Sección de Matemáticas
Universidad de La Laguna

Sandra Carmona García

# *Decodificación de códigos lineales utilizando conjuntos de información*

## Information-Set Decoding

Trabajo Fin de Grado
Grado en Matemáticas
La Laguna, Junio de 2019

Dirigido por
*Irene Márquez Corbella*

*Irene Márquez Corbella*
*Departamento de Matemáticas, Estadística e Investigación Operativa*
*Universidad de La Laguna*
*38200 La Laguna, Tenerife*

# Agradecimientos

Cuando se culmina una etapa o se alcanza un éxito como puede ser la finalización de este trabajo, lo propio es agradecer a las personas que de un modo u otro han intervenido y han puesto su grano de arena para que ésta meta llegue a lograrse.

En este sentido, me gustaría agraceder en primer lugar a mi tutora Irene Márquez Corbella por haberme dado esta oportunidad, su inestimable ayuda y gran dedicación.

A mi familia, en especial a mi padre, por ser incondicionalmente el camino hacia unos nuevos pasos. Por alegrarse de mis éxitos y sufrir mis fracasos. Por ser el respaldo de mis sueños, de mis metas y objetivos, y mi ancla. Por el gran esfuerzo sin el que yo no estaría aquí. Y por estar siempre, paciente.

A mis amigos, los nuevos y los de siempre, por saber estar siempre, ser el apoyo perfecto para superar cualquier obstáculo y quienes dan sentido a los momentos de felicidad.

A Abel, por haberme acompañado siempre, en los buenos y malos momentos. Y por confiar siempre en mi.

A todos, gracias.

Sandra Carmona García
La Laguna, 11 de junio de 2019

# Resumen · Abstract


### *Resumen*

*La criptografía nace de la necesidad de establecer comunicaciones seguras. En la actualidad, está muy presente tanto en nuestra vida personal como institucional, ya que, al mismo tiempo que crece la libertad para comunicarnos, se multiplican los riesgos para la privacidad. Además, la necesidad actual de utilizar y transferir datos electrónicos hace que el uso de la criptografía no sea opcional.*

*El principal problema es que la posible aparición del ordenador cuántico tendría efectos devastadores en la criptografía que se utiliza actualmente. Por eso, estudiamos una posible solución: el estudio de la criptografía basada en códigos, cuya seguridad se basa en la dificultad de decodificar códigos lineales. En este trabajo, estudiamos los algoritmos de decodificación más eficientes conocidos, que son aquellos basados en conjuntos de información. En particular, trabajaremos los algoritmos de Prange (1962), Lee-Brickell (1988) y Stern (1989).*



**Palabras clave:** *Teoría de códigos – Criptografía basada en códigos – Conjunto de información.*

## *Abstract*

*Cryptography arises from the needs to keep safe communications. In a world where the use of electronic data plays a larg part in both personal and institutional life, the proper archiving, retention and encryption is no longer optional.*

*However, the construction of large quantum computers would have dramatically effects on the currently used cryptography. Therefore, we study a possible solution: Code-based Cryptography, whose security is based on the difficulty of decoding linear codes. In particular, we study the best known technique to decode random linear codes: information set decoding algorithms. We will study in this work the decoding algorithms of Prange (1962), Lee-Brickell (1988) and Stern (1989).*

**Keywords:** *Coding Theory – Code based Cryptography – Information Set.*

# Contenido

**Agradecimientos** ..... III

**Resumen/Abstract** ..... V

**1. Introducción** ..... 1
- 1.1. Criptografía de clave secreta ..... 2
- 1.2. Criptografía de clave pública ..... 3
  - 1.2.1. RSA ..... 4
  - 1.2.2. Criptografía basada en códigos ..... 5

**2. Teoría de códigos correctores** ..... 7
- 2.1. Introducción ..... 7
- 2.2. Códigos correctores de errores ..... 7
- 2.3. Códigos lineales ..... 9
  - 2.3.1. Matriz generatriz de un código lineal ..... 10
  - 2.3.2. Matriz de paridad ..... 12
  - 2.3.3. Código dual ..... 15
  - 2.3.4. Código perforado ..... 16
- 2.4. Problema de decodificación ..... 17
- 2.5. Complejidad ..... 18
  - 2.5.1. Algoritmos de decodificación básicos ..... 20
  - 2.5.2. El problema de decodificación de códigos lineales es NP ..... 24
- 2.6. Otras cotas definidas para códigos lineales. ..... 25
  - 2.6.1. Los conjuntos $A_q(n,d)$ y $B_q(n,d)$ ..... 25
  - 2.6.2. Cota de Singleton ..... 28
  - 2.6.3. Empaque de esferas, radio de cobertura y códigos perfectos ..... 28
  - 2.6.4. Cota de Hamming ..... 29
  - 2.6.5. Cota de Gilbert-Varshamov ..... 30
  - 2.6.6. Cotas asintóticas ..... 31

**3. Decodificación usando conjuntos de información** ..... 37
3.1. Preliminares ..... 37
3.2. Algoritmo de Prange ..... 39
3.3. Algoritmo de Lee-Brickell ..... 41
3.4. Algoritmo de Stern ..... 42
3.5. Más algoritmos de decodificación utilizando conjuntos de información ..... 47

**Bibliografía** ..... 49

**Poster** ..... 51

# 1

# Introducción

Las raíces etimológicas de la palabra **criptografía** son *kriptos*, que significa oculto, y *graphos*, que significa escribir, lo que da una clara idea de su definición cotidiana: escribir mensajes enigmáticamente.
La criptografía empezó a ser considerada una ciencia aplicada (debido a su relación con la teoría de números, la teoría de la información, la teoría de la complejidad computacional, etc) en el año 1949, cuando Shannon publicó la *Teoría de las comunicaciones secretas*, aunque este arte se remonta muy atrás en el tiempo. Shannon, en 1949, mencionó la relación que existe entra la teoría de códigos y la criptografía. Ambas ciencias comparten muchos conceptos y resultados, pero son dos materias esencialmente distintas.

- Teoría de códigos: el objetivo es enviar un mensaje de manera fiable y eficiente a través de canales afectados por ruido y que, por lo tanto, pueden distorsionar la información.

  **mensaje enviado** $\xrightarrow{\text{canal ruidoso}}$ **mensaje recibido**

- Criptografía: el objetivo es ocultar, hacer confusa la información.

  **texto original** $\xrightarrow{\text{transformación}}$ **texto cifrado**

En el primer caso, la distorsión del mensaje no es intencionada, sino que se debe al ruido que hay en el canal, que produce errores en el mensaje. En teoría de códigos, el objetivo es determinar un sistema de codificación/decodificación que permita transmitir de forma rápida y eficiente y que también nos permita recuperar la información emitida a pesar de las alteraciones sufridas en la transmisión.
En la vida cotidiana utilizamos los códigos correctores de forma muy frecuente. Los ejemplos más comunes son el código con un dígito de control como el ISBN, el DNI o los códigos de barra que permiten detectar si se ha cometido un error pero no corregirlo.

En el caso de la criptografía, la distorsión del mensaje sí es intencionada, con la finalidad de transmitir un mensaje que sólo pueda ser entendido por las personas autorizadas. Además, se trabaja con la posible existencia de una tercera persona (el adversario) que intenta obtener el mensaje original. En resumen, en teoría de códigos se intenta transmitir el mensaje dando la mayor cantidad de información posible, mientras que en criptografía se trata de todo lo contrario; es decir, hacer el mensaje lo más incomprensible posible para que el adversario no pueda descifrarlo.
Nos centraremos en este capítulo en criptografía. Nuestro objetivo es explicar la criptografía basada en códigos correctores, un ejemplo de criptografía de clave pública que resiste ataques utilizando el ordenador cuántico. Antes, vamos a realizar un resumen de los conceptos claves.

## 1.1. Criptografía de clave secreta

El cifrado simétrico (también conocido como cifrado de clave secreta) consiste en utilizar la misma clave para el cifrado del mensaje que para el descrifrado. Este tipo de cifrados requieren que se utilice un canal seguro para intercambiar la clave, lo que hace que estos cifrados sean vulnerables a robos. Es decir, los participantes sólo pueden comunicarse con un acuerdo previo, además, este cifrado requiere de una clave para cada par de personas que quieran comunicarse y por lo tanto, el número de claves aumenta considerablemente a medida que aumenta el número de participantes. Este factor hace inviable el uso único de estos criptosistemas hoy en día. Shannon sugirió dos métodos básicos que detallaremos a continuación para cifrar un mensaje mediante clave secreta: la difusión y la confusión (o dicho de otra forma, utilizar técnicas de transposición y sustitución).

- El propósito de la difusión es anular la influencia del lenguaje sobre el texto original. La forma más sencilla de conseguirlo es con la técnica de transposición, que consiste en crear el texto cifrado simplemente desordenando las unidades que forman el texto original, es decir, alteramos el orden de los elementos del texto según una clave.
- El objetivo de la confusión es hacer que la relación entre la clave y el texto cifrado sea lo más compleja posible, buscando que el texto cifrado no se parezca al texto original. Esto se consigue normalmente con la técnica de sustitución, que consiste en sustituir los elementos de cada palabra por otras, es decir, reemplazamos según una clave.

Como ejemplo sencillo de cifrado de clave secreta tenemos el conocido **cifrado de César**. Corresponde a una función **T** que sustituye las letras de un alfabeto de cardinal 27 según la fórmula: $T(M) \equiv M + 3 \pmod{27}$. Este criptosistema debe su nombre a Julio César, quien lo utilizó.
Actualmente es fácil su criptoanálisis pero en la época eran pocos los que sabían

leer y aún menos los que habrían podido hacer uso de técnicas de criptoanálisis. El sistema anterior puede generalizarse de forma que si tenemos un alfabeto cualquiera con $m$ símbolos y un entero fijo $b$, entonces podemos definir la **transformación desplazamiento** como $T_b(M) \equiv M + b \pmod{m}$. Si queremos descifrarlo, basta con calcular $M = T_b^{-1}(C) \equiv C - b \pmod{m}$. Este tipo de criptosistemas son muy inseguros hoy en día, pues la clave secreta es $b$, para la que sólo hay 27 posibilidades. La potencia de los ordenadores actuales permite probar todas las claves de forma rápida. Este sistema puede mejorarse usando lo que se conoce como **transformación afín** que viene dada como $T_{(a,b)}(M) = C \equiv aM + b \pmod{m}$, donde $a$ y $b$ son enteros fijos y juntos forman la clave. Augusto, el sobrino de Julio César, lo utilizó haciendo un desplazamiento de una letra. Más recientemente, ha sido utilizado en secciones de anuncios de periódicos, incluso por el capo mafioso Bernardo Provenzano quien lo llegó a utilizar en pleno siglo XXI.

## 1.2. Criptografía de clave pública

En el siglo XX, las nuevas tecnologías electrónicas y digitales se adaptaron a las máquinas criptográficas. De esta forma, se dio paso a sistemas criptográficos más modernos y mucho más fiables que la sustitución y transposición clásica. Además, a partir de la segunda mitad del siglo XX con el desarrollo de internet y expansión de la informática han surgido nuevas aplicaciones de la criptografía propulsadas por la necesidad de proteger datos e información durante su transmisión y almacenamiento, ya que, al mismo tiempo que crece la libertad de comunicarse, se multiplican los riesgos para la privacidad. Hoy en día, en un mundo en que la utilización de datos electrónicos es imprescindible tanto en la vida personal como en la institucional, el uso de la criptografía ya no es opcional. La criptografía de clave pública fue creada en 1976 por Diffie y Hellman y resuelve uno de los problemas clave que tenía la criptografía hasta el momento: el intercambio de claves.

El cifrado de clave pública utiliza un par de claves relacionadas matemáticamente. Se llama cifrado asimétrico porque no puede utilizarse la misma clave para cifrar y descifrar el mensaje. Cada participante en este criptosistema dispone de un par de claves: una que designa como **clave privada** y se mantiene secreta y la otra clave es **pública** y se reparte a quien lo desee.

Si consideramos un conjunto amplio de usuarios que utilizan criptografía asimétrica, cada usuario dispondrá ahora de dos claves: una secreta que deberá conservar $E_k$ y una pública que debe difundir entre el resto de usuarios $D_k$. Así, se publica un listado con todos los usuarios y su clave pública. Cuando el usuario A quiere enviar un mensaje $x$ a otro miembro B basta con que localice su clave pública y cifre el mensaje utilizando esta clave ($D_k$). Ahora, sólo B es capaz de descifrar el mensaje porque es el único que conoce su clave privada necesaria para el des-

cifrado.
Los principios teóricos que debe satisfacer un criptosistema asimétrico:

- La obtención de la clave secreta $E_k$ debe ser computacionalmente imposible a partir de la clave pública $D_k$ y del texto cifrado.
- El cálculo de $E_k$ y $D_k$ debe ser computacionalmente sencillo.

Para asegurar estás propiedades hay que considerar una función unidireccional, que son funciones matemáticas de cálculo fácil en un sentido (one-way functions), esta parte de la función se utiliza para cifrar; pero que su sentido inverso sea computacionalmente imposible si no se tiene información extra (función unidireccional con trampa), es decir descifrar sin conocer la clave secreta sea impracticable (computacionalmente).

**Aplicaciones de la criptografía de clave pública**

Este tipo de criptografía ocupa un lugar destacado en nuestro día a día, entre otras aplicaciones destacamos:

- Se utiliza en la tecnología blockchain y criptomoneda. (Bitcoin utiliza un tipo de cifrado asimétrico -firma digital con curvas elípticas ECDSA-para verificar sus transacciones).
- El protocolo SSL que hace posible las conexiones seguras en los sitios web también emplea criptografía asimétrica.
- Aplicaciones como WhatsApp, Telegram... cifran los mensajes de sus usuarios utilizando criptografía asimétrica.

### 1.2.1. RSA

El sistema RSA fue desarrollado por Ronald Rivest, Adi Shamir y Leonard Adleman en 1976 (un año después del intercambio de clave propuesto por Diffie y Hellman), de ahí el nombre RSA que son las iniciales de los apellidos de cada uno. Se trata de un sistema de clave pública y el más utilizado actualmente.
La seguridad de este sistema se basa en el hecho de que no existe ninguna forma eficiente de factorizar números enteros. En el algoritmo 1, describimos el algoritmo que se usa para cifrar y descifrar mensajes usando el sistema RSA.
Consideramos un ejemplo sencillo con la finalidad de aclarar el algoritmo RSA.

*Ejemplo 1.1.* Tomamos $p = 3$ y $q = 5$. De esta forma, tenemos que $n = 3{\cdot}5 = 15$ y $\phi(n) = 8$. Podemos tomar ahora $d = 3$ ya que m.c.d$(3, 8) = 1$ y calculamos su inverso (mod $\phi(n)$). Consideramos ahora $e = 3$ ya que $3{\cdot}3 = 9 \equiv 1 \pmod 8$. Supongamos ahora que tenemos un mensaje codificado en base decimal, nuestro mensaje es $M = 7$. Su cifrado es $C = 13 \equiv 7^3 \pmod{15}$. Efectivamente, si desciframos el mensaje $C = 13$ tenemos que $13^3 \pmod{15} \equiv 7$, que es nuestro mensaje original.

**Algoritmo 1:** RSA.

**Generación de claves**:
- Elegir dos números primos grandes $p$ y $q$ ($\simeq 256$ bits).
- Calcular $n = pq$.
- Calcular $\phi(n) = (p-1)(q-1)$.
- Elegir $d$ primo con $\phi(n)$.
- Calcular $e$ tal que $1 \leq e \leq \phi(n)$ y $e{\cdot}d \equiv 1 \pmod{\phi(n)}$.

**Clave pública**: $(e, n)$.
**Clave privada**: $(\phi(n), d)$ o, lo que es lo mismo, $(p, q, d)$.
**CIFRADO**:
Cada mensaje $M$ se escribe en un sistema numérico en base $b$ y se divide en bloques de tamaño $j-1$ tal que $b^{j-1} < n < b^j$.
Cada bloque $M_i$ se cifra utilizando la expresión:

$$C_i \equiv M_i^e (\bmod n).$$

**DESCIFRADO**: Para cifrar $C_i$ se utiliza la clave privada
$M_i = C_i^d \equiv (M_i^e)^d \equiv M_i \pmod{n}$, pues $e{\cdot}d \equiv 1 \pmod{\phi(n)}$.

Las principales dificultades en el uso del sistema del RSA son: cálculo de potencias modulares y búsqueda de números primos adecuados para evitar que el sistema pueda ser interceptado con facilidad.
Además, el algoritmo de Shor es un algoritmo cuántico que permite descomponer en tiempo polinomial un entero. Es decir, si el ordenador cuántico llega a ser real, tendrá fatídicas consecuencias en la criptografía asimétrica que utilizamos hoy en día. El tema es serio, el NIST (agencia que se encarga de proponer los estándares criptográficos) hizo una convocatoria en 2017 para proponer algoritmos matemáticos que se puedan utilizar en criptografía de clave pública y sean computacionalmente difíciles para el ordenador cuántico. En la siguiente página web https://csrc.nist.gov/Projects/Post-Quantum-Cryptography se pueden ver las últimas noticias del concurso de NIST. Actualmente, están en la segunda ronda y todos los algoritmos propuestos están basados en tres problemas matemáticos: decodificación de códigos lineales, el problema del vector más cercano en retículos y resolución de ecuaciones no lineales en varias variables. En este trabajo vamos a estudiar el primer problema.

### 1.2.2. Criptografía basada en códigos

Esta técnica fue propuesta en 1978 por Robert McEliece, primera persona que unificó los términos de criptografía y teoría de códigos. El sistema de McEliece se trata de un sistema de clave pública y es considerado uno de los más eficientes y seguros hasta el momento, aunque debido al tamaño de sus claves

no se utiliza todavía.
En los criptosistemas de tipo McEliece la idea es utilizar un código lineal para hacer ininteligible el mensaje. Como veremos en el capítulo 2, codificar un mensaje con un código lineal es un proceso rápido que sólo requiere multiplicar un vector por una matriz. A este mensaje codificado se le añaden errores para que el mensaje no se comprenda aunque sea interceptado. Finalmente, el proceso de descifrado requiere de decodificar un mensaje, un proceso computacionalmente difícil si se utiliza un código lineal aleatorio.
La clave de estos criptosistemas radica en utilizar una familia de códigos lineales del que se conozca un algoritmo eficiente de decodificación. En particular, McEliece propone utilizar la familia de códigos Goppa y su propuesta, 40 años más tarde, sigue siendo resistente a cualquier ataque. Es más, es uno de los candidatos que todavía se mantiene en el concurso NIST.
El primer argumento que justifica la seguridad del criptosistema de McEliece es la gran dificultad que supone la decodificación de un código lineal aleatorio. Para este problema sólo se conocen soluciones de tiempo exponencial. Los mejores algoritmos de decodificación conocidos son algoritmos que utilizan conjuntos de información y estudiaremos los más básicos en el capítulo 3. El segundo argumento que justifica la seguridad es que la familia de códigos que se utilice no sea distiguible de un código lineal aleatorio. Todos los conceptos de Teoría de

**Algoritmo 2:** El criptosistema de McEliece

**Input:** $n$, $t \in \mathbb{N}$, con $t \ll n$.
**Generación de claves**: Dados los parámetros $n$ y $t$, generamos una matriz $G$ de tamaño $k \times n$ generatriz de un código $\mathcal{C}$ sobre el cuerpo $\mathbb{F}$ con dimensión $k$ y distancia mínima $d \geq 2t + 1$. En el caso de McEliece $\mathcal{C}$ pertenece a la familia de códigos Goppa.
**Clave pública**: $(G, t)$.
**Clave privada**: $D_{\mathcal{C}}$, un decodificador eficiente para $\mathcal{C}$ que corrija $t$ errores.
**Cifrado**: Para cifrar un mensaje $m \in \mathbb{F}_q^k$ elegir de forma aleatoria un vector $e \in \mathbb{F}_q^n$ de peso menor o igual que $t$. El cifrado del mensaje es: $c = mG + e$.
**Descifrado**: Aplicar $D_{\mathcal{C}}$ al mensaje cifrado $c$.

Códigos que utiliza este algoritmo se estudiarán en el capítulo 2. El capítulo 3 está dedicado a intentar atacar este criptosistema, es decir estudiar diferentes algoritmos de decodificación eficientes de códigos lineales.

# 2

# Teoría de códigos correctores

## 2.1. Introducción

El principal objetivo de la Teoría de Códigos es transferir de forma eficiente y sin errores un mensaje entre un emisor y un receptor. El principal problema es que los mensajes que envíamos pueden ser dañados por interrupciones en el canal de envío. Por lo tanto, debemos corregir los errores producidos para que así los mensajes enviados y recibidos coincidan y el proceso se realice correctamente. El proceso que se realiza es el siguiente:

- Añadimos información redundante al mensaje, proceso que se conoce como **codificación**.
- Posteriormente, se envía el mensaje por un canal que contiene ruido, por lo que, al receptor, le puede llegar el mensaje perturbado (con errores).
- Se recupera el mensaje original enviado, a pesar de los errores que se hayan cometido, proceso que se conoce como **decodificación**.

Para poder realizar este proceso con éxito, es necesario que el mensaje tenga una tasa de información alta y que la capacidad correctora de errores sea alta. Estos dos conceptos los introduciremos más adelante.

## 2.2. Códigos correctores de errores

Supongamos que tenemos un conjunto de símbolos $\mathcal{A}$. Si llamamos $\mathcal{A}^n$ al conjunto de $n$-uplas $(x_1, ..., x_n)$ con $x_i \in \mathcal{A}$. Un **código en bloque** $\mathcal{C}$ de longitud $n$ sobre $\mathcal{A}$ es un subconjunto no vacío de $\mathcal{A}^n$. Los elementos de $\mathcal{C}$ se llaman palabras del código.

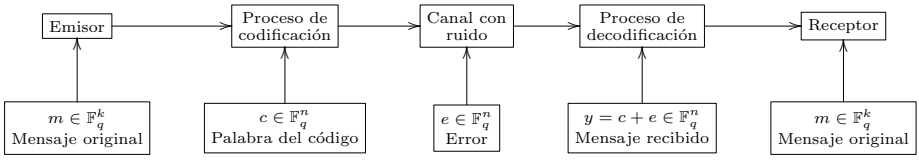
*Observación 2.1.* En nuestro caso, nuestro alfabeto $\mathcal{A}$ será un cuerpo finito con $q$ elementos. Recordamos que todos los cuerpos finitos tienen $q = p^n$ elementos,

donde $p$ es un número primo. Denotaremos al cuerpo finito con $q$ elementos como $\mathbb{F}_q$.

El proceso de comunicación utilizando códigos y el alfabeto $\mathcal{A} = \mathbb{F}_q$ y un código en bloque $\mathcal{C} \subseteq \mathbb{F}_q^n$ es el siguiente:

- En primer lugar, nuestro mensaje $m$ se presenta como $k-$tuplas del alfabeto $\mathbb{F}_q$ y se codifica con una transformación que le añade información redundante $(n-k$ símbolos) hasta convertirlo en una palabra $c$ del código $\mathcal{C} \subseteq \mathbb{F}_q^n$.
- Posteriormente, esta palabra es enviada a través de un canal donde el mensaje puede ser dañado.
- Tenemos que la palabra recibida será $c+e$ donde $e$ representa los errores mencionados anteriormente. Llamaremos $y = c + e$ al mensaje recibido. Tenemos que $e, y \in \mathbb{F}_q^n$ y que $c \in \mathcal{C} \subseteq \mathbb{F}_q^n$.
- A continuación, viene el proceso más delicado, el mensaje $y$ se somete a un proceso de decodificación donde se intenta recuperar el mensaje original $m \in \mathbb{F}_q^k$.
- Después de este proceso, obtendremos el mensaje original. Si no es así, no hemos realizado correctamente el proceso de decodificación o se han cometido más errores de los que nuestro código puede corregir.

La totalidad del proceso es resumida en la siguiente imagen:




Si $\mathcal{C}$ es un código sobre $\mathbb{F}_q$ con $M$ palabras, decimos que $M$ es el tamaño del código. Si $n$ es la longitud de $\mathcal{C}$ y $M$ es el tamaño del código decimos que $\mathcal{C}$ es un $(n, M)_q$ código.

**Definición 2.2.** *Sea $\mathcal{C}$ un $(n, M)_q$-código definido sobre $\mathbb{F}_q$, llamamos* ***redundancia*** *al valor $n - \log_q(M)$.*

**Definición 2.3.** *Para $x = (x_1, ..., x_n)$, $y = (y_1, ..., y_n) \in \mathbb{F}_q^n$. Se llama* ***distancia de Hamming*** *al número de posiciones en las que $x$ e $y$ difieren. Se denota $d_H(x, y)$.*

**Definición 2.4.** *Sea $\mathcal{C}$ un $(n, k)_q$-código en bloque con $k = \lfloor \log_q(M) \rfloor$. Un* ***codificador*** *de $\mathcal{C}$ es una aplicación: $\varepsilon : \mathbb{F}_q^k \longrightarrow \mathbb{F}_q^n$ tal que $\varepsilon(\mathbb{F}_q^k) = \mathcal{C}$, es decir, la imagen de nuestra aplicación es nuestro código en bloque $\mathcal{C}$ ($Im(\varepsilon) = \mathcal{C}$).*

Si $c \in \mathcal{C}$ es una palabra del código, entonces existe un único mensaje $m \in \mathbb{F}_q^k$ tal que $c = \varepsilon\ (m)$.

**Proposición 2.5.** *La distancia de Hamming es una métrica y, por tanto, cumple las siguientes propiedades:*

1. $d_H(x,y) \geq 0$ *y* $d_H(x,y) = 0 \iff x{=}y$ *(Definida positiva)*
2. $d_H(x,y) = d_H(y,x)$ *(Simetría)*
3. $d_H(x,y) \leq d_H(x,z) + d_H(z,y)$ *(Desigualdad triangular)*

**Definición 2.6.** *La* ***distancia mínima*** *de un código de longitud $n$ se define como:* $d(\mathcal{C}) = \min\{d_H(x,y) : x, y \in \mathcal{C}, x \neq y\}$.

## 2.3. Códigos lineales

**Definición 2.7.** *Un* ***código lineal*** $\mathcal{C}$ *es un subespacio lineal de* $\mathbb{F}_q^n$*. En los códigos lineales podemos hablar de un nuevo parámetro: su dimensión, que es la dimensión de $\mathcal{C}$ como $\mathbb{F}_q$-espacio vectorial.*

La redundancia de este tipo de códigos es $n - k$.

*Observación 2.8.* Denotamos un código $\mathcal{C}$ sobre $\mathbb{F}_q$ de longitud $n$ y dimensión $k$ como un $[n,k]_q$ código. Si, además, la mínima distancia del código $d$ es conocida, nos referimos al código $\mathcal{C}$ como un $[n,k,d]_q$-código.

*Observación 2.9.* Si $\mathcal{C}$ es un código lineal y sea $\{g_1, \ldots, g_k\}$ es una base de $\mathcal{C}$, se tiene que para toda palabra del código $c \in \mathcal{C}$, se escribe como $c = \lambda_1 g_1 + \ldots \lambda_k g_k$ con $\lambda_i \in \mathbb{F}_q$, es decir cualquier elemento $c \in \mathcal{C}$ se puede escribir como combinación lineal de los elementos de la base. Por lo tanto, un código lineal sobre $\mathbb{F}_q$ de dimensión $k$ tiene tamaño $M = q^k$.

**Definición 2.10.** *Para todo vector $x \in \mathbb{F}_q^n$, su* ***soporte*** *denotado por* $\mathrm{supp}(x)$ *se define como el conjunto de posiciones de $x$ donde hay elementos distintos de cero, esto es,* $\mathrm{supp}(x) = \{i : x_i \neq 0\}$.

**Definición 2.11.** *El* ***peso de Hamming*** *de un vector $x \in \mathbb{F}_q^n$ es el número de elementos de su soporte, y se denota como* $w_H(x) = \#\mathrm{supp}(x) = \#\{i : x_i \neq 0\}$*, donde $\#A$ denota el cardinal del conjunto $A$.*

**Definición 2.12.** *El* ***mínimo peso*** *de un código $\mathcal{C}$, denotado por* $\min(w_H(\mathcal{C}))$*, se define como el mínimo valor de los pesos de las palabras distintas de cero del código $\mathcal{C}$. Esto es:* $\min(w_H(\mathcal{C})) = \min\{w_H(c) : c \in \mathcal{C}, c \neq 0\}$.

**Proposición 2.13.** *Sea $\mathcal{C}$ un código lineal, entonces:* $d(\mathcal{C}) = \min(w_H(\mathcal{C}))$.

*Demostración.* Tenemos que $\mathcal{C}$ es un código lineal, es decir, un subespacio vectorial de $\mathbb{F}_q^n$. Existen dos palabras $c_1$ y $c_2 \in \mathcal{C}$ tal que $d(\mathcal{C}) = d_H(c_1, c_2)$ con $c_1 \neq c_2$. Además, $d_H(c_1, c_2) = w_H(c_1 - c_2) \geq \min(w_H(\mathcal{C}))$. Análogamente, como $c_1 - c_2 \in \mathcal{C} \setminus \{0\}$, pues $c_1 \neq c_2$, $\min(w_H(\mathcal{C})) = w_H(c)$ con $c \in \mathcal{C} \setminus \{0\}$. Entonces $\min(w_H(\mathcal{C})) = d_H(0, c) \geq d(\mathcal{C})$. □

### 2.3.1. Matriz generatriz de un código lineal

Sea $\mathcal{C}$ un $[n,k]_q$ código lineal. Como $\mathcal{C}$ es un subespacio vectorial de $\mathbb{F}_q^n$ de dimensión $k$, existen $k$ vectores linealmente independientes $g_1, ..., g_k$ donde $g_i \in \mathcal{C}$. Denotamos $g_i = (g_{i1}, ..., g_{in}) \in \mathbb{F}_q^n$.

**Definición 2.14.** *Una **matriz generatriz** del código $\mathcal{C}$ viene dada por:*

$$\begin{pmatrix} g_1 \\ g_2 \\ \vdots \\ g_k \end{pmatrix} = \begin{pmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k1} & g_{k2} & \cdots & g_{kn} \end{pmatrix} \in \mathbb{F}_q^{k\times n}$$

Como cada palabra del código puede expresarse de forma única como combinación lineal de los elementos de la base entonces, cada $c \in \mathcal{C}$, puede escribirse como $c = \lambda_1 \cdot g_1 + ... + \lambda_k \cdot g_k$ donde $\lambda_i \in \mathbb{F}_q$. Si nuestro mensaje $m$ es una $k-$upla de $\mathbb{F}_q$, es decir $m = (m_1, ..., m_k) \in \mathbb{F}_q^k$. Entonces tenemos $c = m \cdot G$ con $c \in \mathcal{C}$. Es decir, el proceso de codificación de un código lineal se puede describir con una multiplicación entre un vector y una matriz.

*Observación 2.15.* La matriz generatriz se utiliza durante el proceso de codificación de un código lineal. El proceso de codificación de un código lineal se define como:

$$\begin{aligned} \varepsilon : \mathbb{F}_q^k &\longrightarrow \mathbb{F}_q^n \\ m &\longmapsto \varepsilon(m) = c = mG \end{aligned}$$

donde $G$ es una matriz generatriz del código $\mathcal{C}$.

*Observación 2.16.* Dado un código $[n,k]_q$, la matriz generatriz no es única, aunque todas son matrices de tamaño $k \times n$ y rango $k$. Del mismo modo, tenemos que toda matriz $k \times n$ de rango $k$ es una matriz generatriz de un $[n,k]_q$-código lineal.

**Proposición 2.17.** *Supongamos que $G$ es una matriz generatriz del código $\mathcal{C}$. Podemos transformar $G$ (haciendo operaciones elementales por filas) en otra matriz generatriz de $\mathcal{C}$. Es decir, se permiten las siguientes operaciones: intercambiar dos filas, multiplicar una fila por una constante distinta de cero y sumar dos filas.*

Diremos que 2 matrices del mismo código son matrices equivalentes. Dada una matriz $G$, hay exactamente una matriz en forma escalonada reducida equivalente a G, denotada por $\text{rref}(G)$.

**Proposición 2.18.** *Sea $G$ una matriz generatriz del código $\mathcal{C}$, entonces* rref(G) *también es una matriz generatriz de $\mathcal{C}$ y* rref(G) $= MG$ *donde $M$ es una matriz invertible $k \times k$ cuyos elementos son de $\mathbb{F}_q$.*

*Demostración.* La forma escalonada de la matriz $G$ es obtenida a través de operaciones elementales. El código $\mathcal{C}$ es el espacio generado por las filas de $G$ y esto no cambia con operaciones elementales. Con lo cual, $\text{rref}(G)$ genera el mismo código $\mathcal{C}$. Además $\text{rref}(G) = E_1 \dots E_l \cdot G$ donde $E_i$ son las matrices elementales que corresponden a las operaciones elementales por filas que realizamos para obtener $\text{rref}(G)$ a partir de $G$. Si tomamos $M = E_1 \dots E_l$, entonces $M$ es invertible, pues las matrices $E_i$ son invertibles y tenemos que $\text{rref}(G) = MG$. □

**Proposición 2.19.** *Sean $G_1$, $G_2$ dos $k \times n$ matrices generatrices de los códigos $\mathcal{C}_1$ y $\mathcal{C}_2$ definidos en $\mathbb{F}_q$. Entonces las siguientes afirmaciones son equivalentes:*

1. $\mathcal{C}_1 = \mathcal{C}_2$
2. $\text{rref}(G_1) = \text{rref}(G_2)$
3. *Existe una matriz $M \in \mathbb{F}_q^{k \times k}$ invertible tal que $G_2 = MG_1$.*

*Demostración.* "(1) ⇒ (2)" Los espacios generados por las filas de $G_1$ y $G_2$ son el mismo ya que $\mathcal{C}_1 = \mathcal{C}_2$. Con lo cual, $G_1$ y $G_2$ son equivalentes y, por tanto, $\text{rref}(G_1) = \text{rref}(G_2)$.

"(2) ⇒ (3)" Existe una matriz $k \times k$ invertible tal que $G_i = M_i \text{rref}(G_i)$ para $i = 1, 2$. Tomamos $M = M_2 M_1^{-1}$ y tenemos que:

$$MG_1 = M_2 M_1^{-1} M_1 \text{rref}(G_1) = M_2 \text{rref}(G_2) = G_2$$

"(3) ⇒ (1)" Supongamos que $G_2 = MG_1$ para alguna matriz $M$ invertible $k \times k$. Entonces, toda palabra del código $\mathcal{C}_2$ es combinación lineal de filas de $G_1$ que está en $\mathcal{C}_1$. Con lo cual, tenemos que $\mathcal{C}_2$ es un subcódigo de $\mathcal{C}_1$, esto es, $\mathcal{C}_2 \subseteq \mathcal{C}_1$. De la misma forma puede probarse que $\mathcal{C}_1 \subseteq \mathcal{C}_2$ ya que $G_1 = M^{-1} G_2$. Por lo tanto, $\mathcal{C}_1 = \mathcal{C}_2$.

□

**Definición 2.20.** *Una matriz generatriz $G$ del código $\mathcal{C}$ se dice sistemática en las posiciones $I = \{j_1, \dots, j_k\}$ si la submatriz $G_I \in \mathbb{F}_q^{k \times k}$ definidas por las $k$ columnas de $G$ indexadas por $I$ forman la matriz identidad. Si $G$ es sistemática en $I = \{i_1, \dots, i_k\}$, la transformación de codificación asociada se llama* codificación sistemática.

*Observación 2.21.* Si una matriz generatriz $G$ del código $\mathcal{C}$ es sistemática en las posiciones $\{j_1, \dots, j_k\}$ y $c \in \mathcal{C}$ es una palabra del código, entonces $c = mG$ para un único mensaje $m \in \mathbb{F}_q^k$ y $c_{ij} = m_i$ para cada $i = 1, \dots, k$.
Supongamos que $j_i$, $1 \leq j_1 < \dots < j_k \leq n$, indica la posición del pivote de $\text{rref}(G)$. Entonces el código $\mathcal{C}$ y la matriz generatriz $\text{rref}(G)$ se dicen sistemáticas en las posiciones $\{j_1, \dots, j_k\}$.

**Proposición 2.22.** *Sea $\mathcal{C}$ un $[n, k]_q$ código lineal y $G \in \mathbb{F}_q^{k \times n}$ una matriz generatriz, entonces $\mathcal{C}$ es sistemático en las posiciones $j_1, \dots, j_k$ si, y solo si, las $k$ columnas de $G$ en las posiciones $j_1, \dots, j_k$ son linealmente independientes.*

*Demostración.* Sea $G$ una matriz generatriz de $\mathcal{C}$. Tomamos $G_I$ la submatriz $k \times k$ dada por las $k$ columnas indexadas por $\{j_1, \ldots, j_k\}$.
Como $\mathcal{C}$ es sistemático en las posiciones $\{j_1, \ldots, j_k\}$. Entonces, la aplicación $x \longmapsto xG_I$ es inyectiva y, por tanto, las columnas de $G_I$ son linealmente independientes.

Recíprocamente, sea $I = \{j_1, \ldots, j_k\}$. Si las columnas de $G_I$ son linealmente independientes, entonces existe una matriz $M$, $k \times k$ invertible tal que $MG_I$ es la matriz identidad. Por tanto, $MG$ es la matriz generatriz de un código $\mathcal{C}$ sistemático en las posiciones de $I$.

□

**Definición 2.23.** *En el caso de la Definición 2.20, al conjunto $I = \{j_1, \ldots, j_k\}$ se le llama* ***conjunto de información****.*

Esta definición será muy utilizada en el capítulo 3.

### 2.3.2. Matriz de paridad

Hemos descrito anteriormente la matriz generatriz de un código lineal y cómo viene definido dicho código a través de su matriz generatriz. Sin embargo, no es la única matriz que nos sirve para caracterizar un código. En este sentido, recordemos que tenemos dos formas de describir un subespacio: explícitamente, dando una base o implícitamente, descrito como la solución de un conjunto de ecuaciones lineales homogéneas. Por lo tanto, hay dos formas de describir un código lineal: explícitamente, utilizando la matriz generatriz, o implícitamente dando un conjunto de ecuaciones lineales homogéneas cuya solución es el código.

**Definición 2.24.** *Sea $H \in \mathbb{F}_q^{(n-k)\times n}$ una matriz de rango $n-k$. Se dice que $H$ es una matriz de paridad de un $[n,k]_q$ código $\mathcal{C}$ si el código es el espacio nulo de esa matriz. Es decir: $\mathcal{C} = \{c \in \mathbb{F}_q^n : Hc^T = 0\}$.*

*Observación 2.25.* La matriz de paridad de un código puede usarse para detectar si la palabra recibida $y \in \mathbb{F}_q^n$ pertenece o no al código. Tenemos que:

$$Hy^T = 0 \text{ si } y \in \mathcal{C} \quad \text{y} \quad Hy^t \neq 0 \text{ si } y \notin \mathcal{C}.$$

**Definición 2.26.** *Sea $H \in \mathbb{F}_q^{(n-k)\times n}$ una matriz de paridad de un $[n,k]_q$ código $\mathcal{C}$. Para todo vector $y \in \mathbb{F}_q^n$, el vector $S(y) = Hy^T$ se llama* ***síndrome*** *de la palabra recibida $y \in \mathbb{F}_q^n$.*

*Observación 2.27.* Sea $y = c + e$ el vector recibido con $c \in \mathcal{C}$ y $e \in \mathbb{F}_q^n$ un vector error. Entonces: $S(y) = Hy^T = H(c+e)^T = Hc^T + He^T = He^T = S(e)$. Es decir, el síndrome de la palabra recibida coincide con el síndrome del error cometido.

**Proposición 2.28.** *Sea $\mathcal{C}$ un $[n,k]_q$-código. Sea $G \in \mathbb{F}_q^{k\times n}$ una matriz generatriz de $\mathcal{C}$ y sea $H \in \mathbb{F}_q^{(n-k)\times n}$ una matriz de rango $n-k$. Entonces, $H$ es una matriz de paridad de $\mathcal{C}$ si y solo si $GH^T = 0$.*

*Demostración.* Suponemos que $H$ es una matriz de paridad de $\mathcal{C}$. Entonces, $H(mG)^T = HG^T m^T = 0$, para todo mensaje $m \in \mathbb{F}_q^k$, luego: $GH^T = 0$.
Recíprocamente, supongamos que $GH^T = 0$. Por la Observación 2.27, tenemos que $H$ es una matriz de paridad de un $[n,k]_q$ código que llamaremos $\mathcal{C}_2$. Además, como $G$ es una matriz generatriz de $\mathcal{C}$, tenemos que para cada $c \in \mathcal{C}$, $c = mG$ para cierto mensaje $m \in \mathbb{F}_q^k$. Ahora, $Hc^T = H(mG)^T = 0$. Luego $c = mG \in \mathcal{C}_2$. Esto implica que $\mathcal{C} \subseteq \mathcal{C}_2$. Como ambos códigos tienen dimensión $k$ se deduce que $\mathcal{C} = \mathcal{C}_2$. Por lo tanto, $H$ es una matriz de paridad de $\mathcal{C}$. □

Veamos cómo obtener la matriz de paridad $H$ a partir de una matriz generatriz $G$ de $\mathcal{C}$ en ciertos casos particulares.

**Proposición 2.29.** *Sea $\mathcal{C}$ un $[n,k]_q-$código lineal. Denotamos $I_k \in \mathbb{F}_q^{k\times k}$ la matriz identidad de tamaño $k$. Sea $P \in \mathbb{F}_q^{(n-k)\times n}$. Entonces $G = (I_k|P)$ es una matriz generatriz de $\mathcal{C}$ si y solo si $H = (-P^T|I_{n-k}) \in \mathbb{F}_q^{(n-k)\times n}$ es una matriz de paridad del código $\mathcal{C}$.*

*Demostración.* Cada palabra del código $\mathcal{C}$ es de la forma $c = mG$ con $m \in \mathbb{F}_q^k$. Supongamos que la matriz generatriz $G$ es sistemática en las primeras $k$ posiciones. Entonces $c = mG = (m, mP)$ con $m \in \mathbb{F}_q^k$, $r = mP \in \mathbb{F}_q^{n-k}$ y se tiene que:

$$-mP + r = 0 \iff -P^T m^T + r^T = 0 \iff (-P^T|I_{n-k})(m,r)^T = 0$$
$$\iff (-P^T|I_{n-k})c^T = 0.$$

Por lo tanto, $(-P^T|I_{n-k})$ es una matriz de paridad del código $\mathcal{C}$. □

*Observación 2.30.* Si la matriz identidad no está al principio, basta con multiplicar $G$ por la matriz permutación correspondiente y luego deshacer la permutación.

*Ejemplo 2.31.* Consideramos $\mathcal{C}$ un $[8,4]_2$ código con la siguiente matriz generatriz:

$$G = \begin{pmatrix} 1\,0\,1\,0\,1\,0\,1\,0 \\ 1\,1\,0\,0\,1\,1\,0\,0 \\ 1\,1\,0\,1\,0\,0\,1\,0 \\ 1\,1\,0\,1\,0\,0\,1\,1 \end{pmatrix} \in \mathbb{F}_2^{8\times 4}$$

Vamos a calcular la matriz escalonada reducida de $G$.

$$G = \begin{pmatrix} 1\,0\,1\,0\,1\,0\,1\,0 \\ 1\,1\,0\,0\,1\,1\,0\,0 \\ 1\,1\,0\,1\,0\,0\,1\,0 \\ 1\,1\,0\,1\,0\,0\,1\,1 \end{pmatrix} \xrightarrow[F_4 = F_4 + F_1]{\substack{F_2 = F_2 + F_1 \\ F_3 = F_3 + F_1}} \begin{pmatrix} 1\,0\,1\,0\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0\,1\,1\,0 \\ 0\,1\,1\,1\,1\,0\,0\,0 \\ 0\,1\,1\,1\,1\,0\,0\,1 \end{pmatrix} \xrightarrow[F_4 = F_4 + F_2]{F_3 = F_3 + F_2}$$

$$\begin{pmatrix} 1\,0\,1\,0\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0\,1\,1\,0 \\ 0\,0\,0\,1\,1\,1\,1\,0 \\ 0\,0\,0\,1\,1\,1\,1\,1 \end{pmatrix} \xrightarrow{F_4 = F_4 + F_3} \begin{pmatrix} 1\,0\,1\,0\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0\,1\,1\,0 \\ 0\,0\,0\,1\,0\,1\,1\,0 \\ 0\,0\,0\,0\,1\,0\,1\,1 \end{pmatrix} = \operatorname{rref}(G).$$

Llamamos $G_1 = \operatorname{rref}(G)$. Observamos que $G_1$ es sistemática en las posiciones $\{1, 2, 4, 8\}$. Construimos ahora la matriz $G_2 = (I_4|P) \in \mathbb{F}_2^{8\times 4}$. Aplicamos para ello la permutación $\pi = (348765)$.
Tenemos:

$$G_2 = \left(\begin{array}{cccc|cccc} 1&0&0&0&1&1&0&1 \\ 0&1&0&0&1&0&1&1 \\ 0&0&1&0&0&1&1&1 \\ 0&0&0&1&0&0&0&0 \end{array}\right) \xrightarrow{\text{Aplicamos Prop. 2.29}} \left(\begin{array}{cccc|cccc} 1&1&0&0&1&0&0&0 \\ 1&0&1&0&0&1&0&0 \\ 0&1&1&0&0&0&1&0 \\ 1&1&1&0&0&0&0&1 \end{array}\right) = H_2 \in \mathbb{F}_2^{8\times 4}$$

Aplicamos ahora la permutación inversa $\pi^{-1} = (356784)$ para obtener la matriz de paridad de $\mathcal{C}$ y tenemos:

$$H = \pi^{-1} H_2 = \begin{pmatrix} 1\,1\,1\,0\,0\,0\,0\,0 \\ 1\,0\,0\,1\,1\,0\,0\,0 \\ 0\,1\,0\,1\,0\,1\,0\,0 \\ 1\,1\,0\,1\,0\,0\,1\,0 \end{pmatrix} \in \mathbb{F}_2^{8\times 4}$$

que es la matriz de paridad del código $\mathcal{C}$.

La siguiente proposición nos permite obtener la distancia mínima de un código lineal $\mathcal{C}$, $d(\mathcal{C})$, a partir de una matriz de paridad $H$ de $\mathcal{C}$.

**Proposición 2.32.** *Sea $H$ una matriz de paridad del código $\mathcal{C}$. Entonces, la distancia mínima del código es el menor entero $d$ tal que $d$ columnas de $H$ son linealmente dependientes.*

*Demostración.* Sean $h_1, \ldots, h_n \in \mathbb{F}_q^{n-k}$ las columnas de $H$. Sea

$$c = (c_1, \ldots, c_n) \in \mathcal{C} \text{ con } c \neq 0 \text{ y } \operatorname{supp}(c) = \{j_1, \ldots, j_w\}$$

es decir $w_H(c) = w$. Entonces, como $Hc^T = 0$, $c_{j_1} h_{j_1} + \cdots + c_{j_w} h_{j_w} = 0$, con $c_{ji} \neq 0$, para $i = 1, \ldots, w$, se tiene que las columnas $h_{j_1}, \ldots, h_{j_w}$ son linealmente

dependientes.
Recíprocamente, si $h_{j_1}, \ldots, h_{j_w}$ son dependientes, entonces existen constantes $a_1, \ldots, a_w$ distintos de cero tal que $a_1 h_{j_1} + \cdots + a_w h_{j_w} = 0$. Sea $c \in \mathcal{C}$ definida como:

$$c_i = \begin{cases} 0 & \text{i} \notin \{j_1, \ldots, j_w\}, \\ a_i & \text{i} \in \{j_1, \ldots, j_w\}. \end{cases}$$

Entonces $Hc^T = 0$. Por lo tanto, $c \in \mathcal{C}$ y $w_H(c) \geq w$.

**Corolario 2.33.** *Se tiene que* $d \leq n - k + 1$.

*Demostración.* Como $\text{rang}(H) = n - k$, por la proposición anterior sabemos que $d$ es el menor entero tal que $d$ columnas de $H$ son linealmente dependientes, con lo cual, se obtiene directamente que $d - 1 \leq n - k$. □

Este corolario aporta una cota a la máxima distancia mínima que puede tener un código lineal si fijamos la longitud $n$ y la dimensión $k$. Esta cota recibe el nombre de *"Cota de Singleton"*.

**Definición 2.34.** *Sea* $\mathcal{C}$ *un* $[n, k, d]_q$ *código lineal. Si se satisface que* $d = n-k+1$ *se dice que* $\mathcal{C}$ *es un código* MDS *(del inglés, Maximum Distance Separable), es decir, un código que tiene la mayor distancia mínima posible.*
*Fijado* $n$ *y* $k$*, un código* MDS *es un código lineal con la mayor capacidad correctora posible.*

### 2.3.3. Código dual

Utilizaremos la siguiente notación para el producto escalar en $\mathbb{F}_q^n$:

$$x \cdot y = x_1 y_1 + \ldots + x_n y_n \text{ para } x, y \in \mathbb{F}_q^n.$$

**Definición 2.35.** *Para un código* $\mathcal{C}$ *de parámetros* $[n, k]_q$*, definimos su código dual* $\mathcal{C}^\perp$*, como:*

$$\mathcal{C}^\perp = \{\text{x} \in \mathbb{F}_q^n : c\text{x} = 0, \forall c \in \mathcal{C}\}.$$

**Proposición 2.36.** *Sea* $\mathcal{C}$ *un código lineal. Entonces:*

1. *$G$ es una matriz generatriz de $\mathcal{C}$ $\iff$ $G$ es una matriz de paridad de $\mathcal{C}^\perp$.*
2. *$H$ es una matriz de paridad del $\mathcal{C}$ $\iff$ $H$ es una matriz generatriz de $\mathcal{C}^\perp$.*

*Demostración.* De la definición de código dual tenemos que las siguientes afirmaciones son equivalentes:

$$x \in \mathcal{C}^\perp \iff c\text{x} = 0, \forall c \in \mathbb{F}_q^n \iff mGx^T = 0, \forall m \in \mathbb{F}_q^k \iff Gx^T = 0$$

Esto implica que $\mathcal{C}^\perp$ es el espacio nulo de $G$. Es decir $G \in \mathbb{F}_q^{k \times n}$ es una matriz de paridad de $\mathcal{C}^\perp$. Esto demuestra (1), la demostración de (2) es análoga. □

*Observación 2.37.* De la proposición anterior se deduce que si $\mathcal{C}$ es un $[n,k]_q$ código lineal, $\mathcal{C}^\perp$ es un $[n,n-k]_q$ código lineal.

**Teorema 2.38.** *Sea $\mathcal{C}$ un $[n,k,d]$-código sobre $\mathbb{F}_q$ con $k \geq 1$. Entonces, las siguientes afirmaciones son equivalentes:*

1. *$\mathcal{C}$ es* MDS.
2. *Cada conjunto de $k$ coordenadas es un conjunto de información en $\mathcal{C}$.*
3. *$\mathcal{C}^\perp$ es* MDS.
4. *Cada conjunto de $n-k$ coordenadas es un conjunto de información en $\mathcal{C}^\perp$.*

*Demostración.* Como la mínima distancia de $\mathcal{C}$ es $d$, cualesquiera $d-1$ columnas de $H$ son linealmente independientes. Ahora, por la cota de Singleton sabemos que $d \leq n-k+1$. Entonces, $d = n-k+1$ si, y solo si, cada conjunto de $n-k$ columnas de $H$ son independientes. Por lo tanto, hemos probado que (1) $\Leftrightarrow$ (4). Si aplicamos el razonamiento anterior a $\mathcal{C}^\perp$ obtenemos que (3) $\Leftrightarrow$ (2).
Ahora, supongamos que (2) es cierto. Fijamos $I = \{i_1, \ldots, i_k\}$ un conjunto de información. Sea $c \in \mathcal{C}$ con $c_i = 0$, $i \in I$.
Entonces $c = mG$ para cierto $m \in \mathbb{F}_q^k$. Por la definición de conjunto de información por la Oservación 2.21, se tiene que $m = 0$. Por lo tanto $c = 0$. Luego, cualquier palabra $c \in \mathcal{C} \setminus \{0\}$ verifica que $w_H(c) \geq n-(k-1)$. Junto con la cota de Singleton se tiene que el código $\mathcal{C}$ es MDS. Por lo tanto (2) $\Longrightarrow$ (1).
Ahora supongamos que $\mathcal{C}$ es MDS. Sea $G$ una matriz generatriz de $\mathcal{C}$ y $\hat{G} \in \mathbb{F}_q^{k\times k}$ una submatriz de $G$ formada por $k$ columnas. Buscamos $x \in \mathbb{F}_q^k$ tal que $x\hat{G} = 0$. Si $x \neq 0$, entonces hemos encontrado una palabra $c = xG$ con peso, a lo sumo, $n-k$, lo que contradice la cota de Singleton. Luego, hemos visto que cualquiera $k$ columnas de $G$ son linealmente independientes. □

### 2.3.4. Código perforado

**Definición 2.39 (Código perforado).** *Sea $\mathcal{C}$ un código lineal con parámetros $[n,k,d]_q$. El proceso de eliminar una o más coordenadas se llama perforar. Definimos el código perforado en la posición $i \in \{1,\ldots,n\}$ como:*

$$\mathcal{C}_i = \{(c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_n) \in \mathbb{F}_q^{n-1} : c = (c_1, \ldots, c_n) \in \mathcal{C}\}.$$

**Proposición 2.40.** *Sea $\mathcal{C}$ un $[n,k,d]_q$ código lineal. Si perforamos el código $\mathcal{C}$ en la posición $i$ se tiene que el código $\mathcal{C}_i$ es un código lineal de parámetros $[n-1,\hat{k},\hat{d}]_q$ con $d-1 \leq \hat{d} \leq d$ y $k-1 \leq \hat{k} \leq k$. Además, si $d > 1$, $\hat{k} = k$.*

*Demostración.* Supongamos que $d(\mathcal{C}) = w_H(c)$ con $c \in \mathcal{C} \setminus \{0\}$. Definimos $\hat{c} = (c_1, \ldots, c_{i-1}, c_{i+1}, \ldots, c_n)$, es decir, eliminamos de $c$ la coordenada $i$. Entonces, si $i \in \operatorname{supp}(c)$ tenemos que $\hat{d} = d-1$. En caso contrario, tenemos que $d = \hat{d}$.

La cota superior de la dimensión está clara. Sea $G$ una matriz generatriz de $\mathcal{C}$ con rango $k$. Definimos $\hat{G}$ la submatriz de $G$ eliminando la columna $i$. Entonces $\hat{G}$ es una matriz generatriz de $\mathcal{C}_i$. Luego:

$$rank(\hat{G}) \geq rank(G) - 1 = k - 1.$$

Concluimos entonces que $\hat{k} \geq k - 1$.

Supongamos ahora que $d > 1$. Sean $c_1, c_2 \in \mathcal{C}$ dos palabras distintas del código. Definimos:

$$\begin{cases} \hat{c}_1 = (c_1^1, \ldots, c_{i-1}^1, c_{i+1}^1, \ldots, c_n^1) \\ \hat{c}_2 = (c_1^2, \ldots, c_{i-1}^2, c_{i+1}^2, \ldots, c_n^2) \end{cases}$$

Como $d > 1$ se tiene que $\hat{c}_1, \hat{c}_2 \in \mathcal{C}_i$ y son palabras distintas en $\mathcal{C}_i$. Luego, $\mathcal{C}$ y $\mathcal{C}_i$ tienen el mismo número de palabras y por lo tanto podemos concluir que $k = \hat{k}$.

□

## 2.4. Problema de decodificación

Supongamos que queremos enviar un mensaje $m \in \mathbb{F}_q^k$ utilizando el $[n,k]_q-$código $\mathcal{C}$. Es decir, sea $G \in \mathbb{F}_q^{k\times n}$ una matriz generatriz de $\mathcal{C}$, enviamos la palabra $c = mG \in \mathbb{F}_q^n$. Esta palabra se transmite a través de un canal (aire, luz, ...) que puede transformar el mensaje.

**Definición 2.41.** *Llamamos decodificador a la aplicación:*

$$\begin{aligned} \mathrm{Dec} : \mathbb{F}_q^n &\longrightarrow \mathbb{F}_q^n \\ y &\longmapsto \mathrm{Dec}(y) = c \end{aligned}$$

El receptor recibe $y = c + e \in \mathbb{F}_q^n$ es decir, la palabra $c$ con algunos errores que se recogen en el vector error $e \in \mathbb{F}_q^n$. El objetivo del proceso de decodificación es recuperar $c \in \mathcal{C}$. Es decir, conseguir una aplicación como la anterior. Se estudian dos tipos de decodificadores:

- **Decodificación por probabilidad maximal:** MLD (Maximum likelihood decoding, en inglés). Este método consiste en averiguar qué palabra del código $c \in \mathcal{C}$ tiene más probabilidad de haber sido enviada tras recibir el mensaje $y = c+e$. Lo que buscamos es minimizar la probabilidad $P_r(\frac{y}{c})$. Esta notación representa la probabilidad de que $y$ sea la palabra recibida condicionada a que $c$ sea la enviada. Es decir, $\mathrm{Dec}(y)$ devuelve $x$ si $x$ maximiza $P_r(\frac{y}{x})$ Se usa la ley de Laplace que establece que la probabilidad de que ocurra un suceso es el cociente entre los casos favorables entre los casos totales.

- **Decodificación por mínimas distancias:** MDD (Minimum distance decoding, en inglés). Este método consiste en, recibido el vector $y \in \mathbb{F}_q^n$, encontrar una palabra $c \in \mathcal{C}$ que sea lo más cercana posible a $y$. Es decir, el problema de decodificación consiste en minimizar $d_H(y, c)$, buscando $c$ entre todas las palabras $c \in \mathcal{C}$. Por lo tanto, en este caso $\mathrm{Dec}(y)$ devuelve $x$ si $x$ minimiza $d_H(x, y)$.

**Definición 2.42 (Canal simétrico).** *Un canal simétrico es aquel canal donde los errores son independientes y con la misma probabilidad p en cada coordenada, con* $0 \leq p \leq \frac{q-1}{q}$*, tal que todos los* $q-1$ *símbolos erroneos ocurren con la misma probabilidad* $\frac{p}{q-1}$*. Entonces, un símbolo es transmitido correctamente con probabilidad* $1-p$*.*

**Proposición 2.43.** *Si tenemos un canal simétrico binario con probabilidad de error* $p \leq \frac{1}{2}$*, ambos problemas de decodificación son equivalentes, es decir,* MLD = MDD*.*

*Demostración.* Haremos la prueba para un caso paticular, canales simétricos binarios con probabilidad de error $p < \frac{1}{2}$. Sea $d_H(x, y) = d$. Entonces

$$P_r\left(\frac{y}{c}\right) = (1-p)^{n-d}p^d = (1-p)^n\left(\frac{p}{1-p}\right)^d$$

Tenemos que se maximiza la probabilidad cuando se minimiza $d$. □

La decodificación es un problema complejo. De hecho, uno de los campos más importantes en el estudio de la teoría de códigos consiste en encontrar técnicas eficientes para realizar el proceso de decodificación. El término de "técnicas eficientes"nos conduce a la siguiente sección.

## 2.5. Complejidad

**Definición 2.44.** *Un algoritmo es un procedimiento computacionalmente bien definido que consisten en una serie de ejecuciones donde se toman una o varias variables que llamamos* ***input*** *y devuelve un resultado como* ***output****.*

**Definición 2.45.** *Una operación aritmética elemental es una suma, comparación o multiplicación de dos elementos* $x$*,* $y \in \{0, 1\}$*.*

Si tenemos un algoritmo $\mathcal{A}$ que tiene como **input** una palabra binaria de tamaño $n$, entonces la complejidad $C_T(\mathcal{A}, n)$ es el número de operaciones elementales que hay que hacer en el algoritmo para obtener el **output** como una función de $n$.

*Ejemplo 2.46.* Sea $\mathcal{C}$ un código lineal de parámetros $[n,k]_2$. Sea $G$ una matriz generatriz de $\mathcal{C}$. Entonces, el proceso de codificación se puede entender como el siguiente algoritmo:

$$(a_1,\cdots,a_k)\longmapsto(a_1,\cdots,a_k)G$$

Para cada ejecución del algoritmo, el **input** es un vector de longitud $k<n$ que representa un mensaje. El **output** es una palabra del código de longitud $n$. Para computar una iteracción, el algoritmo tiene que hacer $k$ multiplicaciones y $k-1$ sumas $n$ veces. La complejidad viene dada entonces por $n(2k-1)\simeq\mathcal{O}(n^2)$.

*Ejemplo 2.47.* En teoría de códigos, la longitud normalmente se toma como parámetro de entrada del algoritmo. Para códigos sobre $\mathbb{F}_q$ de longitud $n$, diremos que las entradas tienen un tamaño de $\lceil n\log_2(q)\rceil$.

*Observación 2.48 (Coste de la eliminación de Gauss).* El método de Gauss o eliminación gaussiana nos permite resolver sistemas lineales o, de forma equivalente, calcular la inversa de una matriz. Este método consiste en transformar un sistema dado $AX=B$ en otro equivalente $UX=C$ en el que la matriz $U$ es una matriz triangular superior. El coste computacional de este método es: multiplicar $n$ elementos de la primera fila para anular $n-1$ elementos (debajo del primer término líder) de la primera columna. Repetir este proceso en toda la matriz. Por lo tanto, el número de productos que tenemos que realizar es:

$$n(n-1)+(n-1)(n-2)+\cdots+2\cdot1=\sum_{i=2}^{n}i(i-1)=\sum_{i=2}^{n}i^2-\sum_{i=2}^{n}i=$$

$$=\frac{n(n+1)(2n+1)}{6}-\sum_{i=2}^{n}i\simeq\mathcal{O}\left(\frac{n^3}{3}\right)\simeq\mathcal{O}(n^3)$$

**Definición 2.49.** *Un problema que tiene como respuestas SÍ o NO es llamado un problema de decisión.*

**Definición 2.50.** *La clase de complejidad* ***P*** *es el conjunto de todos los problemas de decisión que se resuelven en tiempo polinomial, es decir la complejidad es del orden de un polinomio con variable $n$.*
*La clase de complejidad* ***NP*** *es el conjunto de problemas de decisión para los que la respuesta SÍ puede ser verificada en tiempo polinomial, con alguna información extra.*

*Ejemplo 2.51.* Consideramos el problema de decisión que pretende responder a la pregunta ¿$d(\mathcal{C})\leq w$?, donde $\mathcal{C}$ es un código de longitud $n$ y $w$ es un entero positivo. Entonces en el caso de que la respuesta sea SÍ, existirá una palabra del código $c$ de peso menor que $w$. Si nos dan como información extra la palabra $c$, verificar que $w_H(c)\leq w$ tiene complejidad $\mathcal{O}(n)$. Por lo tanto, este problema de decisión es **NP**.

**Definición 2.52.** *Sean $D_1$ y $D_2$ dos problemas computacionales y supongamos que el algoritmo $\mathcal{A}_1$ resuelve $D_1$ y $\mathcal{A}_2$ resuelve $D_2$. Diremos que $D_1$ no es más difícil de resolver que $D_2$ y además ambos pertenecen a la clase de complejidad **P**, denotado como $D_1 \leq_P D_2$, si $\mathcal{A}_1$ utiliza $\mathcal{A}_2$ y $\mathcal{A}_1$ tiene una complejidad polinomial. En este caso diremos que ambos son computacionalmente equivalentes.*

Es natural preguntarse si **P** = **NP**. Muchos expertos piensan que la respuesta es NO, pero no hay pruebas válidas actualmente y se trata de uno de los problemas del milenio. Veremos en la sección 2.5.2 que la decodificación de códigos lineales es un problema un problema **NP**.

### 2.5.1. Algoritmos de decodificación básicos

En las siguientes líneas vamos a explicar las dos primeras ideas de algoritmos de decodificación. Uno de ellos utiliza la matriz generatriz y el otro la matriz de paridad. Antes, veamos cuál es la capacidad de correción de un código lineal.

**Definición 2.53.** *Sea $\mathcal{C}$ un código, diremos que el código es $t-$corrector de errores si para cualesquiera dos palabras del código $c_1$ y $c_2$ y para cualesquiera dos vectores de error $e_1$ y $e_2$ de peso menor o igual que $t$ tenemos que $c_1+e_1 \neq c_2+e_2$.*

**Proposición 2.54.** *Un $[n,k,d]_q-$código es $t-$corrector de errores si, y solo si, $t \leq \left\lfloor \frac{d-1}{2} \right\rfloor$.*

*Demostración.* Supongamos que se verifica que $t < \lfloor \frac{d-1}{2} \rfloor$ y que tenemos $c_1, c_2 \in \mathcal{C}$ dos palabras distintas del código y dos vectores de errores $e_1$ y $e_2$ de peso menor o igual que $t$ tal que $c_1 + e_1 = c_2 + e_2$. Entonces $c_1 - c_2 = e_2 - e_1$ y $w_H(e_2 - e_1) = w_H(c_1 - c_2) \leq 2t < d - 1$ y esto contradice el hecho de que la distancia mínima de $\mathcal{C}$ sea $d$.

Recíprocamente, si procedemos por reducción al absurdo suponiendo que $d(\mathcal{C}) \leq 2t$, existen dos palabras distintas $x, y \in \mathcal{C}$ tal que $d_H(x,y) = d(\mathcal{C}) \leq 2t$. Es decir,

$$d_H(x,y) = \#\{i : x_i \neq y_i\} = \{i_1, \ldots, i_{N_1}\} \cup \{j_1, \ldots, j_{N_2}\} \text{ con } N_1, N_2 \leq t.$$

Por lo tanto, podemos definir $z \in \mathbb{F}_q^n$ como:

$$\begin{cases} z_i = x_i = y_i & \forall i : x_i = y_i, \\ z_i = x_i & \forall i \in \{i_1, \ldots, i_{N_1}\} \\ z_i = y_i & \forall i \in \{j_1, \ldots, j_{N_2}\} \end{cases}$$

tal que $d_H(x,z) = N_2 \leq t$ y $d_H(y,z) = N_1 \leq t$, lo que contradice que $\mathcal{C}$ sea $t$-corrector. □

La idea intuitiva que se nos presenta como primer algoritmo de decodificación es la decodificacion por fuerza bruta.

### Decodificación por fuerza bruta

- **Decodificador:** Supongamos que conocemos $G \in \mathbb{F}_q^{k\times n}$ matriz generatriz del código. Tenemos entonces que $\mathcal{C} = \{xG : x \in \mathbb{F}_q^k\}$.
  En este caso, tenemos el siguiente decodificador:

$$\begin{array}{rcl} \phi : \mathbb{F}_q^n \times \mathbb{F}_q^{k\times n} & \longleftrightarrow & \mathbb{F}_q^k \\ (y, G) & \longleftrightarrow & x \end{array}$$

  que tiene como argumento el mensaje recibido $y$ ($y = xG + e$) y la matriz generatriz del código y devuelve el mensaje $x \in \mathbb{F}_q^k$.

Supongamos que recibimos el vector $y \in \mathbb{F}_q^n$ (que sabemos que puede descomponerse como $y = c + e$ con $c \in \mathcal{C}$ palabra del código enviada y $e$ vector de error). El método de decodificación por fuerza bruta calcula la distancia de Hamming entre $y \in \mathbb{F}_q^n$ y todas las palabras de nuestro código $\mathcal{C}$. Devuelve la palabra $c \in \mathcal{C}$ que minimice la distancia $d_H(y, c)$. La complejidad de este algoritmo es $\mathcal{O}(nq^k)$ ya que el número de operaciones que debemos realizar es comparar $q^k$ palabras de longitud $n$.

### Decodificación por fuerza bruta usando la matriz de paridad

Para esta sección recordamos la definición de síndrome dada en la Definición 2.26. Este algoritmo utiliza la matriz de paridad del código. Variantes de esta idea es lo que estudiaremos en el siguiente capítulo. Es uno de los algoritmos más usados en la investigación de algoritmos eficientes de decodificación de la teoría de códigos.
Conocida la matriz de paridad del código $H \in \mathbb{F}_q^{(n-k)\times n}$. Tenemos que $\mathcal{C} = \{c \in \mathbb{F}_q^n : Hc^T = 0\}$.
Un decodificador genérico que utiliza el síndrome del vector recibido es:

$$\begin{array}{rcl} \psi : \mathbb{F}_q^{n-k} \times \mathbb{F}_q^{n-k\times n} & \longleftrightarrow & \mathbb{F}_q^n \\ (s, H) & \longleftrightarrow & e \end{array}$$

toma como argumento un síndrome y la matriz de paridad del código. La idea de este algoritmo es devolver el error con menor peso que tenga el mismo síndrome que la palabra recibida.
En esta sección presentamos un algoritmo que permite pre-cálculo de una tabla que una vez realizada permite decodificar de forma eficiente varios vectores recibidos. Aún con este pre-cálculo el algoritmo de decodificación sigue siendo costoso por el tamaño de la tabla.
Supongamos ahora que tenemos $H$ matriz de paridad del código $\mathcal{C}$, teniendo en cuenta la definición de síndrome, podemos establecer una relación de equivalencia en $\mathcal{C}$: sea $x, y \in \mathbb{F}_q^n$,

$$x \sim y \iff S(x) = S(y) \tag{2.1}$$

Suponemos $a + \mathcal{C}$ clase de equivalencia de $a \in \mathbb{F}_q^n$. Entonces

$$x, y \in a + \mathcal{C} \iff Hx^T = Hy^T.$$

*Observación 2.55.* Con esta clase de equivalencia podemos particionar $\mathbb{F}_q^n$ en $q^{n-k}$ partes diferentes donde cada clase tiene $q^k$ elementos tal y como se demuestra en el siguiente resultado.

**Teorema 2.56 (Teorema de Lagrange).** *Sea $\mathcal{C}$ un $[n,k]_q-$código. Consideramos la clase de equivalencia definida en Ecuación (2.1) y tenemos que:*

1. *Cada clase de equivalencia tiene exactamente $q^k$ elementos.*
2. *Dos clases de equivalencia o son iguales o son disjuntas.*

*Demostración.* 1. Consideramos la aplicación:

$$\begin{aligned} f : \mathbb{F}_q^n &\longrightarrow \mathbb{F}_q^n \\ x &\longmapsto x + a \end{aligned}$$

con $a \notin \mathcal{C}$. Entonces $\#(a + \mathcal{C}) = \#\mathcal{C} = q^k$.

2. Supongamos que existe $v \in (a+\mathcal{C}) \cap (b+\mathcal{C})$, esto es, $v = a + c_1 = b + c_2$ con $c_1$ y $c_2 \in \mathcal{C}$. Entonces $b = a + (c_1 - c_2) \in (a + \mathcal{C})$. Por lo tanto, tenemos un contenido. De la misma manera se demuestra el otro contenido, $a+\mathcal{C} \subseteq b+\mathcal{C}$. Con lo cual, tenemos que $a + \mathcal{C} = b + \mathcal{C}$.

□

**Definición 2.57.** *Llamamos **elementos líderes** a las palabras que tienen menor peso de Hamming en cada clase de equivalencia. Al peso del elemento líder lo llamamos **peso de la clase de equivalencia**.*

*Observación 2.58.* El elemento líder no tiene por qué ser único. Si hubiera más de uno, podemos tomar cualquiera de ellos.

**Proposición 2.59.** *Toda clase de equivalencia con peso $w_H \leq t = \lfloor \frac{d-1}{2} \rfloor$ tiene un único elemento líder, siendo $d = d(\mathcal{C})$.*

*Demostración.* Procedemos por reducción al absurdo. Supongamos que existen dos elementos $a, b \in x + \mathcal{C}$, es decir, $a = x + c_1$ con $c_1 \in \mathcal{C}$ y $b = x + c_2$ con $c_2 \in \mathcal{C}$, tales que $w_H(a) \leq t$ y $w_H(b) \leq t$. Entonces:
$d_H(c_1, c_2) = d_H(a-x, b-x) = w_H(a-b) \leq w_H(a)+w_H(b) \leq 2t = 2\left(\frac{d-1}{2}\right) = d-1$
lo que contradice la minimalidad de $d$. □

Supongamos $y = c + e$ es una palabra recibida con $c \in \mathcal{C}$ y $e$ vector de error. El algoritmo que se lleva a cabo en este proceso de decodificación es el siguiente:

**Algoritmo 3:** Algoritmo de decodificación usando pre-cálculo.

**Input:** Los parámetros del código $\mathcal{C}$, $[n, k, d]$ en el que estamos trabajando, la palabra recibida $y \in \mathbb{F}_q^n$ y una matriz de paridad $H$.

**Output:** La palabra enviada $c$ del código.

**PASO 1**: Realizamos una tabla donde aparecen todos los posibles síndromes del código y los elementos líder de cada clase de equivalencia.

**PASO 2**: Tomamos la palabra recibida y calculamos su síndrome.

**PASO 3**: Buscamos en la tabla el elemento líder $e \in \mathbb{F}_q^n : S(y) = S(e)$.

**PASO 4**: Devolver el vector $e$. Decodificamos $y \in \mathbb{F}_q^n$ sabiendo que $y - e \in \mathcal{C}$.

**Complejidad**: La complejidad de este método de decodificación es $\mathcal{O}((n-q)q^{n-k})$ ya que tenemos $q^{n-k}$ síndromes, es decir, vectores de longitud $n-k$. Veamos en las siguientes líneas un ejemplo de decodificación por síndrome usando precálculo.

*Ejemplo 2.60.* Supongamos que tenemos un $[6, 3, 3]_2-$código lineal con matriz generatriz y de paridad respectivamente:

$$G = \begin{pmatrix} 1\,0\,0\,0\,1\,1 \\ 0\,1\,0\,1\,0\,1 \\ 0\,0\,1\,1\,1\,0 \end{pmatrix} \in \mathbb{F}_2^{3\times 6} \quad \text{y} \quad H = \begin{pmatrix} 0\,1\,1\,1\,0\,0 \\ 1\,0\,1\,0\,1\,0 \\ 1\,1\,0\,0\,0\,1 \end{pmatrix} \in \mathbb{F}_2^{3\times 6}$$

Construimos la tabla de síndromes, que puede verse en la Tabla 2.1.
Ahora, supongamos que recibimos la palabra $v = (1, 1, 1, 1, 0, 1)$. Sabemos que $v = c + e$ donde $c$ es una palabra del código y $e$ es el error que se ha producido. Calculamos el síndrome de $v$: $eH^T = vH^T = (1, 0, 1)$. De la tabla se deduce que el menor error con dicho síndrome es $e = (0, 1, 0, 0, 0, 0)$. Luego, podemos concluir que:

$$c = v - e = (1, 1, 1, 1, 0, 1) - (0, 1, 0, 0, 0, 0) = (1, 0, 1, 1, 0, 1).$$

Supongamos ahora que recibimos la palabra $v = c + e = (1, 0, 0, 1, 0, 0)$. Calculamos el síndrome de $e$: $eH^T = vH^T = (1, 1, 1)$.
De la tabla se deduce que el líder de la clase de todos los elementos que tienen el mismo síndrome no es único. Por lo tanto, tenemos tres errores posibles con el mismo síndrome, que son:

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

| Síndrome | Elementos de la clase | Elemento líder |
|---|---|---|
| $(0,0,0)$ | $(0,0,0,0,0,0)$, $(0,1,1,0,1,1)$, $(1,1,1,0,0,0)$, $(1,0,1,1,0,1)$, $(1,1,0,1,1,0)$, $(1,0,0,0,1,1)$, $(0,0,1,1,1,0)$, $(0,1,0,1,0,1)$ | (0,0,0,0,0,0) |
| $(0,0,1)$ | $(0,0,0,1,0,0)$, $(1,1,1,1,0,0)$, $(1,1,0,0,1,0)$, $(0,0,1,0,1,0)$, $(0,1,0,0,0,1)$, $(1,0,1,0,0,1)$, $(1,0,0,1,1,1)$, $(0,1,1,1,1,1)$ | (0,0,0,1,0,0) |
| $(0,1,0)$ | $(0,1,0,0,0,0)$, $(1,0,1,0,0,0)$, $(1,0,0,1,1,0)$, $(0,1,1,1,1,0)$, $(0,0,0,1,0,1)$, $(1,1,1,1,0,1)$, $(1,1,0,0,1,1)$, $(0,0,1,0,1,1)$ | (0,1,0,0,0,0) |
| $(0,1,1)$ | $(0,1,0,1,0,0)$, $(1,0,1,1,0,0)$, $(1,0,0,0,1,0)$, $(0,1,1,0,1,0)$, $(0,0,0,0,0,1)$, $(1,1,1,0,0,1)$, $(1,1,0,1,1,1)$, $(0,0,1,1,1,1)$ | (0,0,0,0,0,1) |
| $(1,0,0)$ | $(1,0,0,0,0,0)$, $(0,1,1,0,0,0)$, $(0,1,0,1,1,0)$, $(1,0,1,1,1,0)$, $(1,1,0,1,0,1)$, $(0,0,1,1,0,1)$, $(0,0,0,0,1,1)$, $(1,1,1,0,1,1)$ | (1,0,0,0,0,0) |
| $(1,0,1)$ | $(1,0,0,1,0,0)$, $(0,1,1,1,0,0)$, $(0,1,0,0,1,0)$, $(1,0,1,0,1,0)$, $(1,1,0,0,0,1)$, $(0,0,1,0,0,1)$, $(0,0,0,1,1,1)$, $(1,1,1,1,1,1)$ | (1,0,0,1,0,0), (0,1,0,0,1,0),(0,0,1,0,0,1) |
| $(1,1,0)$ | $(1,1,0,0,0,0)$, $(0,0,1,0,0,0)$, $(0,0,0,1,1,0)$, $(1,1,1,1,1,0)$, $(1,0,0,1,0,1)$, $(0,1,1,1,0,1)$, $(0,1,0,0,1,1)$, $(1,0,1,0,1,1)$ | (0,0,1,0,0,0) |
| $(1,1,1)$ | $(1,1,0,1,0,0)$, $(0,0,1,1,0,0)$, $(0,0,0,0,1,0)$, $(1,1,1,0,1,0)$, $(1,0,0,0,0,1)$, $(0,1,1,0,0,1)$, $(0,1,0,1,1,1)$, $(1,0,1,1,1,1)$ | (0,0,0,0,1,0) |

Tabla 2.1: Tabla precalculada del Ejemplo 2.60.

La mínima distancia de este código es $d = 3$. Por la Proposición 2.54 sabemos que somos capaces de corregir $t = \lfloor \frac{d-1}{2} \rfloor = 1$ errores. Por tanto, no podemos asegurar decodificar bien porque el error en este caso tiene peso 2. Con lo cual, podemos dar tres posibles palabras enviadas:

- Si tomamos como error $e_1 = (1,0,0,1,0,0)$ tenemos que la palabra enviada fue $c = v - e = (1,1,1,1,0,1) - (1,0,0,1,0,0) = (0,1,1,0,0,1)$.
- Si consideramos $e_2 = (0,1,0,0,1,0)$, tomaremos como palabra enviada $c = v - e = (1,1,1,1,0,1) - (0,1,0,0,1,0) = (1,0,1,1,1,1)$.
- Tomando $e_3 = (0,0,1,0,0,1)$, tomamos la palabra enviada $c = v - e = (1,1,0,1,0,0)$.

### 2.5.2. El problema de decodificación de códigos lineales es NP

El problema que intentamos resolver es el siguiente:

**Problema de decodificación**:
Dada una matriz de paridad $H \in \mathbb{F}_2^{(n-k)\times n}$, un vector $s \in \mathbb{F}_2^{n-k}$ y un entero $w > 0$, encontrar un vector $x \in \mathbb{F}_2^n$ que satisfaga que $Hx^T = s$.

Veamos que el problema de decisión anterior es un problema **NP**. Probaremos esta propiedad comprobando que el problema anterior es computacionalmente

equivalente al problema de emparejamiento 3-dimensional, que se conoce que es un problema **NP**.

**Problema de emparejamiento 3-dimensional**:
Dado un conjunto $T \subseteq S_1 \times S_2 \times S_3$ donde $S_1$, $S_2$ y $S_3$ son conjuntos finitos de tamaño $a$, encontrar un subconjunto $U \subseteq T$ con $\#U = a$ de forma que $d_H(u_1, u_2) = 3$ para todo $u_1, u_2 \in U$.

Al problema anterior se le puede asociar una matriz $\mathcal{M}$ de la siguiente forma:

1. Elegimos un orden en $T = \{t_1, \ldots, t_N\}$ donde $t_i = (t_{i1}, t_{i2}, t_{i3}) \in S_1 \times S_2 \times S_3$ y $N = \#T$.
2. Elegimos un orden en los conjuntos $S_j = \{b_{j1}, \ldots, b_{ja}\}$ con $j = 1, 2, 3$.
3. Definimos $e_i$ como el vector de longitud $a$ unitario que tiene un 1 en la posición $i$ y cero en el resto.
   Nos construimos la matriz $\mathcal{M}$ de tamaño $N \times 3a$ donde cada fila $m_i \in \mathbb{F}_2^{3a}$ representa la tripleta $t_i = (t_{i1}, t_{i2}, t_{i3}) \in T$ de la siguiente forma:

   $$m_i = (e_x, e_y, e_z) \text{ con } t_{i1} = b_{1x} \text{ , } t_{i2} = b_{2y} \text{ y } t_{i3} = b_{3z}$$

   El problema de emparejamiento 3-dimensional tiene solución si existen $a$ filas de la matriz $\mathcal{M}$ tal que su suma mod 2 sea el vector todo 1. Es decir, existe un vector $x \in \mathbb{F}_2^N$ con $w_H(x) = a$ tal que $x\mathcal{M} = (1, \ldots, 1) \in \mathbb{F}_2^{3a}$.

**Proposición 2.61.** *El problema de decisión de decodificar códigos lineales es un problema **NP**.*

*Demostración.* El problema de emparejamiento 3-dimensional es un problema **NP**. Tal y como hemos descrito el problema de emparejamiento 3-dimensional en las líneas anteriores, se deduce que si existe un algoritmo polinomial que resuelve el problema de decisión de decodificar códigos lineales en tiempo polinomial, este mismo algoritmo serviría para resolver el problema de emparejamiento 3-dimensional. □

## 2.6. Otras cotas definidas para códigos lineales.

### 2.6.1. Los conjuntos $A_q(n, d)$ y $B_q(n, d)$

La mínima distancia $d$ mide la capacidad correctora de un código. Para una longitud dada y un número de palabras fijado, un problema fundamental en teoría de códigos es construir un código con la máxima distancia mínima posible. Alternativamente, dada la longitud $n$ y como cota inferior de la distancia mínima un entero $d$, nos gustaría encontrar un código con el máximo número posible de palabras. Vamos a estudiar en esta sección tanto códigos lineales como códigos no lineales.

- Denotamos por $A_q(n,d)$ al máximo número de palabras que puede contener un código en $\mathbb{F}_q$ de longitud $n$ y distancia mínima, al menos, $d$. Denotaremos por $a_q(n,d) = \log_q(A_q(n,d))$.
- De forma similar, para códigos no lineales, denotamos como $B_q(n,d)$ al mayor número de palabras que puede contener un código lineal en $\mathbb{F}_q^n$ de longitud $n$ y distancia mínima, al menos $d$. Denotaremos por $b_q(n,d) = \log_q(B_q(n,d))$.

**Definición 2.62.** *Un código de longitud $n$ y distancia mínima $d$ sobre $\mathbb{F}_q$ se dice* ***óptimo*** *si tiene $A_q(n,d)$ palabras (o $B_q(n,d)$ en el caso de ser lineal).*

*Observación 2.63.* Se tiene que $B_q(n,d) \leq A_q(n,d)$ (ya que si tenemos el conjunto de todos los códigos bloque, los lineales siempre serán un subconjunto propio de ellos). Además, como $B_q(n,d)$ es potencia de $q$ (por la observación 2.9), aplicando propiedades de logaritmos sabemos que $b_q(n,d)$ es un entero no negativo.

**Teorema 2.64.** $A_q(n,n) = B_q(n,n) = q$.

*Demostración.* El código lineal sobre $\mathbb{F}_q$ que consta de todos los múltiplos del vector de uno de longitud $n$ tiene mínima distancia $n$. Luego, por la Observación 2.63 tenemos que $A_q(n,n) \geq B_q(n,n) \geq q$. Si $A_q(n,n) > q$, existe un código con más de $q$ palabras y mínima distancia $n$. Por lo tanto, al menos dos palabras coinciden en una coordenada, lo que implicaría que tienen distancia menor que $n$, lo cual es una contradiccón. Luego $A_q(n,n) = B_q(n,n) = q$. □

**Teorema 2.65.** $A_q(n,d) \leq qA_q(n-1,d)$ *y* $B_q(n,d) \leq qB_q(n-1,d)$.

*Demostración.* Vamos a estudiar primero el caso de códigos lineales. Sea $\mathcal{C}$ un $[n,k,d]_q$ código lineal, entonces $\#\mathcal{C} = q^k = M$. Consideramos el subcódigo $\mathcal{C}(0) \subseteq \mathcal{C}$ formado por todas las palabras de $\mathcal{C}$ que tienen un cero en la primer coordenada, es decir:

$$\mathcal{C}(0) = \{c \in \mathcal{C} : c = (0, c_2, \ldots, c_n)\}.$$

Es fácil comprobar que si $G = \begin{pmatrix} g_1 \\ \vdots \\ g_k \end{pmatrix}$ es una matriz generatriz del código $\mathcal{C}$, entonces aplicando eliminación gaussiana en la primera columna a esta matriz tenemos una matriz

$$G' = \left(\begin{array}{c|c} 1 & g_1' \\ \hline \bar{0} & \bar{G} \end{array}\right) \text{ con } \bar{0} \in \mathbb{F}_q^{(k-1)\times 1}, \bar{G} \in \mathbb{F}_q^{(k-1)\times(n-1)} \text{ y } g_1' \in \mathbb{F}_q^{1\times(n-1)}.$$

tal que $(0|\bar{G})$ es una matriz generatriz de $\mathcal{C}(0)$ y, por lo tanto, $\#\mathcal{C}(0) = q^{k-1}$. Por la Proposición 2.40, si perforamos $\mathcal{C}(0)$ en la primera coordenada obtenemos

un código de longitud $n-1$, dimensión $k-1$ y distancia mínima $\hat{d} \geq d-1$. Si este código no tiene distancia mínima $d$, es decir $\min(w_H(\mathcal{C})) = w_H(g_1)$, repitiendo este proceso en otra coordenada distinta del cero que no pertenezca al soporte de $g_1$ debemos obtener un código lineal de longitud $n-1$, dimensión $k-1$ y distancia mínima $d$.
Por lo tanto $q^{k-1} \leq B_q(n-1,d)$ y esto implica que $B_q(n,d) \leq qB_q(n-1,d)$.
Un proceso similar se puede realizar para códigos no lineales. Sea $\mathcal{C}$ un código óptimo de longitud $n$ y distancia mínima $d$ sobre $\mathbb{F}_q$. Es decir, $\mathcal{C}$ tiene $M = A_q(n,d)$ palabras. Definimos $\mathcal{C}(\alpha)$ como el conjunto de palabras de $\mathcal{C}$ que empiezan por $\alpha$. Esto es:

$$\mathcal{C}(\alpha) = \{c \in \mathcal{C} : c = (\alpha, c_2, \ldots, c_n)\}.$$

Entonces existe algún $\alpha \in \mathbb{F}_q$ tal que $\#\mathcal{C}(\alpha) \geq \frac{M}{q}$ ya que si para todo $\alpha \in \mathbb{F}_q$, $\#\mathcal{C}(\alpha) < \frac{M}{q}$ tenemos que

$$\sum_{\alpha \in \mathbb{F}_q} \#\mathcal{C}(\alpha) = \#\mathcal{C}(\alpha_1) + \#\mathcal{C}(\alpha_2) + \cdots + \#\mathcal{C}(\alpha_q) < q\frac{M}{q} = M$$

y esto es una contradicción ya que $\cup_{\alpha \in \mathbb{F}_q} \mathcal{C}(\alpha) = \mathcal{C}$.
Por lo tanto, si perforamos $\mathcal{C}(\alpha)$ en la primera coordenada conseguimos un código con longitud $n-1$ y distancia mínima al menos $d$. Para estudiar la distancia mínima del código perforado estudiamos tres casos:

1. Supongamos que $d(\mathcal{C}) = d_H(c_1, c_2)$ con $c_1, c_2 \in \mathcal{C}(\alpha)$. Entonces $d(\mathcal{C}) = d_H(\hat{c}_1, \hat{c}_2) = d$ con $c_1 = (\alpha, \hat{c}_1)$ y $c_2 = (\alpha, \hat{c}_2)$.
2. Si $c_1, c_2 \notin \mathcal{C}(\alpha)$ entonces la distancia mínima del código perforado es $d$.
3. Si $d_H(c_1, c_2) = d(\mathcal{C})$ con $c_1 \in \mathcal{C}(\alpha)$ y $c_2 \notin \mathcal{C}(\alpha)$, entonces $d_H(\mathcal{C}(\alpha)) \leq d(\mathcal{C})$. En este caso, repetimos el proceso en otra coordenada.

Así, tenemos que $\frac{M}{q} \leq A_q(n-1,d)$. Luego, $A_q(n,d) = M \leq qA_q(n-1,d)$. □

**Lema 2.66.** *Sea $\mathcal{C}$ un $[n,k]_q-$código, entonces el número de palabras de peso menor o igual que $t = \lfloor \frac{d-1}{2} \rfloor$ donde $d = d(\mathcal{C})$ es:*

$$1 + (q-1)\binom{n}{1} + (q-1)^2\binom{n}{2} + \cdots + (q-1)^t\binom{n}{t}.$$

*Demostración.* Es claro que la única palabra $c$ del código $\mathcal{C}$ de peso 0 es aquella que tiene 0 en todas las posiciones del vector.
Ahora, si queremos contar el número de palabras de peso 1 tenemos que tener en cuenta que: existen $(q-1)$ elementos distintos de cero en $\mathbb{F}_q$ y hay $\binom{n}{1}$ posiciones para elegir un elemento distinto de cero en un vector de longitud $n$. Luego, tenemos que, existen $(q-1)\binom{n}{1}$ palabras de peso 1.
Si ahora queremos saber cuántas palabras hay de peso 2 tenemos que tener en cuenta que:

- Existen $(q-1)$ elementos distintos de cero.
- Hay $\binom{n}{2}$ opciones para elegir dos posiciones en un vector de longitud $n$.
- Hay $(q-1)$ valores diferentes de cero para la primera y segunda coordenada.

Luego, tenemos que existen $(q-1)^2\binom{n}{2}$ palabras de peso 2. Procedemos de la misma forma hasta obtener que el número de palabras de peso exactamente $t$ es $(q-1)^t\binom{n}{t}$. Con lo cual, podemos concluir que el número de palabras de peso menor o igual que $t$ es:

$$1+(q-1)\binom{n}{1}+(q-1)^2\binom{n}{2}+\cdots+(q-1)^t\binom{n}{t}.$$

□

### 2.6.2. Cota de Singleton

Para códigos lineales, por el Corolario 2.33 sabemos que $d \leq n-k+1$. Como consecuencia de esto, tenemos la siguiente proposición:

**Proposición 2.67.** *Para $d \leq n$, se tiene que $B_q(n,d) = q^k \leq q^{n-d+1}$.*

*Demostración.* Sea $\mathcal{C}$ un código lineal con parámetros $[n,k,d]$. Sabemos por el Corolario 2.33 que $d \leq n-k+1$ y por la Observación 2.9 que $B_q(n,d) = q^k$. Luego, obtenemos que $B_q(n,d) = q^k \leq q^{n-d+1}$. □

Ahora, para el caso no lineal tenemos la siguiente proposición:

**Proposición 2.68.** *Para $d \leq n$, se tiene que $A_q(n,d) \leq q^{n-d+1}$.*

*Demostración.* Sabemos por la Proposición 2.64 que $A_q(n,n) = q$. Asumimos que $d < n$ y tenemos, por el Teorema 2.65 que $A_q(n,d) \leq qA_q(n-1,d)$. Si seguimos aplicando el teorema inductivamente tenemos que $A_q(n,d) \leq q^{n-d}A_q(d,d)$. Como $A_q(d,d) = q$, se tiene que $A_q(n,d) = q^{n-d+1}$. □

### 2.6.3. Empaque de esferas, radio de cobertura y códigos perfectos

**Definición 2.69.** *Definimos las esferas de radio $t$ alrededor de un vector $x \in \mathbb{F}_q^n$ prefijado como:*

$$V_q(x,n,t) = \{y \in \mathbb{F}_q^n : d_H(x,y) \leq t\}$$

Cuando no es necesario fijar el vector $x$, utilizaremos la notación $V_q(n,d)$ para referirnos a la esfera de radio $t$ centradas en cualquier $x \in \mathbb{F}_q^n$.

**Proposición 2.70.** *Se tiene que $\#V_q(n,t) = \sum_{i=0}^{t}\binom{n}{i}(q-1)^i$.*

*Demostración.* La demostración es similar a la dada en el Lema 2.66. Sea $I = \{j : x_j \neq y_j\}$ un subconjunto de índices con $\#I = i$. Ahora, se tiene que las posibilidades de elegir ese conjunto $I$ en $\{1, \ldots, n\}$ es $\binom{n}{i}$ y, además, el número de vectores $y \in \mathbb{F}_q^n$ con $x_j \neq y_j$, para todo $j \in I$ es $(q-1)^i$. Luego, con esto queda demostrado que $\#V_q(n,t) = \sum_{i=0}^{t} \binom{n}{i}(q-1)^i$.

**Definición 2.71.** *El **radio de cobertura** $\rho(\mathcal{C})$ de un código $\mathcal{C}$ de longitud $n$ sobre $\mathbb{F}_q$ es definido como el entero $t$ más pequeño tal que:*

$$\bigcup_{c \in \mathcal{C}} \#V_q(c, n, t) = \mathbb{F}_q^n.$$

***Definición 2.72.*** *Un código con radio de recubrimiento $\rho(\mathcal{C})$ es perfecto si $V_q(c, n, \rho(c))$ con $c \in \mathcal{C}$ son disjuntas.*

### 2.6.4. Cota de Hamming

**Teorema 2.73 (Cota de Hamming).** *Si consideramos un código $\mathcal{C}$ de longitud $n$ y distancia mínima, al menos, $d$. Se tiene que:*

$$B_q(n,d) \leq A_q(n,d) \leq \frac{q^n}{\#V_q(n,t)}, \text{ con } t = \left\lfloor \frac{d-1}{2} \right\rfloor$$

*Demostración.* Sea $\mathcal{C}$ un código óptimo sobre $\mathbb{F}_q$ (puede que no sea lineal) de longitud $n$ y distancia mínima $d$. Suponemos que $\mathcal{C}$ tiene $M = A_q(n,d)$ palabras. Como la distancia entre dos palabras cualesquiera es mayor o igual que $d \geq 2t+1$, las esferas de radio $t$ centradas en $x$ son todas disjuntas. Además hay $\alpha = \sum_{i=0}^{t} \binom{n}{i}(q-1)^i$ vectores en cualquiera de estas esferas. Luego $M\alpha$ no puede exceder $q^n$ (que es el número total de vectores de $\mathbb{F}_q^n$). Así, se tiene que $M = A_q(n,d) \leq \frac{q^n}{\sum_{i=0}^{t}\binom{n}{i}(q-1)^i} = \frac{q^n}{\#V_q(n,t)}$. □

**Teorema 2.74.** *Sea $C$ un $[n,k]_q-$código, entonces el número de vectores de peso menor o igual que $t = \lfloor \frac{d-1}{2} \rfloor$, con $d = d(\mathcal{C})$ es menor o igual que $q^{n-k}$. Esto es:*

$$\sum_{j=0}^{t} \binom{n}{j}(q-1)^j \leq q^{n-k}.$$

*Demostración.* Sabemos por el Lema 2.66 que el número de elementos de peso menor o igual que $t$ es:

$$\sum_{j=0}^{t} \binom{n}{j}(q-1)^j.$$

Además, el número total de clases de equivalencia es $q^{n-k}$. Ahora, todos los vectores de peso menor o igual que $t$ son el único líder de su clase de equivalencia. Luego, el número de vectores de peso menor o igual que $t$ es menor o igual que el número total de clases de equivalencia por la Proposición 2.59. □

### 2.6.5. Cota de Gilbert-Varshamov

**Proposición 2.75 (Cota de Gilbert).** $a_q(n,d) \geq n - \log_q(V_q(n,d-1))$

*Demostración.* Sea $\mathcal{C}$ un código óptimo (no necesariamente lineal) con longitud $n$ y distancia mínima $d$ y $M = A_q(n,d)$ el número de palabras de $\mathcal{C}$.
Si $M\#V_q(n,d-1) < q^n$, entonces las esferas de radio $d-1$ centradas en las palabras de $\mathcal{C}$ no recubren todo $\mathbb{F}_q^n$. Sea $x \in \mathbb{F}_q^n$ un elemento que no está en la unión de todas las esferas centradas en una palabra de $\mathcal{C}$ de radio $d-1$. Es decir, $d(x,c) \geq d$, para toda palabra $c \in \mathcal{C}$. Tenemos que $\mathcal{C} \cup \{x\}$ es un código con longitud $n$, número de palabras $M+1$ y distancia mínima $d$. Pero esto es absurdo ya que contradice la maximalidad de $A_q(n,d)$.
Por lo tanto, se tiene que $A_q(n,d)\#V_q(n,d-1) \geq q^n$. Aplicamos logaritmos a esta ecuación y tenemos que: $a_q(n,d) \geq n - \log_q(V_q(n,d-1))$. □

**Proposición 2.76 (Cota de Varshamov).** $b_q(n,d) \geq n - \lceil \log_q(V_q(n,d-1)) \rceil$

*Demostración.* Sea $\mathcal{C}$ un código lineal con parámetros $[n,k,d]$, tenemos que el número de palabras es $q^k$. Suponemos ahora que $q^k\#V_q(n,d-1) < q^n$. Con lo cual, se tiene que existe un vector $x \in \mathbb{F}_q^n \setminus \mathcal{C}$ tal que $d(x,c) \geq d$, para toda palabra $c \in \mathcal{C}$.
Definimos ahora el código lineal $\hat{\mathcal{C}} = \mathcal{C} + \langle x \rangle$ y veamos a continuación que $\hat{\mathcal{C}}$ es efectivamente un código lineal.
Consideramos $G = \begin{pmatrix} g_1 \\ \vdots \\ g_k \end{pmatrix}$ matriz generatriz de $\mathcal{C}$ donde $g_i \in \mathbb{F}_q^n$. Tenemos que:

$$\forall c \in \mathcal{C}, c = \lambda_1 g_1 + \cdots + \lambda_k g_k$$

Ahora, toda palabra $\hat{c} \in \mathcal{C} + \langle x \rangle$, se tiene que $\hat{c} = \lambda_1 g_1 + \cdots + \lambda_k g_k + \beta x$. Como $x$ no pertenece al código, tenemos que $x$ es independiente de $(g_1, \ldots, g_k)$. Con lo cual, se deduce que $\dim(\hat{\mathcal{C}}) = k+1$.
Ahora, procedemos por reducción al absurdo suponiendo que, para cualquiera dos palabras $c_1$ y $c_2 \in \hat{\mathcal{C}}$ se tiene que $d(c_1,c_2) < d$.
Escribimos:

$$c_1 = \lambda_1 g_1 + \cdots + \lambda_k g_k + \beta_1 x$$

$$c_2 = \alpha_1 g_1 + \cdots + \alpha_k g_k + \beta_2 x$$

Como $w_H(c_1 - c_2) < d$ tenemos que $d_H(c, \langle x \rangle) < d$, lo que contradice nuestra suposición. Con lo cual, podemos concluir que $\hat{\mathcal{C}}$ es un código de distancia mínima $d$, longitud $n$ y dimensión $k+1$, pero esto es absurdo porque contradice la maximalidad de $B_q(n,d)$. Aplicando logaritmos a la ecuación anterior se tiene que: $b_q(n,d) = \log_q(B_q(n,d)) > k$. Por tanto, $b_q(n,d) \geq n - \lceil \log_q(\#V_q(n,d-1)) \rceil$.
□

A continuación, enunciamos una proposición que explica cómo construir un código $[n,k]_q$ con distancia mínima $d$.

**Proposición 2.77.** *Si se verifica que:*

$$\sum_{i=0}^{d-2}(q-1)^i\binom{n-1}{i}<q^{n-k}$$

*Existe un $[n,k,d]_q$ código lineal, que denotamos como $\mathcal{C}$.*

*Demostración.* Sea $H$ una matriz de paridad del código $\mathcal{C}$. Sabemos que cada $d-1$ columnas de $H$ son linealmente independientes. Construimos por inducción las columnas $h_1,\ldots,h_n\in\mathbb{F}_q^{n-k}$ de $H$. Elegimos: $h_1\in\mathbb{F}_q^{n-k}$ cualquier vector distinto de cero, $h_2\in\mathbb{F}_q^{n-k}$ cualquier vector que no sea múltiplo de $h_1$. Así sucesivamente, elegimos $h_{j-1}\in\mathbb{F}_q^{n-k}$ cualquier vector que no sea múltiplo de ninguna combinación lineal de a lo sumo $d-2$ vectores de $\{h_1,\ldots,h_{j-2}\}$. Tenemos que:

$$\sum_{i=0}^{d-2}(q-1)^i\binom{n-1}{i}$$

es el número de combinaciones lineales de, a lo sumo, $d-2$ vectores en el conjunto $\{h_1,\ldots,h_{j-2}\}$. Además, $q^{n-k}$ es el número de vectores de $\mathbb{F}_q^{n-k}$.
Ahora, tenemos que podremos definir el vector $h_j$ de la misma forma anterior si:

$$\sum_{i=0}^{d-2}(q-1)^i\binom{n-1}{i}<q^{n-k}-1.$$

□

**Definición 2.78.** *La **distancia de Gilbert-Varshamov** de un $[n,k]_q$-código lineal es el mayor entero $d_0$ que verifica que:*

$$\sum_{i=0}^{d_0-1}\binom{n}{i}(q-1)^i\leq q^{n-k}.$$

### 2.6.6. Cotas asintóticas

Estudiaremos ahora algunas de las cotas que ya han sido definidas anteriormente cuando la longitud del código tiende a infinito. Las cotas resultantes se llamarán **cotas asintóticas**. Primero, necesitamos introducir dos nuevos conceptos:

**Definición 2.79.** *Sea $\mathcal{C}$ un código (no necesariamente lineal) sobre $\mathbb{F}_q$ con $M$ palabras. La **tasa de información** del código viene dada por $\frac{log_q(M)}{n}=R(\mathcal{C})$.*

Si el código fuera lineal, contiene $M = q^k$ palabras y $R(\mathcal{C}) = \frac{log_q(M)}{n} = \frac{k}{n}$. En el caso de códigos lineales, la tasa de información es una medida del número de coordenadas de información relativo al total de coordenadas. Cuanto mayor sea la tasa de información, mayor será la proporción de palabras del código que contienen información relevante.

**Definición 2.80.** *Sea $\mathcal{C}$ un código (lineal o no) de longitud $n$ y distancia mínima $d$. Entonces la **distancia relativa** del código viene dada por $\frac{d}{n} = \delta(\mathcal{C})$.*

La distancia relativa es una medida de la capacidad de corregir errores del código relativo a su longitud.

**Definición 2.81.** *Consideramos una secuencia infinita de $\mathbb{F}_q$-códigos lineales $B = \{\mathcal{C}_i\}_{i=1}^{\infty}$ donde cada código $\mathcal{C}_i$ tiene parámetros $[n_i, k_i, d_i]_q$.*
*Decimos que $B$ es una secuencia asintótica si $\lim_{i\to\infty} n_i = \infty$ y los siguientes límites existen:*

$$R(B) = \lim_{i\to\infty} \frac{k_i}{n_i} = \lim_{i\to\infty} R(\mathcal{C}_i) \text{ y } \delta(B) = \lim_{i\to\infty} \frac{d_i}{n_i} = \lim_{i\to\infty} \delta(\mathcal{C}_i).$$

*Si además ambos límites son positivos, entonces la secuencia de códigos se dice que es asintóticamente buena.*

Buscamos cotas superiores e inferiores para la mayor tasa de información posible de una familia de códigos (posiblemente no lineales) sobre $\mathbb{F}_q$ cuando la longitud $n$ tiende a infinito y la distancia relativa se aproxima a $\delta$. La función que determina la tasa de información viene dada por:

$$\alpha_q(\delta) = \lim_{n\to\infty} \sup \frac{1}{n} \log_q A_q(n, \delta n)$$

Una cota superior de $\alpha_q(\delta)$ indica que todas las familias de códigos con distancia relativa próxima a $\delta$ tiene tasa de información como máximo su cota superior. Una cota inferior de $\alpha_q(\delta)$ indica que existe una familia de códigos cuya longitud tiende a infinito y distancia reativa próxima a $\delta$ cuya tasa de información es, como mínimo, dicha cota.

**Proposición 2.82 (Cota asintótica de Singleton).** *Sea $C$ una secuencia de códigos. Se tiene que: $R(C) + \delta(C) \leq 1$.*

*Demostración.* De la Proposición 2.67 y 2.68 se deduce que

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

Aplicando logaritmos a esta desigualdad obtenemos que $\log_q(M) \leq n - d + 1$ o, lo que es equivalente, $\log_q(M) + d \leq n + 1$. Dividimos entre $n$ y aplicamos límites a la desigualdad anterior a una familia de códigos y se tiene que

$$\lim_{i\to\infty} \frac{\log_q(M_i)+d}{n_i} \leq 1+\frac{1}{n_i}.$$

Usando que $\lim_{n\to\infty} \frac{1}{n} = 0$ obtenemos que $R(C)+\delta(C) \leq 1$. □

**Definición 2.83 (Función de entropía).** *Llamamos función de entropía a:*

$$H_q(x) = x\log_q(q-1) - x\log_q x - (1-x)\log_q(1-x) \text{ con } 0 < x < 1.$$

*Observación 2.84.* Se tiene que: $H_q(0) = 0$ y $H_q(1) = \log_q(q-1)$

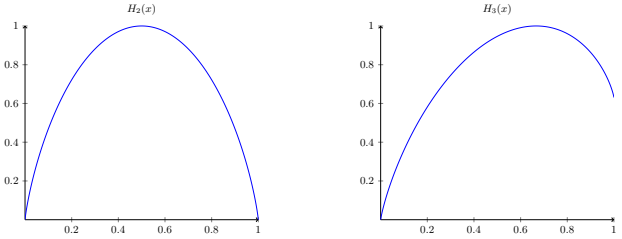



Figura 2.1: Funciones $H_2(x)$ y $H_3(x)$

**Lema 2.85.** *Si denotamos* $\log_q(H_q(x)) = h_q(x)$*, tenemos que:*

$$q^{(h_q(\theta)-o(1))n} \leq \#V_q(n,\theta n) \leq q^{h_q(\theta)n} \text{ con } 0 \leq \theta \leq 1-\frac{1}{q} = \frac{q-1}{q}.$$

*Demostración.* Comenzamos probando la segunda desigualdad:

$$\begin{aligned}
\frac{\#V_q(n,\theta n)}{q^{h_q(\theta)n}} &= \frac{\sum_{j=0}^{\theta n}\binom{n}{j}(q-1)^j}{(q-1)^{\theta n}\theta^{-\theta n}(1-\theta)^{-(1-\theta)n}} \\
&= \sum_{j=0}^{\theta n}\binom{n}{j}(q-1)^j(q-1)^{-\theta n}\theta^{\theta n}(1-\theta)^{(1-\theta)n} \\
&= \sum_{j=0}^{\theta n}\binom{n}{j}(q-1)^j(1-\theta)^n\left(\frac{\theta}{(q-1)(1-\theta)}\right)^{\theta n} \\
&\leq \sum_{j=0}^{\theta n}\binom{n}{j}(q-1)^j(1-\theta)^n\left(\frac{1}{q(1-\theta)}\right)^{\theta n}.
\end{aligned}$$

Por hipótesis $\frac{1}{q(1-\theta)} \leq 1$. Con lo cual, se tiene que $(\frac{1}{q(1-\theta)})^{\theta n}$ es una función exponencial del tipo $a^x$ con $a < 1$. Así, tenemos que, a medida que aumentamos el exponente, la función decrece. Por tanto, se tiene que:

$$\sum_{j=0}^{\theta n} \binom{n}{j} (q-1)^j (1-\theta)^n \left(\frac{\theta}{(q-1)(1-\theta)}\right)^{\theta n} \leq \sum_{j=0}^{\theta n} \binom{n}{j} (q-1)^j (1-\theta)^n \left(\frac{\theta}{(q-1)(1-\theta)}\right)^{j}$$
$$= \sum_{j=0}^{\theta n} \binom{n}{j} (1-\theta)^{n-j} \theta^j = 1$$

La última desigualdad es consecuencia de que los coeficientes corresponden a una distribución binomial. Por tanto, podemos concluir que $\#V_q(n, \theta n) \leq q^{h_q(\theta)n}$. Para la demostración de la segunda desigualdad usaremos la **fórmula de Stirling**, que establece que:

$$\log(n!) \simeq n\log(n) - n + \frac{1}{2}\log(2\pi \mathrm{n}) = \left(\mathrm{n} + \frac{1}{2}\right)\log(\mathrm{n}) - \mathrm{n} + \frac{1}{2}(\log(2\pi)).$$

Usando la fórmula anterior, tenemos que: $\log\binom{n}{m} = \frac{\mathrm{n!}}{\mathrm{m!(n-m)!}}$

$$\simeq \left(n + \tfrac{1}{2}\right)\log(n) - \left(m + \tfrac{1}{2}\right)\log(m) - \left(n - m + \tfrac{1}{2}\right)\log(n-m) - \tfrac{1}{2}\log(2\pi)$$
$$= n\log(n) - m\log(m) - (n-m)\log(m) + \tfrac{1}{2}\log(n) - \tfrac{1}{2}\log(m) - \tfrac{1}{2}\log(n-m) - \tfrac{1}{2}\log(2\pi)$$
$$= n\log(n) - m\log(m) - (n-m)\log(m) + \tfrac{1}{2}\log\left(\tfrac{nm}{(n-m)2\pi}\right).$$

Ahora, sabemos que:

$$\#V_q(n, \theta n) = \sum_{j=0}^{\theta n} \binom{n}{j} (q-1)^j \geq \binom{n}{\theta n} (q-1)^{\theta n}.$$

Esto implica que:

$$\log_q(\#V_q(n, \theta n)) \geq \log_q \binom{n}{\theta n} + \theta n \log_q(q-1).$$

Si aplicamos la fórmula de Stirling obtenemos que:

$$\log_q \binom{n}{\theta n} + \theta n \log_q(q-1) \simeq n\log(n) - \theta n \log(\theta n)$$

$$- (n-\theta n)\log(\theta n) + \frac{1}{2}\log\left(\frac{n\theta n}{(n-\theta n)(2\pi)}\right) + \theta n \log_q(q-1).$$

Aplicando ahora propiedades de logaritmos se obtiene que:

$$\begin{gathered}\log_q\binom{n}{\theta n} + \theta n \log_q(q-1) \\ = n\theta\log_q(q-1) - n\theta\log_q(\theta) - n(1-\theta)\log_q(1-\theta) + \theta n\log_q(q-1) - o(n) \\ = H_q(\theta)n - o(n)\end{gathered}$$

donde o(n) denota el error de aproximación. □

**Corolario 2.86.** *Sean $q \geq 2$ y $0 \leq \theta \leq \frac{q-1}{q}$. Sea $\lim_{n\to\infty} \frac{\lfloor \theta n \rfloor}{n} = \delta$. Entonces:*

$$\lim_{n\to\infty} \frac{1}{n} \# V_q(n, \lfloor \theta n \rfloor) = H_q(\delta)$$

*Demostración.* Es consecuencia directa del lema anterior. □

**Teorema 2.87 (Cota asintótica de Hamming).** *Sea $C$ una secuencia asintótica de códigos. Entonces:*

$$R(C) \leq 1 - H_q\left(\frac{\delta(C)}{2}\right)$$

*Demostración.* Sabemos por el Teorema 2.73 que $B_q(n,d) \leq \frac{q^n}{\# V_q(n,t)}$. Si aplicamos logaritmos a esta desigualdad obtenemos que:

$$b_q(n,d) \leq \log_q q^n - \log_q V_q(n,t).$$

Si aplicamos límites tenemos que:

$$\lim_{i\to\infty} \frac{1}{n_i} b_q(n_i, d_i) \leq \lim_{i\to\infty} \frac{1}{n_i} n_i - \lim_{i\to\infty} \frac{1}{n_i} \log_q \# V_q(n_i, t_i).$$

Si ahora aplicamos el Teorema 2.73 obtenemos que $R(C) \leq 1 - H_q(\frac{\delta(C)}{2})$. □

**Proposición 2.88 (Cota asintótica de Gilbert-Varshamov).** *Si $0 < \delta \leq 1 - \frac{1}{q}$ donde $q \geq 2$, entonces existe una secuencia $C$ de códigos lineales sobre $\mathbb{F}_q$ asintóticamente buena tal que:*

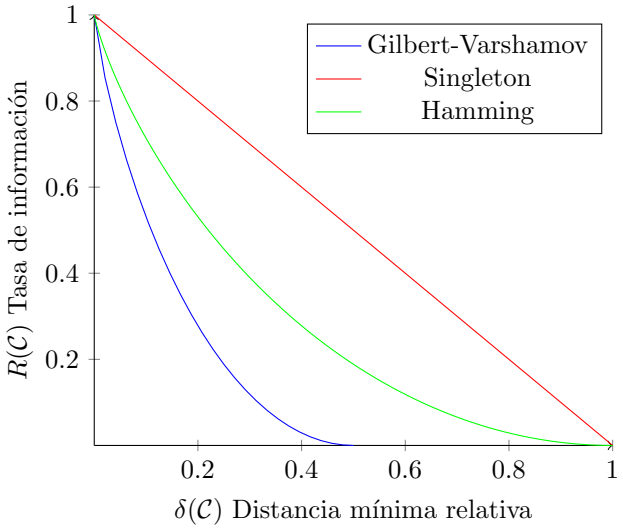
$$R(C) = 1 - H_q(\delta(C))$$

*Demostración.* Sea $0 < \theta < \frac{q-1}{q}$. Sea $\{n_i\}_{i=1}^{\infty}$ una secuencia de números enteros no negativos con $\lim_{i\to\infty} n_i = \infty$. Sea $d_i = \lfloor \theta n_i \rfloor$ y $k_i = n_i - \lceil (\log_q(V_q(n_i, d_i - 1))) \rceil$. Ahora, por la Cota de Varshamov (Teorema 2.76) y la Proposición 2.77 sabemos que existe una secuencia $C = \{\mathcal{C}_i\}_{i=1}^{\infty}$ de $[n_i, k_i, d_i]$-códigos sobre $\mathbb{F}_q$.
Tenemos que $\delta(C) = \theta > 0$ para esta secuencia de códigos. Con lo cual, el Lema 2.87 implica que:

$$R(C) = \lim_{i\to\infty} \frac{k_i}{n_i} = 1 - \lim_{i\to\infty} \frac{1}{n_i} \log_q(V_q(n_i, d_i - 1)) = 1 - H_q(\theta)$$

y $1 - H_q(\theta) > 0$ ya que $\theta < \frac{q-1}{q}$. □

Se muestra a continuación una gráfica donde se representan las cotas asintóticas de Gilbert-Varshamov(cota inferior), Singleton y Hamming (cotas superiores) para códigos binarios. En el eje horizontal se representa la distancia relativa $\delta \in [0, 1]$ y el eje vertical representa la tasa de información $R \in [0, 1]$.
Todo punto $(R, \delta)$ por debajo de la curva de Gilbert-Varshamov o por encima de la cota de Hamming no se pueden alcanzar.



# 3

# Decodificación usando conjuntos de información

## 3.1. Preliminares

En este capítulo vamos a trabajar con algoritmos de decodificación que utilizan conjuntos de información. Esta técnica de decodificación ha tenido toda la atención de los criptoanalistas (profesionales dedicados a atacar los criptosistemas) en los últimos años, pues se trata de la mejor técnica conocida para atacar criptosistemas del estilo McEliece. O dicho de otra forma: para decodificar códigos lineales de los que no conocemos más información que su matriz generatriz.
Los códigos que utilizaremos en este capítulo serán códigos binarios (suficientes para muchas aplicaciones criptográficas), aunque la generalización a códigos lineales no binarios se puede ver en [11].
Vamos a suponer, además, que cualquier atacante o cualquier persona que utilice estos algoritmos de decodificación no tiene ninguna información de la estructura algebraica que pueda tener el código. Es decir, trabajaremos con códigos lineales de los que sólo se conoce una matriz generatriz o de paridad.
Sea $\mathcal{C}$ un código lineal de parámetros $[n,k]_2$ y sea $H$ una matriz de paridad de $\mathcal{C}$. Definimos la aplicación síndrome (relacionada con la matriz $H$) como:

$$\begin{aligned} S_H : \mathbb{F}_2^n &\longrightarrow \mathbb{F}_2^{n-k} \\ y &\longrightarrow Hy^T \end{aligned}$$

Denotaremos al conjunto de vectores con síndrome $s$ como:

$$S_H^{-1}(s) = \{y \in \mathbb{F}_2^n : Hy^T = s\}$$

Ya sabemos que $S_H^{-1}(0) = \mathcal{C}$. Además, para cada síndrome $s \in \mathbb{F}_2^{n-k}$:

$$S_H^{-1}(s) = y + \mathcal{C} = \{y + c : c \in \mathcal{C}\} \text{ donde } Hy^T = s.$$

A estos conjuntos los hemos llamado clases de equivalencia del código $\mathcal{C}$. Además, hemos visto en la Observación 2.55 que estas clases particionan el conjunto $\mathbb{F}_2^n$.

**Proposición 3.1.** *Dado un vector $s \in \mathbb{F}_2^{n-k}$, encontrar un vector $y \in \mathbb{F}_2^n$ tal que $S_H^{-1}(s) = y + \mathcal{C}$ se puede realizar en tiempo polinomial.*

*Demostración.* Para cualquier $y, z \in S_H^{-1}(s)$, tenemos que $yH^T = zH^T$. Es decir, $(y+z)H^T = 0$ y, por tanto, $y + z \in \mathcal{C}$. De aquí se sigue que $S_H^{-1}(s) = y + \mathcal{C}$. Ahora, para obtener un elemento particular de $S_H^{-1}(s)$ vamos a considerar la forma sistemática $H_0$ de la matriz de paridad $H$. $H_0$ es una matriz $n-k \times n$ de la forma $[\mathrm{Id}|X]$ donde Id es la matriz identidad de tamaño $n-k$ y $X$ es una matriz $n-k \times k$ tal que $H_0 = UH$ con $U$ matriz no singular.
Podemos obtener esta matriz $U$ en tiempo $\mathcal{O}((n-k)^3)$ (utilizando eliminación Gaussiana, véase Observación 2.48).

Ahora, sea $y = [sU^T|0] \in \{0,1\}^n$, como $(U^T)^{-1} = (U^{-1})^T$. se tiene que:

$$yH^T = y(U^{-1}H_0)^T = yH_0^T(U^{-1})^T = (sU^T|0)\begin{bmatrix}\mathrm{Id}\\ X^T\end{bmatrix} = (U^T)^{-1} = s.$$

Con esto, hemos conseguido un vector $y \in \mathbb{F}_2^n \in S_H^{-1}(s)$ en tiempo polinomial (coste de una eliminación gaussiana). □

### Problema de decodificación

Dado un vector $y \in \mathbb{F}_2^n$, el problema de decodificación es equivalente a resolver uno de los siguientes problemas:

- Encontrar la palabra $c \in \mathcal{C}$ más cercana (en distancia de Hamming) al vector $y$. En este caso, se trabajaría con la matriz generatriz.
- Encontrar el error $e \in S_H^{-1}(s)$ con $s = Hy^T$ de menor peso de Hamming. Se necesita en este caso la matriz de paridad.

En la práctica, es difícil comprobar que un error es el de peso más pequeño en un conjunto relativamente grande (o de forma equivalente, comprobar que la palabra $c \in \mathcal{C}$ es la más cercana al vector $y$). Por ello, vamos a considerar en el resto de capítulo un problema similar al anterior.

**Problema de decodificación (más sencillo)**:
Dada una matriz de paridad $H \in \mathbb{F}_2^{(n-k)\times n}$, un síndrome $s \in \mathbb{F}_2^{n-k}$ y un entero $w > 0$, encontrar un vector $e \in S_H^{-1}(s)$ con $w_H(e) \leq w$.

*Observación 3.2.* El problema de encontrar palabras de peso pequeño en un código se puede simplificar de forma similar al definido anteriormente. Este problema se puede establecer como:

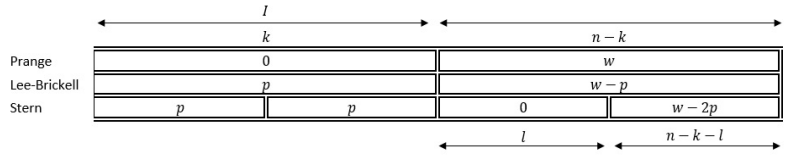
Dada $H \in \mathbb{F}_2^{(n-k)\times n}$ una matriz de paridad y fijado un entero $w > 0$, encontrar palabras de peso de Hamming menor o igual que $w$ en $S_H^{-1}(0)$.

*Observación 3.3.* El valor del parámetro $w$ afecta de forma significativa la dificultad de este problema. En teoría de códigos, la decodificación tiene sentido si $w$ es elegido de forma que el problema tiene una única solución. Es decir, si $w \leq d_{GV}$ (distancia de Gilbert-Varshamov). Para aplicaciones criptográficas, el valor $w$ debe ser elegido para que este problema sea difícil.

## 3.2. Algoritmo de Prange

El algoritmo que nos ocupa a continuación fue introducido por Prange en 1962, tal y como se muestra en [12].
Cualquier algoritmo de decodificación utilizando un conjunto de información $I \subseteq \{1, \ldots, n\}$ presupone que el vector de error tenga un cierto patrón con respecto a dicho conjunto. Se muestra a continuación un esquema de la distribución de errores en los diferentes algoritmos que veremos en este capítulo:

Figura 3.1: Distribución de errores




*Observación 3.4.* Recordemos la Definición 2.23 que nos describe un conjunto de información $I$ de un código $\mathcal{C}$ de parámetros $[n, k]_q$ con matriz generatriz $G$ como un conjunto de $k$ posiciones tal que $G$ contiene una submatriz invertible $G_I$, la cual está formada por las columnas indexadas por las posiciones de $I$.

Una decodificación completa usando el algoritmo de Prange espera que no hayan errores en las posiciones del vector $y$ que están indexadas por $I$ (siguiendo el patrón que se refleja en la Figura 3.1). Sea $G$ una matriz generatriz, entonces $y_I G_I^{-1}$ es la pre-imagen de una palabra del código $c \in \mathcal{C}$ y podemos obtener $c$ calculando $y_I G_I^{-1} G$.

*Ejemplo 3.5.* Supongamos que tenemos un $[8, 4]_2-$código $\mathcal{C}$ con matriz generatriz:

$$G = \begin{pmatrix} 1\,1\,0\,0\,0\,0\,1\,0 \\ 1\,1\,1\,0\,1\,1\,0\,1 \\ 0\,0\,0\,1\,1\,1\,1\,0 \\ 0\,1\,1\,0\,1\,0\,0\,0 \end{pmatrix} \in \mathbb{F}_2^{4\times 8}$$

Fijamos $w = 2$ y enviamos $c = (0,1,1,0)G = (1,1,1,1,0,0,1,1)$.
Supongamos que recibimos $y = c + \underbrace{(0,0,0,0,0,0,1,1)}_{e} = (1,1,1,1,0,0,0,0)$.

Nuestro objetivo es usar el algoritmo de Prange para encontrar el vector $e \in \mathbb{F}_2^n$. Elegimos como conjunto de información $I = \{1,2,3,5\}$, hemos tenido suerte en nuestra elección y este conjunto no tiene errores. Luego, tenemos que:

$$y_I = (1,1,1,0),\ G_I = \begin{pmatrix} 1&1&0&0 \\ 1&1&1&1 \\ 0&0&0&1 \\ 0&1&1&1 \end{pmatrix} \text{ y } G_I^{-1}G = \begin{pmatrix} 1&0&0&0&0&1&0&1 \\ 0&1&0&0&0&1&1&1 \\ 0&0&1&1&0&0&0&1 \\ 0&0&0&1&1&1&1&0 \end{pmatrix}.$$

Ahora, como las posiciones de error son $\{7,8\}$ y éstas son disjuntas de $I$ tenemos que $c = \left(y_I G_I^{-1}\right) G = (1,1,1,1,0,0,1,1)$. Luego $e = y - c = (0,0,0,0,0,0,1,1)$.

**Complejidad**

Todos los algoritmos de decodificación que utilizan conjuntos de información siguen un mismo patrón a la hora de cacular el coste medio del algoritmo:

1. Calcular la probabilidad $\mathcal{P}$ de que un vector error siga el patrón establecido por el algoritmo.
   Esta probabilidad coincide con la probabilidad de devolver el vector $e$ correcto (o que cumple las condiciones del problema) en la primera iteración.
2. Como las iteraciones son independientes entre sí, el número medio de iteraciones necesarias para tener éxito es: $\frac{1}{\mathcal{P}}$.
3. Con esta información el coste medio del algoritmo es:

$$\text{COSTE TOTAL } = \frac{\text{Coste de una iteración}}{\mathcal{P}}$$

Estudiamos ahora la complejidad en el algoritmo de Prange:

1. Sea $e \in \mathbb{F}_2^n$ con $w_H(e) = w$ un vector elegido de forma aleatoria y uniforme. Sea $I$ un subconjunto de $\{1,\ldots,n\}$ de tamaño $k$.
   Entonces la probabilidad de que el peso de $e$ sea cero en el conjunto $I$ es:

$$\mathcal{P}_{\text{PRANGE}}(n,k,w) = \frac{\binom{n-k}{w}}{\binom{n}{w}}$$

   siendo el numerador la posibilidad de elegir $w$ índices entre los $n-k$ que no están en $I$ y el denominador la posibilidad de elegir $w$ índices entre todos los posibles.
2. El coste de una iteración está dominado por aplicar GAUSS a la matriz $G_I^{-1}$ :, esto es, $k^3$.

## 3.3. Algoritmo de Lee-Brickell

El algoritmo de Lee-Brickell fue creado como algoritmo de decodificación. Se trata de una extensión del Algoritmo de Prange, ya que Lee y Brickell quisieron generalizarlo permitiendo $p$ errores en el conjunto de información (tal y como se establece en la Figura 3.1). Este algoritmo toma como entradas un vector $y \in \mathbb{F}_2^n$, un entero $w$ y una matriz generatriz $G$ de un código lineal $\mathcal{C}$ de parámetros $[n, k]_2$.
Según la figura 3.1, el algoritmo de Lee-Brickell permite $p$ errores en las posiciones que están en el conjunto de información. Estos $p$ errores pueden corregirse encontrando las $p$ filas de $G$ que corresponden a los índices con error en $I$.
Vamos a denotar por $G_a$ a la única fila de $G_I^{-1}G$ que tiene un 1 en la posición $a$ con $a \in I$. Sea $p$ un entero tal que $0 \leq p < w$.
El parámetro $p$ es elegido como un número pequeño. El algoritmo consiste en una serie de iteraciones independientes donde cada iteración hace elecciones aleatorias.
Si el conjunto $I$ elegido en el **PASO 1** no nos conduce a una palabra de peso $w$ en el **PASO 3**, entonces repetimos el proceso con una nueva elección del conjunto $I$. El algoritmo se describe en las siguientes líneas:

**Algoritmo 4:** Algoritmo de Lee-Brickell.

**Input:** Una matriz generatriz $G \in \mathbb{F}_2^{k \times n}$ de un código lineal $\mathcal{C}$ de parámetros $[n, k]_2$, un vector $y \in \mathbb{F}_2^n$ y un entero $w \geq 0$.
**Output:** Una palabra $e \in \mathbb{F}_2^n$ de peso $w$ tal que $y - e \in \mathcal{C}$, si existe.
**PASO 1**: Elegir aleatoriamente un conjunto de información $I$.
**PASO 2**: Reemplazar $y$ por $e = y - y_I G_I^{-1} G$.
Si $w_H(e) = w$, ir al **PASO 4**.
**PASO 3**: Para cada subconjunto $A \subseteq I$ de tamaño $p$, caculamos :

$$e = y - \sum_{a \in A} G_a$$

Si $e$ tiene peso $w$, ir al **PASO 4**, en caso contrario volver al **PASO 1**.
**PASO 4**: Devolver el vector $e$.

### Complejidad

Sea $e \in \mathbb{F}_2^n$ con $w_H(e) = w$ un vector elegido de forma aleatoria y uniforme. Sea $I$ es un subconjunto de $\{1, \ldots, n\}$ de tamaño $k$. Entonces la probabilidad de que el peso de $e$ sea $p$ en el conjunto $I$ y sea $w - 2p$ en el resto de índices es:

$$\mathcal{P}_{\mathrm{LB}}(n, k, w, p) = \frac{\binom{n-k}{w-p}\binom{k}{p}}{\binom{n}{w}}$$

El coste de una iteración es el coste de aplicar eliminación gaussiana y enumerar todos los subconjuntos de $p$ elementos en $I$:

$$\text{Coste de una iteración} : k^3 + \binom{k}{p}$$

*Ejemplo 3.6.* Tomamos ahora el Ejemplo 3.5 pero considerando que hay un error en las posiciones indexadas por $I$ (es decir, $p = 1$).
Tenemos un $[8, 4]_2-$código $\mathcal{C}$ con matriz generatriz:

$$G = \begin{pmatrix} 1\,1\,0\,0\,0\,0\,1\,0 \\ 1\,1\,1\,0\,1\,1\,0\,1 \\ 0\,0\,0\,1\,1\,1\,1\,0 \\ 0\,1\,1\,0\,1\,0\,0\,0 \end{pmatrix} \in \mathbb{F}_2^{4\times 8}$$

Tomamos $w = 2$ y el conjunto de información $I = \{1, 2, 3, 5\}$.
Sea $c = (0, 1, 1, 0)G = (1, 1, 1, 1, 0, 0, 1, 1)$ y supongamos que recibimos la palabra $y = c + e = c + (1, 0, 0, 0, 0, 0, 0, 1) = (0, 1, 1, 1, 0, 0, 1, 0)$.
Si hacemos $y_I G_I^{-1} G$ tenemos:

$$y_I G_I^{-1} G = (0, 0, 1, 1) \begin{pmatrix} 1\,1\,0\,0\,0\,0\,1\,0 \\ 1\,1\,1\,0\,1\,1\,0\,1 \\ 0\,0\,0\,1\,1\,1\,1\,0 \\ 0\,1\,1\,0\,1\,0\,0\,0 \end{pmatrix} = (0, 1, 1, 1, 0, 1, 1, 0).$$

Esto es: $e = y - y_I G_I^{-1} G = (0, 0, 0, 0, 0, 1, 0, 0)$. Como $w(e) \neq 2$, tenemos que la palabra encontrada no es la $c$ enviada. Por tanto, siguiendo el algoritmo 4, hacemos:

- $e_1 = y - y_I G_I^{-1} G - G_1 = (1, 0, 0, 0, 0, 0, 0, 1)$ y $w = 2$
- $e_2 = y - y_I G_I^{-1} G - G_2 = (0, 1, 0, 0, 0, 0, 1, 1)$ y $w \neq 2$
- $e_3 = y - y_I G_I^{-1} G - G_3 = (0, 0, 1, 1, 0, 1, 0, 1)$ y $w \neq 2$
- $e_4 = y - y_I G_I^{-1} G - G_5 = (0, 0, 0, 1, 1, 0, 1, 0)$ y $w \neq 2$

Como $e_1 = (1, 0, 0, 0, 0, 0, 0, 1)$ y $w = 2$, concluimos que $e = (1, 0, 0, 0, 0, 0, 0, 1)$.

## 3.4. Algoritmo de Stern

El algoritmo de Stern fue presentado en [13] en 1989. Su objetivo era buscar palabras de poco peso de Hamming en un código lineal. Stern usa la idea de Lee-Brickell, pero permite $2p$ errores en el conjunto de información (véase Figura 3.1). Este algoritmo se plantea en esta sección utilizando una matriz de paridad del código, en lugar de una matriz generatriz como en los casos anteriores, aunque una versión similar podría presentar el algoritmo de Stern

usando la matriz generatriz.
Dada una matriz de paridad $H$, un error de peso $w$ y un síndrome $s$, el algoritmo busca un error $e \in \mathbb{F}_2^n$ de peso $w$ tal que $He^T = s$. Stern ideó este algoritmo para el caso $s = 0$ ya que su objetivo no era decodificar sino buscar palabras de peso pequeño en el código. Sin embargo, en esta sección vamos a generalizar el algoritmo para encontrar cualquier vector que tenga el síndrome $s \in \mathbb{F}_2^n$ de la palabra recibida.

*Observación 3.7.* El algoritmo trata de construir una palabra $e$ de peso $w$ tal que $He^T = 0$. Tenemos que para toda palabra del código $c \in \mathcal{C}$, $Hc^T = 0$ donde $H = (h_1, \ldots, h_n)$ es una matriz de paridad y $h_i$ denotan las columnas de $H$ para $i = 1, \ldots, n$. Como $Hc^T = 0$, esto nos dice que $h_1c_1 + \cdots + h_nc_n = 0$ con $h_i \in \mathbb{F}_2^{n-k}$ y $c_i = \{0, 1\}$. Queremos además que $w_H(c) \leq w$. Por tanto, este problema es equivalente a encontrar $w$ columnas de $H$ que sumen cero. Los correspondientes índices de esas columnas indican las posiciones donde debemos colocar un 1 en la palabra de peso $w$ que devuelve el algoritmo. Para el caso de buscar errores $e \in \mathbb{F}_2^n$ con $w_H(e) \leq w$ y tales que $He^T = s$ el problema es equivalente a encontrar $w$ columnas de $H$ que sumen $s$.

El primer paso del algoritmo es elegir de forma aleatoria un conjunto de información $I$. Para no complicar la explicación, podemos suponer que $I$ son las $k$ primeras posiciones.

*Observación 3.8.* Si $I$ es un conjunto de información, se tiene que $\hat{G} = (I_k|P) = G \cdot G_I^{-1}$. Por la Proposición 2.29 se tiene que una matriz de paridad viene dada por $\hat{H} = (-P^T|I_{n-k}) = H_I^{-1}H$, donde $P \in \mathbb{F}_2^{(n-k)\times k}$. Trabajaremos el algoritmo con esta matriz de paridad (en forma sistemática).

*Observación 3.9.* La matriz $H_I^{-1}H$ también podemos escribirla como $Q = \begin{pmatrix} Q_1 \\ Q_2 \end{pmatrix}$ con $Q_1 \in \mathbb{F}_2^{l\times k}$, $Q_2 \in \mathbb{F}_2^{(n-k-l)\times k}$.

El algoritmo de Stern precisa de dos parámetros iniciales que son $p \in \{0, 1, \ldots, \frac{w}{2} - 1\}$ y $l \in \{0, 1, \ldots, n-k\}$. Una vez que tenemos los parámetros iniciales. Definimos los siguientes conjuntos:

$$X = \left\{1, \ldots, \left\lfloor \frac{k}{2} \right\rfloor\right\}, \; Y = \left\{\left\lfloor \frac{k}{2} \right\rfloor + 1, \ldots, k\right\}, \text{ y } Z \subseteq \bar{I}, \text{ con } \#Z = l.$$

Tenemos con esto que $X \cup Y = I$. Sin pérdida de generalidad, suponemos que $Z = \{k+1, \ldots, k+l\}$. Es decir, tenemos la situación de la matriz $\hat{H}$ definida en 3.2 Tenemos que el conjunto $X$ indexa las columnas $\{1 \ldots, \frac{k}{2}\}$; el conjunto $Y$ indexa las columnas $\{\frac{k}{2} + 1, \ldots, k\}$ y el conjunto $Z$ indexa las columnas $\{k+1, \ldots, k+l\}$. (Nótese que $Z$ también indexa las primeras $l$ filas de $H_I^{-1}H$, en particular, $Z$ indexa las filas de $Q_1$).

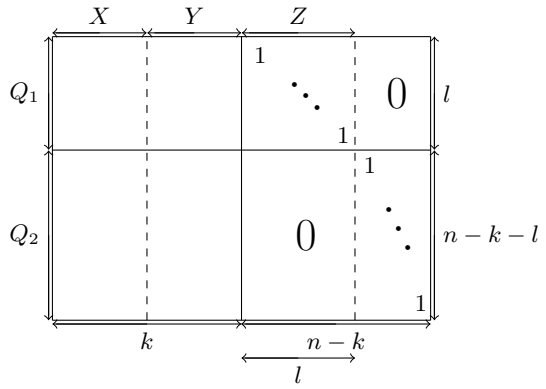



Figura 3.2: Esquema de la matriz de paridad $H$.

El algoritmo trata de encontrar $e$ que tenga el patrón de la Figura 3.1, es decir, con peso $(p, p, 0)$ en las posiciones indexadas por $X, Y, Z$ respectivamente, esto es, $w_H(e_{|X}) = p, w_H(e_{|Y}) = p$ y $w_H(e_{|Z}) = 0$, donde $e_{|S}$ representa el vector $e$ restringido a las posiciones de $S$.
Como todos los algoritmos de decodificación utilizando conjuntos de información, este algoritmo consiste en una serie de iteraciones donde, en cada una de ellas, se hacen elecciones aleatorias. Si el conjunto $I$ elegido en el **PASO 1** no devuelve una palabra de peso $w$ en el **PASO 8**, debe ejecutarse otra iteración del algoritmo.
Una vez que se han definido los conjuntos $X$, $Y$ y $Z$, tomamos los conjuntos: $A$ con $p$ columnas de $X$ y $B$ con $p$ columnas de $Y$. De esta forma tenemos que $\#A \cup B = 2p$.
Definimos entonces:

$$\phi(A) = \sum_{a \in A} (s - h_a)_{|Q_1} \qquad \Psi(B) = \sum_{b \in B} (h_b)_{|Q_1}$$

donde $(h_i)_{|Q_j}$ es la restricción de la columna $h_i$ a las posiciones de $Q_j$, para $j = 1, 2$. Si una colisión puede encontrarse entre $p$ columnas de $Q_1$ indexadas por $X$ y $p$ columnas de $Q_1$ indexadas por $Y$, es decir, $\phi(A) = \Psi(B)$, se tiene que:

$$\sum_{c \in A \cup B} (h_c)_{|Q_1} = s_{|Q_1}$$

donde $s = \sum_{c \in A \cup B} (h_c)$.
Ahora comprobamos si $w_H(s) = w - 2p$. En tal caso, tenemos que $w_H(s_{|Q_1}) = 0$ y $w_H(s_{|Q_2}) = w - 2p$. Es decir, las posiciones distintas de cero provienen de filas de $Q_2$. Por lo tanto, si $s_j \neq 0$, entonces sumamos al vector $s$ la columna $j$ de la identidad $I_{n-k}$.

*Observación 3.10.* $s_j \neq 0 \Leftrightarrow j \in \{k+l+1, \ldots, n\}$ ya que las posiciones distintas de cero provienen de $Q_2$.

Y tenemos que $\sum_{i \in A \cup B \cup C} h_i = s$.
Después de estos pasos, devolvemos $e \in \mathbb{F}_2^n$ con $e_j \neq 0$ cuando $j \in A \cup B \cup \text{supp}(s)$.
Así, se obtiene que:

$$w_H(e) = w = \underbrace{\#A \cup B}_{2p} + \underbrace{\text{supp}(s)}_{w-2p}$$

Por lo tanto, hemos encontrado un vector $e \in \mathbb{F}_2^n$ con peso $w$ tal que $He^T = s$.
El algoritmo se define de forma esquemática en Algoritmo 5.

**Algoritmo 5:** Algoritmo de Stern

**Input:** Una matriz de paridad $H \in \mathbb{F}_2^{(n-k)\times n}$ para un código lineal, un vector columna $s \in \mathbb{F}_2^{n-k}$, un número entero $w > 0$ y los parámetros $p$ y $l$.
**Output:** Un vector $e \in \mathbb{F}_2^n$ de peso $w$ que cumple $He^T = s$, si existe.
**PASO 1**: Elegir de forma uniforme y aleatoria un conjunto de información $I$ y encontrar una matriz $H_I^{-1} \in \mathbb{F}_2^{(n-k)\times(n-k)}$ tal que la submatriz de $H_I^{-1}H$ indexada por $\{1, \ldots, n\} - I$ sea la identidad $(n-k) \times (n-k)$.
**PASO 2**: Seleccionar de forma uniforme y aleatoria un subconjunto $X$ de I de tamaño $\lfloor \frac{k}{2} \rfloor$.
**PASO 3**: Definimos $Y = I - X$.
**PASO 4**: Seleccionar un subconjunto $Z$ de tamaño $l$ en $\{1, \ldots, n\} - I$.
**PASO 5**: Para cada subconjunto $A \subseteq X$ de tamaño $p$ consideramos las $p$ columnas $(H_I^{-1}H)_a$ indexadas por cada $a \in A$ y computamos $\phi(A) = s - \sum_{a \in A}(H_I^{-1}H)_a$ restringido a las $l$ filas indexadas por $Z$.
**PASO 6**: Para cada subconjunto $B \subseteq Y$ de tamaño $p$, consideramos las $p$ columnas $(H_I^{-1}H)_b$ indexadas para cada $b \in B$ y computamos $\psi(B) = \sum_{b \in B}(H_I^{-1}H)_b$ restringido a las $l$ filas indexadas por $Z$.
**PASO 7**: Para cada par $(A, B)$ tal que $\phi(A) = \psi(B)$, calculamos $s' = s - \sum_{i \in A \cup B}(H_I)^{-1}H$.
**PASO 8**: Si $wt(s') = w - 2p$ entonces, sumar las correspondientes $w - 2p$ columnas en la submatriz identidad $(n-k) \times (n-k)$ para hacer $s' = 0$.
**PASO 9**: Devolver el vector $e \in \mathbb{F}_2^n$ con soporte las columas del **PASO 8** y las columnas indexadas por $A \cup B$.

### Complejidad

- Probabilidad de que el error siga el patrón de Stern:

$$\mathcal{P}_{\text{STERN}} = \frac{\binom{k+l}{p}\binom{n-k-l}{w-p}}{\binom{n}{w}}$$

- El coste de una iteración es: $k(n-k)^2 + 2lp\binom{k/2}{p} + (2p+1)(n-k)\frac{\binom{k/2}{p}^2}{2^l}$, donde:
  - El coste del **PASO 1** es aplicar Gauss: $\mathcal{O}(k(n-k)^2)$.
  - **PASO 5 y 6** tienen el coste de sumar $p$ vectores de longitud $l$ y repetir este proceso para cada subconjunto de $p$ elementos de $X$ y de $Y$. Luego el coste es $\mathcal{O}(2lp\binom{k/2}{p})$.
  - Para el coste del **PASO 7 y 8** el problema es contar el número de colisiones. Si suponemos que $\phi(A)$ y $\psi(B)$ siguen una distribución uniforme la probabilidad de que sean iguales es $\frac{\binom{k/2}{p}^2}{2^l}$. Luego, en cada colisión hay que calcular la suma de $2p+1$ vectores de longitud $n-k$.

*Ejemplo 3.11.* Supongamos que tenemos un código $\mathcal{C}$ cuya matriz generatriz es $G$ y aplicando la Proposición 2.29 se obtiene que una matriz de paridad es $H$.

$$G = \begin{pmatrix} 1\,0\,0\,0\,0\,1\,0\,1 \\ 0\,1\,0\,0\,0\,1\,1\,1 \\ 0\,0\,1\,0\,1\,1\,1\,1 \\ 0\,0\,0\,1\,1\,1\,1\,0 \end{pmatrix} \text{ y } H = \begin{pmatrix} 0\,0\,1\,1\,1\,0\,0\,0 \\ 1\,1\,1\,1\,0\,1\,0\,0 \\ 0\,1\,1\,1\,0\,0\,1\,0 \\ 1\,1\,1\,0\,0\,0\,0\,1 \end{pmatrix}$$

Consideramos $I = \{1,2,3,4\}$, $X = \{1,2\}$, $Y = \{3,4\}$ y $Z = \{5\}$. Tomamos $p = 1$, $w = 3$ y $l = 1$. Supongamos que la palabra del código enviada es $c = (1,0,0,0,0,1,0,1)$ pero debido a los errores producidos en el canal de envío, la palabra recibida es $y = c + e = c + (1,0,1,0,0,1,0,0) = (0,0,1,0,0,0,0,1)$. Tenemos que el síndrome de la palabra recibida (que coincide con el síndrome del error) es $Hy^T = s = (1,1,1,0)^T$. Ahora, tenemos que:

$$(H_I)^{-1}H = \begin{pmatrix} 0\,0\,1\,1 \\ 1\,1\,1\,1 \\ 0\,1\,1\,1 \\ 1\,1\,1\,0 \end{pmatrix}^{-1} H = \begin{pmatrix} 1\,0\,0\,0\,0\,1\,1\,0 \\ 0\,1\,0\,0\,1\,0\,1\,0 \\ 0\,0\,1\,0\,1\,1\,0\,1 \\ 0\,0\,0\,1\,0\,1\,0\,1 \end{pmatrix}.$$

Hemos realizado los 4 primeros pasos del algoritmo.
**PASO 5**: Tomamos $A_1 = \{1\} \subseteq X$ con $\#A_1 = p = 1$.

Calculamos $\phi(A_1) = s - \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$. Luego $\phi(A_1)_{|Z} = (0)$.

– Si tomáramos $A_2 = \{2\} \subseteq X$ con $\#A_2 = p = 1$.

Calculamos $\phi(A_2) = s - \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}$. Luego $\phi(A_2)_{|Z} = (1)$.

**PASO 6**: Tomamos $B_1 = \{3\} \subseteq Y$ con $\#B_1 = 1$.

Tenemos que $\psi(B_1) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$, es decir, $\psi(B_1)_{|Z} = (0)$.

Si $B_2 = \{4\} \subseteq Y$. Se tiene que $\psi(B_2) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$, es decir, $\psi(B_2)_{|Z} = (0)$.

**PASO 7**: Comparamos las posibles colisiones y se obtiene que:

$$\phi(A_1)_{|Z} = \psi(B_1)_{|Z} \qquad \text{y} \qquad \phi(A_1)_{|Z} = \psi(B_2)_{|Z}$$

Computamos $s_1' = s - \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$.

Por otro lado, tenemos que: $s_2' = s - \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}$.

**PASO 8**: Calculamos $w_H(s_1') = 1 = w - 2p$. Sin embargo, $w_H(s_2') = 3 \neq w - 2p$. De esta forma tenemos que:

$$s_1' = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} - \underbrace{\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}}_{\text{columna 2 de la submatriz identidad de } H} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Por último, devolvemos el vector error $e$ (**PASO 9**): $e = (1,0,1,0,0,1,0,0)$. Haciendo $y - e$ obtenemos la palabra del código que fue enviada y terminamos el proceso de decodificación.

*Observación 3.12.* Si se aplica con otro conjunto de información $I$, como el número de errores es superior al que podemos corregir $\left(\lfloor \frac{d-1}{2} \rfloor = \lfloor \frac{3-1}{2} \rfloor = 1\right)$ puede suceder que el algoritmo devuelva otro error posible.

## 3.5. Más algoritmos de decodificación utilizando conjuntos de información

En este capítulo hemos trabajado 3 algoritmos de decodificación utilizando conjuntos de información. Estos algoritmos fueron introducidos por Prange en

1962, estudiado en la sección 3.2 (Pra-ISD), pero existen numerosos trabajos que mejoran este algoritmo (en este capítulo hemos presentado el Algoritmo de Lee-Brickell (LB-ISD) y el algoritmo de Stern (S-ISD)). Otras variantes posteriores y más complicadas de estudiar fueron presentadas por: Dumer (DU-ISD) [4] en 1991, May, Meurer y Thomae (MMT-ISD) [8] en 2011 y Becker, Joux, May y Meurer (BJMM-ISD) [1] en 2012.
En el artículo [3] se presenta un análisis asintótico de todos estos algoritmos. Es decir, se considera una familia de códigos de longitud $n$ (con $n$ creciente), que tienen una tasa de información $R$ (con $0 < R < 1$) y se estudia el problema de encontrar errores con peso $w$ (siendo $w$ proporcional a $n$). El estudio que se realiza sobre el coste de estos algoritmos cuando $n$ es creciente se resume en el siguiente resultado (que presentamos sin demostración).

**Proposición 3.13 (Proposición 2. Artículo [3]).** *Sean $k$ y $w$ parámetros dados en función de $n$ tales que:*

$$\lim_{n\to\infty} \frac{k}{n} = R \quad con\ 0 < R < 1 \qquad y \qquad \lim_{n\to\infty} \frac{w}{n} = 0.$$

*Cualquier algoritmo $A$ entre los mencionados anteriormente (Pra-ISD, LB-ISD, S-ISD, DU-ISD, MMT-ISD, BJMM-ISD) verifican que:*

$$\text{Coste}(A(n,k,w)) = 2^{cw(1+o(1))}.\ con\ c = \log_2\left(\frac{1}{1-R}\right).$$

Es decir, el coste de todos los algoritmos de decodificación utilizando conjuntos de información tienen un coste exponencial. En la siguiente tabla se presenta una comparación entre los diferentes métodos. Nótese que, aunque la complejidad sigue siendo exponencial, se ha obtenido una mejora de casi un 20 % entre al algoritmo Pra-ISD y BJMM-ISD.

| | $c = \lim_{n\to\infty} \frac{\log_2(\text{Coste}A)}{n}$ con $A = 2^{cn(1+o(1))}$, $c$ constante |
|---|---|
| PRANGE | 0.1198 |
| STERN | 0.1154 |
| DUMER | 0.1151 |
| MMT | 0.1101 |
| BJMM | 0.1000 |

Tabla 3.1: La tabla se ha obtenido de la presentación de N. Sendrier en el MOOC con título "Code-based cryptography" de la plataforma FUN. Para $k = 0.5n$ y $w = 0.11n$

# Bibliografía


[1] A. Becker, A. Joux, A. May, and A. Meurer. *Decoding random binary linear codes in* $2^{\frac{n}{20}}$*: How* $1+1=0$ *improves information set decoding.* In D. Pointcheval and T. Johansson, editors, Advances in Cryptology - EUROCRYPT 2012, volume 7237 of LNCS, pages 520-536. Springer, 2012.

[2] Caballero Gil, P. *Introducción a la Criptografía.* Textos Universitarios, Ra-Ma, 2002.

[3] R. Canto-Torres, and N. Sendrier. *Analysis of Information Set Decoding for a Sub-Linear Error Weight* PQCrypto 2016.

[4] I. Dumer. *On minimum distance decoding of linear codes.* In: Proceedings of 5th Joint Soviet-Swedish International Workshop on Information Theory, Moscow, pp.50-52 (1991).

[5] W.C. Huffman, and V. Pless. *Fundamentals of Error-Correcting Codes.* Cambridge University Press, 2010.

[6] J. Justesen, and T. Høholdt.*A Course in Error-correcting Codes.* European Mathematical Society, EMS textbooks in mathematics, 2004.

[7] P.J. Lee, and E.F. Brickell. *An observation on the security of McEliece's public-key cryptosystem.* In C.G. Günther, editor, Advances in Cryptology - EUROCRYPT 88, volume 330 of LNCS, pages 275-280. Springer, 1988.

[8] A. May, A. Meurer, and E. Thomae. *Decoding random linear codes in* $\mathcal{O}(2^{0.054n})$. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 107-124. Springer, Heidelberg (2011).

[9] R. Overbeck, and N. Sendrier. *Code-based Cryptography.* In D. J. Bernstein, J. Buchmann, E. Dahmen, editors. Post-Quantum Cryptography, pp. 95-145, Springer Berlin Heidelberg, 2009.

[10] R. Pellikaan, X-W. Wu, R. Jurrius, and S. Bulygin. *Codes, Cryptology and Curves with Computer Algebra.* Cambridge University Press, 2017.

[11] C. Peters. *Information-set decoding for linear codes over* $\mathbb{F}_q$. In Post-Quantum Cryptography, Lecture Notes in Computer Science, Vol. 6061, pp. 81-94. Springer-Verlag Berlin Heidelberg, 2010.

[12] E. Prange. *The use of information sets in decoding cyclic codes.* IRE Transactions, IT-8:S5.29, 1962.

[13] J. Stern. *A method for finding codewords of small weight.* In G. Cohen and J. Wolfmann, editors, Coding theory and applications, volume 388 of LNCS, pages 106-113. Springer, 1989.

# *Information-Set Decoding*

Sección de Matemáticas
Universidad de La Laguna

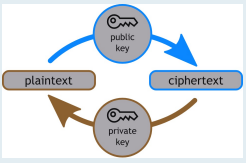
**Sandra Carmona García**
Facultad de Ciencias · Sección de Matemáticas
Universidad de La Laguna
alu0100964790@ull.edu.es

**Abstract**

*Cryptography arises from the needs to keep safe communications. In a world where the use of electronic data plays a larg part in both personal and institutional life, the proper archiving, retention and encryption is no longer optional. However, the construction of large quantum computers would have dramatically effects on the currently used cryptography. Therefore, we study a possible solution: Code-based Cryptography, whose security is based on the difficulty of decoding linear codes. In particular, we study the best known technique to decode random linear codes: information set decoding algorithms. We will study in this work the decoding algorithms of Prange (1962), Lee-Brickell (1988) and Stern (1989).*

## 1. Code based Cryptography

**Public-key cryptography** is a cryptographic system that uses a pairs of keys: public keys which may be disseminated widely, and private keys which are known only to the owner. The generation of such keys depends on cryptographic algorithms based on mathematical problems that produce one-way functions - wich is a function that is easy to compute in one direction and difficult to find its inverse. Effective security only requires keeping the private key safe; the public key can be openly distributed without problem. In this type of cryptography, any person can encrypt a message using the receiver's public key, but the encrypted message can only be decrypted with the receiver's private key.




McEliece was the first one to unify the concepts of cryptography and coding theory on 1978. He published this algorithm that is still very potent and safe.

**Algorithm 1:** *McEliece's Algorithm*

**Input**: $n$, $t \in \mathbb{N}$, with $t \ll n$.

**Key Generation**: Let $n$ y $t$, we take a generator matrix $G$ of an $[n,k,d]_q$ code $\mathscr{C}$ with $d \geq 2t+1$.

**Public key**: $(G,t)$.

**Private key**: $D_{\mathscr{C}}$, an efficient decoding algorithm for $\mathscr{C}$ that corrects $t$ errors.

**Encryption**: To encode a message $m \in \mathbb{F}_q^k$ choose a vector $e \in \mathbb{F}_q^n$ randomly of Hamming weight lower than $t$. The encryption of the plaintext is: $c = mG + e$.

**Decryption**: Apply $D_{\mathscr{C}}$ to the ciphertext $c$.

## 2. Glossary

- A **linear code** is a linear subespace of $\mathbb{F}_q^n$. We denote a linear code $\mathscr{C}$ over $\mathbb{F}_q$ of length $n$ and dimension $k$ by $[n,k]_q$.
- A **generator matrix** of a $[n,k]_q$ code is a $k \times n$ matrix whose rows are linearly independent.
- An **encoder** of a linear $[n,k]_q$–code $\mathscr{C}$ is an one-to-one map

$$\begin{aligned} \varepsilon : \mathbb{F}_q^k &\longrightarrow \mathbb{F}_q^n \\ m &\longmapsto c = mG \end{aligned}$$

such that $\varepsilon(\mathbb{F}_q^k) = \mathscr{C}$. Let $c \in \mathscr{C}$ be a codeword, then there exists a unique $m \in \mathbb{F}_q^k$ with $c = \varepsilon(m)$.

Encoding a message has a cost of $\mathcal{O}(n^2)$ operations in $\mathbb{F}_q$ (i.e **encoding is very fast**).

- An $(n-k) \times n$ matrix of rank $n-k$ is called a **parity check matrix** of an $[n,k]_q$ code $\mathscr{C}$ if $\mathscr{C}$ is the null space of this matrix.
- The **minimum distance** of a code $\mathscr{C} \subseteq \mathbb{F}_q^n$ is defined as:

$$d(\mathscr{C}) = \min\{d_H(x,y) : x, y \in \mathscr{C}, x \neq y\}.$$

- A linear $[n,k,d]_q$–code **corrects** $t$ errors if ond only if, $t < \lfloor \frac{d-1}{2} \rfloor$.
- The problem of **decoding** is an **NP problem**.

## 3. Decoding Algorithms

Let $\mathscr{C}$ be an $[n,k]_2$ linear code. If $c \in \mathscr{C}$ is the transmitted codeword and $y \in \mathbb{F}_2^n$ is the received word, then $\{i : y_i \neq c_i\}$ is the set of error positions. Let $e = y - c$, then $e$ is called the error vector. All known decoding algorithms that works for linear codes have exponential complexity. We consider some of the most effective ones, those based on information sets. We are going to explain the first algorithm of this type, introduced by Prange in 1962. This algorithm suppose that there are no errors in the choosen information set. Otherwise, we have to repeat until finding a good information set. Other decoding algorithms that we have studied are: Lee-Brickell (1988) and Stern (1989) that allows another error pattern (see Figure 2).

**Algorithm 2:** *Prange's Algorithm*

**Input**: A $[n,k]_2$ linear code $\mathscr{C}$, a generator matrix $G \in \mathbb{F}_2^{k \times n}$ of $\mathscr{C}$, a received vector $y \in \mathbb{F}_2^n$ and an integer $w \geq 0$.

**Output**: A vector $e \in \mathbb{F}_2^n$ of weight $w$ such that $y - e \in \mathscr{C}$.

**Step 1**: Choose an information set $I$ randomly.

**Step 2**: Replace $y \to e = y - y_I G_I^{-1} G$. If $w_H(e) > w$, go to **Step 1**.

**Step 3**: Give the vector $e$ back.

**Figure 2:** *Distribution of errors*