las esferas $S_q(c, r)$ de radio r son disjuntas y las esferas $S_q(c, r+1)$ de radio $r+1$ cubren a A^n.

Ejemplo 2.19. Sea $C = \{0000, 1111\}$ sobre $\mathbb{F}_2$. Como $d(C) = 4$ entonces C no es perfecto y $pr(C) = 1$. Por otro lado, como

$$S_2(0000, 2) = \{0000, 0001, 0010, 0100, 1000, 0011, 0101, 1001, 0110, 1010, 1100\},$$

$$S_2(1111, 2) = \{1111, 1110, 1101, 1011, 0111, 1100, 1010, 0110, 1001, 0101, 0011\},$$

entonces $S_2(0000, 2) \bigcup S_2(1111, 2) = \mathbb{F}_2^4$ y así $cr(C) = pr(C) + 1$. Luego C es quasi-perfecto.

Algunos códigos de repetición de longitud par también son códigos quasi-perfectos.

Capítulo 3

Códigos Lineales

Una vez definidos los conceptos básicos en el Capítulo 2, se presenta a continuación una clase importante de códigos, denominados Códigos Lineales. Como se verá en el desarrollo de este capítulo, éstos presentan ciertas ventajas en los procesos de codificación y decodificación con respecto a otros códigos que no presentan la estructura algebraica de los códigos lineales.

3.1. Matriz Generadora

Un concepto que permite entender el proceso de codificación y decodificación con códigos lineales es la matriz generadora. Sin embargo introducimos primero algunos conceptos básicos sobre códigos lineales.

Definición 3.1. (Código lineal). Un *código lineal* L de longitud n sobre $\mathbb{F}_q$ es un subespacio vectorial de $\mathbb{F}_q^n$.

Como L es un subespacio vectorial, entonces la dimensión de L, es su dimensión como espacio vectorial sobre $\mathbb{F}_q$. De esta manera, si L tiene dimensión k sobre $\mathbb{F}_q$, se dice que L es un $[n, k]$-código lineal. Además, si L tiene distancia mínima d, L se denomina un $[n, k, d]$-código lineal.

Ejemplo 3.1. Los siguientes conjuntos son ejemplos de subespacios vectoriales sobre $\mathbb{F}_q$ y por lo tanto, códigos lineales.

- $L_1 = \mathbb{F}_q^n$ y $L_2 = \{\mathbf{0}\}$, donde $\mathbf{0}$ es el vector nulo en $\mathbb{F}_q^n$.
  L_1 se utiliza principalmente como conjunto de información, mientras que L_2 es el código lineal con un único elemento, en realidad, este código no es muy útil en la práctica.

- $L_3 = \{(\lambda, \ldots, \lambda) : \lambda \in \mathbb{F}_q\}$.
  L_3 es un código lineal de repetición, tiene dimensión $k = 1$, q elementos y distancia mínima n, es decir, L_3 es un $[n, 1, n]$-código lineal. Cuando el valor de n es relativamente grande,

la capacidad de este código para detectar y corregir errores es muy buena, de acuerdo a los Teoremas 2.3 y 2.4, respectivamente. Más precisamente, L_3 es un código que detecta $n-1$ errores y corrige $\lfloor \frac{n-1}{2} \rfloor$ errores.

- $L_4 = \{(0,0,0,0),(1,0,1,0),(0,1,0,1),(1,1,1,1)\}$.
  L_4 es un subespacio de $\mathbb{F}_2^4$. Este código tiene dimensión 2 (como espacio vectorial es generado por 1010 y 0101), longitud 4 y distancia mínima $d(L_4) = 2$, es decir, L_4 es un $[4,2,2]$-código lineal. De acuerdo al Teorema 2.3 L_4 detecta 1 error, pero, por el Teorema 2.4 L_4 no corrige errores.

- $L_5 = \{(0,0,0,0),(1,1,0,0),(2,2,0,0)\}$.
  L_5 es un subespacio de $\mathbb{F}_3^4$. Este código tiene dimensión 1 (como espacio vectorial es generado por 1100), longitud 4 y distancia mínima $d(L_5) = 2$, es decir, L_5 es un $[4,2,2]$-código lineal.

Observación 3.1. Un código que no sea lineal se denomina *código no lineal*.

Un código lineal definido sobre el campo $\mathbb{F}_2$, $\mathbb{F}_3$ ó $\mathbb{F}_4$ se denomina *código lineal binario*, *ternario* ó *cuaternario*, respectivamente.
En general, para un (n, M)-código arbitrario, determinar su distancia mínima puede llevar mucho tiempo, debido a que es necesario calcular $\binom{M}{2}$ distancias de Hamming. Sin embargo para códigos lineales este cálculo se simplifica considerablemente. El siguiente teorema trata esta cuestión.

Teorema 3.1. *Sea L un código lineal sobre $\mathbb{F}_q$. Entonces $d(L) = wt(L)$.*

Demostración. Por la Definición 2.6, existen $x, y \in L$ tales que $d(L) = d(x,y)$. Por la parte 1 del Lema 2.1 se tiene $d(x,y) = wt(x-y)$. Pero $x - y \in L$, entonces $wt(x-y) \geq wt(L)$, es decir $d(L) \geq wt(L)$.
Por otro lado, existe una palabra código no nula $z \in L$, tal que $wt(z) = wt(L)$. Pero por la ecuación (2.2.4), $wt(z) = d(z,0)$ y además, $d(z,0) \geq d(L)$, luego $wt(L) \geq d(L)$. Por lo tanto, $d(L) = wt(L)$. ■

El teorema anterior, permite el cálculo de la distancia de un código lineal L, a partir del cálculo de los $M-1$ pesos posibles de las palabras no nulas del código, donde M es el tamaño de L. El menor de estos valores, en vista del Teorema 3.1 es igual a la distancia de L.

Ejemplo 3.2. Considere el código lineal binario $L = \{0000, 1000, 0100, 1100\}$. Como

$$\begin{aligned} wt(1000) &= 1, \\ wt(0100) &= 1, \\ wt(1100) &= 2. \end{aligned}$$

Entonces $d(L) = 1$.

Observación 3.2. Un caso particular de la Proposición 1.1 cuando el campo es $K = \mathbb{F}_2$ implica que la condición suficiente y necesaria para que un subconjunto no vacío S del espacio vectorial $\mathbb{F}_2^n$ sobre $\mathbb{F}_2$ sea un subespacio vectorial es: si $\mathbf{x}, \mathbf{y} \in L$, entonces $\mathbf{x} + \mathbf{y} \in L$. Esto debido, a que tanto λ como μ de la Proposición 1.1, toman únicamente los valores de 0 y 1.

Ejemplo 3.3. Mostremos que el código $L = \{000, 100, 001, 101\}$ sobre $\mathbb{F}_2$ es lineal.
Por lo mencionado en el párrafo anterior, es suficiente mostrar que todas las sumas de cualquier dos elementos no nulos de L es nuevamente un elemento de L. Debido a que el espacio es conmutativo para la suma, es suficiente mostrar las sumas en un solo sentido. Las sumas posibles son:

$$\begin{aligned} 101 + 101 &= 000, \quad & 100 + 100 &= 000, \\ 100 + 001 &= 101, \quad & 100 + 101 &= 001, \\ 001 + 001 &= 000, \quad & 001 + 101 &= 100. \end{aligned}$$

Como todas estas sumas están en L, entonces L es un código lineal.

Una característica importante de los espacios vectoriales sobre campos finitos, es que todos ellos tienen un número finito de elementos y por lo tanto un número finito de bases, cuestión que no se cumple en los espacios vectoriales sobre campos infinitos. El siguiente resultado caracteriza el número de elementos de un espacio vectorial sobre $\mathbb{F}_q$, así mismo el número de bases diferentes que tiene.

Teorema 3.2. *Sea L un espacio vectorial sobre $\mathbb{F}_q$. Si $\dim(L) = k$ entonces*

1. *L tiene q^k elementos.*
2. *L tiene*

$$\frac{1}{k!}\prod_{i=0}^{k-1}(q^k - q^i) \tag{3.1.1}$$

bases diferentes.

Demostración.

1. Sea $\{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ una base para L, entonces

$$L = \{\lambda_1\mathbf{v}_1 + \cdots + \lambda_k\mathbf{v}_k : \lambda_i \in \mathbb{F}_q\}.$$

Como $|\mathbb{F}_q| = q$, entonces existen q elecciones diferentes para cada una de las $\lambda_i \in \mathbb{F}_q$, por lo tanto L tiene q^k elementos.

2. Sea $B = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ una base para L. Dado que B es linealmente independiente entonces el vector nulo $\mathbf{0}$ no pertenece a B. Luego, por la parte 1, para $\mathbf{v}_1$ existen $q^k - 1$ elecciones posibles. Como B es una base, entonces se debe cumplir que $\mathbf{v}_2 \notin \langle\mathbf{v}_1\rangle$. Es decir existen $q^k - q$ elecciones

para $\mathbf{v}_2$. Similarmente, para cada i, con $2 \leq i \leq k$, se debe cumplir $\mathbf{v}_i \notin \langle \mathbf{v}_1, \ldots, \mathbf{v}_{i-1} \rangle$, de manera que existen $q^k - q^{i-1}$ elecciones posibles para $\mathbf{v}_i$. Por el principio de la multiplicación, existen $\prod_{i=0}^{k-1}(q^k - q^i)$ k-uplas ordenadas distintas $(\mathbf{v}_1, \ldots, \mathbf{v}_k)$. Como el orden de los elementos $\mathbf{v}_1, \ldots, \mathbf{v}_k$ para una base es irrelevante, entonces el número de bases distintas es

$$\frac{1}{k!}\prod_{i=0}^{k-1}(q^k - q^i).$$

■

Del resultado anterior, un código C cuyo tamaño sea distinto de una potencia de un primo, no puede ser lineal, sin embargo, el recíproco es falso. El código $C = \{001, 100\}$ tiene 2 elementos (potencia de 2), sin embargo, este código no es lineal, ya que no contiene la palabra código $\mathbf{0}$. Además, despejando k de la ecuación (3.1.1), se tiene la expresión $k = \log_q \mid L \mid$, la cual permite determinar la dimensión de un código lineal a partir de su tamaño.

Corolario 3.1. *Sea L un código lineal de longitud n sobre $\mathbb{F}_q$. Entonces* $\dim(L) = \log_q |L|$.

Debido a la estructura de los códigos lineales, no resulta difícil construir uno, a partir de un subconjunto no vacío $S \subseteq \mathbb{F}_q^n$. La idea principal es encontrar el espacio generado por S. Este conjunto por la Definición 1.12 es un espacio vectorial y por lo tanto un código lineal.
Este procedimiento, se resume en los siguientes pasos:

- Formar la matriz A, cuyas filas son los elementos de S.
- Aplicar operaciones elementales de fila, para expresar A en la forma escalonada por filas.
- Identificar las filas no nulas de A, éstas por Álgebra Lineal, forman una base para el espacio fila de A, el cual es un subespacio vectorial.

El espacio generado por esas filas es el código L, es decir $L = \langle S \rangle$. De igual forma, este procedimiento permite determinar una base y por lo tanto la dimensión para el código lineal L, el cual está formado por las filas no nulas de la matriz A en la forma escalonada.

Ejemplo 3.4. Dado el subconjunto $S = \{1000, 0110, 0010, 0001, 1001\}$ de $\mathbb{F}_2^4$. Determinemos el código lineal L generado por S.
Aplicando los pasos anteriormente citados se tiene

$$A = \begin{pmatrix} 1000 \\ 0110 \\ 0010 \\ 0001 \\ 1001 \end{pmatrix} \xrightarrow{E_{51}(1)} \begin{pmatrix} 1000 \\ 0110 \\ 0010 \\ 0001 \\ 0001 \end{pmatrix} \xrightarrow{E_{54}(1)} \begin{pmatrix} 1000 \\ 0110 \\ 0010 \\ 0001 \\ 0000 \end{pmatrix}$$

Las filas no nulas de A son 1000, 0110, 0010 y 0001. Para determinar explícitamente los elementos de L, utilizamos la función en GAP, `GeneratorMatCode`, que genera los parámetros del código como longitud y distancia. Seguidamente, con la función `AsSSortedList`, GAP presenta los elementos de L. El código fuente se presenta a continuación.

```
> G := GeneratorMatCode([[1,0,0,0],[0,1,1,0],[0,0,1,0],[0,0,0,1]],
  "demo code", GF(2) );
[  a linear [4,4,1]0 demo code over GF(2)
> AsSSortedList( G);
[ [ 0 0 0 0 ], [ 0 0 0 1 ], [ 0 0 1 0 ], [ 0 0 1 1 ], [ 0 1 0 0 ],
  [ 0 1 0 1 ], [ 0 1 1 0 ], [ 0 1 1 1 ], [ 1 0 0 0 ], [ 1 0 0 1 ],
  [ 1 0 1 0 ], [ 1 0 1 1 ], [ 1 1 0 0 ], [ 1 1 0 1 ], [ 1 1 1 0 ],
  [ 1 1 1 1 ] ]
```

Como un código lineal es un subespacio vectorial, sus elementos quedan totalmente descritos por los elementos de una base. Una matriz formada por estos elementos recibe un nombre especial y es de gran utilidad en los procesos de codificación y decodificación. Para profundizar en esta idea, se presenta la siguiente definición.

Definición 3.2. (Matriz generadora). Sea L un $[n, k]$-código lineal. Una matriz G de tamaño $k \times n$ cuyas filas forman una base para L es llamada una *matriz generadora* para el código L.

Ejemplo 3.5. La matriz G presentada a continuación, es una matriz generadora para el código L del Ejemplo 3.4.

$$G = \begin{pmatrix} 1000 \\ 0110 \\ 0010 \\ 0001 \end{pmatrix}.$$

Observación 3.3. Si L es un $[n, k]$-código lineal con matriz generadora G, entonces

$$L = \{x \cdot G : \, x \in \mathbb{F}_q^k\}.$$

Es decir, una matriz generadora G para un $[n, k]$-código lineal, permite realizar el proceso de codificación del canal, multiplicando por la izquierda de G las cadenas $x \in \mathbb{F}_q^k$, realizando de esta manera la adición de los $n - k$ símbolos de redundancia a estas cadenas. Sin embargo, este proceso de codificación, puede llevarse de manera más eficiente, si se tiene una forma particular de la matriz generadora. Este propósito conduce a la siguiente definición.

Definición 3.3. (Matriz generadora en la forma estándar). Sea L un $[n, k]$-código. Una matriz generadora G para L en la forma $G = (I_k \mid A)$, se llama *matriz generadora en la forma estándar*.

Por el Teorema 3.2 existen $\frac{1}{k!}\prod_{i=0}^{k-1}(q^k - q^i)$ bases diferentes para un $[n,k]$-código lineal L y por lo tanto $\prod_{i=0}^{k-1}(q^k - q^i)$ matrices generadoras. Sin embargo, no todo código lineal tiene una matriz generadora en la forma estándar, como lo ilustra el siguiente ejemplo.

Ejemplo 3.6. Determine todas las matrices generadoras para el código lineal

$$L = \{000, 001, 100, 101\}.$$

Por el Corolario 3.1, $\dim(L) = \log_2 4 = 2$. Por Teorema 3.2, el número de bases diferentes para L es

$$\frac{1}{2!}\prod_{i=0}^{2-1}(2^2 - 2^i) = \frac{1}{2!}(2^2 - 1)(2^2 - 2) = 3.$$

Las bases de L son:

$$\{001, 100\}, \quad \{001, 101\}, \quad \{101, 101\}.$$

Las matrices generadoras para L son

$$\begin{pmatrix} 001 \\ 100 \end{pmatrix}, \begin{pmatrix} 100 \\ 001 \end{pmatrix}, \begin{pmatrix} 001 \\ 101 \end{pmatrix}, \begin{pmatrix} 101 \\ 001 \end{pmatrix}, \begin{pmatrix} 100 \\ 101 \end{pmatrix}, \begin{pmatrix} 101 \\ 100 \end{pmatrix}.$$

Como se puede observar, ninguna de estas matrices está en la forma estándar.

Esta dificultad se puede solucionar retomando el concepto de códigos equivalentes y haciendo uso del siguiente resultado.

Teorema 3.3. *Cualquier código lineal L es equivalente a un código lineal L' con matriz generadora en la forma estándar.*

Demostración. Supongamos que G es una matriz generadora para el código L que no está expresada en la forma estándar. Por operaciones elementales de fila, G puede expresarse en la forma escalonada reducida por filas. Permutando las columnas con pivote es posible formar la matriz identidad de orden k, donde $\dim(C) = k$. El resto de columnas se ubican enseguida de la matriz identidad formada, obteniendo así una matriz generadora G' en la forma estándar para un código L', el cual es equivalente al código L. ■

Para el Ejemplo 3.6, elijamos la matriz generadora en la forma escalonada reducida por filas

$$G = \begin{pmatrix} 100 \\ 001 \end{pmatrix}.$$

Permutando la segunda columna por la tercera se obtiene la matriz generadora en la forma estándar

$$G' = \begin{pmatrix} 100 \\ 010 \end{pmatrix}.$$

Las filas de esta matriz generan el código lineal $L' = \{000, 010, 100, 110\}$, el cual es equivalente al código L.
Si G está en la forma estándar, entonces L es sistemático en las primeras k coordenadas. En este caso codificar y decodificar resulta bastante sencillo.
Si $x \in \mathbb{F}_q^k$ entonces x se codifica como $x \cdot G = x \cdot (I_k \mid A) = (x \mid x \cdot A)$, siendo $x \cdot A$, los símbolos de redundancia. Similarmente, si $(x \mid x \cdot A)$ es recibido entonces esta palabra se decodifica a x.

Ejemplo 3.7. La matriz G mostrada a continuación es una matriz generadora en la forma estándar para un código lineal L.

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Veamos como es el proceso de codificación. Para cada $\mathbf{x} = x_1x_2x_3x_4$ en $\mathbb{F}_2^4$

$$\begin{aligned} \mathbf{x} \cdot G &= \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \\ &= \begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_2+x_3+x_4 & x_1+x_3+x_4 & x_1+x_2+x_4 \end{pmatrix}. \end{aligned}$$

Note que las primeras cuatro coordenadas del vector $\mathbf{x}G$ son iguales a la palabra $\mathbf{x}$, lo cual proporciona una ventaja a la hora de decodificar un mensaje recibido, dado que la información está precisamente en esas posiciones.

De manera general se presenta el siguiente teorema.

Teorema 3.4. *Sea L un $[n, k]$-código lineal. Para cualquier k coordenadas dadas, existe un código equivalente a L que es sistemático en estas posiciones.*

Demostración. Sea G una matriz generadora para L. Por operaciones elementales de fila, se expresa G en la forma escalonada reducida por filas. Sean $\mathbf{c}_{i_j}$, para $j = 1, \ldots, k$, las columnas con pivote de la matriz escalonada reducida. L es sistemático en estas posiciones. Sean $v_1, v_2, \ldots, v_k$ las coordenadas de las palabras del código L. Por permutación de las columnas $\mathbf{c}_{i_j}$, en las columnas $\mathbf{c}_{v_1}, \mathbf{c}_{v_2}, \ldots, \mathbf{c}_{v_k}$, se obtiene una matriz generadora G' para un código L' equivalente al código L, que además, por construcción es sistemático en estas k posiciones. ■

3.2. El Código Dual

Dado que un código lineal es un subespacio vectorial, por la Definición 1.21, su complemento ortogonal también es un código lineal. Este código y algunas de sus propiedades serán tratadas en la presente sección.

Definición 3.4. (Código dual). Sea L un $[n, k]$-código lineal. El conjunto

$$L^{\perp} = \{\mathbf{x} \in \mathbb{F}_q^n : \mathbf{x} \cdot \mathbf{c} = \mathbf{0} \text{ para todo } \mathbf{c} \in L\}$$

es el *código dual* de L.

Dicho de otra forma, el código dual es el complemento ortogonal del código lineal L. Algunas características de los códigos duales se establecen en el siguiente teorema.

Teorema 3.5. *Sea L un $[n, k]$-código lineal sobre $\mathbb{F}_q$.*

1. *Si G es una matriz generadora para L entonces*

$$L^{\perp} = \{\mathbf{x} \in \mathbb{F}_q^n : \mathbf{x} \cdot G^{\top} = \mathbf{0}\}.$$

2. *$L^{\perp}$ es un $[n, n-k]$-código lineal, por lo tanto $dim(L) + dim(L^{\perp}) = n$.*

3. *$(L^{\perp})^{\perp} = L$.*

Demostración. Para la primera parte, se probará que el conjunto de la Definición 3.4, es igual al conjunto

$$M = \{\mathbf{x} \in \mathbb{F}_q^n : \mathbf{x} \cdot G^{\top} = \mathbf{0}\}.$$

Sea $\mathbf{x} \in L^{\perp}$, por definición $\mathbf{x}$ es ortogonal a cada palabra código de L, en particular es ortogonal a las filas de la matriz generadora. Luego $L^{\perp} \subseteq M$. Para $\mathbf{x} \in M$, sea $\mathbf{c} \in L$, determinemos el valor de $\mathbf{x} \cdot \mathbf{c}$. Sean $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ las filas de G, por definición de matriz generadora $\mathbf{c} = \lambda_1\mathbf{v}_1 + \lambda_2\mathbf{v}_2 + \cdots + \lambda_k\mathbf{v}_k$ para algunos $\lambda_1, \lambda_2, \ldots, \lambda_k \in \mathbb{F}_q$. Luego

$$\begin{aligned} \mathbf{x} \cdot \mathbf{c} &= \mathbf{x} \cdot (\lambda_1\mathbf{v}_1 + \lambda_2\mathbf{v}_2 + \cdots + \lambda_k\mathbf{v}_k) \\ &= \lambda_1\mathbf{x} \cdot \mathbf{v}_1 + \lambda_2\mathbf{x} \cdot \mathbf{v}_2 + \cdots + \lambda_k\mathbf{x} \cdot \mathbf{v}_k \\ &= \mathbf{0}. \end{aligned}$$

Luego $M \subseteq L^{\perp}$ y así $L^{\perp} = M$.
Para la segunda parte, como L es el espacio fila y $L^{\perp}$ es el espacio nulo de la matriz G por la observación 1.17 se tiene el resultado deseado.
Para 3, usando la parte 2 se tiene $\dim((L^{\perp})^{\perp}) = n - (n-k) = k$. Pero $L \subseteq (L^{\perp})^{\perp}$ y ambos tienen la misma dimensión k, por lo tanto ellos tienen que ser iguales, es decir $(L^{\perp})^{\perp} = L$. ■

Ejemplo 3.8. Determinemos el código dual $L^\perp$ del código lineal binario $L = \{000, 101, 010, 111\}$. Como $\dim(L) = \log_2 4 = 2$ entonces dos elementos linealmente independientes de L son 101 y 010. Por lo tanto una matriz generadora para L es

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

Ahora, determinemos una base para el espacio nulo del sistema $\mathbf{x} \cdot G^\top = \mathbf{0}$, para $\mathbf{x} = x_1x_2x_3 \in \mathbb{F}_2^3$. El generado por estos elementos por definición es el código dual $L^\perp$ de L.

$$\mathbf{x} \cdot G^\top = \begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} x_1 + x_3 & x_2 \end{pmatrix} = \begin{pmatrix} 0 & 0 \end{pmatrix}.$$

De ahí que una base para $L^\perp$ es 101 y por lo tanto $L^\perp = \{000, 101\}$.

Note que L y $L^\perp$ son subespacios ortogonales y por lo tanto por la Proposición 1.6 su intersección es el vector nulo, sin embargo $L \cap L^\perp \neq \{000\}$. La contradicción surge por el campo elegido en estos espacios, el cual es finito como lo hemos visto. Esta idea conduce a la siguiente definición.

Definición 3.5. (Código auto-ortogonal y auto-dual). Sea L un código lineal.

1. L es *auto-ortogonal* si $L \subseteq L^\perp$.
2. L es *auto-dual* si $L = L^\perp$.

Ejemplo 3.9.

1. Considere el código lineal generado por el conjunto $S = \{01201\} \subset \mathbb{F}_3^5$. Este código es $L = \{00000, 01201, 02102\}$ y es auto-ortogonal, ya que

   $$01201 \cdot 01201 = 0, \quad 01201 \cdot 02102 = 0, \quad 02102 \cdot 02102 = 0.$$

   Note que este código no es auto-dual, ya que la palabra $00111 \in L^\perp$, pero $00111 \notin L$.

2. El $[4, 2]$-código lineal binario $L = \{0000, 1100, 0011, 1111\}$ es auto-dual. Para mostrarlo, sea G la matriz generadora para L formada por las filas 1100 y 0011. Determinemos el código dual, lo cual es equivalente a encontrar una base que lo genere. Sea $\mathbf{x} = x_1x_2x_3x_4$, entonces al solucionar el sistema

   $$\mathbf{x} \cdot G^\top = \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} x_1 + x_2 & x_3 + x_4 \end{pmatrix} = \begin{pmatrix} 0 & 0 \end{pmatrix}.$$

Se encuentra que $x_1 = x_2$ y $x_3 = x_4$, lo cual implica que el conjunto $\{1100, 0011\}$, es una base para el dual. Pero estos elementos coinciden con las filas de la matriz generadora G para L. Por lo tanto, generan el mismo espacio y así $L = L^{\perp}$.

Un resultado que caracteriza la dimensión de los códigos auto-ortogonales y auto-duales se formula en la siguiente proposición.

Proposición 3.1. *La dimensión de un código auto-ortogonal de longitud n, es menor o igual que $\frac{n}{2}$, mientras que la dimensión de un código auto-dual de longitud n es $n/2$, donde n es un número par.*

Demostración. Si L es un $[n, k]$-código q-ario auto-ortogonal entonces $L^{\perp}$ es un $[n, n-k]$-código lineal. Como L es auto-ortogonal entonces $L \subseteq L^{\perp}$, lo cual implica que $\mid L \mid = q^k \leq q^{n-k} = \mid L^{\perp} \mid$, es decir $k \leq n - k$ y de ahí que $k \leq n/2$.
Si L es auto-dual, entonces $k = n-k$ es decir $k = n/2$, pero como k es entero, entonces n es par. ■

Algunos resultados con respecto a esta clase de códigos lineales son los siguientes.

Teorema 3.6. *Sea G una matriz generadora para un código lineal L sobre $\mathbb{F}_q$, donde $q = 2$ ó $q = 3$. L es auto-ortogonal si y sólo si todas las filas distintas de G son ortogonales y tienen peso divisible por q.*

Demostración. Por definición, todas las filas de G son ortogonales. Probemos por separado para $q = 2$ y $q = 3$ que las filas de G tienen peso divisible por q.
Para el caso $q = 2$, supongamos por contradicción, que una fila $\mathbf{x}$ de G tiene peso impar, es decir $wt(\mathbf{x}) = 2t + 1$, para algún entero no negativo t. Luego, $\mathbf{x} \cdot \mathbf{x} = 2t + 1 = 1 \neq 0$, pero esto es una contradicción, ya que L es auto-ortogonal.
Para $q = 3$, supongamos por contradicción que existe una fila $\mathbf{x}$ de G tal que $wt(\mathbf{x}) = 3t + 1$ ó $wt(\mathbf{x}) = 3t + 2$. En el primer caso, sean x_i, para $i = 1, \ldots, 3t + 1$, las coordenadas no nulas de $\mathbf{x}$. Entonces $x_i = 1$ ó $x_i = 2$. En ambos casos, se tiene $x_i^2 = 1$, ya que $x_i \in \mathbb{F}_3^*$. Por lo tanto $\mathbf{x} \cdot \mathbf{x} = 3t + 1 = 1 \neq 0$, lo cual contradice la hipótesis. Similarmente, si $wt(\mathbf{x}) = 3t + 2$ entonces $\mathbf{x} \cdot \mathbf{x} = 3t + 2 = 2 \neq 0$. Luego $wt(\mathbf{x}) = 3t$.
Recíprocamente, sean $\mathbf{x}, \mathbf{c} \in L$, $\dim(L) = k$ y $g_1, g_2, \ldots, g_k$ las filas de la matriz G. Entonces $\mathbf{x}$ y $\mathbf{c}$ se pueden escribir en la forma

$$\mathbf{x} = \lambda_1 g_1 + \lambda_2 g_2 + \cdots + \lambda_k g_k,$$

$$\mathbf{c} = \beta_1 g_1 + \beta_2 g_2 + \cdots + \beta_k g_k,$$

para algunos $\lambda_1, \ldots, \lambda_k, \beta_1, \ldots, \beta_1 \in \mathbb{F}_q$.

Luego

$$\begin{aligned}\mathbf{x}\cdot\mathbf{c} &= (\lambda_1 g_1 + \lambda_2 g_2 + \cdots + \lambda_k g_k)\cdot(\beta_1 g_1 + \beta_2 g_2 + \cdots + \beta_k g_k)\\ &= \sum_{i=1}^{k} \lambda_1\beta_i g_1 \cdot g_i + \sum_{i=1}^{k} \lambda_2\beta_i g_2 \cdot g_i + \cdots + \sum_{i=1}^{k} \lambda_k\beta_i g_k \cdot g_i\\ &= \sum_{i=1}^{k}\sum_{j=1}^{k} \lambda_i\beta_j g_i \cdot g_j = \sum_{i=1}^{k} \lambda_i\beta_i g_i \cdot g_i\\ &= \mathbf{0},\end{aligned}$$

dado que las filas de G son ortogonales y tienen peso divisible por q. ■

Teorema 3.7. *Si las filas distintas de una matriz generadora para un código lineal binario L son ortogonales y tienen peso divisible por 4, entonces L es auto-ortogonal y todos los pesos en L son divisibles por 4.*

Demostración. Si las filas de la matriz generadora tienen peso divisible por 4, entonces también son divisibles por 2 y por el teorema anterior L es auto-ortogonal. Para la segunda parte, sean $g_1, g_2, \ldots, g_k$, las filas de la matriz generadora G para L. Probemos por inducción que todas las combinaciones lineales no nulas de G tienen peso divisible por 4.
Claramente para $x = g_i$ con $i = 1, \ldots, k$ se tiene

$$wt(x) = wt(g_i),$$

es divisible por 4 ya que todos los pesos $wt(g_i)$ son divisibles por 4.
Para $g_i + g_j$ con $i, j = 1, \ldots, k$, $i \neq j$, se tiene

$$\begin{aligned}wt(x) &= wt(g_i + g_j)\\ &= wt(g_i) + wt(g_j) - 2wt(g_i \cap g_j).\end{aligned}$$

Pero $wt(g_i \cap g_j)$ es siempre par, ya que de lo contrario la palabra código $w = g_i + g_j$ tendría peso impar y por lo tanto $w \cdot w = 1 \neq 0$, contradiciendo la primera parte. Por hipótesis $wt(g_i)$ y $wt(g_j)$ son divisibles por 4, por lo tanto $wt(x)$ es divisible por 4. Siguiendo un razonamiento similar, para cualquier combinación no nula de las filas de G el peso es divisible por 4. ■

El siguiente teorema establece algunos criterios para garantizar la existencia de códigos auto-duales.

Teorema 3.8. *Un $[n, n/2]$ código auto-dual q-ario existe si y sólo si una de las siguientes condiciones se satisface*

1. *q y n son pares.*

2. $q \equiv 1 \,(mód\, 4)$ *y* n *es par.*

3. $q \equiv 3 \,(mód\, 4)$ *y* n *es divisible por* 4.

Demostración. Esta demostración requiere nuevos conocimientos, por lo cual se omite. Sin embargo, el lector interesado puede consultar [12, p. 633]. ■

Observación 3.4. Sea L un código lineal con matriz generadora en la forma estándar $G = (I_k \mid A)$ de tamaño $k \times n$. Consideremos la matriz H de la forma $H = (-A^\top \mid I_{n-k})$, con $A^\top$ la traspuesta de A. Entonces

$$GH^\top = (I_k \mid A) \begin{pmatrix} -A \\ I_{n-k} \end{pmatrix} = -A + A = \mathbf{0}.$$

Es decir las filas de H son ortogonales a las filas de G y como $\mathrm{r}(H) = n - k = \dim(L^\perp)$, entonces H es una matriz generadora para el código dual $L^\perp$. Esta matriz recibe un nombre especial y es muy importante en el proceso de decodificación.

Definición 3.6. (Matriz de control de paridad). Una *matriz de control de paridad* H para un código lineal L, es una matriz generadora para el código dual $L^\perp$. Una matriz generadora para el código dual en la forma $H = (-A^\top \mid I_{n-k})$ se llama *matriz de control de paridad en la forma estándar.*

De ahí que para un código lineal L, una palabra $x \in L$ si y sólo si $x \cdot H^\top = \mathbf{0}$.

Ejemplo 3.10. Sea

$$H = \begin{pmatrix} 1001000 \\ 1010100 \\ 0110010 \\ 1100001 \end{pmatrix}$$

la matriz de control de paridad para un código lineal binario L. Determinemos explícitamente los elementos de L. Sea $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5, x_6, x_7)$ un elemento de L. Por la observación 3.4, $\mathbf{x} \in L$ si y sólo si $\mathbf{x} \cdot H^\top = \mathbf{0}$, es decir

$$\begin{aligned} \mathbf{x} \cdot H^\top &= (x_1, x_2, x_3, x_4, x_5, x_6, x_7) \cdot \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ &= (x_1 + x_2 + x_4, x_1 + x_3 + x_5, x_2 + x_3 + x_6, x_1 + x_2 + x_7) \\ &= (0, 0, 0, 0). \end{aligned}$$

El sistema reducido proporciona las ecuaciones

$$\begin{aligned} x_1 + x_2 + x_4 &= 0, \\ x_1 + x_3 + x_5 &= 0, \\ x_2 + x_3 + x_6 &= 0, \\ x_1 + x_2 + x_7 &= 0, \end{aligned}$$

las cuales son llamadas en general *ecuaciones de control de paridad*. De estas ecuaciones obtenemos una base $B = \{1001101, 0101011, 0010110\}$ para el código lineal L. Con GAP listamos los elementos de L explícitamente.

```
> L := GeneratorMatCode([[1,0,0,1,1,0,1],[0,1,0,1,0,1,1],[0,0,1,0,1,1,0]],
  "demo code", GF(2) );
[  a linear [7,3,1..3]2..3 demo code over GF(2)
> AsSSortedList(L);
[ [ 0 0 0 0 0 0 0 ], [ 0 0 1 0 1 1 0 ], [ 0 1 0 1 0 1 1 ], [ 0 1 1 1 1 0 1 ],
  [ 1 0 0 1 1 0 1 ], [ 1 0 1 1 0 1 1 ], [ 1 1 0 0 1 1 0 ], [ 1 1 1 1 0 0 0 ] ]
```

es decir $L = \{0000000, 0010110, 0101011, 0111101, 1001101, 1011011, 1100110, 1110000\}$.

La construcción de un código dual para un código lineal L a partir de un subconjunto $S \subseteq \mathbb{F}_q^n$ se realiza mediante la construcción de una matriz generadora en la forma estándar para el código L, seguida por la construcción de una matriz generadora para el código dual de acuerdo a la observación 3.4.

Ejemplo 3.11. Encontrar, a partir del conjunto $S = \{110000, 011000, 001100, 000110, 000011\}$, el código dual de $L = \langle S \rangle$.
Una matriz generadora en la forma estándar para $L = \langle S \rangle$ está formada por una base para el espacio fila de la matriz A cuyas filas son los elementos de S

$$A = \begin{pmatrix} 110000 \\ 011000 \\ 001100 \\ 000110 \\ 000011 \end{pmatrix} \xrightarrow{E_{45}(1)} \begin{pmatrix} 110000 \\ 011000 \\ 001100 \\ 000101 \\ 000011 \end{pmatrix} \xrightarrow{E_{34}(1)} \begin{pmatrix} 110000 \\ 011000 \\ 001001 \\ 000101 \\ 000011 \end{pmatrix} \xrightarrow{E_{23}(1)} \begin{pmatrix} 110000 \\ 010001 \\ 001001 \\ 000101 \\ 000011 \end{pmatrix} \xrightarrow{E_{12}(1)} \begin{pmatrix} 100001 \\ 010001 \\ 001001 \\ 000101 \\ 000011 \end{pmatrix}.$$

La última matriz, es una matriz generadora en la forma estándar para el código lineal $L = \langle S \rangle$ y por lo tanto, una matriz generadora para el código dual es

$$H = \begin{pmatrix} 111111 \end{pmatrix}.$$

En caso de tener un código lineal que no tenga una matriz generadora en la forma estándar, se permutan las columnas de tal forma que se construya una matriz generadora en la forma estándar para un código equivalente al original.

En general, no existe una manera eficiente de determinar la distancia de un código lineal L a partir de una matriz generadora. Sin embargo, si H es una matriz de control de paridad para L esto se puede realizar como lo muestra el siguiente teorema.

Teorema 3.9. *Sea L un $[n, k, d]$-código lineal con matriz de control de paridad H. Entonces*

1. *L tiene distancia mayor o igual a d si y sólo si cualquier $d-1$ columnas de H son linealmente independientes.*

2. *L tiene distancia menor o igual a d si y sólo si H tiene d columnas que son linealmente dependientes.*

Demostración. Sea $\mathbf{v} = v_1v_2, \ldots, v_n \in L$ una palabra de peso $e > 0$. Supongamos que las coordenadas no cero de $\mathbf{v}$ están en las posiciones $i_1, i_2, \ldots, i_e$, así que, $v_j = 0$ si $j \notin \{i_1, i_2, \ldots, i_e\}$. Sea $\mathbf{c}_i$, para $1 \leq i \leq n$, la i-ésima columna de H. Entonces L contiene una palabra no cero $\mathbf{v} = v_1v_2, \ldots, v_n \in L$ de peso $e > 0$ si y sólo si

$$\mathbf{0} = \mathbf{v}H^\top = v_{i_1}\mathbf{c}_{i_1}^\top + v_{i_2}\mathbf{c}_{i_2}^\top + \cdots + v_{i_e}\mathbf{c}_{i_e}^\top,$$

lo cual es cierto, si y sólo si e columnas de H son linealmente dependientes.
Como $wt(\mathbf{v}) \geq d(L)$, entonces la distancia de L es mayor o igual que d si y sólo si L no contiene palabras no cero de peso menor o igual a $d-1$, ya que de ser así, el peso de esta palabra sería menor que la distancia d, lo cual es una contradicción. Luego, cualquier $d-1$ columnas de H son linealmente independientes.
Para la segunda parte, la distancia de L es menor o igual a d si y sólo si L contiene una palabra no cero de peso menor o igual a d lo cual ocurre si y sólo si H tiene a lo más d columnas que son linealmente dependientes, es decir, si y sólo si H tiene d columnas linealmente dependientes. ■

El siguiente resultado es consecuencia inmediata del teorema anterior.

Corolario 3.2. *Sea L un código lineal y H una matriz de control de paridad para L. Las siguientes proposiciones son equivalentes.*

1. *L tiene distancia d.*

2. *Cualquier $d-1$ columnas de H son linealmente independientes y H tiene d columnas que son linealmente dependientes.*

Ejemplo 3.12. Para la matriz de control H del ejemplo 3.10 cualquier par de columnas son linealmente independientes, pero H tiene 3 columnas que son linealmente dependientes como por ejemplo 1101, 1011, 0110. Luego $d(L) = 2$, como puede verificarse fácilmente en el ejemplo tratado.

3.3. Decodificación por Síndrome

Veamos ahora una manera general de decodificación con códigos lineales equivalente a la decodificación por distancia mínima. Primero se introduce algunos conceptos claves.

Definición 3.7. (Síndrome). Sea L un $[n,k]$-código lineal con matriz de control de paridad H. Para cualquier $x \in \mathbb{F}_q^n$, la palabra $x \cdot H^\top$ se llama el *síndrome* de x y se denota por $s(x)$.

Claramente, el síndrome es una palabra de longitud $n-k$ y $x \in L$ si y sólo si $s(x) = \mathbf{0}$.

Definición 3.8. Clases Laterales
Sean L un código lineal de longitud n sobre $\mathbb{F}_q$, $x \in \mathbb{F}_q^n$ un vector de longitud n; se define la *clase lateral de L determinada por x* como

$$L + x = \{c + x : c \in L\}.$$

Teniendo en cuenta que L es un grupo abeliano se tiene $L + x = x + L$. Además, este conjunto es un espacio vectorial sobre $\mathbb{F}_q$ con las operaciones

$$a(x + L) = ax + L \quad y \quad (x + L) + (y + L) = (x + y) + L.$$

Por lo tanto, $x + L = y + L$ si y sólo si $x - y \in L$. A partir de esto, se presenta el siguiente resultado.

Teorema 3.10. *Sea L un $[n,k]$-código, con matriz de control de paridad H. Entonces $x, y \in \mathbb{F}_q^n$ tienen el mismo síndrome si y sólo si, ellos pertenecen a la misma clase lateral de L.*

Demostración.

$$\begin{aligned} x + L = y + L &\iff x - y \in L \\ &\iff (x-y)H^\top = 0 \\ &\iff xH^\top - yH^\top = 0 \\ &\iff xH^\top = yH^\top \\ &\iff s(x) = s(y). \end{aligned}$$

Veamos ahora como utilizar el teorema anterior para decodificar con códigos lineales. ■

Teorema 3.11. *Sea L un código lineal con matriz de control de paridad H. Decodificar por distancia mínima es equivalente a decodificar la palabra recibida x como la palabra código $c = x - a$, donde a es una palabra de menor peso en la clase lateral $x + L$, o equivalentemente, a es una palabra de menor peso con igual síndrome que x.*

Demostración. Supongamos que una palabra código es enviada y la palabra x es recibida. La decodificación por distancia mínima decodifica x a la palabra código c tal que

$$d(x, c) = \text{mín}\{d(x, c), c \in C\},$$

lo cual es equivalente a que $d(x - c, 0)$ sea mínima, es decir que el peso de $a = x - c$ sea mínimo. Por lo tanto, cuando c varía sobre L, a varía sobre la clase lateral $x + L$ y de ahí que la decodificación por distancia mínima requiere encontrar la palabra de peso mínimo entre todas las palabras de una clase lateral, equivalentemente, la palabra de menor peso entre todas las palabras con el mismo síndrome de x. ■

El proceso de decodificación descrito en el teorema anterior puede describirse en términos de lo que se llama *arreglo estándar* o *arreglo Slepiano* (debido a Slepian (1960)) para L. Es decir

$$\begin{array}{lllll} 0 & c_1 & c_2 & \dots & c_m \\ a_1 & c_1 + a_1 & c_2 + a_1 & \dots & c_m + a_1 \\ a_2 & c_1 + a_2 & c_2 + a_2 & \dots & c_m + a_2 \\ \vdots & \vdots & \vdots & & \vdots \\ a_s & c_1 + a_s & c_2 + a_s & \dots & c_m + a_s \end{array}$$

donde $r = q^k - 1$ y $s = q^{n-k} - 1$. La primera fila consta en todas las palabras código de L. La segunda fila consta de la suma de cada palabra de la primera fila con una palabra $a_1 \in \mathbb{F}_q$ de menor peso que no este en L, es decir la clase lateral $a_1 + L$. De manera general, la i-ésima fila del arreglo, se forma por la suma de cada palabra de la primera fila con una palabra a_i de menor peso que no está aún en el arreglo, es decir la clase lateral $a_i + L$. Este proceso continua hasta que el arreglo contenga todas las palabras de $\mathbb{F}_q^n$, es decir cuando se tenga $s = q^{n-k} - 1$ filas. Las palabras a_i se llaman *líderes de las clases laterales.*

Como cada fila del arreglo estándar es una clase lateral de L, entonces dos palabras en $\mathbb{F}_q^n$, tienen el mismo síndrome si y sólo si están en la misma fila del arreglo (Teorema 3.11).

Teniendo en cuenta la forma como los líderes de la clase lateral fueron seleccionados, entonces una palabra recibida x esta en la j-ésima columna del arreglo, es decir, $x = c_j + a_i$, para alguna palabra a_i de menor peso en la clase lateral $x + L$. La decodificación por distancia mínima decodifica a x como c_j, es decir, la palabra código de la columna j.

Ejemplo 3.13. Consideremos el $[5, 2, 3]$-código binario $L = \{00000, 10110, 01011, 11101\}$. Las clases laterales son

$$00000 + L = \{00000, 10110, 01011, 11101\},$$

$$00001 + L = \{00001, 10111, 01010, 11100\},$$

$$00010 + L = \{00010, 10100, 01001, 11111\},$$

$$00100 + L = \{00100, 10010, 01111, 11001\},$$
$$01000 + L = \{01000, 11110, 00011, 10101\},$$
$$10000 + L = \{10000, 00110, 11011, 01101\},$$
$$00101 + L = \{00101, 10011, 01110, 11000\},$$
$$01100 + L = \{00000, 11010, 00111, 10001\}.$$

El arreglo estándar queda

00000	10110	01011	11101
00001	10111	01010	11100
00010	10100	01001	11111
00100	10010	01111	11001
01000	11110	00011	10101
10000	00110	11011	01101
00101	10011	01110	11000
01100	11010	00111	10001

Si se recibe la palabra $x = 11011$, se detecta un error ya que x no está en la primera fila. Como x está en la sexta fila, la única palabra de menor peso en esta clase lateral es 10000, por lo tanto x se decodifica como $c = 01011$, la palabra que está en la parte superior, en la misma columna que x. En cambio, si se recibe la palabra $x = 11010$, se detecta dos errores. Como x está en la octava columna y en esta clase lateral hay dos palabras de peso mínimo (01100 y 10001, con 01100 como líder de la clase lateral), según el arreglo mostrado x se decodifica a $c = 10110$, sin embargo, si hubiésemos considerado a 10001 como líder de la clase lateral, en cuyo caso la octava fila sería

10001	00111	11010	01100

se tiene que x se decodifica como $c' = 01011$, es decir $c \neq c'$. La razón de esta situación, es porque L corrige exactamente 1-error, pero no dos errores como ocurre en este caso. En general, los errores ocurridos durante la transmisión que este método corrige dependen del líder de la clase lateral escogido en el caso en que haya más de dos palabras de peso mínimo en la misma clase lateral. Es decir, si $x = c + e$ es una palabra recibida, donde e es el error y e es un líder de alguna clase lateral, entonces $x \in e + L$ y por lo tanto al decodificar x se obtiene la palabra código correcta $c = x - e$. Por otra parte, si e no es un líder de una clase lateral y se encuentra en una fila j, entonces x también está en la fila j y es decodificada incorrectamente como $x - a_j \neq x - e = c$.

La realización del arreglo estándar resulta bastante dispendiosa para códigos de longitud relativamente grande. Sin embargo, no es necesario completar todo el arreglo, basta con construir una tabla de líderes de las clases laterales a_i con sus correspondientes síndromes, ya que cada fila

está determinada por éstos (es decir, cada fila tiene diferente síndrome). Por lo tanto, si x es una palabra recibida, se calcula su síndrome $s(x)$ y se busca en la tabla el líder de la clase lateral a_i con igual síndrome que x. Luego, x se decodifica como $c = x - a_i$. Este proceso se conoce como *decodificación por síndrome*.

Ejemplo 3.14. En GAP, con la función `GeneratorMatCode` generamos el código lineal L, generado por $S = \{0121, 1022\} \subset \mathbb{F}_3^4$. Con la función `AsSSortedList(L)`, obtenemos sus elementos y con la función `MinimumDistance(L)` obtenemos su distancia, es decir $d(L) = 3$. Luego, L detecta 2 errores y corrige 1.

```
> L := GeneratorMatCode([[0,1,2,1],[1,0,2,2]],"demo code", GF(3) );
[ a linear [4,2,1..3]1 demo code over GF(3)
> AsSSortedList( L);
[ [ 0 0 0 0 ], [ 0 1 2 1 ], [ 0 2 1 2 ], [ 1 0 2 2 ], [ 1 1 1 0 ],
  [ 1 2 0 1 ], [ 2 0 1 1 ], [ 2 1 0 2 ], [ 2 2 2 0 ] ]
> MinimumDistance(L);
[ 3
```

Con la función `StandardArray(L)`, se obtiene un arreglo estándar para L, el cual consta de 9 filas. Los líderes de las clases laterales son 0000,0001,0002,0010,0020,0100,0200,1000,2000, como se puede observar en el siguiente código fuente.

```
> StandardArray(L);
[ [ [ 0 0 0 0 ], [ 0 1 2 1 ], [ 0 2 1 2 ], [ 1 0 2 2 ], [ 1 1 1 0 ],
      [ 1 2 0 1 ], [ 2 0 1 1 ], [ 2 1 0 2 ], [ 2 2 2 0 ] ],
  [ [ 0 0 0 1 ], [ 0 1 2 2 ], [ 0 2 1 0 ], [ 1 0 2 0 ], [ 1 1 1 1 ],
      [ 1 2 0 2 ], [ 2 0 1 2 ], [ 2 1 0 0 ], [ 2 2 2 1 ] ],
  [ [ 0 0 0 2 ], [ 0 1 2 0 ], [ 0 2 1 1 ], [ 1 0 2 1 ], [ 1 1 1 2 ],
      [ 1 2 0 0 ], [ 2 0 1 0 ], [ 2 1 0 1 ], [ 2 2 2 2 ] ],
  [ [ 0 0 1 0 ], [ 0 1 0 1 ], [ 0 2 2 2 ], [ 1 0 0 2 ], [ 1 1 2 0 ],
      [ 1 2 1 1 ], [ 2 0 2 1 ], [ 2 1 1 2 ], [ 2 2 0 0 ] ],
  [ [ 0 0 2 0 ], [ 0 1 1 1 ], [ 0 2 0 2 ], [ 1 0 1 2 ], [ 1 1 0 0 ],
      [ 1 2 2 1 ], [ 2 0 0 1 ], [ 2 1 2 2 ], [ 2 2 1 0 ] ],
  [ [ 0 1 0 0 ], [ 0 2 2 1 ], [ 0 0 1 2 ], [ 1 1 2 2 ], [ 1 2 1 0 ],
      [ 1 0 0 1 ], [ 2 1 1 1 ], [ 2 2 0 2 ], [ 2 0 2 0 ] ],
  [ [ 0 2 0 0 ], [ 0 0 2 1 ], [ 0 1 1 2 ], [ 1 2 2 2 ], [ 1 0 1 0 ],
      [ 1 1 0 1 ], [ 2 2 1 1 ], [ 2 0 0 2 ], [ 2 1 2 0 ] ],
  [ [ 1 0 0 0 ], [ 1 1 2 1 ], [ 1 2 1 2 ], [ 2 0 2 2 ], [ 2 1 1 0 ],
      [ 2 2 0 1 ], [ 0 0 1 1 ], [ 0 1 0 2 ], [ 0 2 2 0 ] ],
```

```
  [ [ 2 0 0 0 ], [ 2 1 2 1 ], [ 2 2 1 2 ], [ 0 0 2 2 ], [ 0 1 1 0 ],
      [ 0 2 0 1 ], [ 1 0 1 1 ], [ 1 1 0 2 ], [ 1 2 2 0 ] ] ]
```

La función `SyndromeTable(L)` genera una tabla de líderes de las clases laterales con sus respectivos síndromes. Por ejemplo, en la fila 5 mostrada a continuación, se tiene el líder de la clase lateral 2000 junto con su respectivo síndrome la palabra 22.

```
> SyndromeTable(L);
[ [ [ 0 0 0 0 ], [ 0 0 ] ], [ [ 0 0 0 1 ], [ 0 1 ] ],
  [ [ 0 0 0 2 ], [ 0 2 ] ], [ [ 0 0 1 0 ], [ 1 0 ] ],
  [ [ 1 0 0 0 ], [ 1 1 ] ], [ [ 0 1 0 0 ], [ 1 2 ] ],
  [ [ 0 0 2 0 ], [ 2 0 ] ], [ [ 0 2 0 0 ], [ 2 1 ] ],
  [ [ 2 0 0 0 ], [ 2 2 ] ] ]
```

El síndrome de cada líder de la clase lateral se obtuvo de la matriz de control de paridad H para L

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

como puede verificarse fácilmente. Supongamos que se envió la palabra código $c = 1201$ y la palabra $x = 1211$ con un error ha sido recibida. Con la función `Syndrome(L,Codeword( "1211"))`, determinamos su síndrome, es decir, $s(x) = 10$, como puede verse a continuación.

```
> Syndrome( L, Codeword( "1211" ) );
[ [ 1 0 ]
```

Este síndrome puede determinarse a partir de la matriz H para verificar la respuesta de GAP, es decir

$$\begin{pmatrix} 1 & 2 & 1 & 1 \end{pmatrix} \cdot H^{\top} = \begin{pmatrix} 1 & 2 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} = 10.$$

De la tabla de síndromes mostrada por GAP, el líder de la clase lateral con síndrome igual a 10 se encuentra en la segunda fila y es la palabra 0010. Por lo tanto, $x = 1211$ se decodifica como $c = 1211 - 0010 = 1201$, lo cual es correcto.

Observación 3.5. Para un código lineal L con distancia mínima d, todas las palabras de peso menor o igual a $t = \left\lfloor \frac{d-1}{2} \right\rfloor$ son líderes de las clases laterales. En efecto, supongamos por contradicción que dos palabras distintas $\mathbf{x}$ y $\mathbf{y}$ tiene peso menor o igual a t y ambas están en la misma clase lateral. Por la desigualdad triangular

$$d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{0}) + d(\mathbf{0}, \mathbf{y}) \leq wt(\mathbf{x}) + wt(\mathbf{y}) \leq 2t \leq d - 1,$$

lo cual no es posible, ya que $d(\mathbf{x}, \mathbf{y}) \geq d$.

3.4. Construcción de Códigos

En esta sección se mostrará algunas de las más conocidas técnicas usadas para obtener nuevos códigos, a partir de la combinación o modificación de otros códigos ya conocidos. Aunque las definiciones tratadas en esta sección son en general para cualquier código, se prestará mayor atención a los códigos lineales.

Definición 3.9. (Perforar un código (Puncturing a code)). Proceso que consiste en remover una o más coordenadas de las palabras de un código. Si C es un (n, M, d)-código sobre $\mathbb{F}_q$ y si $d \geq 2$, entonces el *código perforado* C^* obtenido a partir de la eliminación de la misma coordenada de todas las palabras de C tiene como parámetros

$$n^* = n - 1,\ M^* = M,\ d^* = d \text{ ó } d^* = d - 1.$$

Si C es un código lineal entonces el código perforado C^* también es lineal. En efecto, sean $x^*, y^* \in C^*$, $\lambda, \beta \in \mathbb{F}_q$. Por definición, x^*, y^* se obtuvieron de algunas $x, y \in C$ por eliminación de la misma coordenada de estas palabras. Como $\lambda x + \beta y \in C$ entonces $\lambda x^* + \beta y^* \in C^*$, dado que esta palabra resulta de la eliminación de la misma coordenada de la palabra $\lambda x + \beta y \in C$. Además, claramente $C^* \neq \phi$, ya que la palabra cero está en C^*.
Algunas características de los códigos lineales obtenidos por perforación se resumen en el siguiente teorema.

Teorema 3.12. *Sea L un $[n, k, d]$ código lineal sobre $\mathbb{F}_q$ y sea L^* el código obtenido por la eliminación de la i-ésima coordenada de todas las palabras de L. Entonces*

1. *Si $d > 1$, L^* es un $[n-1, k, d^*]$-código lineal, donde $d^* = d-1$, si L tiene una palabra de peso mínimo cuya i-ésima coordenada no es cero y $d^* = d$ en cualquier otro caso.*

2. *Cuando $d = 1$, L^* es un $[n-1, k, 1]$-código lineal, si L no tiene una palabra de peso 1 cuya entrada no cero está en la i-ésima coordenada; de otra manera, si $k > 1$ entonces L^* es un $[n-1, k-1, d^*]$-código lineal con $d^* \geq 1$.*

Demostración. Por lo dicho anteriormente, L^* es lineal. Además, su longitud $n-1$ es clara a partir de la definición. Por otro lado, ya que L contiene q^k palabras, entonces la única manera para que L^* contenga menos palabras, es que dos palabras de L sean iguales en todas, excepto en la i-ésima coordenada. En este caso, $d(L) = 1$ y L tiene una palabra de peso 1 cuya i-ésima coordenada es distinta de cero. Por lo tanto, $\dim(L^*) = k$ y así L^* es un $[n-1, k, 1]$-código lineal. Si $k > 1$ y L contiene una palabra de peso 1, cuya i-ésima coordenada es distinta de cero, entonces $d(L^*) = d^* \geq 1$ y $\dim(L^*) = k - 1$ ya que esta palabra genera q elementos y no puede pertenecer a una base de L^*, dado que es nula. Luego L^* es un $[n-1, k-1, d^*]$. Esto prueba la parte 1.

Para la parte 2, por lo dicho anteriormente, si $d(L) > 1$ entonces $\dim(L^*) = k$. Para la distancia, sean L_1 y L_2 subcódigos de L cuyas i-ésimas componentes son distintas de cero e iguales a cero respectivamente, por lo tanto $L^* = L_1^* \cup L_2^*$. Si $c \in L_1$ tal que $d(L) = wt(c) = d$, entonces $d(L^*) = wt(c^*) = d - 1$, donde c^* se obtiene de c por eliminación de la i-ésima coordenada. Si no existe tal $c \in L_1$, entonces $d(L) = wt(x) = d$ ó $d(L) = wt(y) = d$, para algunas $x \in L_1$ y $y \in L_2$. En cualquier caso, $d(L) = wt(x) = wt(x^*) = d$ ó $d(L) = wt(y) = wt(y^*) = d$, donde x^* y y^* son palabras obtenidas de x y y respectivamente, por eliminación de la i-ésima coordenada. Lo anterior implica que $d \geq wt(L^*)$. Pero $wt(L^*) \geq d$, ya que de lo contrario, existe $x \in L_1$ tal que $d(L) = wt(x) = d$, lo cual es una contradicción. Esto prueba la parte 2. ■

Ejemplo 3.15. Sea L el $[6, 3, 2]$-código lineal binario con matriz generadora

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

Sean L_1^* y L_5^*, los códigos obtenidos por eliminación de la primera y quinta coordenada de L respectivamente.
Estos tienen matrices generadoras

$$G_1^* = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix} \text{ y } G_5^* = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

De esta manera L_1^* es un $[6, 3, 1]$-código lineal, mientras que L_1^* es un $[6, 3, 2]$-código lineal.

De manera general, para códigos lineales, un código L puede ser perforado sobre un conjunto T de coordenadas, por eliminación de las componentes etiquetadas por este conjunto en todas las palabras de L. Si T tiene tamaño t, el código denotado por L^T, es un $[n - t, k^*, d^*]$-código lineal, con $k^* \geq k - t$ y $d^* \geq d - t$. La prueba se realiza aplicando inducción sobre el Teorema 3.12.

Definición 3.10. (Extender un código). Es un proceso que consiste en añadir una o más coordenadas adicionales a las palabras de un código. Existen varias formas de extender un código, sin embargo, aquí se presentará aquella que se obtiene al adicionar una o más coordenadas de control de paridad. Por lo tanto, si C es un (n, M, d)-código sobre $\mathbb{F}_q$, se define el *código extendido* $\widehat{C}$ por

$$\widehat{C} = \left\{ c_1 c_2 \cdots c_n c_{n+1} / c_1 c_2 \cdots c_n \in C \text{ y } \sum_{k=1}^{n+1} c_k = 0 \right\}.$$

Si $\widehat{C}$ es un $(\widehat{n}, \widehat{M}, \widehat{d})$-código, entonces

$$\widehat{n} = n + 1,\ \widehat{M} = M,\ \widehat{d} = d \text{ ó } \widehat{d} = d + 1.$$

Para el caso lineal, el código extendido $\widehat{L}$ de cualquier código lineal L, también es lineal. En efecto, sean G y H matrices generadora y de control, respectivamente para el código L. Por definición de código extendido, la matriz $\widehat{G}$ obtenida de G por adición de una columna que verifique que la suma de las coordenadas de cada fila de $\widehat{G}$ sea igual a cero, es una matriz generadora para el código $\widehat{L}$. Además, una matriz de control para $\widehat{L}$ es

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

ya que para cada $x = x_1 \cdots x_n x_{n+1} \in \widehat{L}$ se tiene

$$x \cdot \widehat{H}^\top = x \cdot \mathbf{1} + x \cdot (H|\mathbf{0}) = \sum_{k=1}^{n+1} c_k + (x_1 \cdots x_n) \cdot H^\top = \mathbf{0}.$$

Ejemplo 3.16. El código binario de Hamming $H_2(r)$ tiene parámetros $n = 2^r - 1$, $M = 2^{2^r-1-r}$, $d = 3$. El código extendido de Hamming $\widehat{H}_2(r)$, obtenido por adición de una coordenada de control de paridad para $H_2(r)$, tiene parámetros

$$\widehat{n} = 2^r,\ \widehat{M} = 2^{2^r-1-r},\ \widehat{d} = 4.$$

Con GAP, obtenemos el código de Hamming $H_2(3)$ el cual denotamos por $C1$. La función `AsSSortedList(C1)` presenta elementos de $C1$. Con la función `ExtendedCode(C1)` se construye el *código extendido de Hamming* $H_2(3)$, denotado como $\widehat{H}_2(3)$. Nuevamente, con la función `AsSSortedList(C2)`, GAP presenta los elementos de este código. Lo dicho anteriormente se resume a continuación.

```
> C1 := HammingCode( 3, GF(2) );
[ a linear [7,4,3]1 Hamming (3,2) code over GF(2)
> AsSSortedList( C1 );
  [ [ 0 0 0 0 0 0 0 ], [ 0 0 0 1 1 1 1 ], [ 0 0 1 0 1 1 0 ], [ 0 0 1 1 0 0 1 ],
    [ 0 1 0 0 1 0 1 ], [ 0 1 0 1 0 1 0 ], [ 0 1 1 0 0 1 1 ], [ 0 1 1 1 1 0 0 ],
    [ 1 0 0 0 0 1 1 ], [ 1 0 0 1 1 0 0 ], [ 1 0 1 0 1 0 1 ], [ 1 0 1 1 0 1 0 ],
    [ 1 1 0 0 1 1 0 ], [ 1 1 0 1 0 0 1 ], [ 1 1 1 0 0 0 0 ], [ 1 1 1 1 1 1 1 ] ]
> C2 := ExtendedCode( C1 );
[ a linear [8,4,4]2 extended code
> AsSSortedList( C2 );
  [ [ 0 0 0 0 0 0 0 0 ], [ 0 0 0 1 1 1 1 0 ], [ 0 0 1 0 1 1 0 1 ],
    [ 0 0 1 1 0 0 1 1 ], [ 0 1 0 0 1 0 1 1 ], [ 0 1 0 1 0 1 0 1 ],
    [ 0 1 1 0 0 1 1 0 ], [ 0 1 1 1 1 0 0 0 ], [ 1 0 0 0 0 1 1 1 ],
```

```
[ 1 0 0 1 1 0 0 1 ], [ 1 0 1 0 1 0 1 0 ], [ 1 0 1 1 0 1 0 0 ],
[ 1 1 0 0 1 1 0 0 ], [ 1 1 0 1 0 0 1 0 ], [ 1 1 1 0 0 0 0 1 ],
[ 1 1 1 1 1 1 1 1 ] ]
```

Note que para el caso binario, el código extendido contiene sólo palabras de peso par. Por lo tanto, si la distancia de un código binario L es impar, entonces la distancia del código extendido es par. Si la distancia del código L es par, entonces la distancia del código extendido también es par e igual a la distancia de L. Sin embargo, esto no se cumple cuando el código no es binario.

Al extender un código, pueden suceder dos situaciones con respecto a la distancia. Si la distancia del código extendido se mantiene igual, entonces no es conveniente extender el código, ya que éste cuenta con la misma capacidad de corrección. Por otra parte, si la distancia del código extendido aumenta, entonces su capacidad de detección aumenta, aunque su capacidad de corrección pueda ser la misma que la del código original.
Para códigos lineales binarios, el proceso de perforar y extender un código puede utilizarse para probar el siguiente resultado.

Teorema 3.13. *Un $(n, M, 2t+1)$-código binario existe si y sólo si un $(n+1, M, 2t+2)$-código binario existe.*

Demostración. Supongamos que C es un $(n, M, 2t+1)$-código binario y que $\widehat{C}$ es el código obtenido de C por adición de una coordenada de control de paridad. Por lo dicho en el Ejemplo 3.16, para cada $x \in \widehat{C}$ su peso es par. Por la segunda parte del Lema 2.1, para cada $x, y \in \widehat{C}$ se tiene

$$d(\widehat{x}, \widehat{y}) = wt(\widehat{x}) + wt(\widehat{y}) - 2wt(\widehat{x} \cap \widehat{y}).$$

Luego $d(\widehat{C})$ es par, es decir $d(\widehat{C}) = 2t+2$.
Recíprocamente, si C es un $(n+1, M, 2t+2)$-código binario, entonces existen $x, y \in C$ tal que $d(x, y) = 2t+2$. Perforando una de las $2t+2$ coordenadas en las que x y y no coinciden en todas las palabras código de C se tiene $d(\widehat{x}, \widehat{y}) = 2t+1 = d(\widehat{C})$ y por lo tanto $\widehat{C}$ es un $(n, M, 2t+1)$-código. ■

Definición 3.11. (Expurgar un código (expunging a code)). Este proceso consiste en simplificar un código mediante la eliminación de algunas palabras código.
Un ejemplo de este proceso es el siguiente.

Ejemplo 3.17. Suponga que L es un (n, M, d)-código lineal binario. Si L contiene al menos una palabra de peso impar, entonces exactamente la mitad de las palabras de L deben tener peso impar. En efecto, sean L_1 y L_2, subcódigos de L que tienen peso impar y par, respectivamente. Si $x \in L_1$ entonces por la linealidad de L se tiene $x + x \in L_2$ y por lo tanto $|x + L_1| \leq |L_2|$ y $|x + L_2| \leq |L_1|$. Claramente $|L_1| = |x + L_1|$ y $|L_2| = |x + L_2|$, por lo tanto $|L_1| = |L_2| = M/2$. Sin tener en cuenta las palabras de peso impar, se tiene un $(n, M/2, d')$-código, donde $d' \geq d$. Este código tiene palabras de peso par, distancia mínima par (Lema 2.1) y si d es impar, entonces $d' > d$.

Definición 3.12. (Aumentar un código (augmenting a code)). El proceso opuesto a expurgar un código es aumentar un código, lo cual consiste en agregar palabras adicionales al código.

Para un código binario C, una manera común de aumentar el código C consiste en añadir el *complemento* x^c de cada palabra binaria $x \in C$, donde x^c es la palabra obtenida de x intercambiando los ceros por unos y los unos por ceros. El conjunto formado por todos los complementos de C se denota por C^c.

Ejemplo 3.18. Sea $C = \{00000, 00101, 10001, 11011, 10100, 11110, 01010, 01111\}$ el cual es un $[5, 3, 2]$-código lineal binario, entonces $C^c = \{11111, 11010, 01110, 00100, 01011, 00001, 10101, 10000\}$ es un $(5, 8, 2)$-código. Note que C^c no es un código lineal, dado que no contiene la palabra código $\mathbf{0}$.

Lema 3.1. *Si $x, y \in \mathbb{F}_2^n$, entonces $d(x, y^c) = n - d(x, y)$.*

Demostración. Sean $x, y \in \mathbb{F}_2^n$, como $d(x, y^c)$ es el número de posiciones en las cuales x y y^c difieren, el cual es igual al número de posiciones en las cuales x y y coinciden (ya que $q = 2$), es decir $n - d(x, y)$, entonces $d(x, y^c) = n - d(x, y)$. ■

Usando el resultado anterior no es difícil ver que para $x, y \in C$, se tiene $d(x, y) = d(x^c, y^c)$. Por lo tanto $d(C) = d(C^c)$ y como $|C| = |C^c|$, entonces C y C^c son códigos equivalentes.
El siguiente resultado permite determinar la distancia para el código binario $C \cup C^c$.

Teorema 3.14. *Sea C un (n, M, d)-código binario. Entonces*

$$d(C \cup C^c) = \min\{d, n - d_{max}\},$$

donde d_{max} es la máxima distancia entre las palabras en C.

Demostración. Como $d(C) = d(C^c)$, entonces

$$d(C \cup C^c) = \min\left\{d(C), \min d(x, y), \text{ con } x \in C,\ y \in C^c\right\}.$$

Por el Lema 3.1

$$\min_{\substack{x \in C \\ y \in C^c}} d(x, y) = \min_{\substack{x \in C \\ y \in C}} d(x, y^c) = \min_{\substack{x \in C \\ y \in C}} \{n - d(x, y)\} = n - \max_{\substack{x \in C \\ y \in C}} d(x, y).$$

■

Para el caso lineal sobre $\mathbb{F}_2$, se tiene el siguiente resultado.

Teorema 3.15. *Sea L un (n, M, d)-código lineal binario que no contiene la palabra código $\mathbf{1} = 11 \cdots 1$, entonces el código $L \cup L^c$ es un $(n, 2M, d')$-código lineal binario, donde*

$$d' = \min\{d, n - w_{max}\}$$

y w_{max} es el máximo peso de alguna palabra en L.

Demostración. Dado que $\mathbf{1} = 11\cdots 1 \notin L$, entonces $\mathbf{1}^c = \mathbf{0} \in L$ y así $\mathbf{0} \in L \cup L^c$. Luego $L \cup L^c \neq \phi$. Por la observación 3.2 es suficiente mostrar que $\mathbf{x} + \mathbf{y} \in L^c$ para todo $\mathbf{x}, \mathbf{y} \in L \cup L^c$. Consideremos los siguientes casos:

- Si $\mathbf{x}, \mathbf{y} \in L$, claramente $\mathbf{x} + \mathbf{y} \in L \cup L^c$.
- Si $\mathbf{x}, \mathbf{y} \in L^c$ entonces $\mathbf{x}+\mathbf{1}, \mathbf{y}+\mathbf{1} \in L$ y así $\mathbf{x}+\mathbf{y} = (\mathbf{x}+\mathbf{1})+(\mathbf{y}+\mathbf{1}) \in L$. Luego $\mathbf{x}+\mathbf{y} \in L \cup L^c$.
- Si $\mathbf{x} \in L$ y $\mathbf{y}^c \in L^c$, entonces $\mathbf{y} + \mathbf{1} \in L$ y por lo tanto $\mathbf{x} + \mathbf{y} = (\mathbf{x} + \mathbf{y} + \mathbf{1}) + \mathbf{1} = \mathbf{z} + \mathbf{1} \in L^c$, donde $\mathbf{z} = \mathbf{x} + \mathbf{y} + \mathbf{1} \in L$, ya que L es lineal. Por lo tanto $\mathbf{x} + \mathbf{y} \in L \cup L^c$.
- La prueba para $\mathbf{x} \in L^c$, $\mathbf{y} \in L$ se realiza de manera similar.

Lo anterior prueba que $L \cup L^c$ es lineal. Su longitud es claramente igual a n. Por otro lado, mostremos que $L \cap L^c = \phi$. Supongamos que existe $x \in L \cap L^c$, entonces $x \in L$ y $x \in L^c$. Luego $x + \mathbf{1} \in L$ y así $x + (x + \mathbf{1}) = \mathbf{1} \in L$, lo cual es una contradicción. Como $M = |L| = |L^c|$ y $L \cap L^c = \phi$, entonces $|L \cup L^c| = 2M$.
Por el Teorema 3.15, $d(L \cup L^c) = \text{mín}\{d, n - d_{max}\}$, pero como $L \cup L^c$ es lineal entonces $d_{max} = w_{max}$. ■

En términos de la dimensión, dado que $M = 2^k$, con $k = \dim(L)$ entonces $2M = 2^{k+1}$ y por lo tanto $L \cup L^c$ es un $[n, k+1, d']$-código lineal binario.

Ejemplo 3.19. Sea $L = \{0000, 0101, 1100, 1001\}$ un $[4, 2, 2]$-código lineal binario, entonces $L^c = \{1111, 1010, 1100, 0110\}$. Luego $L \cup L^c = \{0000, 0101, 1100, 1001, 1111, 1010, 1100, 0110\}$ es un $[4, 3, 2]$-código lineal binario.

Definición 3.13. (Acortar un código (shortening a code)). Es un proceso que mantiene aquellas palabras de un código que tienen un símbolo común en una posición dada (por ejemplo, un **0** en la primera posición) y luego eliminar esa coordenada. De manera general, si C es un (n, M, d)-código sobre $\mathbb{F}_q$ y T es cualquier conjunto de t coordenadas, consideremos el conjunto $C(T)$ de palabras que tienen coordenada igual a cero en T. Perforando $C(T)$ sobre las t coordenadas se obtiene un código sobre $\mathbb{F}_q$ de longitud $n - t$ llamado el *código acortado* sobre T y se denota por C_T. El código acortado obtenido de las palabras del código C con una s en la i-ésima coordenada, se le denomina *sección transversal* $x_i = s$ (cross-section). Para este caso, el código acortado tiene longitud $n - 1$ y distancia mínima al menos d.

Observación 3.6. No es difícil probar que para un código lineal L, el código acortado L_T sobre un conjunto de coordenadas T, también es lineal.

Ejemplo 3.20. Con GAP, se obtiene el código de Hamming $H_2(3)$, el cual denotamos por $L1$. Con la función `AsSSortedList(L1)`, determinamos sus elementos explícitamente y con la función `ShortenedCode(L1)` se construye el código acortado de sección transversal $x_1 = 0$, el cual denotamos por $L2$. Nuevamente, con la función `AsSSortedList(L2)` podemos ver sus elementos. GAP permite también acortar el código original en un conjunto de coordenadas, en este caso la notación es la siguiente `L3:=ShortenedCode(L1,[1,3,4])` y $T = \{1, 2, 3\}$. El código acortado obtenido denotado por $L3$ se presenta con lo dicho anteriormente en el siguiente código fuente.

```
>L1 := HammingCode( 3 );
a linear [7,4,3]1 Hamming (3,2) code over GF(2)
>AsSSortedList( L1 );
[ [ 0 0 0 0 0 0 0 ], [ 0 0 0 1 1 1 1 ], [ 0 0 1 0 1 1 0 ], [ 0 0 1 1 0 0 1 ],
  [ 0 1 0 0 1 0 1 ], [ 0 1 0 1 0 1 0 ], [ 0 1 1 0 0 1 1 ], [ 0 1 1 1 1 0 0 ],
  [ 1 0 0 0 0 1 1 ], [ 1 0 0 1 1 0 0 ], [ 1 0 1 0 1 0 1 ], [ 1 0 1 1 0 1 0 ],
  [ 1 1 0 0 1 1 0 ], [ 1 1 0 1 0 0 1 ], [ 1 1 1 0 0 0 0 ], [ 1 1 1 1 1 1 1 ] ]
>L2 := ShortenedCode( L1 );
a linear [6,3,3]2 shortened code
>AsSSortedList( L2 );
[ [ 0 0 0 0 0 0 ], [ 0 0 1 1 1 1 ], [ 0 1 0 1 1 0 ], [ 0 1 1 0 0 1 ],
  [ 1 0 0 1 0 1 ], [ 1 0 1 0 1 0 ], [ 1 1 0 0 1 1 ], [ 1 1 1 1 0 0 ] ]
>L3 := ShortenedCode( L1, [ 1, 3, 4 ]);
a linear [4,1,3]2 shortened code
>AsSSortedList( L3 );
[ [ 0 0 0 0 ], [ 1 1 0 1 ] ]
```

Teorema 3.16. *Si C es un (n, M, d)-código lineal binario, entonces la sección transversal $x_i = 0$, donde no todas las x_i son iguales a cero, es un $(n-1, M/2, d')$-código lineal binario, donde $d' \geq d$.*

Demostración. Es claro que éste código es lineal y de longitud $n-1$. Probemos que tiene tamaño $M/2$. Sean L_1 y L_2 subcódigos de L cuya i-ésima componente es igual a cero y distinta de cero respectivamente. Estos conjuntos son disjuntos. Sean $x \in L_1$ y $y \in L_2$, entonces como L es lineal se tiene

$$|x + L_1| = |y + L_2|,$$

es decir

$$|L_1| = |L_2|.$$

Como L_1 y L_2 son disjuntos y la union es L entonces

$$|L_1| + |L_2| = M,$$

lo cual implica que $|L_1| = \frac{M}{2}$. Luego al eliminar la posición x_i de todas las palabras de L_1, se obtiene el código L_1' el cual es un $(n-1, M/2, d')$-código lineal binario. Para la distancia, dado que L_1', es lineal se tiene

$$d = wt(L) \leq wt(L_1) = wt(L_1') = d'$$

y así $d' \geq d$. ■

Ejemplo 3.21. Sea $L = \{00000, 11000, 00111, 11111\}$ un $[5, 2, 2]$-código lineal binario, entonces al considerar la sección transversal $x_2 = 0$ se obtiene el código lineal binario $L' = \{0000, 0111\}$, un $[4, 1, 3]$-código lineal binario.

El siguiente resultado relaciona los códigos lineales perforados y acortados con sus duales.

Teorema 3.17. *Sea L un $[n, k, d]$-código lineal sobre $\mathbb{F}_q$. Sea T un conjunto de t coordenadas. Entonces $(L^\perp)_T = (L^T)^\perp$ y $(L^\perp)^T = (L_T)^\perp$.*

Demostración. La linealidad de éstos códigos está garantizada por el Teorema 3.12 y la Observación 3.6, de ahí que tenga sentido hablar de sus duales. Sean $\mathbf{c} \in L^\perp(T)$ y $\mathbf{c}'$ la palabra obtenida de $\mathbf{c}$ cuyas coordenadas sobre T han sido removidas, es decir $\mathbf{c}' \in (L^\perp)_T$. Si $\mathbf{x} \in L$, entonces $0 = \mathbf{x} \cdot \mathbf{c} = \mathbf{x}' \cdot \mathbf{c}'$, donde $\mathbf{x}'$ es la palabra $\mathbf{x}$ perforada sobre T. Luego, $(L^\perp)_T \subseteq (L^T)^\perp$. Por otro lado, cualquier palabra $\mathbf{c} \in (L^T)^\perp$ puede extenderse a una palabra $\widehat{\mathbf{c}}$ añadiendo ceros sobre T. Si $\mathbf{x} \in L$, perforando $\mathbf{x}$ sobre T se obtiene $\mathbf{x}'$. Como $0 = \mathbf{x}' \cdot \mathbf{c} = \mathbf{x} \cdot \widehat{\mathbf{c}}$, entonces $\mathbf{c} \in (L^\perp)_T$. Luego $(L^T)^\perp \subseteq (L^\perp)_T$ y así $(L^\perp)_T = (L^T)^\perp$. Para la otra parte, reemplazando L por $L^\perp$ en la anterior igualdad se obtiene

$$\begin{aligned}((L^\perp)^\perp)_T &= ((L^\perp)^T)^\perp \\ L_T &= ((L^\perp)^T)^\perp.\end{aligned}$$

Como estos códigos son iguales, entonces sus duales también lo son, es decir

$$(L_T)^\perp = (((L^\perp)^T)^\perp)^\perp = (L^\perp)^T.$$

■

Ejemplo 3.22. Sea L un $[6, 3, 2]$-código lineal binario con matriz generadora

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

$L^\perp$ es también un $[6, 3, 2]$-código lineal binario con matriz generadora

$$G^\perp = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

Si las coordenadas son etiquetadas como $1, 2, \ldots, 6$ y $T = \{4, 5\}$, entonces matrices generadoras para el código acortado L_T y el código perforado L^T son

$$G_T = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix} \quad \text{y} \quad G^T = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}.$$

Acortando y perforando el código $L^\perp$ sobre T, se obtiene respectivamente, los siguientes códigos $(L^\perp)_T$ y $(L^\perp)^T$, los cuales tienen como matrices generadoras

$$(G^\perp)_T = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix} \quad \text{y} \quad (G^\perp)^T = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

De las matrices G_T y G^T, se tienen matrices generadoras para L_T y L^T, respectivamente, las cuales son

$$(G_T)^\perp = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{y} \quad (G^T)^\perp = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix}.$$

Es decir, $(L^\perp)_T = (L^T)^\perp$ y $(L^\perp)^T = (L_T)^\perp$.

Definición 3.14. (Sumas directas). Si C_1 es un (n_1, M_1, d_1)-código y C_2 es un (n_2, M_2, d_2)-código, ambos sobre $\mathbb{F}_q$, la *suma directa* de C_1 y C_2 es el código

$$C_1 \oplus C_2 = \{(\mathbf{c}, \mathbf{d}) | \mathbf{c} \in C_1, \mathbf{d} \in C_2\}.$$

Claramente, $C_1 \oplus C_2$ es un $(n_1 + n_2, M_1 M_2, d')$-código sobre $\mathbb{F}_q$, donde $d' = \text{mín}\{d_1, d_2\}$.

Para el caso lineal, esta construcción se enuncia como un teorema.

Teorema 3.18. *Sea L_i un $[n_i, k_i, d_i]$-código lineal sobre $\mathbb{F}_q$, para $i = 1, 2$. La suma directa de L_1 y L_2 definida por*

$$L_1 \oplus L_2 = \{(\mathbf{c}_1, \mathbf{c}_2) | \mathbf{c}_1 \in L_1, \mathbf{c}_2 \in L_2\}.$$

es un $[n_1 + n_2, k_1 + k_2, \text{mín}\{d_1, d_2\}]$-código lineal sobre $\mathbb{F}_q$.

Demostración. Sean $\lambda, \beta \in \mathbb{F}_q$, $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$, $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2)$ elementos de $L_1 \oplus L_2$, con $\mathbf{x}_1, \mathbf{y}_1 \in L_1$ y $\mathbf{x}_2, \mathbf{y}_2 \in L_2$. Entonces

$$\lambda\mathbf{x} - \beta\mathbf{y} = \lambda(\mathbf{x}_1, \mathbf{x}_2) - \beta(\mathbf{y}_1, \mathbf{y}_2) = (\lambda\mathbf{x}_1 - \beta\mathbf{y}_1, \lambda\mathbf{x}_2 - \beta\mathbf{y}_2).$$

Como $\lambda\mathbf{x}_i - \beta\mathbf{y}_i \in L_i$, para $i = 1, 2$, entonces $\lambda\mathbf{x} - \beta\mathbf{y} \in L_1 \oplus L_2$. Además, $L_1 \oplus L_2 \neq \phi$, ya que $\mathbf{0} \in L_1 \oplus L_2$. Luego $L_1 \oplus L_2$ es un $[n, k, d]$-código lineal.

La longitud de $L_1 \oplus L_2$ es clara, es decir $n = n_1 + n_2$. Como el tamaño de $L_1 \oplus L_2$ es igual al producto de los tamaños de L_1 y L_2 se tiene

$$k = \dim(L_1 \oplus L_2) = \log_q(|L_1 \oplus L_2|) = \log_q(|L_1| \cdot |L_2|) = k_1 + k_2.$$

Para la distancia mínima, supongamos sin pérdida de generalidad que $d_1 \leq d_2$. Sea $\mathbf{u} \in L_1$ tal que $wt(\mathbf{u}) = d_1$. Por definición $(\mathbf{u}, \mathbf{0}) \in L_1 \oplus L_2$ y $wt((\mathbf{u}, \mathbf{0})) = d_1$, por lo tanto

$$d(L_1 \oplus L_2) \leq wt((\mathbf{u}, \mathbf{0})) = d_1.$$

Por otro lado, para cualquier palabra no cero $(\mathbf{c}_1, \mathbf{c}_2) \in L_1 \oplus L_2$, con $\mathbf{c}_1 \in L_1$ y $\mathbf{c}_2 \in L_2$ se tiene que $\mathbf{c}_1 \neq \mathbf{0}$ ó $\mathbf{c}_2 \neq \mathbf{0}$. Por la ecuación (2.2.5) se tiene

$$wt((\mathbf{c}_1, \mathbf{c}_2)) = wt(\mathbf{c}_1) + wt(\mathbf{c}_2) \geq d_1.$$

En particular, para $(\mathbf{c}_1, \mathbf{c}_2) \in L_1 \oplus L_2$ tal que $d((\mathbf{c}_1, \mathbf{c}_2), \mathbf{0})) = d(C) = wt((\mathbf{c}_1, \mathbf{c}_2))$ se tiene $d(C) \geq d_1$, es decir, $d = \text{mín}\{d_1, d_2\}$. ■

Observación 3.7. Si L_i para $i = 1, 2$ es un código lineal que tiene como matriz generadora a G_i y como matriz de control de paridad a H_i, entonces

$$G_1 \oplus G_2 = \begin{pmatrix} G_1 & \mathbf{0} \\ \mathbf{0} & G_2 \end{pmatrix} \quad \text{y} \quad H_1 \oplus H_2 = \begin{pmatrix} H_1 & \mathbf{0} \\ \mathbf{0} & H_2 \end{pmatrix}$$

son matrices generadora y de control, respectivamente, para $L_1 \oplus L_2$.

Ejemplo 3.23. Sea $L_1 = \{000, 110, 101, 011\}$ un $[3, 2, 2]$-código lineal binario y $L_2 = \{0000, 1111\}$ un $[4, 1, 4]$-código lineal binario. Entonces

$$L_1 \oplus L_2 = \{0000000, 1100000, 1010000, 0110000, 0001111, 1101111, 1011111, 0111111\},$$

es un $[7, 3, 2]$-código lineal binario.

Dado que la distancia mínima de un código obtenido por suma directa no excede la distancia mínima de los códigos que lo forman, este código es generalmente de poco uso en la práctica por lo que sólo tiene interés teórico.
La última construcción consiste en combinar dos códigos de la misma longitud para formar un tercer código cuya longitud es igual al doble de la longitud de los códigos dados.

Definición 3.15. (La construcción $(\mathbf{u} + \mathbf{v})$). Si C_1 es un (n, M_1, d_1)-código y C_2 es un (n, M_2, d_2)-código, ambos sobre $\mathbb{F}_q$, entonces se define el código C por

$$C = \{(\mathbf{u}, (\mathbf{u} + \mathbf{v}))|\mathbf{u} \in C_1, \mathbf{v} \in C_2\},$$

el cual es un $(2n, M_1 M_2, \text{mín}\{2d_1, d_2\})$-código.

Para el caso lineal la definición anterior se enuncia como un teorema.

Teorema 3.19. *Sea L_i un $[n, k_i, d_i]$-código lineal sobre $\mathbb{F}_q$, para $i = 1, 2$. El código L definido por*

$$L = \{(\mathbf{u}, (\mathbf{u} + \mathbf{v})) : \mathbf{u} \in L_1, \mathbf{v} \in L_2\},$$

es un $[2n, k_1 + k_2, \min\{2d_1, d_2\}]$-código lineal sobre $\mathbb{F}_q$.

Demostración. Dado que $(\mathbf{0}, \mathbf{0} + \mathbf{0}) \in L$, entonces $L \neq \phi$. Sean $(\mathbf{u}_1, \mathbf{u}_1 + \mathbf{v}_1), (\mathbf{u}_2, \mathbf{u}_2 + \mathbf{v}_2) \in L$ y $\alpha, \beta \in \mathbb{F}_q$, entonces

$$\begin{aligned}
\alpha(\mathbf{u}_1, \mathbf{u}_1 + \mathbf{v}_1) + \beta(\mathbf{u}_2, \mathbf{u}_2 + \mathbf{v}_2) &= (\alpha\mathbf{u}_1, \alpha\mathbf{u}_1 + \alpha\mathbf{v}_1) + (\beta\mathbf{u}_2, \beta\mathbf{u}_2 + \beta\mathbf{v}_2) \\
&= (\alpha\mathbf{u}_1 + \beta\mathbf{u}_2, \alpha\mathbf{u}_1 + \alpha\mathbf{v}_1 + \beta\mathbf{u}_2 + \beta\mathbf{v}_2) \\
&= (\alpha\mathbf{u}_1 + \beta\mathbf{u}_2, \alpha\mathbf{u}_1 + \beta\mathbf{u}_2 + \alpha\mathbf{v}_1 + \beta\mathbf{v}_2) \\
&= (\mathbf{u}, \mathbf{u} + \mathbf{v}),
\end{aligned}$$

donde $\mathbf{u} = \alpha\mathbf{u}_1 + \beta\mathbf{u}_2 \in L_1$ y $\mathbf{v} = \alpha\mathbf{v}_1 + \beta\mathbf{v}_2 \in L_2$, ya que L_1 y L_2 son lineales. Esto prueba que L es lineal. Claramente la longitud de L es $2n$ y el tamaño es el producto del tamaño de L_1 y L_2, por lo tanto, la dimensión es $k = k_1 + k_2$.
Por otro lado, para cualquier palabra no cero $(\mathbf{c}_1, \mathbf{c}_1 + \mathbf{c}_2) \in L$ con $\mathbf{c}_1 \in L_1$ y $\mathbf{c}_2 \in L_2$ se tiene $\mathbf{c}_1 \neq 0$ ó $\mathbf{c}_2 \neq 0$.

Caso (1) Si $\mathbf{c}_2 = 0$ entonces $\mathbf{c}_1 \neq 0$. Luego

$$wt((\mathbf{c}_1, \mathbf{c}_1 + \mathbf{c}_2)) = wt((\mathbf{c}_1, \mathbf{c}_1)) = 2wt(\mathbf{c}_1) \geq 2d_1 \geq \min\{2d_1, d_2\}.$$

Caso (2) Si $\mathbf{c}_2 \neq 0$, entonces

$$\begin{aligned}
wt((\mathbf{c}_1, \mathbf{c}_1 + \mathbf{c}_2)) &= wt(\mathbf{c}_1) + wt(\mathbf{c}_1 + \mathbf{c}_2) \\
&\geq wt(\mathbf{c}_1) + (wt(\mathbf{c}_2) - wt(\mathbf{c}_1)) && \text{Lema 2.2} \\
&= wt(\mathbf{c}_2) \geq d_2 \geq \min\{2d_1, d_2\}.
\end{aligned}$$

Luego $d(L) \geq \min\{2d_1, d_2\}$. Del otro lado, sean $\mathbf{x} \in L_1$, $\mathbf{y} \in L_2$ tales que $wt(\mathbf{x}) = d_1$ y $wt(\mathbf{y}) = d_2$. Entonces

$$d(L) \leq wt((\mathbf{x}, \mathbf{x})) = 2d_1$$

y

$$d(L) \leq wt((\mathbf{0}, \mathbf{y})) = d_2.$$

Luego $d(L) \leq \min\{2d_1, d_2\}$ y así $d(L) = \min\{2d_1, d_2\}$. ■

Observación 3.8. Si L_i tiene matriz generadora G_i y matriz de control de paridad H_i, para $i = 1, 2$, entonces matrices generadora y de control de paridad para L son

$$\begin{pmatrix} G_1 & G_1 \\ 0 & G_2 \end{pmatrix} \text{ y } \begin{pmatrix} H_1 & 0 \\ -H_2 & H_2 \end{pmatrix},$$

respectivamente.

Ejemplo 3.24. Sean $L_1 = \{000, 110, 101, 011\}$ un $[3, 2, 2]$-código lineal binario y $L_2 = \{000, 111\}$ un $[3, 1, 3]$-código lineal binario. Entonces

$$L = \{000000, 110110, 101101, 011011, 000111, 110001, 101010, 011100\},$$

es un $[6, 3, 3]$-código lineal binario.

El Teorema 3.19 puede utilizarse para demostrar el siguiente corolario.

Corolario 3.3. *Sea A un $[n, k, d]$-código lineal binario. Entonces el código L definido por*

$$L = \{(\mathbf{c}, \mathbf{c})\} \cup \{(\mathbf{c}, \mathbf{1} + \mathbf{c}) : \mathbf{c} \in A\},$$

donde $\mathbf{1} = 11 \cdots 1$, es un $[2n, k + 1, \text{mín}\{n, 2d\}]$-código lineal binario.

Demostración. Del Teorema 3.19, tomando $L_1 = A$ y $L_2 = \{\mathbf{0}, \mathbf{1}\}$, donde $\mathbf{0} = 00 \cdots 0$ se tiene el resultado deseado. ■

Capítulo 4

El problema fundamental de la Teoría de Códigos

Una cuestión fundamental en la Teoría de Códigos es encontrar códigos con el máximo tamaño posible, para una longitud dada, esto con el fin de codificar la máxima cantidad de información posible. Lo ideal sería lograr el anterior objetivo y que la distancia mínima del código sea lo mayor posible, con el fin de detectar y corregir la mayor cantidad de errores que puedan ocurrir en la transmisión. Sin embargo, estos dos objetivos son incompatibles porque entre más palabras tenga un código menor puede ser su distancia mínima.

Sea A un conjunto de orden q, habitualmente se denota por $A_q(n, d)$ al mayor valor posible del tamaño M, para el cual existe un código q-ario con longitud n y distancia mínima d, es decir

$$A_q(n, d) = \text{máx}\{M\text{: existe un } (n, M, d)\text{-código sobre } A\}.$$

El valor de $A_q(n, d)$ tiene un papel fundamental en la Teoría de Códigos y se han dedicado muchos esfuerzos en determinar sus valores. El problema de determinar los valores de $A_q(n, d)$ es conocido como el *problema fundamental de la Teoría de Códigos*. Sin embargo, este es un problema muy complejo y muchos de los resultados obtenidos se centran en determinar $A_q(n, d)$ para valores pequeños de q, n y d, o en encontrar cotas tanto superiores como inferiores para esta función.

Definición 4.1. (Código óptimo). Un (n, M, d)-código para el cual $M = A_q(n, d)$ se denomina un código óptimo.

En lugar de considerar todos los códigos, también se puede restringir este problema a los códigos lineales y obtener la siguiente notación

$$B_q(n, d) = \text{máx}\{q^k\text{: existe un } [n, k, d]\text{-código sobre } \mathbb{F}_q\}.$$

También se podría pensar en buscar el código con la longitud más pequeña que contiene M palabras código y con distancia mínima d, esta longitud más pequeña se denota por $N_q(M, d)$. El

problema de determinar $N_2(M, d)$ es equivalente al de encontrar $A_2(n, d)$, ver [12, pag. 524, problema (1)]. Así como en fijar la longitud n y el tamaño M del código y tratar de construir códigos con distancia mínima tan grande como sea posible, con el fin de poder corregir la mayor cantidad de errores, para códigos lineales este problema es abordado por Brouwer en [2]. Sin embargo, estos problemas no serán tratados en este trabajo.

4.1. Resultados Elementales

A pesar de la dificultad de determinar los valores exactos para $A_q(n, d)$ y $B_q(n, d)$, se tienen algunas propiedades sencillas sobre dichos valores.

Teorema 4.1. *Sea q una potencia de un número primo, entonces*

1. $B_q(n, d) \leq A_q(n, d) \leq q^n$.
2. $B_q(n, 1) = A_q(n, 1) = q^n$.
3. $B_q(n, n) = A_q(n, n) = q$.

Demostración.

1. La primera desigualdad se deduce de las definiciones, dado que $B_q(n, d)$ es el máximo tamaño para un código lineal, mientras que $A_q(n, d)$ lo es para un código cualquiera, en particular uno lineal. La segunda desigualdad es clara, pues el máximo número de palabras q-arias de longitud n es q^n.

2. Claramente, $\mathbb{F}_q^n$ es un $[n, n, 1]$-código lineal sobre $\mathbb{F}_q$. Entonces $q^n \leq B_q(n, 1) \leq A_q(n, 1) \leq q^n$. Por lo tanto, $B_q(n, 1) = A_q(n, 1) = q^n$.

3. Sea C un (n, M, n)-código, entonces dado que la longitud de C es n y la distancia entre cualquier par de palabras distintas de C es a lo menos n, obligatoriamente la distancia entre cualquier par de palabras distintas debe ser exactamente n. Por lo tanto, las palabras de C deben diferir en todas las coordenadas, de donde al considerar únicamente la primera posición de las M palabras de C, esta puede tomar a lo más q posibles valores, $0, 1, \ldots, q-1$, es decir, $M \leq q$. Entonces,

   $$B_q(n, n) \leq A_q(n, n) \leq q.$$

   Por otro lado, el código de repetición de longitud n, es decir, $\{aa\cdots a : a \in \mathbb{F}_q\}$, es un $[n, 1, n]$-código lineal y por tanto un (n, q, n)-código sobre $\mathbb{F}_q$. Por lo tanto,

   $$B_q(n, n) = A_q(n, n) = q.$$

■

Lema 4.1. *Si $d < d^*$, entonces*

$$A_q(n, d^*) \leq A_q(n, d).$$

Demostración. Sea C' un (n, M, d^*)-código q-ario óptimo y sean $x, y \in C'$ tales que $d(x, y) = d^*$. Llamemos y' a la palabra obtenida al cambiar $t = d^* - d$ coordenadas de y en las cuales x y y difieren, de tal manera que ahora coincidan. Claramente $y' \notin C'$, dado que $d(y, y') = t = d^* - d < d^*$, entonces formemos el código C cambiando la palabra código y de C' por y'. Probemos que la distancia del código C es d.

Como $d(x, y') = d^* - t = d$, probemos que $d(z, y') \geq d$, para toda $z \in C$, con $z \neq y'$. Supongamos que $d(y, z) = k \geq d^*$, dado que $y, z \in C'$. Como y' se obtiene al cambiar t coordenadas de y, lo peor que puede pasar al hacer estos t cambios es que z y y' coincidan en estas coordenadas, es decir que $d(y', z) = k - t$. Luego

$$d(y', z) = k - t \leq d^* - t = d,$$

lo que se quería probar.

Así, C es un (n, M, d)-código q-ario. De esta forma,

$$A_q(n, d^*) \leq A_q(n, d).$$

■

Teorema 4.2. *Para todo $n \geq 2$,*

$$A_q(n, d) \leq qA_q(n - 1, d).$$

Demostración. Sea C un (n, M, d)-código q-ario óptimo, es decir que $M = A_q(n, d)$. Supongamos que para toda sección transversal $x_1 = i$, con $i = 1, 2, \ldots, q$, se tiene que $|x_1 = i| < M/q$, donde $|x_1 = i|$ denota el tamaño de la sección transversal $x_1 = i$. Sin embargo, es claro que

$$|C| = \sum_{i=1}^{q} |x_1 = i|.$$

Si consideramos $|x_1 = j| = \text{máx}\{|x_1 = i| \text{ para } i = 1, 2, \ldots, q\}$, tenemos que

$$|C| = \sum_{i=1}^{q} |x_1 = i| \leq q|x_1 = j| < q\left(\frac{M}{q}\right) = M,$$

lo cual es una contradicción. Entonces, alguna de las q secciones transversales $x_1 = i$ de C debe contener al menos M/q palabras código, además, según lo visto en la Definición 3.13, la distancia mínima es al menos d. Es decir que esta sección transversal seria un $(n - 1, M/q, d')$-código q-ario, con $d \leq d'$. Por lo tanto

$$\frac{A_q(n, d)}{q} = \frac{M}{q} \leq A_q(n - 1, d'),$$

pero por el Lema 4.1 se tiene que $A_q(n-1,d') \leq A_q(n-1,d)$. Luego,

$$\frac{A_q(n,d)}{q} \leq A_q(n-1,d)$$
$$A_q(n,d) \leq qA_q(n-1,d).$$

■

Teorema 4.3. *Para códigos binarios,*

$$A_2(n,2t+1) = A_2(n+1,2t+2).$$

En otras palabras, si d es par, entonces $A_2(n,d) = A_2(n-1,d-1)$.

Demostración. Se sigue del Teorema 3.13, según el cual un $(n,M,2t+1)$-código binario existe si y sólo si existe un $(n+1,M,2t+2)$-código binario. ■

Del anterior teorema se sigue que para códigos binarios, es suficiente determinar $A_2(n,d)$ únicamente para los valores impares de d (o sólo para los valores pares).

En seguida se presentan con las respectivas pruebas, algunas cotas para $A_q(n,d)$, al igual que algunos resultados para códigos binarios de peso constante, es decir, donde todas las palabras código tienen el mismo peso y por último algunos ejemplos.

4.2. Cotas para $A_q(n,d)$

En esta sección se presentan algunos de los resultados más conocidos sobre cotas para $A_q(n,d)$. Se iniciará con la cota inferior de Gilbert-Varshamov y luego se pasará a las cotas superiores de Singleton, de Hamming y por último la cota de Plotkin. Para la primera cota se introduce la siguiente definición.

Definición 4.2. (Código máximo). Un (n,M,d)-código se llama máximo si no está contenido en algún $(n,M+1,d)$-código.

Ejemplo 4.1. El código $C = \{000, 111, 333\}$ sobre el alfabeto $A = \mathbb{F}_3$ logra ser más eficiente si se añade la palabra código 222 (en el sentido que se puede enviar más información), la cual incrementa la tasa del código sin cambiar la distancia mínima. Además el código

$$B = \{000, 111, 222, 333\}$$

es un código máximo.

Teorema 4.4. *(Cota de Gilbert-Varshamov).*

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

Demostración. Si C es un (n, M, d)-código máximo sobre $\mathbb{F}_q$, entonces ninguna palabra en $\mathbb{F}_q^n$ tiene distancia d o más con todas las palabras en C. En consecuencia, todas las esferas $S_q(c, d-1)$ con centro en las palabras de C deben cubrir $\mathbb{F}_q^n$, es decir

$$\bigcup_{c_i \in C} S_q(c_i, d-1) = \mathbb{F}_q^n,$$

de donde

$$\begin{aligned} MV_q(n, d-1) &\geq q^n \\ M &\geq \frac{q^n}{V_q(n, d-1)}. \end{aligned}$$

Luego,

$$A_q(n,d) \geq M \geq q^n \Big/ \sum_{i=0}^{d-1} \binom{n}{i} (q-1)^i.$$

■

Teorema 4.5. *(Cota de Singleton).*

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

Demostración. Sea C un (n, M, d)-código q-ario. Si se quitan las últimas $d-1$ coordenadas de cada palabra en C, las M palabras resultantes deben ser diferentes. Dado que la longitud de estas palabras es $n-d+1$ se obtiene que,

$$M \leq q^{n-d+1},$$

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

La cota de Singleton implica que cualquier $[n, k, d]$-código lineal debe satisfacer

$$\begin{aligned} q^k &\leq q^{n-d+1} \\ k &\leq n-d+1. \end{aligned}$$

Así, se tiene la siguiente desigualdad,

$$d \leq n-k+1. \tag{4.2.1}$$

Un código lineal para el cual se tiene la igualdad en la expresión (4.2.1), se denomina un "*código separable por distancia máxima*", o código MDS, dado que este tiene la máxima distancia mínima posible para cualquier código con una longitud y una dimensión dadas. Un código MDS tiene

parámetros $[n, n-d+1, d]$ ó $[n, k, n-k+1]$.

Por otro lado, si se quita cualquier conjunto de $n-k=d-1$ posiciones de las palabras en un código C que sea MDS, se deben obtener q^k palabras distintas de longitud k, es decir que se obtiene $\mathbb{F}_q^k$. Por tanto, cualquier conjunto de k posiciones en C contiene todas las posibles k-uplas, de donde C es sistemático sobre cualquier conjunto de k posiciones.

Teorema 4.6. *(Cota del empaquetamiento de la esfera o Cota de Hamming).*

$$A_q(n,d) \leq q^n \Big/ \sum_{i=0}^{t} \binom{n}{i} (q-1)^i,$$

donde

$$t = \left\lfloor \frac{d-1}{2} \right\rfloor.$$

Demostración. Sea C un (n, M, d)-código q-ario óptimo. Por el Teorema 2.6, las esferas de radio $pr(C) = t = \lfloor \frac{d-1}{2} \rfloor$ con centro en cada palabra de C son disjuntas, luego

$$\begin{aligned} MV_q(n,t) &\leq q^n \\ M &\leq \frac{q^n}{V_q(n,t)} \\ A_q(n,d) = M &\leq q^n \Big/ \sum_{i=0}^{t} \binom{n}{i} (q-1)^i. \end{aligned}$$

■

Teorema 4.7. *(Cota de Plotkin).*
Sea $\theta = \dfrac{q-1}{q}$*, si* $d > \theta n$*, entonces*

$$A_q(n,d) \leq \frac{d}{d-\theta n}.$$

Demostración. Sean C un (n, M, d)-código q-ario, $\theta = \frac{q-1}{q}$. Consideremos las sumas de las distancias entre las palabras código, la cual está dada por

$$S = \sum_{c \in C} \sum_{b \in C} d(c,b).$$

Dado que la distancia mínima de C es d, se tiene que

$$S \geq M(M-1)d. \tag{4.2.2}$$

Por otro lado, sea B la matriz de tamaño $M \times n$, cuyas filas son las palabras de C. Para $1 \leq i \leq n$, sea K_{ij} el número de veces que $j \in \mathbb{F}_q$ aparece en la i-ésima columna de B. Entonces,

$$\sum_{j \in \mathbb{F}_q} K_{ij} = M,$$

para todo $1 \leq i \leq n$.

Además, recordemos que dados $\mathbf{c}, \mathbf{x} \in C$, con $\mathbf{c} = c_1c_2\cdots c_n$ y $\mathbf{x} = x_1x_2\cdots x_n$

$$d(\mathbf{c}, \mathbf{x}) = d(c_1, x_1) + d(c_2, x_2) + \cdots + d(c_n, x_n).$$

Ahora, sea $j \in \mathbb{F}_q$. Si $K_{ij} = 0$ (es decir que j no aparece en la i-sima columna), se tiene que esta columna no suma en S. Lo mismo para el caso cuando $K_{ij} = M$, como j aparece en toda la i-ésima columna, esta columna no aporta a la suma S.
En general, supongamos que $K_{ij} = t$, es decir que j aparece para t filas de B en la i-ésima columna y que no aparece en las $M - t$ filas restantes. Luego, cada una de las t veces en que j aparece en la i-ésima columna, dicha columna aporta $M - t$ unos a la suma S. En consecuencia,

$$S = \sum_{i=1}^{n}\left(\sum_{j=0}^{q-1} K_{ij}(M - K_{ij})\right) = \sum_{i=1}^{n}\left(M^2 - \sum_{j=0}^{q-1} K_{ij}^2\right).$$

Por la desigualdad de Cauchy-Schwarz, Teorema 1.3, tenemos que $|\mathbf{x}\cdot\mathbf{1}| \leq \| \mathbf{x} \| \| \mathbf{1} \|$, donde $\mathbf{x} = K_{i0}K_{i1}\cdots K_{iq-1}$ y $\mathbf{1} = 11\ldots1$, a partir de lo cual se deduce que

$$\begin{aligned}\left(\sum_{j=0}^{q-1} K_{ij}\right)^2 &\leq q\sum_{j=0}^{q-1} K_{ij}^2\\ \frac{1}{q}M^2 &\leq \sum_{j=0}^{q-1} K_{ij}^2.\end{aligned}$$

Luego,

$$S = \sum_{i=1}^{n}\left(M^2 - \sum_{j=0}^{q-1} K_{ij}^2\right) \leq \sum_{i=1}^{n}\left(M^2 - \frac{M^2}{q}\right) = nM^2\left(1 - \frac{1}{q}\right) = n\theta M^2. \tag{4.2.3}$$

Por lo tanto, de las desigualdades (4.2.2) y (4.2.3) se tiene

$$\begin{aligned}M(M-1)d \leq S &\leq \theta M^2 n\\ (M-1)d &\leq \theta M n\\ Md - d &\leq \theta M n\\ M(d - \theta n) &\leq d\\ M &\leq \frac{d}{d-\theta n}.\end{aligned}$$

■

Para códigos binarios, se tiene otra versión de la cota de Plotkin, sin embargo, antes de ésta presentamos el siguiente lema.

Lema 4.2. *Sea $m > 0$ un número impar y supongamos que $m = x + y$, con x, y enteros positivos. Entonces*

$$xy \leq \left(\frac{m-1}{2}\right)\left(\frac{m+1}{2}\right).$$

Demostración. Si uno de los dos es igual a $\frac{m-1}{2}$, podemos suponer que $x = \frac{m-1}{2}$, en consecuencia $y = \frac{m+1}{2}$. Por tanto, en este caso se tiene la igualdad.
En caso contrario, supongamos sin pérdida de generalidad que $x < \frac{m-1}{2}$, digamos $x = \frac{m-1}{2} - t$, con $t > 0$, entonces $y = \frac{m+1}{2} + t$ y por lo tanto

$$\begin{aligned} xy &= \left(\frac{m-1}{2} - t\right)\left(\frac{m+1}{2} + t\right) \\ &= \left(\frac{m-1}{2}\right)\left(\frac{m+1}{2}\right) + \frac{t(m-1)}{2} - \frac{t(m+1)}{2} - t^2 \\ &= \left(\frac{m-1}{2}\right)\left(\frac{m+1}{2}\right) - t - t^2 \\ &< \left(\frac{m-1}{2}\right)\left(\frac{m+1}{2}\right). \end{aligned}$$

■

Teorema 4.8. *(Cota de Plotkin para códigos binarios).*

1. *Si d es par,*

$$A_2(n,d) \leq \begin{cases} 2\lfloor d/(2d-n)\rfloor & \text{para } n < 2d, \\ 4d & \text{para } n = 2d. \end{cases}$$

2. *Si d es impar,*

$$A_2(n,d) \leq \begin{cases} 2\lfloor (d+1)/(2d+1-n)\rfloor & \text{para } n < 2d+1, \\ 4d+4 & \text{para } n = 2d+1. \end{cases}$$

Demostración.

1. Sea C un (n, M, d)-código binario con d un número par. Si $n < 2d$, por el teorema anterior tenemos que $M \leq \frac{2d}{2d-n}$. Si M es par, digamos $M = 2t$, con $t \in \mathbb{Z}^+$, entonces

$$\begin{aligned} 2t &\leq 2\frac{d}{2d-n} \\ t &\leq \left\lfloor \frac{d}{2d-n} \right\rfloor \\ 2t &\leq 2\left\lfloor \frac{d}{2d-n} \right\rfloor \\ M &\leq 2\left\lfloor \frac{d}{2d-n} \right\rfloor. \end{aligned}$$

Por otro lado, si M es impar, en la demostración del teorema anterior obsérvese que

$$S = \sum_{i=1}^{n}[k_{i,0}(M - k_{i,0}) + k_{i,1}(M - k_{i,1})] = \sum_{i=1}^{n} 2k_{i,0}k_{i,1} = 2\sum_{i=1}^{n} k_{i,0}k_{i,1},$$

dado que $k_{i,0} + k_{i,1} = M$. Sin embargo, por el Lema 4.2 $k_{i,0}k_{i,1} \leq \left(\frac{M+1}{2}\right)\left(\frac{M-1}{2}\right)$, en consecuencia se tiene

$$S \leq 2\left(\frac{n(M+1)(M-1)}{4}\right) = \frac{n(M+1)(M-1)}{2}.$$

Luego,

$$\begin{aligned} M(M-1)d \leq S &\leq \frac{n(M+1)(M-1)}{2} \\ Md &\leq \frac{n(M+1)}{2} \\ 2Md &\leq nM + n \\ M(2d-n) &\leq n \\ M &\leq \frac{n}{2d-n} \\ M &\leq \frac{2d}{2d-n} - 1 \end{aligned}$$

y dado que M es impar, por un razonamiento semejante al del caso par, obtenemos que

$$M \leq 2\left\lfloor \frac{d}{2d-n} \right\rfloor.$$

Para $n = 2d$, sea $d = 2k$, entonces por el Teorema 4.2 y por el caso anterior (cuando $n < 2d$), se tiene que

$$A_2(n,d) = A_2(4k,2k) \leq 2A_2(4k-1,2k) \leq 4\left\lfloor \frac{2k}{4k-(4k-1)} \right\rfloor = 8k = 4d.$$

2. Sea d un número impar, supongamos que $n < 2d+1$, entonces por el Teorema 4.3 y por el item 1, se tiene

$$A_2(n,d) = A_2(n+1,d+1) \leq 2\left\lfloor \frac{d+1}{2(d+1)-(n+1)} \right\rfloor = 2\left\lfloor \frac{d+1}{2d+1-n} \right\rfloor.$$

Para $n = 2d+1$, aplicando los teoremas 4.2, 4.3 y el item 1, se tiene

$$A_2(2d+1,d) = A_2(2d+2,d+1) \leq 2A_2(2d+1,d+1) \leq 4\left\lfloor \frac{d+1}{2(d+1)-(2d+1)} \right\rfloor = 4(d+1).$$

■

4.3. Cotas para $A(n, d, w)$

Otra forma de afrontar el estudio de $A_q(n, d)$ es centrarse en el caso $q = 2$, es decir para códigos binarios, pero de peso constante. Por tanto, la notación $A(n, d, w)$ denota el tamaño máximo para un código binario de longitud n, distancia mínima al menos d y peso constante w. A continuación se presentan algunas propiedades básicas para los valores de $A(n, d, w)$.

Observación 4.1. La distancia entre dos palabras de igual peso es siempre par, ya que si fuese impar, una palabra debería tener al menos un uno más que la otra. Así, la distancia mínima en un código de peso constante debe ser par. Por tanto, el hecho que en la notación $A(n, d, w)$ se mencione que la distancia sea "al menos d", garantiza la existencia de códigos de este tipo cuando d es impar.

Lema 4.3.

1. $A(n, 2k, k) = \left\lfloor \frac{n}{k} \right\rfloor$.
2. $A(n, 2k, w) = A(n, 2k, n - w)$.
3. $A(n, 2k - 1, w) = A(n, 2k, w)$.

Demostración.

1. Para construir un (n, M, d)-código binario C, con $d \geq 2k$ y $M = A(n, 2k, k)$, todas las palabras en C deben tener k unos y la distancia al menos $2k$, luego, ninguna palabra puede tener un 1 en la misma posición que otra. Así, el mayor número de palabras binarias, de longitud n que cumplen las anteriores condiciones es t, donde $n = tk + r$, con $0 \leq r < k$. Es decir
$$M = t = \left\lfloor \frac{n}{k} \right\rfloor.$$
2. Sea C un (n, M, d)-código binario, con $d \geq 2k$, tal que $M = A(n, 2k, k)$ y sea C' el código obtenido por permutación de los símbolos cero y uno en las palabras de C. Entonces C y C' tienen la misma distancia por ser códigos equivalentes, ver Teorema 2.5. Además, todas las palabras en C' tienen peso $n - w$, por lo tanto
$$A(n, 2k, w) \leq A(n, 2k, n - w).$$
De forma similar se prueba la otra desigualdad.
3. Dado que la distancia entre palabras de igual peso debe ser par, el número máximo de palabras binarias con w unos que se pueden formar, de tal manera que la distancia sea mayor o igual que $2k - 1$, es el mismo que se requiere para que la distancia sea mayor o igual que $2k$.

Luego,

$$A(n, 2k-1, w) = A(n, 2k, w).$$

■

Teorema 4.9. $A(n, 2k, w) \leq \left\lfloor \dfrac{kn}{w^2 - wn + kn} \right\rfloor$.

Demostración. Sean C un $(n, M, 2k)$-código con todas las palabras de peso w, para el cual $M = A(n, 2k, w)$, B la matriz de tamaño $M \times n$ cuyas filas son las palabras en C y k_i el número de unos en la i-ésima columna de B. Calculemos de dos formas la suma S de los productos escalares $c_i \cdot c_j$ de todos los pares distintos de filas de B.
Por un lado, si $i \neq j$, entonces

$$c_i \cdot c_j = w(c_i \cap c_j) = \frac{1}{2}[w(c_i) + w(c_j) - d(c_i, c_j)] \leq \frac{1}{2}[2w - 2k] = w - k,$$

y por lo tanto

$$S = \sum_{i=1}^{M} \sum_{\substack{j=1 \\ j \neq i}}^{M} c_i \cdot c_j \leq (w-k)M(M-1).$$

Por otro lado, viendo la sumatoria por columnas, la i-ésima columna aporta a la suma S con $k_i(k_i - 1)$, de donde se obtiene que

$$S = \sum_{i=1}^{n} k_i(k_i - 1) = \sum_{i=1}^{n} k_i^2 - \sum_{i=1}^{n} k_i = \sum_{i=1}^{n} k_i^2 - wM,$$

pero por la desigualdad de Cauchy-Schwarz, Teorema 1.3 tenemos

$$\begin{aligned} \left(\sum_{i=1}^{n} k_i \right)^2 &\leq n \sum_{i=1}^{n} k_i^2 \\ \frac{1}{n} \left(\sum_{i=1}^{n} k_i \right)^2 &\leq \sum_{i=1}^{n} k_i^2 \\ \frac{1}{n}(wM)^2 &\leq \sum_{i=1}^{n} k_i^2, \end{aligned}$$

de donde

$$\frac{w^2 M^2}{n} - wM \leq S.$$

Entonces, de las dos formas de ver S obtenemos que

$$\begin{aligned}
\frac{w^2M^2}{n} - wM &\leq S \leq (w-k)M(M-1) \\
\frac{w^2M^2 - wMn}{n} &\leq M(wM - w - Mk + k) \\
w^2M - wn &\leq wMn - wn - Mkn + kn \\
M(w^2 - wn + kn) &\leq kn \\
M &\leq \left\lfloor \frac{kn}{w^2 - wn + kn} \right\rfloor .
\end{aligned}$$

■

A continuación se presenta un resultado que brinda una desigualdad recursiva para $A(n, 2k, w)$.

Teorema 4.10. $A(n, 2k, w) \leq \left\lfloor \frac{n}{w} A(n-1, 2k, w-1) \right\rfloor$.

Demostración. Sea C un $(n, M, 2k)$-código cuyas palabras tienen todas peso w, con $M = A(n, 2k, w)$. Por un lado, el número total de unos de todas las palabras en C es wM.
Por otro lado, la sección transversal $x_1 = 1$ de C tiene longitud $n-1$, distancia mínima al menos $2k$ y peso constante $w-1$, de donde su tamaño es a lo más $A(n-1, 2k, w-1)$. En consecuencia, hay a lo más $A(n-1, 2k, w-1)$ palabras en C con un uno en la primera posición. Aplicando este razonamiento a las $n-1$ posiciones restantes se obtiene que el número total de unos de todas las palabras en C es a lo más $nA(n-1, 2k, w-1)$. Por lo tanto

$$\begin{aligned}
wM &\leq nA(n-1, 2k, w-1) \\
M &\leq \left\lfloor \frac{n}{w} A(n-1, 2k, w-1) \right\rfloor .
\end{aligned}$$

■

Corolario 4.1.

$$A(n, 2k-1, w) = A(n, 2k, w) \leq \left\lfloor \frac{n}{k} \left\lfloor \frac{n-1}{k-1} \left\lfloor \cdots \left\lfloor \frac{n-w+k}{k} \right\rfloor \cdots \right\rfloor \right\rfloor \right\rfloor .$$

Demostración. La igualdad se sigue de la parte 3 del Lema 4.3. Para probar la desigualdad, primero aplicamos el Teorema 4.10 reiteradamente hasta obtener

$$\begin{aligned}
A(n, 2k, w) &\leq \left\lfloor \frac{n}{w} \left\lfloor \frac{n-1}{w-1} \left\lfloor \cdots \left\lfloor \frac{n-(w-k-1)}{w-(w-k-1)} A(n-(w-k), 2k, w-(w-k)) \right\rfloor \cdots \right\rfloor \right\rfloor \right\rfloor \\
&= \left\lfloor \frac{n}{w} \left\lfloor \frac{n-1}{w-1} \left\lfloor \cdots \left\lfloor \frac{n-w+k+1}{k+1} A(n-w+k, 2k, k) \right\rfloor \cdots \right\rfloor \right\rfloor \right\rfloor .
\end{aligned}$$

Entonces, por la parte 1 del Lema 4.3,

$$A(n-w+k, 2k, k) = \left\lfloor \frac{n-w+k}{k} \right\rfloor .$$

Por lo tanto,

$$A(n, 2k-1, w) = A(n, 2k, w) \leq \left\lfloor \frac{n}{w} \left\lfloor \frac{n-1}{w-1} \left\lfloor \cdots \left\lfloor \frac{n-w+k}{k} \right\rfloor \cdots \right\rfloor \right\rfloor \right\rfloor.$$

■

Algunos de los resultados vistos anteriormente, permiten calcular valores exactos para $A_q(n, d)$, como lo muestra el siguiente ejemplo.

Ejemplo 4.2. Para $A_2(6, 4)$, por la cota de Plotkin se tiene

$$A_2(6, 4) \leq 4. \tag{4.3.1}$$

Por otro lado, el código $C = \{00000, 01101, 10110, 11011\}$ es un $(5, 4, 3)$-código binario. De donde $A_2(5, 3) \geq 4$, pero por el Teorema 4.3 tenemos que $A_2(5, 3) = A_2(6, 4)$ y por la desigualdad (4.3.1)

$$4 \leq A_2(5, 3) = A_2(6, 4) \leq 4.$$

Por lo tanto, $A_2(6, 4) = 4$, lo cual corresponde con el valor que se verá en la Tabla 4.1.

También cabe mencionar que el paquete GUAVA de GAP tiene funciones para calcular los valores de algunas de las cotas aquí tratadas, entre otras. Las cotas de Gilbert-Varshamov 4.4, Singleton 4.5, Hamming 4.6 y Plotkin 4.7, se pueden calcular en GAP mediante las siguientes funciones `LowerBoundSpherePacking`, `UpperBoundSingleton`, `UpperBoundHamming` y `UpperBoundPlotkin` respectivamente.

Ejemplo 4.3. Determinemos en GAP estas cotas para $A_3(10, 5)$,

```
> LowerBoundSpherePacking(10,5,3);
[ 13
>UpperBoundSingleton(10,5,3);
[ 729
>UpperBoundHamming( 10,5,3);
[ 293
```

De donde, por la cota de Gilbert-Varshamov 4.4 se tiene que $A_3(10, 5) \geq 13$, por la cota de Singleton 4.5 $A_3(10, 5) \leq 729$ y por la cota de Hamming 4.6 $A_3(10, 5) \leq 293$. Se puede observar que en este caso la cota Hamming da una mejor aproximación que la cota de Singleton. Sin embargo, con estos resultados aún existe una gran diferencia entre la cota inferior (13) y la cota superior (293).

En el ejemplo anterior, es de resaltar que no se puede aplicar la cota de Plotkin 4.7, dado que no se satisface que $d > \theta n$, con $\theta = \frac{q-1}{q}$, pues en este caso $\theta = 2/3$ y $d = 5 < (2/3)10 = \theta n$.

Otra función de gran importancia en GAP es `UpperBound`. Ésta presenta la mejor cota superior que el programa posee para $A_q(n, d)$. Por ejemplo para $A_3(10, 5)$ se tiene

```
> UpperBound( 10, 5, 3);
293
```

En este caso dicha cota coincide con la cota superior de Hamming, que es 293. Otro ejemplo para $A_3(14, 10)$ se muestra a continuación

```
>UpperBoundSingleton(14,10,3);
[ 243
>UpperBoundHamming(14,10,3);
[ 247
>UpperBoundPlotkin(14,10,3);
[15
>UpperBound(14,10,3);
[15
```

Ahora, la mejor cota que el programa tiene para $A_3(14, 10)$ coincide con la cota de Plotkin.

Por último se presenta la Tabla 4.1 con los valores de $A_2(n, d)$ tomada de [10, pag. 54, Tabla 2.1].

n	$d = 4$	$d = 6$	$d = 8$	$d = 10$
6	4	2	1	1
7	8	2	1	1
8	16	2	2	1
9	20	4	2	1
10	40	6	2	2
11	72	12	2	2
12	144	24	4	2
13	256	32	4	2
14	512	64	8	2
15	1024	128	16	4
16	2048	256	32	4
17	2720-3276	256-340	36-37	6
18	5312-6552	512-680	64-72	10
19	10496-13104	1024-1288	128-144	20
20	20480-26208	2048-2372	256-279	40
21	36864-43689	2560-4096	512	42-48
22	73728-87378	4096-6941	1024	50-88
23	147456-173491	8192-13774	2048	76-150
24	294912-344308	16384-24106	4096	128-280

Tabla 4.1: Algunos valores para $A_2(n, d)$.

Una tabla con más valores de $B_2(n, d)$ se puede encontrar en [11, pag. 34, Tabla 1].

Capítulo 5

Otros Códigos

En la primera sección de este capítulo se presenta una clase de códigos lineales bastante interesantes debido a su estructura y a las propiedades que presentan. En la segunda sección, se presentará una clase de códigos no lineales basados en la noción de conjunto B_h que permiten crear códigos con parámetros específicos.

5.1. Códigos Cíclicos

En el Capítulo 3 se vio la importancia de los códigos lineales debido a su estructura algebraica. En esta sección se estudiará una clase especial de códigos lineales, los códigos cíclicos, los cuales tienen una mejor estructura matemática que facilita los procesos de codificación y decodificación. Además, veremos a continuación que un código cíclico de longitud n queda determinado totalmente por un polinomio de grado menor que n.

Se iniciará con la definición de código cíclico basada en su estructura combinatoria, para luego transformarla en una estructura algebraica.

Definición 5.1 (Código cíclico)**.** Un código lineal $L \subset \mathbb{F}_q^n$ es cíclico si

$$c_0c_1\cdots c_{n-2}c_{n-1} \in L \Longrightarrow c_{n-1}c_0c_1\cdots c_{n-2} \in L.$$

De acuerdo a la anterior definición, un código lineal L es cíclico si es cerrado bajo el desplazamiento cíclico

$$c_0c_1\cdots c_{n-2}c_{n-1} \longrightarrow c_{n-1}c_0c_1\cdots c_{n-2},$$

en consecuencia, L es cerrado bajo todos los desplazamientos cíclicos

$$c_0c_1\cdots c_{n-2}c_{n-1} \longrightarrow c_k\cdots c_{n-1}c_0c_1\cdots c_{k-1}.$$

Otra forma de caracterizar un código cíclico consiste en identificar la estructura algebraica que estos poseen. Si L es un código lineal sobre $\mathbb{F}_q$, entonces a cada palabra código

$\mathbf{c} = c_0c_1\cdots c_{n-1}$, se le asocia una clase lateral de $R_n = \mathbb{F}_q[x]/\langle x^n - 1\rangle$ de la siguiente manera,

$$\phi : \mathbb{F}_q^n \longrightarrow R_n \tag{5.1.1}$$
$$\phi(\mathbf{c}) = [c_0 + c_1x + \cdots + c_{n-1}x^{n-1}],$$

donde ϕ es un isomorfismo de espacios vectoriales de L sobre el subespacio $\phi(L)$ de R_n.

Es decir, una palabra de un código lineal puede ser vista como una clase lateral en R_n, aunque en general solo se trabaja con el polinomio representante de la clase lateral como se verá en el resto de la sección. Además, si L es un código cíclico se tiene el siguiente resultado.

Teorema 5.1. *Sea ϕ la aplicación definida en la expresión (5.1.1). Entonces, un subconjunto no vació L de $\mathbb{F}_q^n$ es un código cíclico si y sólo si $\phi(L)$ es un ideal de R_n.*

Demostración. Sea ϕ la función definida en $\mathbb{F}_q^n$ sobre R_n. Si L es un código cíclico, probemos que $\phi(L)$ es un ideal de R_n.

Sean $p(x), q(x) \in \phi(L)$, entonces claramente $p(x) - q(x) \in \phi(L)$, dado que $\phi(L)$ es un subespacio vectorial. Ahora, sea $p(x) \in \phi(L)$, con

$$p(x) = c_0 + c_1x + \cdots + c_{n-1}x^{n-1} = \phi(c_0c_1\cdots c_{n-1}),$$

donde, $c_0c_1\cdots c_{n-1} \in L$. Entonces, el polinomio

$$xp(x) = c_0x + c_1x^2 + \cdots + c_{n-2}x^{n-1} + c_{n-1}x^n \equiv c_{n-1} + c_0x + c_1x^2 + \cdots + c_{n-2}x^{n-1},$$

es también un elemento de $\phi(L)$, dado que L es cíclico. De esta manera, $x^2p(x)$ es también un elemento de $\phi(L)$. Por inducción, se tiene que $x^ip(x)$ pertenece a $\phi(L)$, para todo $i \geq 0$. Por lo tanto, para todo $q(x) = c'_0 + c'_1x + \cdots + c'_{n-1}x^{n-1} \in R_n$, el polinomio

$$q(x)p(x) = \sum_{i=0}^{n-1} c'_i(x^ip(x)),$$

es un elemento de $\phi(L)$. Por lo tanto, $\phi(L)$ es un ideal de R_n.

Recíprocamente, si $\phi(L)$ es un ideal de R_n, entonces el polinomio nulo pertenece a $\phi(L)$ y por tanto el vector nulo a L. Además, para cualquier $\alpha, \beta \in \mathbb{F}_q \subset R_n$ y $a, b \in L$, por la Definición 1.27, se tiene que $\alpha\phi(a) + \beta\phi(b) \in \phi(L)$ y dado que ϕ es un isomorfismo $\phi(\alpha a + \beta b) \in \phi(L)$, de donde, $\alpha a + \beta b$ es una palabra del código L. Luego, L es un código lineal.

Además, si $\mathbf{c} = c_0c_1\cdots c_{n-2}c_{n-1} \in L$, el polinomio

$$\phi(\mathbf{c}) = c_0x + c_1x^2 + \cdots + c_{n-2}x^{n-2} + c_{n-1}x^{n-1},$$

es un elemento de $\phi(L)$. Como $\phi(L)$ es un ideal de R_n, el elemento

$$\begin{aligned} x(\phi(L)) &= c_0x + c_1x^2 + \cdots + c_{n-2}x^{n-1} + c_{n-1}x^n \\ &= c_{n-1} + c_0x + c_1x^2 + \cdots + c_{n-2}x^{n-1}, \end{aligned}$$

pertenece a $\phi(L)$, es decir que $c_{n-1}c_0c_1\cdots c_{n-2}$ es una palabra del código L. Luego, L es un código cíclico. ■

Dado que un código cíclico puede ser visto como un subconjunto de $\mathbb{F}_q^n$ y como un ideal de R_n, a partir de este momento se utilizará indistintamente las dos formas de representación según sea conveniente.

5.1.1. Polinomio generador y polinomio de control

Como ya se ha mencionado, una de las ventajas que presentan los códigos cíclicos, es que pueden ser determinados completamente mediante un polinomio de grado menor que n, donde n es la longitud del código. A continuación se presentan las nociones de polinomio generador y polinomio de control, las cuales se relacionan entre sí. Se presenta también algunas propiedades que muestran las ventajas que ofrecen los códigos cíclicos.

Teorema 5.2. *Existe un único polinomio mónico $g(x)$ de menor grado para cualquier ideal L de R_n. Además, este polinomio genera a L, es decir $L = \langle g(x)\rangle$.*

Demostración. Sea L un ideal de R_n, supongamos que L tiene dos polinomios mónicos $g_1(x)$ y $g_2(x)$ de grado mínimo r. Entonces el polinomio no nulo $g_1(x) - g_2(x)$ pertenece a L. Como ambos polinomios son mónicos y de grado r, el $grad(g_1(x) - g_2(x))$ es menor que r, lo cual contradice el hecho que estos dos polinomios son de grado mínimo en L. Luego, hay un único polinomio $g(x)$ de grado mínimo r en L. Probemos ahora que dicho polinomio genera a L.

Como $g(x) \in L$ y L es un ideal de R_n, entonces $\langle g(x)\rangle \subseteq L$. Por otro lado, sea $c(x) \in L$, por el algoritmo de la división se tiene

$$c(x) = q(x)g(x) + r(x),$$

donde $q(x), r(x) \in R_n$ y $grad(r(x)) < grad(g(x)) = r$. Luego $r(x) = c(x) - q(x)g(x) \in L$ pero ésto contradice la minimalidad del grado de $g(x)$. En consecuencia, $r(x) = 0$ y así $c(x) \in \langle g(x)\rangle$, es decir $L \subseteq \langle g(x)\rangle$. Las dos contenencias implican que $L = \langle g(x)\rangle$. ■

Definición 5.2 (Polinomio generador)**.** El único polinomio mónico de menor grado en un ideal no cero I de R_n se denomina el *polinomio generador* de I. Para un código cíclico L sobre $\mathbb{F}_q$, el polinomio generador de $\phi(L)$ es llamado también el polinomio generador de L.

Observación 5.1. Un código cíclico L puede ser generado por otros polinomios distintos del polinomio generador. Por lo tanto, se adopta la notación $L = \langle\langle p(x)\rangle\rangle$ que significa que L es el ideal generado por $p(x)$ y que $p(x)$ es el polinomio generador de L.

Teorema 5.3. *Sea $L = \langle\langle g(x)\rangle\rangle$, con $r = grad(g(x))$, un ideal no cero de R_n, es decir un código cíclico de longitud n.*

1. *El polinomio $g(x)$ divide a $x^n - 1$.*

2. *Cualquier $c(x) \in L$ puede ser escrito de manera única como $c(x) = r(x)g(x)$ en $\mathbb{F}_q[x]$, donde $r(x) \in \mathbb{F}_q[x]$ tiene grado menor que $n - r$. Además, la dimensión de L es $n - r$. Por lo tanto, el mensaje $r(x)$ se convierte en $r(x)g(x)$.*

3. *Si $g(x) = g_0 + g_1x + \cdots + g_rx^r$. Entonces, $g_0 \neq 0$ y L tiene como matriz generadora*

$$G = \begin{pmatrix} g(x) \\ xg(x) \\ x^2g(x) \\ \vdots \\ x^{n-r-1}g(x) \end{pmatrix} = \begin{pmatrix} g_0 & g_1 & g_2 & \cdots & & g_r & 0 & 0 & \cdots & 0 \\ 0 & g_0 & g_1 & g_2 & \cdots & & g_r & 0 & \cdots & 0 \\ 0 & 0 & g_0 & g_1 & g_2 & \cdots & & g_r & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & & & \ddots & 0 \\ 0 & 0 & \cdots & 0 & g_0 & g_1 & g_2 & \cdots & & g_r \end{pmatrix},$$

donde cada fila de G es un desplazamiento cíclico de la fila anterior (obsérvese que un polinomio se está identificando con un vector).

Demostración.

1. Por el algoritmo de la división

$$x^n - 1 = p(x)g(x) + r(x),$$

con $p(x), r(x) \in \mathbb{F}_q[x]$ y $grad(r(x)) < grad(g(x)) = r$. En R_n esto implica que $r(x) = -p(x)g(x) \in L$, luego $r(x) = 0$, dado que el grado de $g(x)$ es el más pequeño. Por lo tanto, $g(x)|x^n - 1$.

2. Dado que $L = \langle\langle g(x)\rangle\rangle$, entonces

$$L = \langle g(x)\rangle = \{f(x)g(x) : f(x) \in R_n\},$$

con la reducción módulo $x^n - 1$. Probemos que es suficiente restringir $f(x)$ a los polinomios de grado menor que $n - r$. Dado que $g(x)|x^n - 1$, se obtiene que $x^n - 1 = h(x)g(x)$, para algún polinomio $h(x) \in \mathbb{F}_q[x]$ de grado $n - r$. Dividiendo $f(x)$ por $h(x)$ se tiene

$$f(x) = q(x)h(x) + r(x),$$

con $q(x), r(x) \in \mathbb{F}_q[x]$ y $grad(r(x)) < n - r$. Luego

$$f(x)g(x) = q(x)h(x)g(x) + r(x)g(x) = q(x)(x^n - 1) + r(x)g(x),$$

de donde $f(x)g(x) = r(x)g(x)$ en R_n, como se quería probar. Por lo tanto, cualquier $c(x) \in L$ puede escribirse de manera única como $c(x) = r(x)g(x)$ en R_n, donde $r(x) \in R_n$ tiene grado menor que $n - r$. Esto implica también que el conjunto $\{g(x), xg(x), \ldots, x^{n-r-1}g(x)\}$, el cual claramente es linealmente independiente genera a L, es decir $\dim(L) = n - r$.

3. Sea $g(x) = g_0 + g_1x + \cdots + g_rx^r$ el polinomio generador de L. Si $g_0 = 0$, entonces $g(x)$ se puede escribir como $g(x) = xf(x)$, donde $grad(f(x)) < r$. Luego,

$$g_1(x) = 1 \cdot f(x) \equiv x^n f(x) = x^{n-1}g(x)$$

y por lo tanto $f(x) \in L$, lo cual no es posible ya que ningún polinomio en L puede tener grado menor que r. Luego, $g_0 \neq 0$.

El hecho de que la matriz G es una matriz generadora para L se sigue de que el conjunto $\{g(x), xg(x), \ldots, x^{n-r-1}g(x)\}$ es linealmente independiente y genera a L.

■

Teorema 5.4. *Un polinomio mónico $p(x)$ en R_n es el polinomio generador para un código cíclico si y sólo si $p(x)|x^n - 1$.*

Demostración. Una de las implicaciones se tiene por el item 1 del Teorema 5.3. Para la otra implicación, si $p(x)|x^n - 1$, sea L el ideal $\langle p(x)\rangle$ y supongamos que $g(x)$ es el polinomio generador para L. Si $p(x) \neq g(x)$, dado que $p(x)$ y $g(x)$ son ambos mónicos, por el Teorema 5.2 debe cumplirse que $grad(g(x)) < grad(p(x))$. Por otro lado,

$$x^n - 1 = p(x)f(x), \tag{5.1.2}$$

para algún polinomio $f(x) \in \mathbb{F}_q[x]$. Además, dado que $g(x) \in \langle p(x)\rangle$, se tiene que

$$g(x) = a(x)p(x),$$

para algún $a(x) \in R_n$. Multiplicando ambos lados de la anterior igualdad por $f(x)$ y utilizando la igualdad (5.1.2), se obtiene

$$g(x)f(x) \equiv a(x)p(x)f(x) \equiv a(x)(x^n - 1) \equiv 0,$$

sin embargo, $grad(g(x)f(x)) < grad(p(x)f(x)) = n$. En consecuencia, $g(x)f(x) = 0$ en $\mathbb{F}_q[x]$, lo cual implica que $f(x) = 0$, dado que $\mathbb{F}_q[x]$ es un dominio entero. Sin embargo esto contradice la expresión (5.1.2). Luego $p(x) = g(x)$. ■

Corolario 5.1. *Existe una correspondencia uno a uno entre los códigos cíclicos en $\mathbb{F}_q^n$ y los divisores mónicos de $x^n - 1 \in \mathbb{F}_q[x]$.*

Demostración. Sea π la función del conjunto D_n de todos los divisores mónicos de $x^n - 1$ en el conjunto C_n de todos los códigos cíclicos en R_n definida como

$$\pi : D_n \longrightarrow C_n$$
$$\pi(g(x)) \longrightarrow \langle\langle g(x)\rangle\rangle,$$

es decir, π envía cada divisor mónico $g(x)$ de x^n-1 al código cíclico $\langle\langle g(x)\rangle\rangle$. Por los teoremas 5.2 y 5.4, se tiene que la función π es biyectiva, dado que cada divisor mónico de x^n-1 genera un único código cíclico en R_n, además todo código cíclico en R_n debe ser generado por un divisor mónico de x^n-1. ■

El resultado anterior muestra la importancia que tiene la factorización del polinomio x^n-1 sobre $\mathbb{F}_q[x]$, con el fin de determinar los polinomios generadores de los códigos cíclicos que se pueden construir en R_n, es decir permite determinar el número de códigos cíclicos en R_n. Con este propósito se presenta el siguiente resultado.

Teorema 5.5. *Sea $x^n-1 \in \mathbb{F}_q[x]$ con factorización*

$$x^n-1=\prod_{i=1}^{r} p_i^{e_i}(x),$$

donde $p_1(x), p_2(x), \ldots, p_r(x)$ son polinomios mónicos irreducibles diferentes y $e_i \geq 1$ para todo $i=1,2,\ldots,r$. Entonces, existen $\prod_{i=1}^{r}(e_i+1)$ códigos cíclicos de longitud n sobre $\mathbb{F}_q$.

Demostración. Por el Corolario 5.1, determinar el número de códigos cíclicos de longitud n sobre $\mathbb{F}_q$ requiere contar el número de divisores mónicos de x^n-1. Por lo tanto, si

$$x^n-1=\prod_{i=1}^{r} p_i^{e_i}(x),$$

con $p_1(x), p_2(x), \ldots, p_r(x)$ polinomios mónicos irreducibles diferentes y $e_i \geq 1$ para todo $i=1,2,\ldots,r$, entonces cada polinomio p_i tiene e_i+1 divisores mónicos que son $1, p_i, p_i^2, \ldots, p_i^{e_i}$, los cuales claramente también dividen a x^n-1. Por lo tanto, considerando los r polinomios en la factorización de x^n-1 con sus respectivos grados, el número de divisores mónicos será

$$\prod_{i=1}^{r}(e_i+1).$$

■

Observación 5.2. Consideremos dos casos con respecto a los números n y q del teorema anterior. Si n y q no son primos relativos, entonces n se puede escribir como $n=mp^k$, donde $(m,q)=1$ y p la característica de $\mathbb{F}_q$. Entonces,

$$x^n-1=x^{mp^k}-1=(x^m-1)^{p^k}$$

y por tanto x^n-1 tiene factores repetidos en su descomposición factorial.

Por otro lado, si n y q son primos relativos, en polinomio $x^n - 1$ no tiene raíces múltiples en ninguna extensión de $\mathbb{F}_q$, dado que su derivada $D[x^n - 1] = nx^{n-1}$, no tiene factores comunes con $x^n - 1$. Por lo tanto, en este caso $x^n - 1$ se factoriza como

$$x^n - 1 = \prod_{i=1}^{r} p_i(x),$$

donde $p_1(x), p_2(x), \ldots, p_r(x)$ son polinomios mónicos irreducibles diferentes. En consecuencia, para este caso, por el Teorema 5.5 el número de códigos cíclicos de longitud n sobre $\mathbb{F}_q$ es 2^r.

Ejemplo 5.1. Con el fin de determinar todos los códigos cíclicos de longitud 8 sobre $\mathbb{F}_3$, se factoriza el polinomio $x^8 - 1$ sobre $\mathbb{F}_3[x]$ de acuerdo a lo dicho en la Sección 1.2.

$$x^8 - 1 = (x + 2)(x + 1)(x^2 + 1)(x^2 + x + 2)(x^2 + 2x + 2).$$

Claramente los factores son irreducibles dado que ninguno tiene raíces en $\mathbb{F}_3$. Por el Teorema 5.5, existen $2^5 = 32$ códigos cíclicos de longitud 8 sobre $\mathbb{F}_3$.

Por el item 2 del Teorema 5.3 se puede concluir que existen únicamente seis $[8, 4]$-códigos cíclicos sobre $\mathbb{F}_3$, que son los generados por los polinomios

$$\begin{aligned}
g_1(x) &= (x + 2)(x + 1)(x^2 + 1),\\
g_2(x) &= (x + 2)(x + 1)(x^2 + x + 2),\\
g_3(x) &= (x + 2)(x + 1)(x^2 + 2x + 2),\\
g_4(x) &= (x^2 + 1)(x^2 + x + 2),\\
g_5(x) &= (x^2 + 1)(x^2 + 2x + 2),\\
g_6(x) &= (x^2 + x + 2)(x^2 + 2x + 2).
\end{aligned}$$

Por último, un $[8, 2]$-código cíclico sobre $\mathbb{F}_3$ es $\langle\langle(x^2 + 1)(x^2 + x + 2)(x^2 + 2x + 2)\rangle\rangle$. Es decir

$$\begin{aligned}
\left\langle (x^2 + 1)(x^2 + x + 2)(x^2 + 2x + 2) \right\rangle = \{&00000000, 10101010, 01010101, 20202020, 02020202,\\
&21212121, 12121212, 11111111, 22222222\}.
\end{aligned}$$

Definición 5.3 (Polinomio de control)**.** Sea $g(x)$ el polinomio generador de un $[n, n - r]$-código cíclico. Dado que $g(x)$ divide a $x^n - 1$ en R_n, se tiene que

$$x^n - 1 = g(x)h(x),$$

donde $h(x) \in \mathbb{F}_q[x]$ es un polinomio de grado $n - r$ denominado el *polinomio de control* de L.

Antes de mencionar algunas propiedades del polinomio de control de un código cíclico, se presenta la siguiente definición y un lema.

Definición 5.4 (Polinomio recíproco)**.** Sea $h(x) = \sum_{i=0}^{k} a_i x^i$ un polinomio de grado k sobre $\mathbb{F}_q$. Se define el *polinomio recíproco* $h_R(x)$ por

$$h_R(x) = x^k h(1/x) = \sum_{i=0}^{k} a_{k-i} x^i.$$

Lema 5.1. *Si L es un código cíclico, entonces el código dual $L^\perp$ es también un código cíclico.*

Demostración. Sean L un código cíclico y $\mathbf{x} = x_0 x_1 \cdots x_{n-1} \in L^\perp$. Para cualquier $\mathbf{c} = c_0 c_1 \cdots c_{n-1} \in L$ se tiene que $\mathbf{c'} = c_1 c_2 \cdots c_{n-1} c_0$ también está en L y en consecuencia

$$\begin{aligned} 0 = \mathbf{c'} \cdot \mathbf{x} &= c_1 x_0 + c_2 x_1 + \cdots + c_{n-1} x_{n-2} + c_0 x_{n-1} \\ &= c_0 x_{n-1} + c_1 x_0 + c_2 x_1 + \cdots + c_{n-1} x_{n-2}, \end{aligned}$$

Luego, si $\mathbf{x'} = x_{n-1} x_0 x_1 \cdots x_{n-2}$, entonces $\mathbf{c} \cdot \mathbf{x'} = 0$. De esta manera $\mathbf{x'} \in L^\perp$ y por lo tanto $L^\perp$ es también un código cíclico. ■

Teorema 5.6. *Sea $h(x) = h_0 + h_1 x + \cdots + h_{n-r} x^{n-r}$ el polinomio de control para un código cíclico L en R_n.*

1. *El código L puede ser descrito por*

$$L = \{p(x) \in R_n : p(x)h(x) \equiv 0\}.$$

2. *$h_0^{-1} h_R(x)$ es el polinomio generador de $L^\perp$, donde h_0 es el término constante de $h(x)$. Además, una matriz de control de paridad para L es*

$$H = \begin{pmatrix} h_R(x) \\ x h_R(x) \\ x^2 h_R(x) \\ \vdots \\ x^{r-1} h_R(x) \end{pmatrix} = \begin{pmatrix} h_{n-r} & \cdots & & h_0 & 0 & 0 & \cdots & 0 \\ 0 & h_{n-r} & \cdots & & h_0 & 0 & \cdots & 0 \\ 0 & 0 & h_{n-r} & \cdots & & h_0 & & \vdots \\ \vdots & \vdots & \ddots & \ddots & \cdots & & \ddots & 0 \\ 0 & 0 & \cdots & 0 & h_{n-r} & \cdots & & h_0 \end{pmatrix}.$$

Obsérvese que la dimensión del código $L^\perp$ es r, dado que $grad(h_R(x)) = n - r$.

Demostración.

1. Sea $g(x)$ el polinomio generador de L. Si $p(x) \in L$, entonces, $p(x) = f(x)g(x)$ para algún polinomio $f(x) \in R_n$. Por lo tanto,

$$p(x)h(x) = f(x)g(x)h(x) = f(x)(x^n - 1) \equiv 0.$$

Por otro lado, si $p(x) \in R_n$ y $p(x)h(x) \equiv 0$, entonces, por el algoritmo de la división

$$p(x) = q(x)g(x) + r(x),$$

donde $grad(r(x)) < r$. Multiplicando por $h(x)$ se obtiene

$$\begin{aligned} p(x)h(x) &= q(x)g(x)h(x) + r(x)h(x) \\ 0 &= q(x)(x^n - 1) + r(x)h(x), \end{aligned}$$

de donde, $r(x)h(x) \equiv 0$. Sin embargo, $grad(r(x)h(x)) < r + (n - r) = n$. Luego, $r(x)h(x) = 0$ y en consecuencia $r(x) = 0$. Por lo tanto $p(x) = q(x)g(x) \in L$.

2. Sean $g(x) = \sum_{i=0}^{n-1} g_i x^i$ el polinomio generador y $h(x) = \sum_{i=0}^{n-1} h_i x^i$ el polinomio de control para un código cíclico sobre R_n, con $grad(h(x)) = n - r$. Entonces, considerando el producto

$$\begin{aligned} 0 \equiv & g(x)h(x) \\ \equiv & (g_0h_0 + g_1h_{n-1} + \cdots + g_{n-1}h_1) + (g_0h_1 + g_1h_0 + \cdots + g_{n-1}h_2)x \\ & + (g_0h_2 + g_1h_1 + \cdots + g_{n-1}h_3)x^2 + \cdots + (g_0h_{n-1} + g_1h_{n-2} \\ & + \cdots + g_{n-1}h_0) \text{ (mód } x^n - 1). \end{aligned}$$

Luego, los coeficientes de cada potencia de x en la última linea de la anterior expresión deben ser cero. Observando los coeficientes de cada potencia de x, se obtiene $\mathbf{g}_i \cdot h_{n-1}h_{n-2}\cdots h_1h_0 = 0$, para todo $i = 0, 1, \ldots, n-1$, donde $\mathbf{g}_i$ es la secuencia obtenida de $g_0g_1\cdots g_{n-1}$ mediante un desplazamiento cíclico de i posiciones. Por lo tanto, $h_{n-1}h_{n-2}\cdots h_1h_0$ es una palabra del código $L^\perp$, dado que por el Teorema 5.3, el conjunto $\{\mathbf{g}_0, \mathbf{g}_1, \ldots, \mathbf{g}_{n-1}\}$ genera a L. Mediante un desplazamiento cíclico de $n - r + 1$ posiciones de la secuencia $h_{n-1}h_{n-2}\cdots h_1h_0$ se obtiene la secuencia correspondiente a $h_R(x)$, es decir $h_{n-r}h_{n-r-1}\cdots h_1h_0h_{n-1}\cdots h_{n-r+1}$. Esto implica que $h_R(x)$ es una palabra código en $L^\perp$, dado que por el Lema 5.1 $L^\perp$ es también un código cíclico. Como $grad(h_R(x)) = grad(h(x)) = n-r$, el conjunto $\{h_R(x), xh_R(x), \ldots, x^{r-1}h_R(x)\}$ es una base para $L^\perp$. Por lo tanto, el polinomio mónico $h_0^{-1}h_R(x)$ es el polinomio generador del código $L^\perp$.

Además, dado que $h_0^{-1}h_R(x)$ es el polinomio generador de $L^\perp$, por el Teorema 5.3 se sigue que la matriz H es una matriz generadora para $L^\perp$, es decir una matriz de control de paridad para L.

■

5.1.2. Codificación con códigos cíclicos

Existen dos maneras bastante sencillas de codificar mensajes utilizando códigos cíclicos, una sistemática y otra *no sistemática*. Sea $L = \langle\langle g(x)\rangle\rangle$ un $[n, n-r]$-código cíclico, con $grad(g(x)) = r$.

Por el item 2 del Teorema 5.3, L puede codificar mensajes q-arios de longitud $n-r$ y requiere r símbolos de redundancia.

La codificación no sistemática es la misma que se utiliza para los códigos lineales en general, lo cual se presentó en la Observación 3.3. Es decir, un mensaje $m \in \mathbb{F}_q^{n-r}$ se codifica como la palabra código $c = mG \in \mathbb{F}_q^n$, donde G es la matriz generadora para L, la cual está dada por el Teorema 5.3.

Para obtener una codificación sistemática, dado un mensaje fuente $a = a_0 a_1 \cdots a_{n-r-1} \in \mathbb{F}_q^{n-r}$, se forma el polinomio mensaje

$$\bar{a}(x) = a_0 x^{n-1} + a_1 x^{n-2} + \cdots + a_{n-r-1} x^r.$$

Obsérvese que $\bar{a}(x)$ no tiene términos de grado menor que r. Ahora, se divide $\bar{a}(x)$ por $g(x)$,

$$\bar{a}(x) = q(x)g(x) + s(x), \quad \text{con} \quad grad(s(x)) < r$$

y se codifica a $\bar{a}(x)$ como $c(x) = \bar{a}(x) - s(x) = q(x)g(x)$. Dado que $\bar{a}(x)$ y $s(x)$ no tienen términos del mismo grado, la codificación es sistemática. En efecto, si se lee los términos de un polinomio del código, de grado mayor a menor, se observa que las primeras $n-r$ coordenadas son los símbolos de información y que las r restantes son símbolos de redundancia.

Ejemplo 5.2. Sea L el $[8,2]$-código cíclico del Ejemplo 5.1, el cual es generado por $g(x) = (x^2+1)(x^2+x+2)(x^2+2x+2) = x^6+x^4+x^2+1$. Este código es capaz de codificar nueve mensajes de longitud 2 sobre $\mathbb{F}_3$, es decir que codifica el conjunto de información $\mathbb{F}_3^2 = \{00, 10, 01, 20, 02, 12, 21, 11, 22\}$ añadiendo seis dígitos de redundancia para obtener un código de longitud 8.

En este caso, para realizar una codificación sistemática, consideremos los polinomios mensaje $\bar{a}_1(x)$, $\bar{a}_2(x)$, $\bar{a}_3(x)$, $\bar{a}_4(x)$, $\bar{a}_5(x)$, $\bar{a}_6(x)$, $\bar{a}_7(x)$, $\bar{a}_8(x)$ y $\bar{a}_9(x)$, mostrados a continuación y dividimos cada uno entre $g(x)$.

Por el algoritmo de la división se obtiene

$$\begin{aligned}
\bar{a}_1(x) &= 0 = (x^6+x^4+x^2+1)(0) + 0,\\
\bar{a}_2(x) &= x^7 = (x^6+x^4+x^2+1)(x) + (2x^5+2x^3+2x),\\
\bar{a}_3(x) &= x^6 = (x^6+x^4+x^2+1)(1) + (2x^4+2x^2+2),\\
\bar{a}_4(x) &= 2x^7 = (x^6+x^4+x^2+1)(2x) + (x^5+x^3+x),\\
\bar{a}_5(x) &= 2x^6 = (x^6+x^4+x^2+1)(2) + (x^4+x^2+1),\\
\bar{a}_6(x) &= x^7+2x^6 = (x^6+x^4+x^2+1)(x+2) + (2x^5+x^4+2x^3+x^2+2x+1),\\
\bar{a}_7(x) &= 2x^7+x^6 = (x^6+x^4+x^2+1)(2x+1) + (x^5+2x^4+x^3+2x^2+x+2),\\
\bar{a}_8(x) &= x^7+x^6 = (x^6+x^4+x^2+1)(x+1) + (2x^5+2x^4+2x^3+2x^2+2x+2),\\
\bar{a}_9(x) &= 2x^7+2x^6 = (x^6+x^4+x^2+1)(2x+2) + (x^5+x^4+x^3+x^2+x+1).
\end{aligned}$$

Entonces $\bar{a}_i(x)$ se codifica como $c_i(x) = \bar{a}_i(x) - r_i(x)$, donde $r_i(x)$ es el residuo de dividir $\bar{a}_i(x)$ entre $g(x)$. Por lo tanto se tiene la codificación

$$\begin{aligned}
a_1(x) &\longrightarrow c_1(x) = 0\\
a_2(x) &\longrightarrow c_2(x) = x^7 + x^5 + x^3 + x\\
a_3(x) &\longrightarrow c_3(x) = x^6 + x^4 + x^2 + 1\\
a_4(x) &\longrightarrow c_4(x) = 2x^7 + 2x^5 + 2x^3 + 2x\\
a_5(x) &\longrightarrow c_5(x) = 2x^6 + 2x^4 + 2x^2 + 2\\
a_6(x) &\longrightarrow c_6(x) = x^7 + 2x^6 + x^5 + 2x^4 + x^3 + 2x^2 + x + 2\\
a_7(x) &\longrightarrow c_7(x) = 2x^7 + x^6 + 2x^5 + x^4 + 2x^3 + x^2 + 2x + 1\\
a_8(x) &\longrightarrow c_8(x) = x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x + 1\\
a_9(x) &\longrightarrow c_9(x) = 2x^7 + 2x^6 + 2x^5 + 2x^4 + 2x^3 + 2x^2 + 2x + 2,
\end{aligned}$$

es decir

$$\begin{aligned}
00 &\longrightarrow 00000000\\
10 &\longrightarrow 10101010\\
01 &\longrightarrow 01010101\\
20 &\longrightarrow 20202020\\
02 &\longrightarrow 02020202\\
12 &\longrightarrow 12121212\\
21 &\longrightarrow 21212121\\
11 &\longrightarrow 11111111\\
22 &\longrightarrow 22222222.
\end{aligned}$$

Se puede observar que la codificación es sistemática, dado que los dos primeros dígitos en cada palabra código corresponden a la información que se desea transmitir, es decir al mensaje, mientras que los últimos seis son de redundancia, con el fin de detectar y corregir errores.

5.1.3. Decodificación con códigos cíclicos

Dado que todo código cíclico es lineal, se puede utilizar la decodificación por síndrome vista en la Sección 3.3, pero en su forma polinómica. Si $c(x) \in L$ es una palabra código enviada (en forma polinómica) y $u(x)$ es el polinomio recibido, entonces, $e(x) = u(x) - c(x)$ se denomina *polinomio error*. El peso de un polinomio es el número de coeficientes no nulos.

Definición 5.5. (Síndrome de un polinomio) Sea $L = \langle\langle g(x)\rangle\rangle$ un $[n, n-r]$-código cíclico. El *síndrome de un polinomio* $u(x)$, denotado por $s(u(x))$, es el residuo de dividir $u(x)$ por $g(x)$, es

decir

$$u(x) = q(x)g(x) + s(u(x)) \quad \text{con} \quad grad(s(x)) < r.$$

Observación 5.3. La definición de síndrome para un polinomio coincide con la dada para códigos lineales. En efecto, sea $s_i(x)$ el residuo de dividir x^{r+i} entre $g(x)$, es decir

$$x^{r+i} = a_i(x)g(x) + s_i(x), \tag{5.1.3}$$

con $grad(s_i(x) < r)$. Probemos que el conjunto $B = \{x^{r+i} - s_i(x) : i = 0, 1, \ldots, n-r-1\}$ es una base para L.

Sea $f(x) \in \langle B \rangle$, entonces

$$f(x) = \sum_{i=0}^{n-r-1} b_i(x^{r+i} - s_i(x)) = \sum_{i=0}^{n-r-1} b_i a_i(x)g(x),$$

donde $b_i \in \mathbb{F}_q$ para todo $i = 0, 1, \ldots, n-r-1$. Por tanto $f(x) \in L$ y así $\langle B \rangle \subseteq L$.

Probemos ahora que el conjunto B es linealmente independiente. Sean $\alpha_0, \alpha_1, \ldots, \alpha_{n-r} \in \mathbb{F}_q$ tales que

$$\alpha_0(x^r - s_0(x)) + \cdots + \alpha_{n-r-1}(x^{n-1} - s_{n-r-1}(x)) = 0,$$

entonces $\alpha_0 x^r + \cdots + \alpha_{n-r-1}x^{n-1} - t(x) = 0$, donde $t(x) = -\alpha_0 s_0(x) - \cdots - \alpha_{n-r-1}s_{n-r-1}(x)$ es un polinomio de grado menor que r. Luego $\alpha_i = 0$ para todo $i = 0, 1, \ldots, n-r-1$, es decir que el conjunto B es linealmente independiente como se quería probar.

Como $\dim(L) = n - r = \dim(\langle B \rangle)$ y $\langle B \rangle \subseteq L$, entonces $L = \langle B \rangle$ y B es una base para L.

Por lo dicho anteriormente, una matriz generadora para el código L es

$$G = \begin{pmatrix} x^r - s_0(x) \\ x^{r+1} - s_1(x) \\ \vdots \\ x^{n-1} - s_{n-r-1}(x) \end{pmatrix} = \begin{pmatrix} -s_{00} & -s_{01} & \cdots & -s_{0r-1} & 1 & 0 & \cdots & 0 \\ -s_{10} & -s_{11} & \cdots & -s_{1r-1} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -s_{n-r-10} & -s_{n-r-11} & \cdots & -s_{n-r-1r-1} & 0 & 0 & \cdots & 1 \end{pmatrix}.$$

Considerando la matriz anterior como $G = (S \mid I_{n-r})$, mediante un razonamiento similar al de la Observación 3.4, se tiene que una matriz de control para L es $H = (I_r \mid -S^\top)$, donde $S^\top$ es la transpuesta de S, es decir

$$H = \begin{pmatrix} 1 & 0 & \cdots & 0 & s_{00} & s_{10} & \cdots & s_{n-r-10} \\ 0 & 1 & \cdots & 0 & s_{01} & s_{11} & \cdots & s_{n-r-11} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & s_{0r-1} & s_{1r-1} & \cdots & s_{n-r-1r-1} \end{pmatrix}.$$

Luego para cada $\mathbf{u} = u_0 u_1 \cdots u_{n-1} \in \mathbb{F}_q^n$, la j-ésima coordenada de su síndrome es

$$uH_j^\top = u_j + \sum_{i=0}^{n-r-1} u_{r+i}s_{ij}, \tag{5.1.4}$$

donde $H_j^\top$ denota la j-ésima columna de $H^\top$.

En términos de polinomios u puede verse como

$$u(x) = \sum_{j=0}^{r-1} u_j x^j + \sum_{i=0}^{n-r-1} u_{r+i}x^{r+i}.$$

Sin embargo, de la expresión (5.1.3) se obtiene que

$$\begin{aligned} u(x) &= \sum_{j=0}^{r-1} u_j x^j + \sum_{i=0}^{n-r-1} u_{r+i}\left(a_i(x)g(x) + s_i(x)\right) \\ &= g(x)\sum_{i=0}^{n-r-1} u_{r+i}a_i(x) + \left(\sum_{j=0}^{r-1} u_j x^j + \sum_{i=0}^{n-r-1} u_{r+i}s_i(x)\right) \\ &= g(x)q(x) + s(x), \end{aligned}$$

donde $q(x) = \sum_{i=0}^{n-r-1} u_{r+i}a_i(x)$ y el grado del polinomio $s(x) = \sum_{j=0}^{r-1} u_j x^j + \sum_{i=0}^{n-r-1} u_{r+i}s_i(x)$ es menor que r. Por lo tanto $s(x)$ es el síndrome de $u(x)$ y el coeficiente de x^j es

$$u_j + \sum i = 0^{n-r-1}u_{r+i}s_{ij}, \tag{5.1.5}$$

donde s_{ij} denota el coeficiente de x^j en $s_i(x)$.

De la expresiones (5.1.4) y (5.1.5) se concluye que las dos definiciones de síndrome son equivalentes.

Ahora se presenta el proceso de decodificación para un código cíclico. Un polinomio recibido $u(x)$ es una palabra código si y sólo si su síndrome es el polinomio nulo. Además, dos polinomios tienen el mismo síndrome si y sólo si están en la misma clase lateral de L. Luego, la forma polinómica de decodificación por síndrome es análoga a la vectorial, veamos esto con el siguiente ejemplo.

Ejemplo 5.3. Sea

$$L = \{00000000, 10101010, 01010101, 20202020, 02020202, 21212121, 12121212, 11111111, 22222222\},$$

el código cíclico ternario del Ejemplo 5.1. Como L es un código lineal entonces $d(L) = wt(L) = 4$ y en consecuencia L solo es capaz de corregir un error, es decir corrige todos los errores para los cuales el polinomio error tenga peso igual a uno.
Los líderes de las clases laterales con sus correspondientes síndromes se muestran en la Tabla 5.1.

líder	síndrome	líder	síndrome
0	0	2	2
1	1	$2x$	$2x$
x	x	$2x^2$	$2x^2$
x^2	x^2	$2x^3$	$2x^3$
x^3	x^3	$2x^4$	$2x^4$
x^4	x^4	$2x^5$	$2x^5$
x^5	x^5	$2x^6$	x^4+x^2+1
x^6	$2x^4+2x^2+2$	$2x^7$	x^5+x^3+x
x^7	$2x^5+2x^3+2x$		

Tabla 5.1: Tabla líder-síndrome.

Supongamos que se envía la palabra código 10101010 y que durante la transmisión ocurre un error en la sexta coordenada de manera que la palabra recibida es la secuencia $u = 10101110$. En términos de polinomios se recibe $u(x) = x^6 + x^5 + x^4 + x^2 + 1$ y como

$$u(x) = x^6 + x^5 + x^4 + x^2 + 1 = (x^6 + x^4 + x^2 + 1)(1) + (x^5),$$

entonces, $s(u(x)) = x^5$ y de la Tabla 5.1 se observa que el líder de esta clase lateral es $a(x) = x^5$. Por lo tanto $u(x)$ se decodifica como

$$c(x) = u(x) - a(x) = (x^6 + x^5 + x^4 + x^2 + 1) - (x^5) = x^6 + x^4 + x^2 + 1,$$

es decir que $u = 10101110$ se decodifica como $c = 10101010$, corrigiendo efectivamente el error ocurrido.

La principal dificultad de esta forma de decodificación es que las tablas líder-síndrome son bastante grandes, como se observa en el Ejemplo 5.3. Sin embargo, esta dificultad se puede superar usando el hecho de que los códigos con los que se trabaja son cíclicos. Supongamos que tenemos algún método para decodificar el coeficiente principal de cualquier palabra recibida $u(x)$, es decir el coeficiente de x^{n-1}, el cual puede ser o no distinto de cero. Luego, podemos decodificar el coeficiente principal de $u(x)$, realizar un desplazamiento cíclico módulo $x^n - 1$, y decodificar el nuevo coeficiente principal, que es el coeficiente de x^{n-2} en $u(x)$. Repitiendo este proceso un número finito de pasos se decodifica toda la palabra. Este método permite ahorrar tiempo en los cálculos ya que solo necesita las filas de la tabla líder-síndrome que contienen líderes de grado $n - 1$.

Un método para decodificar el coeficiente de x^{n-1}, empleando una tabla construida únicamente con los líderes de la clase lateral de grado $n - 1$, consistiría en calcular el síndrome de la palabra recibida $u(x)$; si éste no aparece en la tabla se concluye que el coeficiente de x^{n-1} es correcto. De lo contrario, si el síndrome de $u(x)$ aparece en la tabla se deduce que existe un error en el coeficiente de x^{n-1}, por lo tanto se cambia dicho coeficiente por uno de los elementos restantes en $\mathbb{F}_q$ y se vuelve a calcular el síndrome de la palabra resultante. El anterior proceso se repite hasta que el síndrome

de la palabra obtenida por el cambio del coeficiente de x^{n-1} en la palabra recibida no aparezca en la tabla, ya que este nuevo coeficiente será el correcto.

Observación 5.4. Se concluye que el coeficiente de x^{n-1} es correcto en una palabra recibida $u(x)$, cuando su síndrome no aparece en la tabla con los líderes de la clase lateral de grado $n-1$, porque el líder de la clase lateral corresponde al error y si el síndrome de $u(x)$ no aparece en la tabla significa que el/los error/es están en los coeficientes de los términos de grado menor que $n-1$.

Ejemplo 5.4. Para el caso del Ejemplo 5.3, los únicos lideres de las clases laterales con peso 1 y grado $n-1=7$ son x^7 y $2x^7$, por lo tanto sólo se necesitan dos filas como se muestra en la Tabla 5.2.

líder	síndrome
x^7	$2x^5+2x^3+2x$
$2x^7$	x^5+x^3+x

Tabla 5.2: Tabla líder-síndrome.

Nuevamente, supongamos que se recibe $u(x)=x^6+x^5+x^4+x^2+1$. Dado que $s(u(x))=x^5$ no está en la Tabla 5.2, se asume que el coeficiente líder de $u(x)$, es decir el de x^7, es correcto. En seguida se realiza un desplazamiento cíclico en $u(x)$

$$x(x^6+x^5+x^4+x^2+1)\,\text{mód}(x^8-1)=x^7+x^6+x^5+x^3+x,$$

y se calcula su síndrome, el cual es x^6. Igualmente, este término no está en la Tabla 5.2, por tanto el coeficiente de x^5 en $u(x)$ es correcto. Nuevamente, se realiza otro desplazamiento cíclico

$$x(x^7+x^6+x^5+x^3+x)\,\text{mód}(x^8-1)=x^7+x^6+x^4+x^2+1 \tag{5.1.6}$$

y se calcula su síndrome, es decir $2x^5+2x^3+2x$ el cual está en la Tabla 5.2. Por lo tanto, el coeficiente de x^5 en $u(x)$ es incorrecto. Se tienen dos opciones, cambiar el coeficiente de x^7 en la parte derecha de la expresión (5.1.6) por 0 ó por 2. Si se cambia por 2, se obtiene la palabra $2x^7+x^6+x^4+x^2+1$, cuyo síndrome es x^5+x^3+x y también aparece en la Tabla 5.2, por lo tanto el coeficiente adecuado debe ser 0, el cual debe ser reemplazado en la palabra recibida por el coeficiente de x^5.

Continuando de esta manera, se decodifica $u(x)$ como $c(x)=x^6+x^4+x^2+1$. Lo anterior coincide con la decodificación del Ejemplo 5.3.

Por último, cabe mencionar que algunas funciones del programa GAP referentes a la construcción, codificación y decodificación con códigos cíclicos se describen detalladamente en el apéndice 5.2 y otras se pueden encontrar en [19].

5.2. Códigos a partir de Conjuntos B_h

Los conjuntos B_h en grupos abelianos finitos son una noción propia de la Teoría de Números Aditiva y ha sido utilizada para abordar los problemas de la Teoría de Códigos. A continuación se presentan la definición de conjunto B_h, dos teoremas que garantizan la existencia de conjuntos B_h con cierto cardinal y por último un resultado que muestra la relación existente entre los conjuntos B_h y la Teoría de Códigos.

Definición 5.6. Sea $\langle G, + \rangle$ un grupo abeliano. Para un entero $h \geq 2$, el conjunto $A = \{g_1, g_2, ..., g_k\} \subseteq G$, es un conjunto B_h, si todas las sumas de h elementos no necesariamente distintos de A, producen elementos distintos en G. En este caso, se dice que A está en la *clase de conjuntos B_h sobre G* y se denota por $A \in B_h(G)$.

Veamos sin prueba, dos teoremas que garantizan la existencia de conjuntos B_h logradas por Bose-Chowla [1] en el año 1962 y la generalización de la misma alcanzada por C. A Trujillo, G. García y J. M. Velásquez [3] en el año 2003.

Teorema 5.7 (Construcción de Bose-Chowla, 1962)**.** *Para toda potencia prima q y todo $h \geq 2$ entero, existe un conjunto $B_h(\text{mód } q^h - 1)$ con q elementos.*

Teorema 5.8 (Trujillo, García y Velásquez,2003)**.** *Sean q potencia prima y $h \geq 2$ entero. Para todo $d|h$ con $d > 1$ entero, existe un conjunto $B_d(\text{mód } q^h - 1)$ con q elementos.*

Los anteriores teoremas, además de garantizar la existencia de conjuntos B_h con cierto cardinal, sus pruebas brindan una forma de construir dichos conjuntos. En [7] y [5] se pueden encontrar las pruebas de estos teoremas y otras construcciones de conjuntos B_h.

Después de garantizar la existencia de conjuntos B_h con cierto cardinal, analizaremos la relación existente entre estos conjuntos y los códigos binarios de peso constante cuyo origen se remonta al año de 1980 con el artículo "*Lower bounds for constant weight codes*"[4], publicado por R. L. Graham y N. J. A. Sloane.

La idea fundamental de esta relación, consiste en considerar la función

$$T : \mathbb{F}_w^n \longrightarrow \mathbb{Z}_m,$$

donde $\mathbb{F}_w^n$ denota el conjunto de los $\binom{n}{w}$ vectores binarios de longitud n y peso constante w. De manera que si existe $A = \{t_1, t_2, \ldots, t_n\}$ un conjunto B_{h-1} en $\mathbb{Z}_m$, entonces para $a = (a_1, a_2, \ldots, a_n) \in \mathbb{F}_w^n$ la función T se define como:

$$T(a) = \sum_{i=1}^{n} a_i t_i. \tag{5.2.1}$$

Antes de continuar con esta idea se presenta el siguiente lema.

Lema 5.2. *Un conjunto B_t es inmediatamente un conjunto B_u para todo $u < t$, con $u, t \in \mathbb{Z}^+$.*

Demostración. Sean A un conjunto B_t y $u < t$ enteros positivos, supongamos que existen dos sumas iguales de u elementos en A, es decir

$$s_{i_1} + s_{i_2} + \cdots + s_{i_u} = s_{j_1} + s_{j_2} + \cdots + s_{j_u}. \tag{5.2.2}$$

Luego, sumando $t - u$ elementos de A en ambos lados de la expresión (5.2.2) se tiene

$$s_{i_1} + s_{i_2} + \cdots + s_{i_u} + s_{k_1} + \cdots + s_{k_{t-u}} = s_{j_1} + s_{j_2} + \cdots + s_{j_u} + s_{k_1} + \cdots + s_{k_{t-u}},$$

lo cual es una contradicción, dado que A es un conjunto B_t. Por lo tanto, A es también un conjunto B_u. ■

Teorema 5.9. *Sea T la función definida en la expresión (5.2.1) y supongamos que existe un conjunto B_{h-1} en $\mathbb{Z}_m$, con $h \geq 3$. Entonces, $C_k = T^{-1}(k)$ para $k \in \mathbb{Z}_m$, es un código binario de longitud n, distancia mínima al menos $2h$ y peso constante w. Es decir, si el tamaño de C_k es M, éste es un (n, M, d)-código binario con $d \geq 2h$ y cada palabra de C_k tiene w unos.*

Demostración. Sea $B = \{s_1, s_2, \ldots, s_n\}$ un conjunto B_{h-1} de orden n sobre $\mathbb{Z}_m$. Claramente para $0 \leq i \leq m-1$, $C_i = T^{-1}(i)$ es un código binario de longitud n y peso constante w (para $C_i \neq \varnothing$). Probemos que $d = d(C_i) \geq 2h$.

La distancia mínima de C_i no puede ser impar, por ser un código de peso constante, como se vio en la Observación 4.1.

Supongamos que existen $a, b \in C_i$ tales que $d(a, b) = 2(h-k)$, con $k = 1, 2, 3, \ldots, h-2$. Entonces, dado que C_i es un código de peso constante debe cumplirse que a y b tienen $h - k$ unos y $h - k$ ceros sobre los cuales difieren, por lo tanto, de la expresión (5.2.1) se tiene

$$T(a) = x + s_1 + \cdots + s_{h-k} = x + s'_1 + \cdots + s'_{h-k} = T(b) \,(\text{mód } m).$$

Luego, $s_1 + \cdots + s_{h-k} = s'_1 + \cdots + s'_{h-k}$ (mód m), lo cual es una contradicción, dado que B es un conjunto B_{h-1} y por el Lema 5.2, también un conjunto B_{h-k}. Por lo tanto $d(C_i) \geq 2h$. ■

Del anterior teorema y teniendo en cuenta el Teorema 5.8, cuya demostración se encuentra en [5, pag. 6, Teorema 2.2] y muestra la forma de construir conjuntos B_h de cardinal igual a la potencia de un número primo, se puede construir códigos binarios con ciertos parámetros, como se muestra en el siguiente ejemplo.

Ejemplo 5.5. Tomando el conjunto $A \in B_2$(mód 168),

$$A = \{1, 17, 21, 51, 58, 64, 66, 75, 76, 97, 102, 137, 166\}$$

de orden 13, cuya existencia se garantiza por el Teorema 5.8 y cuya construcción se logró también a partir de la demostración de este teorema y de algunos cálculos sencillos. Considerando la función $T : \mathbb{F}_4^{13} \longrightarrow \mathbb{Z}_{168}$, definida anteriormente, se tiene que la imagen inversa para $42 \in \mathbb{Z}_{168}$ es

$$T^{-1}(42) = \left\{ \begin{array}{c} 1000100110000, \\ 0101001010000, \\ 0110000101000, \\ 0100100000011, \\ 0001010001001, \\ 0000010100110, \\ 1011000000010 \end{array} \right\},$$

note que en este caso estamos tomando un conjunto $B_2 = B_{3-1}$, es decir que $h = 3$ y por tanto el anterior conjunto es un código binario de longitud 13, distancia mínima al menos $d = 2h = 6$ y peso constante 4 . Es decir, $T^{-1}(42)$ es un $(13, 7, d)$-código binario de peso constante igual a 4, con $d \geq 6$.

Cabe mencionar que se eligió $42 \in \mathbb{Z}_{168}$, dado que éste fue el elemento cuya imagen inversa tiene mayor número de elementos, es decir, el código C_{42} es el que tiene mayor cardinal de todos los C_i, para $i = 0, 1, \ldots, 167$ y uno de los objetivo en la Teoría de Códigos es construir códigos con el mayor tamaño posible con el fin de transmitir la mayor información.

Como se puede observar, los códigos obtenidos mediante esta construcción no son lineales. Sin embargo, existe también una relación entre los Conjuntos B_h y los Códigos Lineales, como se muestra en el artículo "*Sobre Conjuntos S_h de vectores binarios y códigos lineales*"[6], con el siguiente resultado.

Teorema 5.10. *Existe un $[n, k, d]$-código con $d \geq 2h + 1$ si y sólo si existe un conjunto S_h con $n + 1$ elementos en $\mathbb{F}_2^{n-k}$, donde $n - k > 2h$.*

Donde un conjunto S_h es un conjunto B_h en el cual las sumas de h elementos deben ser distintas para que determinen elementos distintos.

Además, la construcción de códigos a partir de la noción de conjunto B_h ha proporcionado resultados con lo que respecta al problema fundamental de la Teoría de códigos, ver Capítulo 4, es decir lo relacionado con los valores de $A_q(n, d)$, lo cual se puede encontrar en [5].

Conclusiones

- En este trabajo se describe de manera detallada en qué consiste la Teoría de Códigos y proporciona bases para animar al lector en el estudio más a fondo de las cuestiones aquí tratadas. Para esto, se organizaron de manera sistemática las nociones que se consideraron de mayor importancia y cabe resaltar que la mayoría de las pruebas presentadas han sido reelaboradas de manera más detallada, dado que en los textos estudiados se omiten pasos que restan claridad a dichas pruebas, además algunas son producción propia de los autores ante la ausencia de las mismas en los textos estudiados. Algunas demostraciones reelaboradas han sido: los teoremas 2.4, 2.7, 3.6, 3.15, los lemas 4.1, 4.3, 5.1, el corolario 4.1 y la observación 5.3. Además, la mayoría de los conceptos tratados en este trabajo, se explicaron con ejemplos para una mayor comprensión.

- El tratamiento detallado de los códigos lineales a partir de algunos conceptos básicos del Álgebra Lineal y la Teoría de Campos Finitos, permitió explicar y justificar la importancia que tienen estos códigos en lo concerniente a su estructura, construcción, capacidad de corrección, codificación y decodificación, en comparación con otros códigos que no tienen esta estructura algebraica. Así mismo, este trabajo centró su atención en la explicación detallada de algunas construcciones de códigos, especialmente lineales y de los resultados relevantes concernientes a ellos.

- La Teoría de Códigos se puede abordar desde dos puntos de vista. Uno de ellos sería abordar el problema fundamental, es decir, determinar el máximo cardinal de un código dados ciertos parámetros, ya sea lineal o no. El otro enfoque consistiría en dedicarse a la construcción de códigos. Obviamente, los dos están íntimamente ligados, dado que para la construcción de códigos se debe tener en cuenta el problema fundamental con el fin de construir códigos que puedan transmitir una gran cantidad de información. De igual manera, a partir de la construcción de códigos se pueden determinar cotas sobre el tamaño de un código, lo cual aporta en el estudio del problema fundamental.

- El paquete GUAVA del software GAP ofrece bastantes funciones que permiten realizar cálculos importantes en Teoría de Códigos, relacionadas tanto con los conceptos como con los resultados de dicha teoría, principalmente se tienen la construcción, la manipulación y cálculo de información sobre los códigos.

Apéndice

Este apéndice tiene el propósito de describir detalladamente algunas funciones del paquete GUAVA de GAP utilizadas en el desarrollo de este trabajo. GAP (Groups, Algorithms, Programming- a System for Computational Discrete Algebra) es un sistema de álgebra computacional discreto, con especial interés en la Teoría de Grupos Computacional. GAP ofrece un lenguaje de programación, una biblioteca con miles de funciones, aplicación de algoritmos algebraicos escritos en el lenguaje de GAP, así como de grandes bibliotecas de datos de objetos algebraicos. GAP se utiliza en la investigación y la enseñanza en el estudio de los grupos y sus representaciones, anillos, espacios vectoriales, estructuras combinatorias y más. Teniendo en cuenta las múltiples aplicaciones que tiene GAP, este funciona por paquetes. El paquete matemático de GAP utilizado en este trabajo es GUAVA, el cual proporciona implementaciones de algunas rutinas diseñadas para la construcción y el análisis en la Teoría de la Códigos correctores de errores. Este paquete se carga como se muestra a continuación.

```
gap> LoadPackage( "guava", "2.1" );
```

Las funciones se dividen en tres categorías.

- Construcción de códigos.
- Manipulación de códigos.
- Cálculo de información sobre códigos.

Algunas de la funciones más interesantes se listan a continuación.

- `AsSSortedList(C)`: muestra en pantalla los elementos del código C.
- `AugmentedCode(C,L)`: aumenta el código lineal C. L es una lista de entrada de palabras código.
- `BoundsMinimumDistance(n,k,F)`: calcula una cota inferior y superior sobre la distancia mínima de un código lineal óptimo de longitud n, dimensión k definido sobre el campo F.
- `CheckMat(C)`: calcula una matriz de control de paridad para el código lineal C.
- `CheckMatCode( H[,name,]F)`: calcula un código lineal con matriz de control de paridad H.
- `CheckPol(C)`: retorna el polinomio de control del código cíclico C.
- `CheckPolCode(h,n[,name,]F)`: crea un código cíclico de longitud n sobre F con polinomio de control h.

- `CodeWeightEnumerator(C)`: calcula el enumerador de peso del código C.
- `Codeword(obj[,n][,][F])`: presenta una palabra o una lista de palabras construidas de un objeto. El objeto *obj* puede ser un vector, una cadena, un polinomio o una palabra. Si el número n es especificado se construyen todas las palabras de longitud n. Si el campo de Galois F es especificado, todas las palabras sobre el campo F son construidas.
- `CosetCode(C,w)`: calcula el código de la clase lateral $w + C$.
- `CoveringRadius(C)`: muestra en pantalla el radio de cubrimiento del código lineal C.
- `CyclicCodes(n,F)`: presenta una lista de todos los códigos cíclicos de longitud n sobre el campo F.
- `(c1,c2)`: evalúa si dos palabras son iguales o no. También se puede utilizar para determinar si dos códigos son iguales o no.
- `(c1+c2)`: evalúa la suma de las palabras $c1$ y $c2$. Las palabras deben estar definidas sobre el mismo campo y tener la misma longitud. Esta función también se utiliza para calcular la suma directa de dos códigos.
- `(c1-c2)` realiza la diferencia de las palabras $c1$ y $c2$.
- `Decode(C,r)`: decodifica la palabra recibida r con respecto al código C y presenta la "palabra mensaje"(es decir los símbolos de información asociada a una palabra $c \in C$ cercana a r).
- `Dimension(C)`: presenta el parámetro k de un código lineal C, es decir su dimensión.
- `DirectSumCode(C1,C2)`: calcula la suma directa de los códigos $C1$ y $C2$.
- `DistanceCodeword( c1, c2 )`: calcula la distancia de Hamming entre $c1$ y $c2$. Ambas palabras deben pertenecer al mismo campo de Galois y tener la misma longitud.
- `DualCode(C)`: calcula el código dual del código lineal C.
- `ElementsCode(L,F)`: genera los parámetros de un código L definido sobre el campo F, así como cotas sobre la distancia mínima y sobre el radio de cubrimiento.
- `EvenWeightSubcode(C)`: presenta un subcódigo de peso par del código C.
- `ExpurgatedCode(C,L)`: expurga todas las coordenadas de la lista L de todas las palabras del código lineal C.
- `ExtendedCode(C[,i])`: esta función extiende i veces un código dado C y calcula sus parámetros, i es igual a uno por defecto.
- `GeneratorMatCode(G[,name,]F)`: retorna un código lineal con matriz generadora G, la cual debe estar definida sobre el campo F, *name* es una descripción corta del código.
- `GeneratorPol(C)`: calcula el polinomio generador del código cíclico C. Si C no es cíclico la función presenta "error".

- `GeneratorPolCode(g,n[,name,]F)`: crea un código cíclico con polinomio generador g, palabras de longitud n y definido sobre el campo F.
- `HammingCode(r,F)`: retorna un código de Hamming con r símbolos de redundancia sobre el campo F. Cuando no se especifica F, GAP asume que es $\mathbb{F}_2$.
- `InformationWord(C,c)`: calcula los símbolos de información que tiene la palabra $c \in C$, donde C es un código lineal.
- `IsCyclicCode(obj)`: comprueba si el objeto *obj* es un código cíclico.
- `IsEquivalent(C1,C2)`: comprueba si $C1$ y $C2$ son códigos equivalentes.
- `IsLinearCode(obj)`: comprueba si el objeto *obj* (no necesariamente un código) es un código lineal.
- `IsPerfectCode(C)`: calcula si C es un código perfecto.
- `IsSelfDualCode(C)`: comprueba si C es auto-dual.
- `IsSelfOrthogonalCode(C)`: comprueba si C es auto-ortogonal.
- `IsSubset(C1,C2)`: presenta "verdadero"ó "falso"si $C2$ es un subcódigo de $C1$.
- `LowerBoundSpherePacking(n,d,q)`: presenta la cota inferior dada por Gilbert y Varshamov.
- `MinimumDistance(C)`: calcula la distancia mínima del código C.
- `NrCyclicCodes(n,F)`: calcula el número de códigos cíclicos de longitud n sobre el campo F.
- `OptimalityCode(C)`: calcula la diferencia entre la cota superior más pequeña conocida y el tamaño real del código C.
- `OptimalityLinearCode`: similar a la anterior, pero para códigos lineales.
- `PrimitiveUnityRoot(F,n)`: calcula la n-ésima raíz primitiva de la unidad en un campo de extensión de F.
- `PolyCodeword(obj)`: presenta un polinomio o una lista de polinomios sobre un campo de Galois obtenido de *obj*. El objeto *obj* puede ser una palabra o una lista de palabras.
- `PuncturedCode(C)`: perfora el código C en la última columna y presenta los parámetros del código resultante.
- `ReciprocalPolynomial(P)`: calcula el polinomio recíproco del polinomio P.
- `Redundancy(C)`: calcula el número de símbolos de redundancia del código C.
- `RepetitionCode(n,F)`: calcula el código cíclico de repetición de longitud n sobre F.
- `RootsOfCode(C)`: calcula todas las raíces del polinomio generador del código cíclico C.

- `ShortenedCode(C)`: retorna un código acortado obtenido del código dado C por una sección transversal. Si C es un código lineal, esto se realiza por medio de la eliminación de todas las palabras del código que comienzan con una entrada no cero, para luego eliminar esa coordenada de todas estas palabras. Si C no es lineal, `ShortenedCode` comprueba en primer lugar que elemento del campo finito aparece más en la primera columna de las palabras código. Las palabras del código que no empiecen con este elemento se eliminan del código. En las restantes se elimina la primera columna para formar el código acortado.

- `ShortenedCode(C,L)`. repite el proceso de acortar cada una de las columnas especificadas por L, donde L es una lista de enteros. La lista L se refiere a las columnas del código C.

- `StandardArray(L)`: muestra el arreglo estándar del código lineal L. Un código no lineal no tiene arreglo estándar. En este caso la función `StandardArray(L)` retorna error.

- `SyndromeTable(L)`: esta función calcula la tabla de síndromes para un código lineal L. Ésta consta de dos columnas. La primera columna representa un vector de error o un líder de una clase lateral como ha sido llamado en este trabajo, mientras que la segunda, es el síndrome del respectivo vector de error.

- `Syndrome(L,v)`: calcula el síndrome de una palabra v con respecto a un código lineal L, v es una palabra del espacio vectorial sobre el cual L esta definido.

- `UpperBoundHamming(n,d,q)`: este comando presenta una cota superior sobre el tamaño de un código de longitud n, distancia mínima d y definido sobre el campo de tamaño q.

- `UpperBound(n,d,q)`: presenta la mejor cota superior conocida para $A_q(n, d)$.

- `UpperBoundJohnson(n,d)`: calcula la cota inferior de Johnson para n y d, longitud y distancia mínima, respectivamente. La cota de Johnson trabaja sólo para códigos binarios.

- `UpperBoundPlotkin(n,d,q)`: presenta la cota superior de Plotkin para un código de longitud n, distancia mínima d y definido sobre un campo de tamaño q.

- `UpperBoundSingleton(n,d,q)`: presenta la cota de Singleton para un código de longitud n, distancia mínima d y definido sobre un campo de tamaño q.

- `WeightCodeword(c)`: calcula el peso de la palabra c.

- `WeightDistribution(C)`: calcula en forma de vector, el peso de distribuciones del código C. La i-ésima componente de este vector presenta el número de elementos de C con peso $i - 1$.

- `WordLength(C)`: calcula la longitud del código C.

Más acerca de GAP y del paquete GUAVA se pueden consultar en [19] y [14], respectivamente.

Referencias

[1] Bose R. C and Chowla S. *Theorems in the additive Theory of Numbers*, Comment Math. Helvet. **37** (1962-63), 141-147.

[2] Brouwer A. E. *Bounds on the minimum distance of linear codes.* http://www.win.tue.nl/~aeb/voorlincod.html, 1997-2006.

[3] García G. Trujillo C. y Velásquez J. *Construcción de conjuntos B_h módulo m y particiones*, Matemáticas: Enseñanza Universitaria, Revista de la Corporación. Escuela Regional de Matemáticas, **XIV** No.2 Dic. (2006), 65-70.

[4] Graham R. L. and Sloane N. J. A. *Lower bounds for constant weight codes*, IEEE Transations on Information Theory **26** (1980) No. 1, 37-43.

[5] Gómez C. *Construcción de conjuntos B_h sobre grupos y códigos*, Tesis de Maestría, Universidad del Valle, Cali, (2008).

[6] Gómez C. y Trujillo C *Sobre conjuntos S_h de vectores binarios y códigos lineales*, Revista Colombiana de Matemáticas **45**(2011), 137-146.

[7] Gómez C. y Trujillo C. *Una nueva construcción de conjuntos B_h modulares*, Matemáticas: Enseñanza Universitaria, Revista de la Corporación. Escuela Regional de Matemáticas. **XIX** No, 1 Jun. (2011), 53-62.

[8] Grossman S. *Álgebra Lineal.* Quinta Edición, Mc Graw Hill, México, (1995).

[9] Hill R. *A First Course in Coding Theory.* Clarendon Press, Oxford Applied Mathematics and Computing Sicence Series. (1986).

[10] Huffman W. C. and Pless V. *Fundamentals of Error-Corrrecting Codes.* Cambridge University Press. (2003).

[11] Kiat S. J. *Construction of Binary Linear Codes.* National University of Singapore. (2000).

[12] MacWilliams F. J. and Sloane N. J. A. *The Theory of Error-Corrrecting Codes*, North-Holland Mathematical Library, (2006).

[13] Podestá R. *Introducción a la Teoría de Códigos Autocorrectores.* Mathematics Subject Classification. 94-01, 94-06, 94B05, 94B15. (1991).

[14] R. Baart, T. Boothby, J. Cramwinckel, J. Fields, D. Joyner, R. Miller, E. Minkes, E. Roijackers, L. Ruscio, C. Tjhai, GUAVA-*A GAP package for computing with error-correcting codes*, Version 3.12, (21/05/2012), http://www.gap-system.org/Packages/guava.html.

[15] R. Lidl and H. Niederreiter, *Finite Fields*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, second edition (1997).

[16] Restrepo P., Rosa F. y Muñoz L. *Álgebra lineal con aplicaciones*. Universidad Nacional de Colombia. (2004).

[17] Roman S. *Coding and Information Theory*. Springer, Graduate Texts in Mathematics **34**. (1992).

[18] Shannon C. E. *A Mathematical Theory of Communication*, Bell System Tech. J. **27**, 379-426, 623-656 (1948).

[19] The GAP Group, GAP – Groups, Algorithms, and Programming, Version 4.6.5; 2013. (http://www.gap-system.org).

[20] Xing, C. and Ling S. *Coding Theory a first course*. Cambridge University Press. (2004).

Índice alfabético