Tema 2. Códigos Lineales

Justo Peralta López

Juan Antonio López Ramos
Dpto. Álgebra y Análisis Matemático

Resumen: *En el primer apartado expresamos los códigos lineales como subespacios vectoriales de V (n, q), el espacio vectorial formado por todos los vectores de longitud n sobre GF(q). El hecho de que sea un subespacio, nos permitir definir el código especificando su base. Además, la codificación se simplifica ya que sólo tenemos que asignar a cada mensaje una combinación de los elementos de la base. En el segundo punto, mostramos cómo modificar la matriz generadora del código para poder realizar una codificación más eficiente. Para ello daremos un algoritmo y realizaremos varios ejemplos con diferentes alfabetos. En el tercer y cuarto punto mostramos cómo se codifica y decodifica los códigos lineales. Observaremos la diferencia de usar un código sistemático o en su forma estandar y analizaremos, mediante un ejemplo, sus diferentes implementaciones en un circuito digital. En cuanto a la decodificación veremos un primer método utilizando array de decodificación y mencionaremos el problema de la decodificación incompleta y su relación con códigos perfectos. Este método será mejorado utilizando síndromes. En ambos casos, dejaremos clara la base matemática que hace posible ambos métodos de decodificación. Finalmente, en el quinto punto, mostramos como se puede calcular la distancia mínima del código observando las columnas de la matriz de paridad. Estos teoremas serán necesarios para la definición de códigos Hamming a partir de su matriz de paridad.*

3.1. Introducción

Definición 3.1.1. *Un código C se dice que es lineal si y solo si la suma de palabras de C pertenece a C.*

Definición 3.1.2. *Sea $V_n = \{(a_0, a_1, \ldots, a_n) | a_i \in GF(q)\}$ un espacio vectorial. Un subconjunto C de V_n es un código lineal si y solo si es un subespacio de V_n.*

Teorema 3.1.1. *Sea C un código lineal y $W(C)$ el menor de los pesos de las palabras de C distintas de cero. Entonces*

$$d_{min}(C) = W(C)$$

Demostración. Sean x, y dos palabras del código tal que su distancia coincida con la distancia mínima. Entonces

$$d_{min}(C) = d(x, y) = w(x - y) \geq W(C)$$

ya que al ser C un código lineal, $x - y \in C$. Por otra parte, existirá una palabra $x \in C$ tal que

$$W(C) = w(x) = d(x, 0) \geq d_{min}(C)$$

Luego, $d_{min}(C) = W(C)$ □

3.2. Ventajas de Códigos Lineales

- Si M es el número de palabras del código, ahora, para calcular su mínima distancia solo hay que hacer $M - 1$ comparaciones. Antes hacia falta $\binom{M}{2}$ comparaciones, es decir, todos con todas de dos en dos.
- Para describir un código no lineal, teníamos que indicar todas sus palabras del código. Ahora al ser un subespacio vectorial solo tenemos que indicar su base.
- Los algoritmos de codificación y decodificación son muy sencillos.

Como desventaja, podemos citar que todo código lineal tiene que ser un q-código, donde q es de la forma p^n con p primo. Esto es debido a que todo código lineal deber ser un K-espacio vectorial, donde K es un cuerpo, y todo cuerpo tiene un tamaño de p^n elementos.

A partir de ahora, para notar a un $[n, M, d]$-código lineal, lo haremos por (n, k, d)-código, donde n es la longitud de las palabras del código, d la distancia mínima y k es la dimensión como k-espacio vectorial del código. Fíjese que en el caso de códigos lineales , $M = q^k$, donde q es el tamaño del alfabeto.

Definición 3.2.1. *Una matriz $k \times n$ cuyas filas es la base de un (n, k)-código es llamada matriz generadora del código.*

Ejemplo 3.2.1. *a)* $\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$ *genera el código* $C = \{000, 011, 101, 110\}$ *con parámetros* $(3, 2, 2)$

b) $\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$ *genera un* $(7, 4, 3)$*-código lineal.*

c) Un q-código de repetición sobre $GF(q)$ es un $(n, 1, n)$-código lineal con matriz generadora $[11 \ldots 1]$.

3.3. Código lineales equivalentes

Definición 3.3.1. *Dos códigos son equivalentes si se pueden obtener uno a partir de otro por medio de permutaciones de la posiciones de las palabras del código y multiplicaciones por un escalar no nulo.*

Teorema 3.3.1. *Sea G la matriz generadora de un $[n, k]$-código lineal. Entonces, realizando operaciones sobre sus filas y columnas, se puede transformar en una matriz estándar*

$$G' = [I_k|A]$$

donde I_k es la matriz identidad y A una matriz de dimensión $k \times (n - k)$. Las operaciones que podemos realizar son:

f1) Permutación de filas.

f2) Multiplicación de una fila por un escalar distinto de cero.

f3) Adición de un múltiplo de una fila a otra.

c1) Permutación de columnas.

c2) Multiplicación de una columna por un escalar distinto de cero.

Si las operaciones que realizamos para llegar a G' solo afectan a las filas (operaciones f1,f2 y f3), obtenemos el mismo código. Y en el caso de que afecten a columnas (operaciones c1 y c2), entonces obtenemos el generador de un código equivalente. Veamos a continuación un algoritmo que nos permitirá realizar los cambios en la matriz de forma sistemática.

Algoritmo:

Sea $G = [g_{ij}]$ una matriz de dimensión $(k \times n)$, donde i indica la fila y j la columna. Obsérvese, que el número de filas siempre será mayor o igual que el número de columnas, ya que en caso contrario la matriz G no será una base. Este algoritmo, para un $1 \leq j \leq k$, realiza las operaciones necesarias para que $g_{jj} = 1$ y la columna C_j tenga ceros por encima y por debajo de g_{jj}

Paso 1
- Si $g_{jj} \neq 0$ ir al Paso 2.
- Si $g_{jj} = 0$ y si existe un $i \geq j$ tal que $g_{ij} \neq 0$ intercambiar la fila i por la fila j (lo notaremos por $f_i \leftrightarrow f_j$).
- Si $g_{jj} = 0$ y $g_{ij} = 0$ para todo $i \geq j$, entonces escoger un $h > j$ tal que $g_{jh} \neq 0$ e intercambiar c_j por c_h ($c_j \leftrightarrow c_h$).

Una vez terminado el primer paso, tendremos en la posición g_{jj} un elemento distinto de cero. Obsérvese, que en caso de que dicho elemento sea 0, primero intentamos buscar un intercambio de filas, y si no es posible un intercambio de columnas. Es decir, si es posible, obtendremos la matriz generadora del mismo código, y en caso contrario, la matriz generadora de un código equivalente.

Paso 2 Sustituir la fila f_j por $f_j.g_{jj}^{-1}$

Después de este paso, en la posición g_{jj} de la matriz, tendremos un 1.

Paso 3 Ahora, $g_{jj} = 1$. Para cada $i = 1, 2, \ldots, k$, con $i \neq j$, reemplazar f_i por $f_i - g_{ij} f_j$

La columna c_j está ahora en la forma deseada. Ahora sólo tenemos que repetir el algoritmo anterior para cada columna, es decir, para $j = 1, 2, \ldots, k$. No olvide que todas las operaciones deben ser realizadas sobre el cuerpo en el cual se define el código.

Ejemplo 3.3.1. *Apliquemos el algoritmo a la siguiente matriz generadora y discutamos sobre los parámetros del código que ésta genera.*

$$\begin{bmatrix} 1&1&1&1&1&1&1\\ 1&0&0&0&1&0&1\\ 1&1&0&0&0&1&0\\ 0&1&1&0&0&0&1 \end{bmatrix} \xrightarrow[f_2 \leftrightarrow f_2 - f_1]{f_3 \leftrightarrow f_3 - f_1} \begin{bmatrix} 1&1&1&1&1&1&1\\ 0&1&1&1&0&1&0\\ 0&0&1&1&1&0&1\\ 0&1&1&0&0&0&1 \end{bmatrix} \xrightarrow[f_1 \leftrightarrow f_1 - f_2]{f_4 \leftrightarrow f_4 - f_2}$$

$$\begin{bmatrix} 1&0&0&0&1&0&1\\ 0&1&1&1&0&1&0\\ 0&0&1&1&1&0&1\\ 0&0&0&1&0&1&1 \end{bmatrix} \xrightarrow{f_2 \leftrightarrow f_2 - f_3} \begin{bmatrix} 1&0&0&0&1&0&1\\ 0&1&0&0&1&1&1\\ 0&0&1&1&1&0&1\\ 0&0&0&1&0&1&1 \end{bmatrix} \xrightarrow{f_3 \leftrightarrow f_3 - f_4}$$

$$\begin{bmatrix} 1&0&0&0&1&0&1\\ 0&1&0&0&1&1&1\\ 0&0&1&0&1&1&0\\ 0&0&0&1&0&1&1 \end{bmatrix}$$

Como se puede observar, sólo se han utilizado operaciones sobre filas, luego la matriz generadora en su forma estándar, genera exactamente el mismo código.

En cuanto a la distancia mínima, cada fila es un elemento de la base del código, y por lo tanto un elemento del código. Si observamos el peso de cada uno de los elementos de la base, la distancia mínima del código será como máximo 3. Ahora, es posible que al combinar dos o más elementos de la base, obtengamos una palabra del código con distancia mínima menor que 3, pero fácilmente observamos que ésto es imposible, ya que al sumar dos elementos de la base, obtendremos en los 4 primeras posiciones dos 1 y algún otro en los bits de paridad. En general, obtendremos tantos 1 en los bits de información como elementos de la base combinemos.

Ejemplo 3.3.2. *Apliquemos el mismo algoritmo pero sobre el alfabeto o cuerpo finito $GF(3)$.*

$$\begin{bmatrix} 0&0&0&1&1&1\\ 0&1&1&0&1&2\\ 1&0&2&0&1&1\\ 1&0&0&2&1&1\\ 0&1&0&1&1&2\\ 0&0&1&0&1&1 \end{bmatrix} \xrightarrow{f1 \leftrightarrow f_3} \begin{bmatrix} 1&0&2&0&1&1\\ 0&1&1&0&1&2\\ 0&0&0&1&1&1 \end{bmatrix} \rightarrow \begin{bmatrix} 1&0&2&0&1&1\\ 0&1&1&0&1&2\\ 0&0&0&1&1&1 \end{bmatrix} \xrightarrow{c_3 \leftrightarrow c_4}$$

Obsérvese, que en el último paso realizamos operaciones sobre columnas, luego el código que se obtiene no es el mismo, sino uno equivalente al generado por G.

3.4. Codificación

Sea C un $[n, k]$-código lineal sobre $GF(q)$, con matriz generadora G. C contiene q^k palabras y por lo tanto se puede utilizar para transmitir q^k mensajes distintos. Identificamos cada mensaje por

una k-tupla del espacio vectorial $V(k,q)$. Si denotamos a las filas de G por $r_1,...,r_k$, codificamos cada mensaje $u = u_1u_2...u_k$ por

$$uG = \sum_{i=1}^{k} u_i r_i$$

Es decir, a cada mensaje le hacemos corresponder una combinación lineal de los elementos de la base del subespacio vectorial C.

Ejemplo 3.4.1. $G_1 = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 2 & 2 \end{bmatrix}$ *y* $G_2 = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 1 & 2 & 0 & 2 \end{bmatrix}$ *son dos matrices que generan el mismo código.*

Si G es la matriz generadora, la codificación se realiza de forma más eficiente si ésta está en su forma estándar. Para ver la razón de esta última afirmación, supongamos que G es de la forma

$$\begin{bmatrix} 1 & 0 & 0 & \dots & 0 & P_{0,0} & P_{0,1} & \dots & P_{0,n-k-1} \\ 0 & 1 & 0 & \dots & 0 & P_{1,0} & P_{1,1} & \dots & P_{1,n-k-1} \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 00 & \dots & 1 & P_{k-1,0} & P_{k-1,1} & \dots & P_{k-1,n-k-1} \end{bmatrix}$$

Sea $d = (d_0, d_1, \dots, d_{k-1})$ el mensaje o dato a codificar. Entonces codificamos por $c = uG$ donde $G = [I_k P]$. Luego la palabra codificada será

$$u = d_0, d_1, \dots, d_{k-1}, P_k, P_{k+1}, \dots, P_{n-1}$$

donde

$$\begin{array}{rcl} P_k & = & d_0P_{0,0} + d_1P_{1,0} + \cdots + d_{k-1}P_{k-1,0} \\ P_{k+1} & = & d_0P_{0,1} + d_1P_{1,1} + \cdots + d_{k-1}P_{k-1,1} \\ \vdots & & \vdots \\ P_k & = & d_0P_{0,n-k-1} + d_1P_{1,n-k-1} + \cdots + d_{k-1}P_{k-1,n-k-1} \end{array}$$

Como se puede observar, cuando codificamos un dato utilizando la matriz generadora en su forma estándar, la codificación se simplifica, ya que sólo tenemos que colocar el dato seguido de sus bits de paridad, y estos últimos se calculan de forma sencilla. Esta sencillez, es todavía más valiosa cuando el proceso de codificación se implementa de forma física en un circuito integrado. Los siguientes ejemplos nos muestran dos códigos, uno de ellos en su forma sistemática y el circuito de éste último.

Ejemplo 3.4.2. *Sea* $G_1 = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 0 & 1 & 2 & 2 \end{bmatrix}$ *y* $G_2 = \begin{bmatrix} 1 & 0 & 2 & 1 \\ 1 & 2 & 0 & 2 \end{bmatrix}$ *los dos matrices generadores del mismo código. La siguiente tabla nos muestra como se realiza la decodificación en ambos casos.*

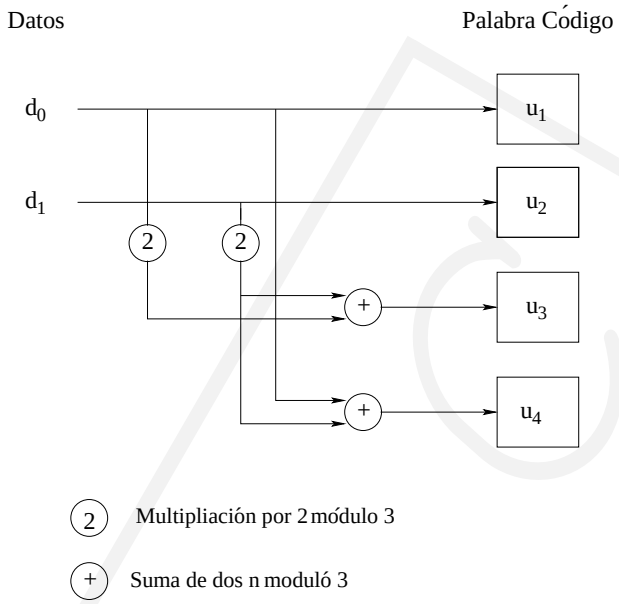

Figura 3.1: Circuito de un $(4, 2)$-código sistemático sobre $GF(3)$

d_0	d_1	u_0	u_1	u_2	u_3	u_0	u_1	u_2	u_3
0	0	0	0	0	0	0	0	0	0
1	0	1	0	2	1	1	0	2	1
2	0	2	0	1	2	2	0	1	2
0	1	0	1	2	2	1	2	0	2
0	2	0	2	1	1	2	1	0	1
1	1	1	1	1	0	0	2	1	1
2	1	2	1	0	1	0	2	1	1
1	2	1	2	0	2	0	1	2	2
2	2	2	2	2	0	1	1	1	0

El circuito que realzaría la codificación del código sistemático sería el de la figura 3.1
El circuito para un código sistemático cualquiera viene dado por la figura 3.2.
Hasta ahora hemos resuelto la primera parte del esquema de transmisión de la figura 3.3.

3.5. Decodificación

Sea C un $[n, k]$-código lineal , $x = (x_1, x_2, ..., x_n) \in C$ la palabra código enviada, $y = (y_1, y_2, ..., y_n)$ la palabra código recibida y $e = x - y = (e_1, e_2, ..., e_n)$ el error cometido durante la transmisión. C tiene estructura de subespacio vectorial de $V(n, q)$, por lo tanto tiene estructura de grupo respecto a la suma de la misma forma que ocurre con $V(n, q)$. Ya que C es un subgrupo de $V(n, q)$, podemos definir la siguiente relación de equivalencia: $x \sim y$ si y solo si $x - y \in C$. Definimos el conjunto

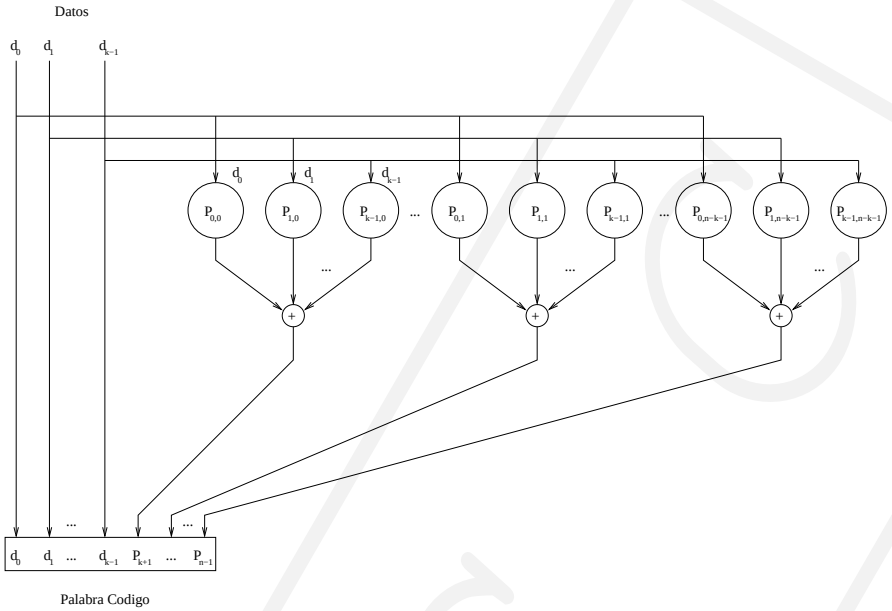

Figura 3.2: Circuito de un (n, k)-código sistemático sobre $GF(q)$

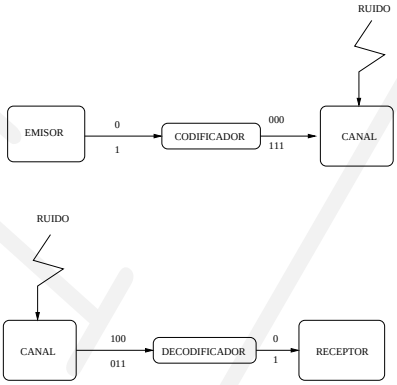

Figura 3.3: Esquema de Transmisión con ruido

$0000+C$	$=$	0000	1011	0101	1110
$0001+C$	$=$	0001	1010	0100	1111
$0010+C$	$=$	0010	1001	0111	1100
$1000+C$	$=$	1000	0011	1101	0110

Cuadro 3.1: Array de decodificación de C

cociente V/C y tomamos como representante de cada clase el vector de esa clase de mínimo peso y lo llamamos líder de la clase.

Definición 3.5.1. *Llamamos array estándar de decodificación al array cuyas filas son las clases de V/C.*

El siguiente algoritmo de decodificación fue introducido por Slepian(1960)

Definición 3.5.2. *Supongamos que C es un $[n,k]$-código lineal sobre $GF(q)$ (cuerpo de Galois de q elementos) y que x es cualquier vector o palabra en $V(n,q)$. Entonces los conjuntos de la forma $x+C=\{x+y|y\in C\}$ son las clases o coclases de C.*

Teorema 3.5.1. *Sea C un $[n,k]$-código sobre $GF(q)$. Entonces*

1. *Todo vector de $V(n,q)$ está en alguna clase de C.*
2. *Toda clase tiene exactamente q^k vectores.*
3. *Dos clases o son disjuntos o coinciden.*

Ejemplo 3.5.1. *Sea $C=\{0000,1011,0101,1110\}$, entonces sus clases vienen dadas por la tabla 3.1*

Algoritmo

1. Buscamos la palabra recibida y, e identificamos la clase a la que pertenece.
2. Tomamos como error cometido e=líder de la clase.
3. Decodificamos por x=y-e

Realmente estamos decodificando la palabra recibida y, por la palabra del código C que más se le parece, es decir, por aquella de C cuya distancia mínima con y es menor. Ya que la probabilidad de error es $p << 1/2$ la palabra de C más parecida con y, será la que tenga mayor probabilidad de haber sido enviada. Este es el motivo por el cual como representante de cada clase se tome el de menor peso, ya que éste indicará el error cometido si la palabra y pertenece a esa clase; y los errores más probables son los de menor peso.

El principal inconveniente de este método es que para código grandes, lo cual es lo normal en la práctica, hay que almacenar demasiados vectores.

3.6. Síndromes

Definición 3.6.1. *Dado un código lineal C, llamamos código dual de C, y lo notamos por $C^{\perp}$, al conjunto de los vectores del espacio vectorial $V(n,q)$ que son ortogonales a toda palabra de C, y notamos por $u.v$ al producto escalar de dos vectores.*

$$C^{\perp} = \{v \in V(n,q) | v.u = 0 \; \forall u \in C\}$$

$C^{\perp}$ es un subespacio vectorial de $V(n,q)$ de dimensión $n-k$, luego será un $[n, n-k]$-código.

Ejemplo 3.6.1.

- $C = \{0000, 1100, 0011, 1111\}$, $C^{\perp} = C$
- $C = \{000, 110, 011, 101\}$, $C^{\perp} = \{000, 111\}$

La matriz de paridad H para un $[n,k]$-código C es el generador de $C^{\perp}$, lo cual nos permite definir un código de la siguiente forma

$$C = \{x \in V(n,q) | xH^t = 0\}$$

Ejemplo 3.6.2. *Si* $H = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$ *entonces*

$$C = \{(x_1, x_2, x_3, x_4) \in V(4,2) | x_1 + x_2 = 0, x_3 + x_4 = 0\}$$

Lema 3.6.1. *Sea C un $[n,k]$-código lineal con matriz generadora G. Entonces un vector $u \in V(n,q)$ pertenece a $C^{\perp}$ si y sólo si es ortogonal a todas las filas de G. Es decir,*

$$u \in C^{\perp} \leftrightarrow vG^t = 0$$

Teorema 3.6.1. *Para todo $[n,k]$-código lineal C, $(C^{\perp})^{\perp} = C$.*

Teorema 3.6.2. *Sea C un $[n,k]$-código lineal sobre $GF(q)$. Entonces el dual de C es un $[n, n-k]$-código lineal. Además, si H es la matriz generadora de $C^{\perp}$, a la cual llamaremos matriz de paridad, entonces $GH^t = 0$*

Teorema 3.6.3. *Si $G = [I_k | A]$ es la matriz generadora de un $[n,k]$-código, entonces la matriz de paridad de C es $H = [-A^t | I_{n-k}]$*

Ejemplo 3.6.3. *Sea* $G = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$

la matriz generadora de C. Su matriz de paridad o matriz generadora de su dual será

$$H = \left[\begin{array}{cccc|ccc} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

Definición 3.6.2. *Sea H una matriz de paridad de un $[n,k]$-código C. Entonces para cualquier vector $y \in V(n,q)$, se define el síndrome de y como sigue*

$$S(y) = yH^t$$

- Si las columnas de H son $h_1, h_2, \ldots, h_{n-k}$, entonces

$$S(y) = \sum y_i * h_i$$

- $S(y) = 0 \Leftrightarrow y \in C$

Lema 3.6.2. *Dos vectores u y v de $V(n,q)$ están en la misma clase de C si y solo si tienen el mismo síndrome.*

Demostración. Sea u, v dos elementos de la misma clase, por definición de clase o coclase, $u+C = v+C$. Por lo tanto, $u - v \in C$ y por ser una palabra del código $(u-v)H^t = 0$ y $uH^t = S(u) = vH^t = S(v)$ □ Luego existe una correspondencia uno a uno entre las clases de V/C y los síndromes. Ahora, en vez de almacenar todo el array de decodificación, sólo almacenamos el líder de las clases y su correspondiente síndrome.

Decodificación

Enviamos x y recibimos y.

1. Calculamos $S(y) = yH^t$
2. Si $S(y) = 0$ no se ha producido ningún error.
3. Si $S(y) \neq 0$ se ha producido algún error. Buscamos un líder con el mismo síndrome y decodificamos por y-líder.

Ejemplo 3.6.4. *Sea C un código lineal generado por* $G = \begin{bmatrix} 1011 \\ 0101 \end{bmatrix}$.

Líder	*Síndrome*
0000	*00*
1000	*11*
0100	*01*
0010	*10*

Entonces la palabra $y = 1001$ *tiene síndrome* 10 *y decodificamos por* 1011.

3.7. Decodificación Incompleta

Si podemos corregir t errores, dividimos el array en dos partes. La primera parte con aquellos lideres de peso $\leq t$, y la segunda parte con el resto. Si el vector o palabra recibida está en la primera parte decodificamos como ya sabemos. Y si está en la segunda parte, sólo podemos detectar el error y no sabemos corregir.Cuando no se produce una decodificación incompleta, es decir, cuando siempre corregimos (aunque en algunos casos se corrija de forma errónea), decimos que el código es un *código perfecto*.

Ejemplo 3.7.1. *Sea C un $[6,2,3]$ código lineal generado por* $G = \begin{bmatrix} 10110 \\ 01011 \end{bmatrix}$

El array de decodificación vendrá dado por

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

Si recibimos 10011 podemos decir que se han producido 2 errores, pero sabemos como corregirlo, ya que la palabra enviada podría ser 01011 o 10110.

3.8. Distancia mínima y matriz de paridad

Hasta ahora, en códigos lineales, para calcular la distancia mínima de un código, calculábamos el menor de los pesos de todas las q^k palabras del código, siendo k la dimensión de éste. Este cálculo se puede hacer de forma más sencilla observando la matriz de paridad del código.

Teorema 3.8.1. *Para cualquier palabra $u \in C$ de pesos d, d columnas de H son linealmente dependientes.*

Demostración. Sea $u = (u_1, u_2, \ldots, u_n)$ una palabra del código con peso $w(u) = d$. Entonces si H es la matriz de paridad de C se verifica $uH^t = (u_1, u_2, \ldots, u_n)(h_1, h_2, \ldots, h_n) = u_1h_1 + u_2h_2 + \cdots + u_nh_n) = 0$ donde h_i es la i-ésima columna de H. Como d de la u_i son distinto de 0, ya que $w(u) = d$, d columnas de H son linealmente dependientes. □

Teorema 3.8.2. *Un código lineal C tiene distancia mínima como mínimo d si y sólo si cualquier $d-1$ o menos columnas de H son linealmente independientes.*

Demostración. Supongamos que la distancia mínima es d. La distancia de un código es el mínimo de sus pesos. Luego si existiera $d-1$ o menos columnas de H linealmente dependientes, entonces en el código C existe una palabra de peso $d-1$ o menos, lo cual contradice la hipótesis. □

3.9. Ejercicios

1. Mostrar que el conjunto de todos los vectores de peso par de $V(n,2)$,E_n, es lineal. ¿Cuáles son los parámetros de E_n ?. Escribir su matriz gene-radora.

2. ¿Es un $[11,24,5]$-código un código lineal?.

3. Probar que en un código binario lineal, todas las palabras tienen peso par o mitad par y mitad impar.

4. Sea C_1 y C_2 dos códigos binarios lineales con matrices generadoras

$$G_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix} \quad G_2 = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

   Cálcular las palabras del código y encontrar la mínima distancia.

5. Sea C un código lineal sobre $GF(3)$ con matriz

$$G = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 2 \end{bmatrix}$$

   Listar las palabras de C, calcular la mínima distancia y decidir si es un código perfecto.

6. Sea C un código lineal con la siguiente matriz generadora

$$G = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

   Encontrar el generador en forma estandar

7. Construir el array de decodificación los códigos generados por

$$G_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad G = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \quad G = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

   Usar el último para

   a) Decodificar las palabras 11111 y 01011.
   b) Dar un ejemplo de un error de peso 2 corregido.
   c) Dar un ejemplo de un error de peso 2 sin corregir.

8. Construir la tabla de síndromes para un $[7, 4, 3]$-código binario perfecto con generador

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

   Usar la tabla para decodificar los vectores

$$0000011 \quad 1111111 \quad 1100110 \quad 1010101$$

9. Sea C un código lineal con generador

$$G = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 2 & 0 & 1 & 1 \end{bmatrix}$$

a) Calcular la matriz generadora estandar

b) Encontrar la matriz de paridad

c) Usar los sindromes para decodificar 2121,1201 y 2222.

10. Sea un $[8, 4]$-código sobre $GF(2)$ cuyas ecuaciones de paridad vienen dadas por

$$P_6 = M_1 + M_2 + M_4$$
$$P_7 = M_1 + M_3 + M_4$$
$$P_8 = M_1 + M_2 + M_3$$
$$P_9 = M_2 + M_3 + M_4$$

Encontrar la matriz generadora, de paridad y d_{min}.