

# DETECCION DE ERRORES

## DISTANCIA DE UN CÓDIGO

Supongamos un código binario de 3 Dígitos  $\Rightarrow u^n \Rightarrow 2^3 = 8$  Palabras Códigos

000	La Distancia de un código es la cantidad de dígitos (Binits) que cambian entre
001	
010	dos palabra del código. Ejemplo:
011	Dist. entre 100 y 101 es $d=1$
100	Dist. entre 000 y 011 es $d=2$
101	Dist. entre 011 y 100 es $d=3$
110	
111	

## DISTANCIA MÍNIMA DE UN CÓDIGO

Es la menor cantidad de dígitos que cambian entre dos palabras del mismo código.

En nuestro caso  **$d_{min} = 1$**

Entonces si al transmitir una palabra código y por efecto de ruido se cambia uno de los dígitos la palabra código se convierte en otra palabra del mismo código  $\Rightarrow$  **ERROR**

Si se transmite la palabra **100** y se recibe la palabra **101** que también pertenece al código no es posible detectar el error

Por lo tanto si se pretende detectar un **error simple** en la palabra código que se recibe se deberá aumentar la Distancia Mínima del Código

## BIT DE PARIDAD

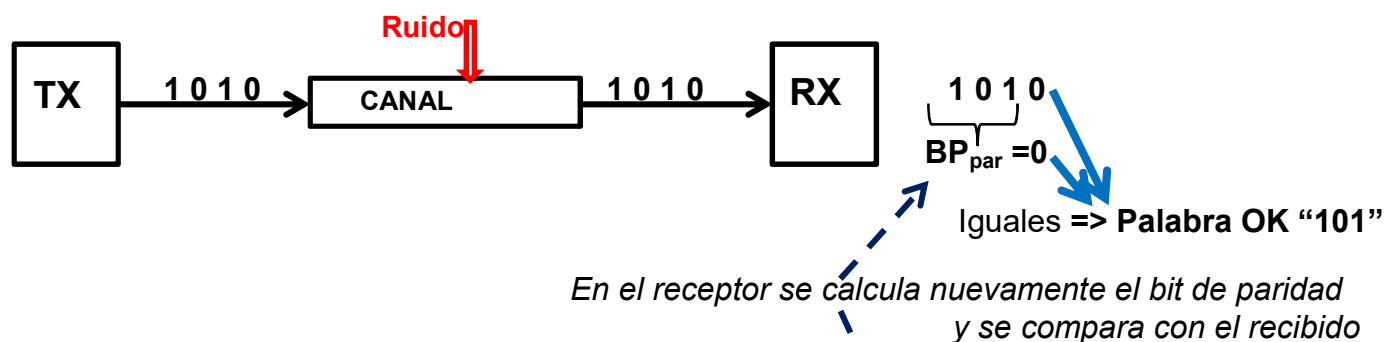
Una manera de aumentar la distancia mínima es agregar un dígito a la derecha que represente la paridad par o impar de unos de la palabra código, es decir la nueva palabra tendrá una cantidad par o impar de unos según:

$I_2 I_1 I_0 BP$	$BP_{par} = I_0 + I_1 + I_2 + \dots + I_n$	$I_2 I_1 I_0 BP$	$BP_{impar} = I_0 + I_1 + I_2 + \dots + I_n + 1$
0 0 0 0	<b>+ : Suma sin acarreo</b>	0 0 0 1	<b>+ : Suma sin acarreo</b>
0 0 1 1		0 0 1 0	
0 1 0 1		0 1 0 0	
0 1 1 0		0 1 1 1	
1 0 0 1		1 0 0 0	
1 0 1 0		1 0 1 1	
1 1 0 0		1 1 0 1	
1 1 1 1		1 1 1 0	

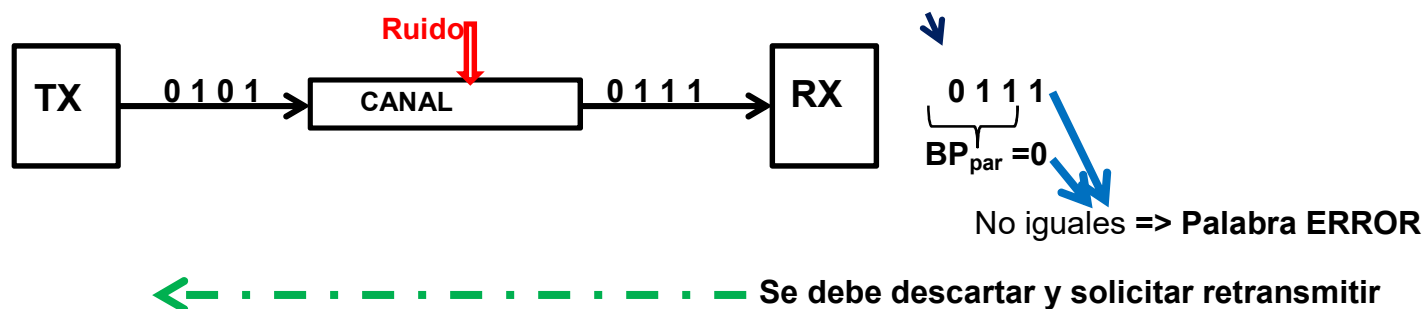
Ahora nuestro código si bien tiene 4 dígitos  $\Rightarrow u^n \Rightarrow 2^4 = 16$  Combinaciones , el código solo usa 8 palabras códigos. Las restantes 8 combinaciones que no se utilizan representan los errores, se dice que esas combinaciones absorben el error simple.

### Ejemplo:

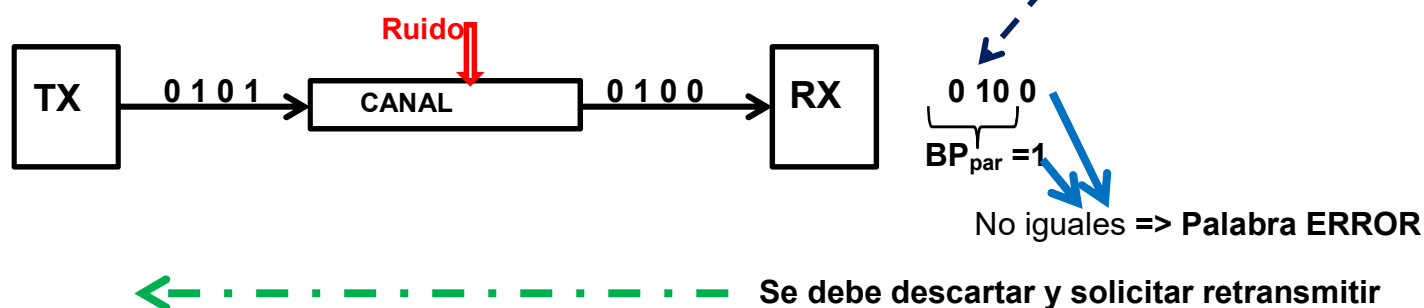
Se transmite 1010 por un canal real:



Se transmite 0101 por un canal real:



Se transmite 1010 por un canal real:



(Aunque la parte de Información está correcta no se puede saber dónde está el error)

## CÁLCULO DEL RENDIMIENTO Y LA REDUNDANCIA

Si consideramos que en el código original los mensajes son equiprobables resulta:

La cantidad de mensajes (palabra código)  $M = u^n = 2^3$

La Probabilidad de uno de mensajes (palabra código) es  $p_{(M)} = 1 / M = 1 / u^n$

Al ser equiprobables la cantidad de Información promedio (Entropía de la fuente) resulta igual a la Información de un mensaje:

$$H(s) = I_{(M)} = \lg 1/p_{(M)} = \lg 1 / 1/M = \lg M = \lg u^n = \lg 2^3 = 3 \cdot \lg 2 = 3 \text{ bit}$$

La longitud promedio del código  $L = \sum p_i \cdot l_i$

Pero como mensajes son equiprobables y el código es de longitud constante resulta

$$L = 3 \text{ binit}$$

En este caso si calculamos el rendimiento  $\text{Rend.} = H / L = 3/3 = 1$  o sea un rendimiento del **100%**

Y la redundancia  $\text{Redun.} = 1 - \text{Rend.} = 1 - 1 = 0$  es decir el código resulta el de menor longitud promedio y no tiene redundancia

Ahora si calculamos estos indicadores para el mismo código con bit de paridad para la detección de errores simples resulta:

$$H(s) = I_{(M)} = \lg 1/p_{(M)} = \lg 1 / 1/M = \lg u^n = \lg 2^3 = 3 \lg 2 = 3 \text{ bit}$$

Ahora si bien también los mensajes también son equiprobables y el código es de longitud constante pero ahora es:

$$L = 4 \text{ binit}$$

Por lo tanto el rendimiento  $\text{Rend.} = H / L = 3/4 = 0,75$  o sea un rendimiento del **75%**

Y la redundancia  $\text{Redun.} = 1 - \text{Rend.} = 1 - 0,75 = 0,25$ , es decir hay una redundancia de **0,25**

Lo cual se ve claramente ya que estamos usando cuatro bit para codificar ocho mensajes.

Esta baja en la calidad del código en cuanto a su rendimiento trae el beneficio de la detección de errores simples.

# CORRECCIÓN DE ERRORES SIMPLES

## PARIDAD POR FILAS Y COLUMNAS

En este caso de arma un bloque de palabras códigos de la siguiente manera:

1001010

1010101

1100110

0100011

Y se agrega una columna y una fila de paridad par por ejemplo:

10010101

10101010

11001100

01000111

10110100

Se transmite el bloque al recibirlo se verifican la paridad de cada fila y cada columna y se comparan

Bloque recibido	Bit Par generado p/fila en el Receptor	Comparación
10010101	1	= Fila Ok
10101010	0	= Fila Ok
11011100	1 ← Hay error en esta fila	≠ Fila c/error
01000111	1	= Fila Ok
10110100	0	= Fila Ok

10100100 Bit Par generado p/columna en el Receptor

===~~×~~=== Comparación

↑ Hay error en esta columna

Por lo tanto el dígito con error se encuentra en la intersección de la fila y columna donde los bits de paridad recibidos y generados en el receptor no coinciden.

Entonces la información correcta es:

1001010

1010101

1100110

0100011

Es decir, recuperada la información correcta, los dígitos de paridad se descartan.

Otro Ejemplo:

Bloque recibido	Bit Par generado p/fila en el Receptor	Comparación
10010101	1	= Fila Ok
<del>10101011</del>	0 ← Hay error en esta fila	<del>=</del> Fila c/error
11001100	0	= Fila Ok
01000111	1	= Fila Ok
<del>10110100</del>	0	= Fila Ok

10100101 Bit Par generado p/columna en el Receptor

~~=====~~ Comparación

↑ Hay error en esta columna

Por lo tanto el dígito con error se encuentra en uno de los bits de paridad.

Entonces la información correcta es:

1001010

1010101

1100110

0100011

Nuevamente los dígitos de paridad se descartan.

El inconveniente de este método es que se debe armar bloques de palabras códigos, no corrige por palabra

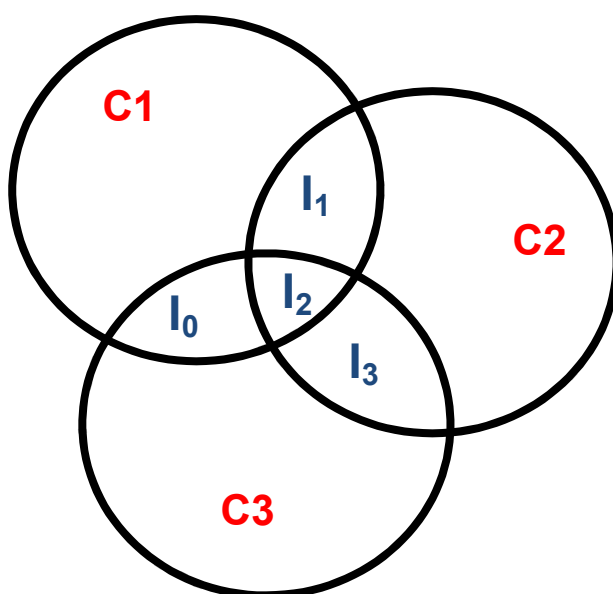
---

## PARIDAD POR ZONAS - CÓDIGO HAMMING

En este caso desarrollaremos “conceptualmente” Códigos Hamming

Consideremos una palabra código de cuatro dígitos  $I_3 I_2 I_1 I_0$

Consideramos tres zonas de control incluyendo un dígito de información en cada intersección de zonas:



Cada dígito de control será la paridad par del conjunto de dígitos de información incluidos en su zona:

$$C1 = I_0 + I_1 + I_2$$

$$C2 = I_1 + I_2 + I_3$$

$$C3 = I_0 + I_2 + I_3$$

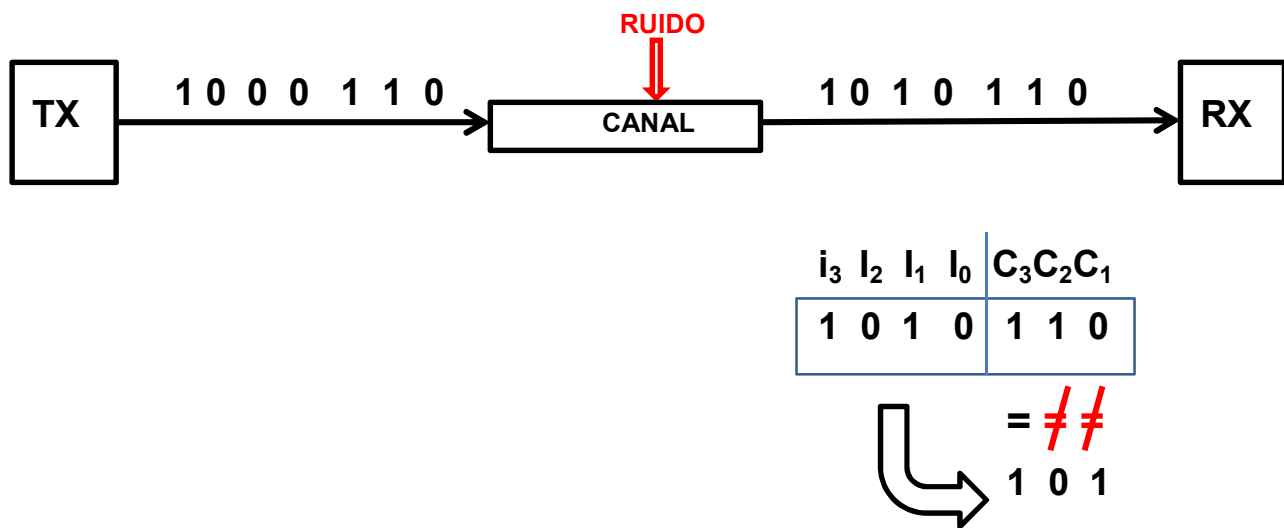
(Cada uno puede armar otras zonas dando por ende otras ecuaciones)

Observar que hay tres dígitos de información que impactan en dos zonas de control respectivamente y el cuarto dígito de información impacta en las tres zonas de control.

Ahora construimos el código completo identificando cada dígito:

$I_3$	$I_2$	$I_1$	$I_0$	$C_3$	$C_2$	$C_1$
0	0	0	0	0	0	0
0	0	0	1	1	0	1
0	0	1	0	0	1	1
0	0	1	1	1	1	0
0	1	0	0	1	1	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	0	0	1
1	0	0	0	1	1	0
1	0	0	1	0	1	1
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
1	1	1	1	1	1	1

Entonces si se transmite las palabras por un canal con ruido se pueden dar las siguientes situaciones :



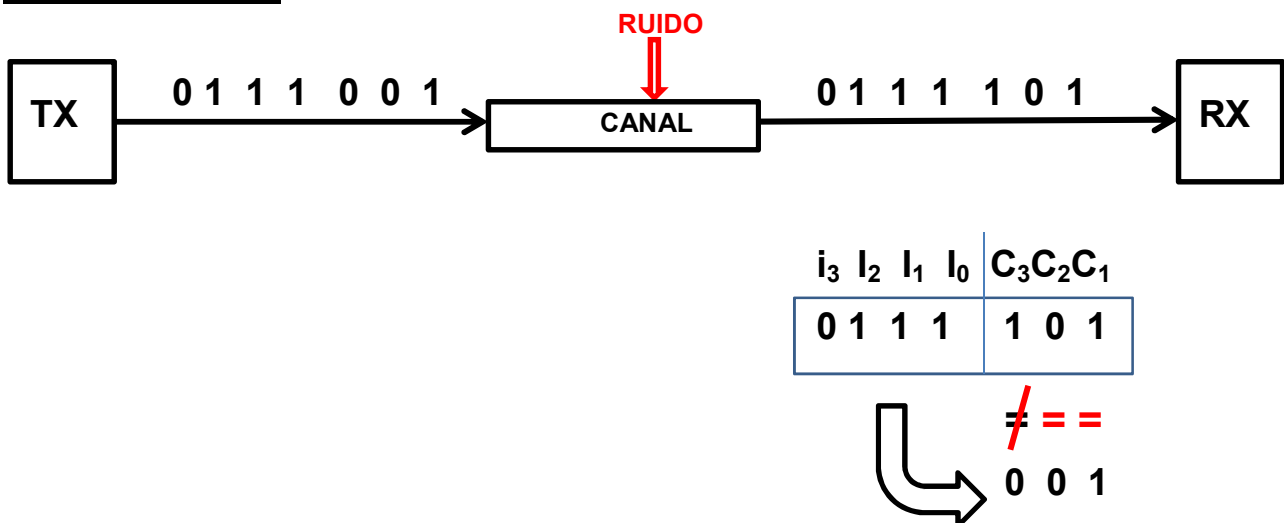
En el receptor genero nuevamente los dígitos de control y los comparo con los dígitos de control recibidos

En este caso los dígitos que cambiaron fueron  $C_1$  y  $C_2$  y observando el diagrama el dígito de Información común a ambos es  $i_1$

Por lo tanto la información correcta es : **1000**

Los dígitos de control se descartan una vez realizada el control y la corrección

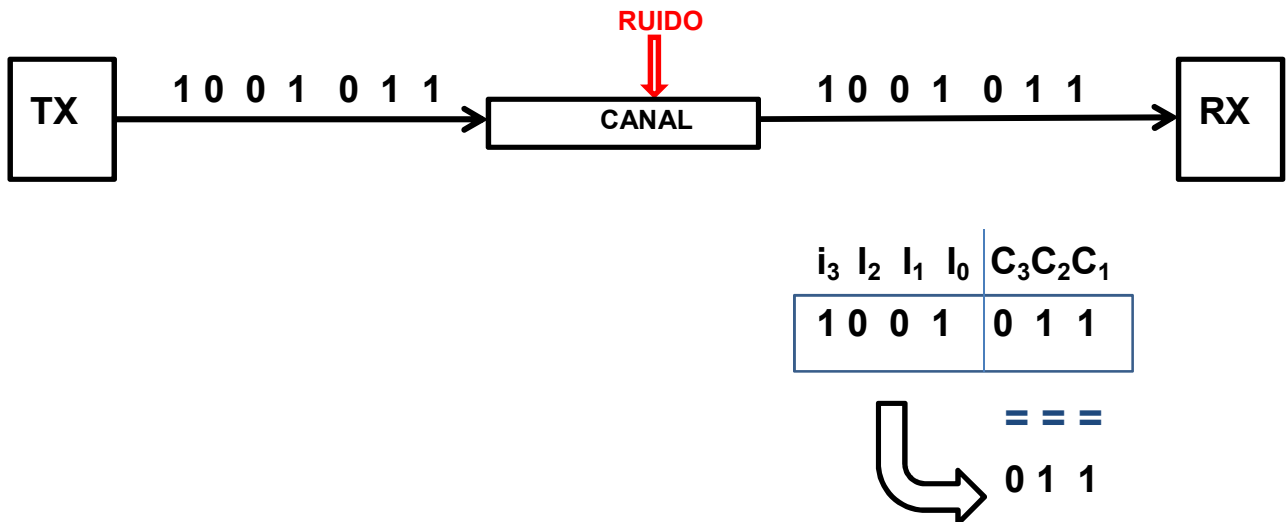
### Otro ejemplo:



Al estar sólo uno de los dígitos de control cambiados , el error no puede estar en ninguno de los dígito de información, por lo que el error está en  $i_3$  y la información correcta es : **0111** (Los dígitos de control se descartan efectuada la verificación).



## Otro Ejemplo:



En este caso como todos los dígitos de control generados en el receptor son iguales a los recibidos , la palabra es correcta : **1001**

## OBSERVACION:

El inconveniente del código Hamming es la de tener dos campos, uno para la información y otro de control.

En nuestro ejemplo en total se transmiten 7 dígitos lo cual daría  $2^7 = 128$  palabras posibles, pero el campo de información es solo de 4 dígitos es decir solo se usan 16 palabras de Información. El campo de control no es información, solo se usa para el control de errores simples.

Si queremos calcular su rendimiento considerando mensajes equiprobables:

$$H(s) = I_{(M)} = \lg 1/p_{(M)} = \lg 1/ 1/M = \lg u^n = \lg 2^4 = 4 \lg 2 = 4 \text{ bit}$$

(Sólo cuatro dig. de inf.,  $i_3 i_2 i_1 i_0$ )

Ahora si bien también los mensajes también son equiprobables y el código es de longitud constante pero ahora es: **L = 7 binits**

Por lo tanto el rendimiento **Rend. = H/ L = 4/7 = 0,571** o sea un rendimiento del **57,1%**

Y la redundancia **Redun. = 1 – Rend. = 1 – 0,571 = 0,43** , es decir hay una redundancia de **0,43**

## CODIGOS LINEALES

Cómo se detalló anteriormente si bien un código de longitud constante de **n** dígitos y **2<sup>n</sup>** palabras equiprobables es compacto con un rendimiento del 100% tiene el inconveniente de que su distancia mínima  $d_m = 1$  con lo cual el error de un dígito en una palabra es decodificada como otra palabra del mismo código, por lo tanto si necesitamos un código para la detección y/o corrección de errores simples se debe aumentar la distancia mínima.

Antes de avanzar haremos un relación entre códigos y Espacios Vectoriales

Consideramos un Espacio Vectorial de **n columnas** y **2<sup>n</sup>** filas con sus elementos binarios

$$\text{E.V} = \begin{matrix} & \text{n columnas} \\ \begin{bmatrix} 0000 \dots 000 \\ 0000 \dots 001 \\ 0000 \dots 010 \\ \dots \dots \dots \\ 1111 \dots 101 \\ 1111 \dots 110 \\ 1111 \dots 101 \end{bmatrix} & \text{2}^n \text{ filas} \end{matrix}$$

Como se necesita un código de n dígitos con posibilidad de detección y corrección de errores simples se debe aumentar la distancia mínima tomado una cantidad menor de palabras códigos (filas)

Esto significa considerar un Sub Espacio Vectorial **V** de **n columnas** y **2<sup>m</sup>** filas con  $m < n$

$$\text{S.E.V.} = \mathbf{V} = \begin{matrix} & \text{n columnas} \\ \begin{bmatrix} 0101 \dots 010 \\ (\dots \mathbf{x}_{ij} \dots) \\ \dots \mathbf{v}_i \dots \\ 1011 \dots 100 \end{bmatrix} & \text{2}^m \text{ filas con } m < n & \text{Filas = Vectores } \mathbf{v}_i \\ & & \text{Vector } \mathbf{v}_i = (\mathbf{x}_{i1}, \mathbf{x}_{i2}, \dots, \mathbf{x}_{ij}, \dots, \mathbf{x}_{in}) \\ & & \text{Siendo el elemento } \mathbf{x}_{ij} = 0 \text{ ó } 1 \end{matrix}$$

Como todo espacio vectorial o sub-espacio vectorial tiene una base generadora:

**n columnas**

$$\mathbf{G} = \begin{bmatrix} \dots\dots\dots \\ \dots \mathbf{g}_i \dots \\ \dots\dots\dots \end{bmatrix} \text{ m filas con } m < n$$

La condición para que **G** sea Base Generadora del **SEV = V** es que sus vectores sean Linealmente Independientes, es decir que ninguno de los vectores **g<sub>i</sub>** resulte de la combinación lineal de los restantes vectores de **G** y que además la combinación lineal de todos los vectores de **G** debe generar el sub espacio vectorial **V**. Es decir:

$$\mathbf{g}_i \neq \sum \mathbf{k}_j \times \mathbf{g}_j \text{ para Todo } j \neq i \quad \text{Siendo } \mathbf{k}_j \text{ una constante que puede vale } 0 \text{ ó } 1$$

$\mathbf{v}_i = \mathbf{k}_1 \times \mathbf{g}_1 + \mathbf{k}_2 \times \mathbf{g}_2 \dots\dots\dots + \mathbf{K}_m \times \mathbf{g}_m$  Es decir cada vector de **V** es una combinación lineal de los vectores de **G**. La base **NO ES UNICA, HAY MÁS DE UNA**  
Recordar que la suma + es suma sin acarreo.

Así mismo existe otro Sub Espacio vectorial que llamaremos **V'** cuyas dimensiones son:

$$\text{S.E.V.} = \mathbf{V}' = \begin{bmatrix} \text{n columnas} \\ 0101\dots010 \\ (\dots\mathbf{x}_{ij}\dots) \\ \dots\mathbf{v}_i\dots \\ 1011\dots100 \end{bmatrix} \quad 2^{n-m} \text{ filas}$$

**Filas = Vectores  $\mathbf{v}'_i$**

**Vector  $\mathbf{v}'_i = (\mathbf{x}'_{i1}, \mathbf{x}'_{i2}\dots\mathbf{x}'_{ij}\dots\mathbf{x}'_{in})$**

Siendo el elemento  $\mathbf{x}'_{ij} = 0 \text{ ó } 1$

Nuevamente todo espacio vectorial o sub-espacio vectorial tiene una base generadora:

**n columnas**

$$\mathbf{H} = \begin{bmatrix} \dots\dots\dots \\ \dots \mathbf{h}_i \dots \\ \dots\dots\dots \end{bmatrix} \text{ n-m filas}$$

La condición para que **H** sea Base Generadora del **SEV = V'** es que sus vectores sean Linealmente Independientes, es decir que ninguno de los vectores **h<sub>i</sub>** resulte de la

combinación lineal de los restantes vectores de **H** y que además la combinación lineal de todos los vectores de **H** debe generar el sub espacio vectorial **V'**. Es decir:

$$h_i \neq \sum k_j \times h_j \text{ para Todo } j \neq i \quad \text{Siendo } k_j \text{ una constante que puede vale 0 ó 1}$$

$v'_i = k_1 \times h_1 + h_2 \times h_2 \dots\dots\dots + K_{n-m} \times h_{n-m}$  Es decir cada vector de **V'** es una combinación lineal de los vectores de **H**. – **Recordar que la base no es única, se puede elegir cualquiera que cumpla las condiciones..**

Este Sub Espacio vectorial **V'** resulta ortogonal al sub espacio vectorial **V** , es decir que se cumple la siguiente condición:

Para todo  $v_i \in V$  resulta  $v_i \times H^T = 0$  siendo  $H^T$  la matriz **transpuesta** de **H**

Para todo  $v'_i \in V'$  resulta  $v'_i \times G^T = 0$  siendo  $G^T$  la matriz **transpuesta** de **G**

Recordar que son productos de matrices ( $v_i$  es una matriz de una sola fila) por lo tanto la propiedad conmutativa no aplica. Solo vale la propiedad asociativa:

A , B y C matrices cualquiera

$$A \times B \neq B \times A$$

$$(A + B) \times C = A \times C + B \times C$$

Dada una matriz A , la matriz  $A^T$  resulta de cambiar filas por columnas.

.....  
Breve descripción de la nomenclatura utilizada:

En mayúscula espacios vectoriales: **V , V' , G, H ,  $G^T$  y  $H^T$**

En minúscula con un subíndice: Vector i de un espacio vectorial:  $v_i , v'_i , g'_i , h_i$

En minúscula con dos subíndices, es un elemento de un vector :  $v_{ij} , v'_{ij} , g_{ij} , h'_{ij}$

K minúscula con un subíndice: Constante de valor 0 ó 1

Vamos a considerar la siguiente identificación de los vectores:  $v_i = (v_{i1} , v_{i2} , \dots\dots\dots v_{in})$

.....  
**APLICACIÓN A CÓDIGOS BINARIOS**

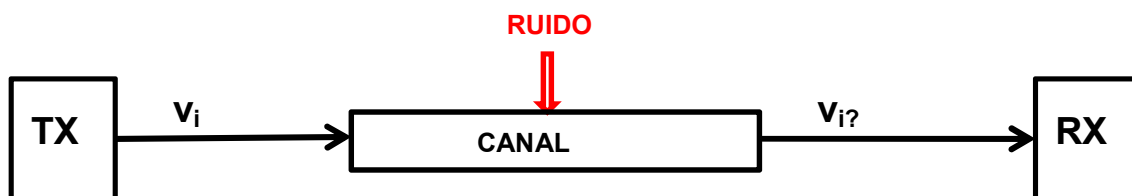
Habiendo hecho el repaso de Algebra, vemos de aplicar lo visto a la transmisión de palabras códigos por un canal con ruido

Consideremos un Código que llamaremos **Código de Grupo V** compuesto por  $2^m$  palabras códigos (vectores ó filas) **n dígitos (n columnas)**

$$V = \begin{bmatrix} 0101 \dots 010 \\ \vdots \\ (\dots x_{ij} \dots) \\ \vdots \\ \dots v_i \dots \\ 1011 \dots 100 \end{bmatrix} \quad \begin{matrix} n \text{ columnas} \\ 2^m \text{ palabras códigos (filas) - con } m < n \end{matrix}$$

Esto permite que V tenga una mayor distancia mínima

Si transmitimos una palabra código  $v_i$  por un canal con ruido:



La palabra  $v_{i?}$ , recibida no puede saberse si tiene error, entonces en el receptor podemos aplicar lo visto anteriormente, es decir si multiplicamos la palabra recibida por  $H^T$  podemos concluir lo siguiente:

$$v_{i?} \times H^T = 0 \Rightarrow \text{Palabra Correcta ya que por la condición de ortogonalidad}$$

**pertenece al Código de Grupo V**

(ó error no detectable, esto lo veremos más adelante)

Si en cambio:

$$v_{i?} \times H^T \neq 0 \Rightarrow \text{Palabra CON ERROR ya que por la condición de ortogonalidad}$$

**NO pertenece al Código de Grupo V**

**Por lo tanto se ha detectado un error simple sin necesidad de agregar dígitos de control, es decir todos los dígitos son de información.**

Ahora bien consideremos un par de palabras códigos pertenecientes a V y modelicemos el error con una palabra que contenga todos ceros menos en la posición que pretendemos generar un error por

ejemplo  $e_a = 000\mathbf{1}00$  (considerando que  $e_a \notin V$ )

Entonces sea  $v_1 \in V$  101110 y  $v_2 \in V$  111001

y el error  $e_a$  000100 modelizo esto mediante la suma sin acarreo (+):

$$\begin{array}{r} v_1 \quad 101110 \\ + \quad + \\ e_a \quad 000100 \\ \hline v_{1ea} \quad 101010 \end{array}$$

$$\begin{array}{r} v_2 \quad 111001 \\ + \quad + \\ e_a \quad 000100 \\ \hline v_{2ea} \quad 111101 \end{array}$$

Entonces aplicamos la detección de errores vista anteriormente suponiendo recibir estas dos palabras códigos con error en el cuarto dígito

$$V_{1ea} \times H^T = (v_1 + e_a) \cdot H^T = \underbrace{v_1 \times H^T}_{= 0 \text{ } (v_1 \in V)} + e_a \times H^T \neq 0$$

$\searrow$   
 $S_{ea}$

$$V_{2ea} \times H^T = (v_2 + e_a) \cdot H^T = \underbrace{v_2 \times H^T}_{= 0 \text{ } (v_2 \in V)} + e_a \times H^T \neq 0$$

$\nearrow$   
 $S_{ea}$

Cómo el error  $e_a$  es el mismo en ambos casos ambos resultados distintos de cero son iguales y se lo denomina Síndrome asociado al error  $e_a$  ( $S_{ea}$ )

En principio cada vez que al recibir una palabra y al multiplicarla por  $H^T$  el resultado dé el síndrome  $S_{ea}$  sabremos que el error está en el cuarto dígito pudiendo entonces corregir , las palabras correctas serán 101110 y 111001

## LA MATRIZ H SE DENOMINA MATRIZ DE CONTROL DE PARIDAD

**Importante:**

Al ser un producto de matrices, puede darse esta situación:

$$e_a \times H^T = S1$$


$$e_b \times H^T = S1$$

Es decir que dos errores simples diferentes generen el mismo Síndrome, en este caso sólo se podrá detectar error  $e_a$  ó  $e_b$ , pero NO corregir.

Si se pretende detectar y corregir errores simples en cada uno de los  $n$  dígitos de la palabra código se debe tener en cuenta lo siguiente:

Dada una palabra código genérica que modelizamos como  $\mathbf{x}_i = (\mathbf{x}_{i1}, \mathbf{x}_{i2}, \dots, \mathbf{x}_{in})$

Si se pretende **detectar y corregir error simple** en **cada uno de los  $n$  dígitos** debe haber tantos **síndromes distintos** como dígitos tenga la palabra código:

$$\mathbf{x}_i = (\mathbf{x}_{i1}, \mathbf{x}_{i2}, \dots, \mathbf{x}_{in})$$

$$\mathbf{s}_{e1}, \mathbf{s}_{e2}, \dots, \mathbf{s}_{en}$$

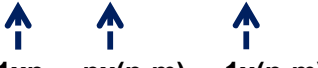
Es decir para detectar y corregir error simple en cada uno de los  $n$  dígitos de la palabra código debe ser la cantidad de síndromes mayor o igual a la cantidad de dígitos.

Veamos esto desde el punto de vista de las dimensiones de los vectores y espacios:

La dimensión de vector que modeliza el error ( $\mathbf{e}_a$ ) es : 1 fila x  $n$  columnas

La dimensión de la matriz de control  $\mathbf{H}^T$  es:  $n$  filas x  $n-m$  columnas

Por lo tanto el Síndrome que es el resultado del producto del vector que modeliza el error por la matriz de control tendrá como dimensión: 1 fila x  $n-m$  columnas

$$\mathbf{e}_a \times \mathbf{H}^T = \mathbf{S}_{ea}$$

$$1 \times n \quad n \times (n-m) \quad 1 \times (n-m)$$

Entonces al ser los elementos binarios, la cantidad de Síndromes distintos será  $2^{n-m} - 1$

(no se considera el cero ya que corresponde a palabra OK )

---

## RESUMEN

**V : Código de Grupo – Dimensión  $2^m$  filas x  $n$  columnas**

**G: matriz generadora de V – Dimensión  $m$  filas x  $n$  columnas**

**V': Espacio Ortogonal a V – Dimensión  $2^{n-m}$  filas x  $n$  columnas**

**H: Matriz generadora de V' llamada matriz Control de paridad – Dimensión  $n-m$  filas x  $n$  columnas**

$$\mathbf{v}_i \times \mathbf{H}^T = 0 \quad \text{siendo } \mathbf{v}_i = (\mathbf{x}_{i1}, \mathbf{x}_{i2}, \dots, \mathbf{x}_{in})$$

$$\mathbf{v}'_i \times \mathbf{G}^T = 0 \quad \text{siendo } \mathbf{v}'_i = (\mathbf{x}'_{i1}, \mathbf{x}'_{i2}, \dots, \mathbf{x}'_{in})$$

$$\mathbf{V}_{1ei} \times \mathbf{H}^T = \mathbf{e}_i \times \mathbf{H}^T = \mathbf{S}_i \quad \text{siendo } \mathbf{e}_i = (0 \dots \mathbf{1} \dots 0)$$

---

**Observación:** Dado el siguiente Código de Grupo, siendo la palabra código  $v_i = (x_1, x_2, x_3, x_4, x_5, x_6)$ :

$$V = \begin{bmatrix} 000000 \\ 001101 \\ 010110 \\ 011011 \\ 100111 \\ 111100 \\ 110001 \end{bmatrix}$$

Observando el Código de Grupo  $V$ , podemos decir que errores **NO SE DETECTAN**, observando lo siguiente: si a la palabra Nula (000000) le generamos un error triple en  $x_1, x_3, x_5$  y si modelizamos el error con una palabra con todos ceros y un uno en las posiciones 1, 3 y 5, es decir  $e_{135} = (101010)$ , resulta:

$$\begin{array}{r} v_1 \\ + \\ e_{135} \\ \hline \end{array} \quad \begin{array}{r} 000000 \\ + \\ 101010 \\ \hline \end{array}$$

$v_1 + e_{135} = 101010$  Y esta palabra pertenece al Código, por lo tanto cuando en el receptor se multiplique por la Matriz  $H^T$  el resultado será CERO como si fuera palabra correcta. Tomemos otra palabra cualquiera del código y generemos el mismo error

$$\begin{array}{r} v_8 \\ + \\ e_{135} \\ \hline \end{array} \quad \begin{array}{r} 110001 \\ + \\ 101010 \\ \hline \end{array}$$

$v_8 + e_{135} = 011011$  Que también pertenece al Código

Esto es porque la palabra generadora de error también pertenece al Código, si probamos con el resto de las palabras código podemos concluir lo siguiente:

**Dado un Código de Grupo, los errores que NO se detectan son aquellos que coinciden con sus palabras códigos o sea donde hay unos en las palabras códigos.**

Entonces observado nuestro código podemos decir que NO se detectan los siguientes errores:

**Error Triple en  $x_3, x_4$  y  $x_6$  . Error Triple en  $x_2, x_4$  y  $x_5$**

**Error Triple en  $x_1, x_3$  y  $x_5$  . Error Triple en  $x_1, x_2$  y  $x_6$**

**Error Cuádruple en  $x_2, x_3, x_5$  y  $x_6$  . Error Cuádruple en  $x_1, x_4, x_5$  y  $x_6$**

**Error Cuádruple en  $x_1, x_2, x_3$  y  $x_4$**

**Todos los demás errores caerán en alguna de las familias asociadas a cada síndrome**