Codificación de Canal

Referencia alternativa: John Proakis, *Digital Communications,* 3ª ed.

Códigos Bloque Lineales

- En la codificación bloque la fuente de información como una secuencia binaria ("0" o "1“) es segmentada en bloques **u** de longitud fija de largo *k* bits y se codifica en bloques **v** de *n* bits de código (>*k*), lo que da una tasa de código $r_c=k/n$. Hay un total de 2^k mensajes o palabras distintas de código válidas (“código bloque”), y un total de 2^n mensajes o palabras distintas posibles de recibir en presencia de posibles errores del canal. Hay correspondencia uno a uno entre **u** y **v**.
- Para facilitar la realización del codificador se utilizan los códigos bloque lineales (*n*,*k*), que se basan en un conjunto de palabras de código independientes, o subespacio ortogonal de dimensión *k*, que al combinarse “linealmente“ (suma módulo 2) ponderadas por el mensaje de datos a codificar generan el mensaje codificado.
- Un código bloque lineal cumple también con que la suma módulo 2 de dos palabras de código genera otra palabra del código.

Codificación de Canal
Códigos Bloque Lineales

Ejemplo de código lineal (7,4)	
Mensajes	**Palabras código**
(0 0 0 0)	(0 0 0 0 0 0 0)
(1 0 0 0)	(1 1 0 1 0 0 0)
(0 1 0 0)	(0 1 1 0 1 0 0)
(1 1 0 0)	(1 0 1 1 1 0 0)
(0 0 1 0)	(1 1 1 0 0 1 0)
(1 0 1 0)	(0 0 1 1 0 1 0)
(0 1 1 0)	(1 0 0 0 1 1 0)
(1 1 1 0)	(0 1 0 1 1 1 0)
(0 0 0 1)	(1 0 1 0 0 0 1)
(1 0 0 1)	(0 1 1 1 0 0 1)
(0 1 0 1)	(1 1 0 0 1 0 1)
(1 1 0 1)	(0 0 0 1 1 0 1)
(0 0 1 1)	(0 1 0 0 0 1 1)
(1 0 1 1)	(1 0 0 1 0 1 1)
(0 1 1 1)	(0 0 1 0 1 1 1)
(1 1 1 1)	(1 1 1 1 1 1 1)

Codificación de Canal
Códigos Bloque Lineales

En un código lineal (*n*,*k*) es posible encontrar *k* palabras de código linealmente independientes, $\mathbf{g}_0$, $\mathbf{g}_1$,..., $\mathbf{g}_{k-1}$ tal que cada palabra código ***v*** es una combinación lineal de esas *k* palabras código indep.:

$$\mathbf{v} = u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \ldots + u_{k-1}\mathbf{g}_{k-1}$$

con u_i= 0 ó 1, i >= 0 e i < *k*.

Representado matricialmente:

$$G = \begin{bmatrix} g0 \\ g1 \\ . \\ . \\ . \\ gk\text{-}1 \end{bmatrix} = \begin{bmatrix} g00 & g01 & g02 & \ldots & g0,n\text{-}1 \\ g10 & g11 & g12 & \ldots & g1,n\text{-}1 \\ . & & & & \\ & & & & \\ & & & & \\ gk\text{-}1,0 & gk\text{-}1,1 & gk\text{-}1,2 & \ldots & gk\text{-}1,n\text{-}1 \end{bmatrix}$$

Codificación de Canal
Códigos Bloque Lineales

Si $\mathbf{u} = (u_0, u_1, \ldots, u_{k-1})$ es el mensaje a codificar, el código correspondiente se puede generar entonces como:

$$\mathbf{v} = \mathbf{u}\,\mathbf{G} = u \cdot \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ . \\ . \\ \mathbf{g}_{k-1} \end{bmatrix} = u_0\mathbf{g}_0 + u_1\mathbf{g}_1 + \ldots + u_{k-1}\mathbf{g}_{k-1}$$

Las filas de la matriz **G** generan el código lineal (*n,k*), y se la llama **matriz generadora** de código. Con esta estructura el codificador solo tiene que almacenar las *k* filas de **G** y formar una combinación lineal de éstas con el mensaje de entrada **u** para genera el mensaje de código a transmitir. La implementación se hace con lógica binaria.

Codificación de Canal
Códigos Bloque Lineales

Ejemplo, el código lineal (7,4) de la tabla anterior, tiene la siguiente matriz generadora:

$$G = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

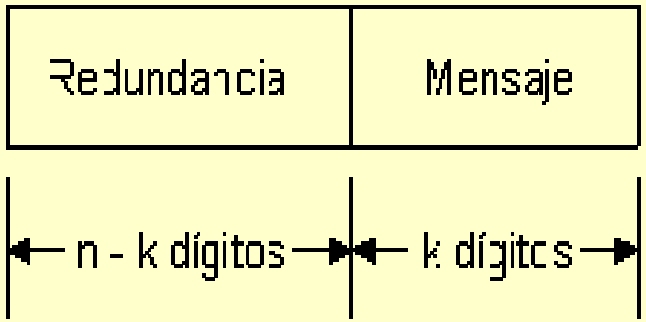
Si el mensaje a codificar es **u** = (1 1 0 1) , el código resultantes es:

$$v = 1\mathbf{g}_0 + 1\mathbf{g}_1 + 0\mathbf{g}_2 + 1\mathbf{g}_3 = (1\ 1\ 0\ 1\ 0\ 0\ 0) + (0\ 1\ 1\ 0\ 1\ 0\ 0) + (1\ 0\ 1\ 0\ 0\ 0\ 1)$$

$$= (0\ 0\ 0\ 1\ 1\ 0\ 1)$$

Codificación de Canal
Códigos Bloque Lineales

Un codificador lineal *sistemático* divide la palabra código en dos partes: la parte del mensaje de *k* bits de información y la parte de redundancia de *n-k* bits de paridad (o comprobación de paridad), como el ejemplo de código (7,4)



Ejemplo de código lineal (7,4)	
Mensajes	**Palabras código**
(0 0 0 0)	(0 0 0 0 0 0 0)
(1 0 0 0)	(1 1 0 1 0 0 0)
(0 1 0 0)	(0 1 1 0 1 0 0)
(1 1 0 0)	(1 0 1 1 1 0 0)
(0 0 1 0)	(1 1 1 0 0 1 0)
(1 0 1 0)	(0 0 1 1 0 1 0)
(0 1 1 0)	(1 0 0 0 1 1 0)
(1 1 1 0)	(0 1 0 1 1 1 0)
(0 0 0 1)	(1 0 1 0 0 0 1)
(1 0 0 1)	(0 1 1 1 0 0 1)
(0 1 0 1)	(1 1 0 0 1 0 1)
(1 1 0 1)	(0 0 0 1 1 0 1)
(0 0 1 1)	(0 1 0 0 0 1 1)
(1 0 1 1)	(1 0 0 1 0 1 1)
(0 1 1 1)	(0 0 1 0 1 1 1)
(1 1 1 1)	(1 1 1 1 1 1 1)

Codificación de Canal
Códigos Bloque Lineales

Un código lineal (*n,k*) sistemático queda completamente definido por una matriz **G***k*x*n* de la siguiente forma:

$$G = \begin{bmatrix} g_0 \\ g_1 \\ \cdot \\ \cdot \\ g_{k-1} \end{bmatrix} = \left[\begin{array}{cccc|ccccc} p_{00} & p_{01} & \cdots & p_{0,n-k-1} & 1 & 0 & 0 & \cdots & 0 \\ p_{10} & p_{11} & \cdots & p_{1,n-k-1} & 0 & 1 & 0 & \cdots & 0 \\ & & & & & & & & \\ & & & & & & & & \\ p_{k-1,0} & p_{k-1,1} & \cdots & p_{k-1,n-k-1} & 0 & 0 & 0 & \cdots & 1 \end{array}\right]$$

Matriz P — Matriz identidad k x k

Con p_{ij} = 0 ó 1, $\mathbf{I}_k$ matriz identidad de dimensión *k*,o bien **G** = [**P** $\mathbf{I}_k$].
Así, la palabra codificada **v** = (v_0 ,v_1 ,v_2 , ... ,v_{n-1}) = (u_0, u_1, ... , u_{k-1})**G** es el mensaje a codificar a la derecha, con $v_{n-k+i} = u_i$ (con 0≤=i<k) y $v_j = u_0p_{0j} + u_1p_{1j}$ + ... + $u_{k-1}p_{k-1j}$ a la izquierda (con 0 ≤ j<*n*–*k*). Es decir los *k* primeros dígitos de código **v** por la derecha son los de información u_0, u_1,..., u_{k-1}, y los *n*–*k* de redundancia son combinaciones lineales de los de información.

Codificación de Canal
Códigos Bloque Lineales

Para el ejemplo:

$$v = u \cdot \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Multiplicando se obtiene el código **v** como (también se puede representar con compuertas lógicas):

$$v_0 = u_0 + u_2 + u_3$$
$$v_1 = u_0 + u_1 + u_2$$
$$v_2 = u_1 + u_2 + u_3$$
$$v_3 = u_0$$
$$v_4 = u_1$$
$$v_5 = u_2$$
$$v_6 = u_3$$

(ej. el código para 1 0 1 1 es 1 0 0 1 0 1 1)

Codificación de Canal
Códigos Bloque Lineales

Decodificación

Sea la matriz de comprobación de paridad de la siguiente forma:

$$\mathbf{H}_{(n-k)\text{x}n} = [\mathbf{I}_{n-k}\ \mathbf{P}_{k\text{x}(n-k)}^{\text{T}}]$$

Y dado que las filas de $\mathbf{G}_{k\text{x}n}$ son independientes u ortogonales (ninguna se genera a partir de la suma modulo 2 de otras dos), se cumple que

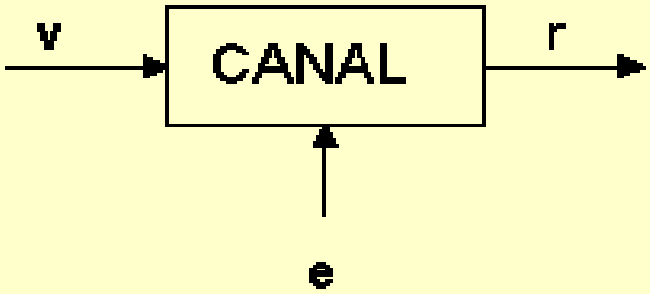
$$\mathbf{G}_{k\text{x}n}\mathbf{H}_{(n-k)\text{x}n}^{\text{T}} = \mathbf{0}_{k\text{x}(n-\text{k})}$$

Luego si el código $\mathbf{v}_{1\text{x}k}$ se comprueba con $\mathbf{H}^{\text{T}}$ se obtiene

$$\mathbf{v}_{1\text{x}n}\mathbf{H}_{(n-k)\text{x}n}^{\text{T}} = \mathbf{u}_{1\text{x}k}\mathbf{G}\mathbf{H}^{\text{T}} = \mathbf{0}_{1\text{x}(n-k)} \qquad (*)$$

Es decir cualquier palabra de código válida arroja un vector de *n-k* 0's como chequeo de los bits de redundancia o paridad, validando los *k* bits de información en **v**. Si los bits de código son afectados por el canal de transmisión, como errores binarios, la operación (*) no da solo 0's.

Codificación de Canal
Códigos Bloque Lineales



Decodificación

Si el código recibido **r** puede contener errores (binarios) **e** "contaminando" a **v**, es decir **r**=**v**+**e** (1's en **e** son errores), la operación (*) aplicada a **r**, llamada síndrome $\mathbf{s}_{1x(n-k)}=\mathbf{r}_{1xn}\mathbf{H}_{(n-k)xn}^{T}$, permite validar los *k* bits de información, o detectar errores (ARQ) y corregirlos (FEC) si es posible. Sin embargo el síndrome puede ser **0** si hay suficientes errores para transformar **v** en otra palabra de código válida, en este caso se produce error por no detección (ni corrección), además en este caso si **s**=**0**, **e**+**v** da otra palabra de código, por lo que **e** también es palabra de código (por diseño), por esto entonces hay 2^k-1 patrones posibles no nulos de errores no detectables. (2^k pal. tot. – 1, la pal. Tx.)

El síndrome depende solo del vector de error, y no de la palabra código transmitida, es decir

$$\mathbf{s}_{1x(n-k)} = \mathbf{r}\mathbf{H}^T = (\mathbf{v}+\mathbf{e})\mathbf{H}^T = \mathbf{v}\mathbf{H}^T(=\mathbf{0}) + \mathbf{e}\mathbf{H}^T = \mathbf{e}\mathbf{H}^T \qquad (**)$$

Codificación de Canal
Códigos Bloque Lineales

Decodificación

Por otra parte, si **s**≠**0** parece posible resolver **e** a partir del sistema de ecuaciones asociado (**), pero dado que hay *n-k* ecuaciones y *n* incógnitas (*n* componentes de **e**), hay *k* grados de libertad para la soluciones de **e**, por lo tanto hay 2^k patrones de error que dan el mismo síndrome. Para minimizar la probabilidad de error, se elige al patrón de error más probable, o $\mathbf{e}_{mín}$ que en un canal BSC es el que tiene el menor número de unos y equivale a elegir la menor distancia de **r** a algún **v**. El mensaje decodificado así es $\mathbf{v}=\mathbf{r}+\mathbf{e}_{mín}$
Si se ha producido un error simple, el síndrome corresponde a una de las filas de $\mathbf{H}^T$, y la posición de la fila es la posición del error.

Cálculo de síndrome para el ejemplo anterior:

$$s_0 = r_0 + r_3 + r_5 + r_6$$
$$s_1 = r_1 + r_3 + r_5 + r_6$$
$$s_2 = r_2 + r_4 + r_5 + r_6$$

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} \qquad \mathbf{s} = \mathbf{r} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

Codificación de Canal
Códigos Bloque Lineales

Distancia mínima de un código.

Cuanto más distintas sean las palabras del código, mejor es la capacidad discriminación del código, es decir de detección y corrección de errores. Esta diferencia se mide como una distancia de Hamming, que es el número de bits distintos entre dos palabras del código $\mathbf{v}_i$ y $\mathbf{v}_j$. Como la distancia es el número de 1's de $\mathbf{v}_k=\mathbf{v}_i+\mathbf{v}_j$, que por diseño es otra palabra de código, si se define al "peso" de $\mathbf{v}_k$, $w(\mathbf{v}_k)$, como el número de 1's de $\mathbf{v}_k$, entonces la distancia mínima $d_{mín}$ de un código es el mínimo peso del código (excluyendo la palabra **0**), y es una cota mínima para la capacidad de detección y corrección de errores. El peso mínimo (o la mínima distancia) de un código corresponde también al número de columnas de la matriz de comprobación **H** que suman 0.

Para el ejemplo se verifica que las columnas 4, 5 y 7 suman 0, así que $d_{mín}=3$.

$$\mathbf{H}=\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Codificación de Canal

Códigos Bloque Lineales

Propiedades de detección y corrección de errores de un código bloque.

Propiedades detectoras.

La distancia de **r** a una palabra de código es el peso de **e**, luego cuando el número de errores en **e** es menor que $d_{mín}$ (y el síndrome no es **0**) a partir del síndrome puede detectarse cualquier patrón de error con $d_{mín}$-1 o menos errores. También es posible detectar una parte de los patrones de error con $d_{mín}$ o más errores, en total son 2^n-2^k patrones de error detectables (que no dan síndrome **0**) por lo tanto hay 2^k-1 patrones de error que no son detectables (que dan otra pal. de cod. – **0**).
A tasa de código constante r_c, la proporción de errores no detectables disminuye con n, es decir $(2^k-1)/(2^n-2^k)\approx 1/2^{n(1-r_c)}$
La capacidad correctora de un código bloque también es de 2^n-2^k patrones de error (palabras no código). Detección, no detección, corrección y no corrección de errores tienen una probabilidad asociada que depende de las características del código en particular.

Codificación de Canal

Códigos Bloque Lineales

Propiedades de detección y corrección de errores de un código bloque.

Propiedades detectoras.

Si el código se usa solo para detectar errores, no hay detección si **e** coincide con una palabra de código (excepto error **0**), luego la probabilidad de no detectar palabras erradas es la suma de las prob. de que esto ocurra, que para BSC es: $P_U(E) = \sum_{i=1}^{n} A_i p^i (1-p)^{n-i}$

Con p la prob. de error por bit de canal, y A_i el número de palabras de código de peso i. Si el código tiene $d_{mín}$, A_i será 0 para $0 < i < d_{mín}$

Para el ejemplo, los A_i del código son (ver tabla, hay 2^k-1 con **e**≠**0**):

$$A_0=1,\ A_1=A_2=0,\ A_3=A_4=7,\ A_5=A_6=0,\ A_7=1.$$

La probabilidad de no detección de palabras erradas es entonces:

$$P_U(E) = 7p^3(1 - p)^4 + 7p^4(1 - p)^3 + p^7.$$

Si $p=10^{-2}$, $P_U \approx 7\text{x}10^{-6}$, es decir en promedio por cada millón de palabras transmitidas apenas hay 7 erradas que no son detectadas.

Codificación de Canal

Códigos Bloque Lineales

Propiedades de detección y corrección de errores de un código bloque.

Propiedades correctoras.

Un código bloque con $d_{mín}$ corrige palabras erradas asociando la secuencia recibida a la palabra de código válida más cercana, esto se puede hacer si el número de errores en **e** (o su peso) es menor que la mitad de $d_{mín}$, parte entera, es decir cuando hay $t=[(d_{mín}-1)/2]$ o menos errores. Si el código se usa solo para corregir, hay error incorregible cuando hay más de t errores, entonces la probabilidad de error de detección es la suma de las probabilidades de que esto ocurra. En un canal BSC con probabilidad de error por bit de canal p, la probabilidad de error por palabra es:

$$P(E) = \sum_{i=t+1}^{n} \binom{n}{i} p^i (1-p)^{n-i}$$

Para el código de ejemplo, $n=7$, $d_{mín}=3 \Rightarrow t=1$. Si $p=10^{-2}$,

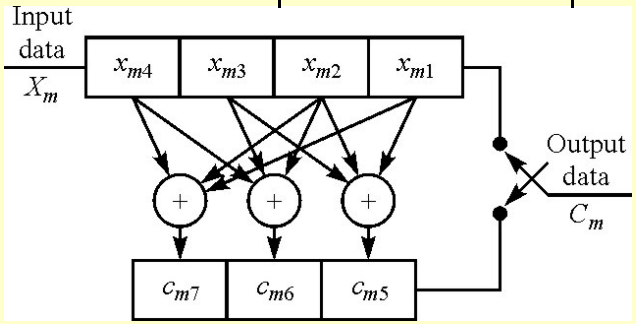
$P(E) \approx 2\text{x}10^{-3}$ (mucho mayor que la prob. de no detección)

Codificación de Canal
Códigos Bloque Lineales

Codificador **v=uG**

Ejemplo

$$v = u \cdot \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$



Decodificador, **G**⇒H, **s**=**r**$\mathbf{H}^T$=**e**$\mathbf{H}^T$

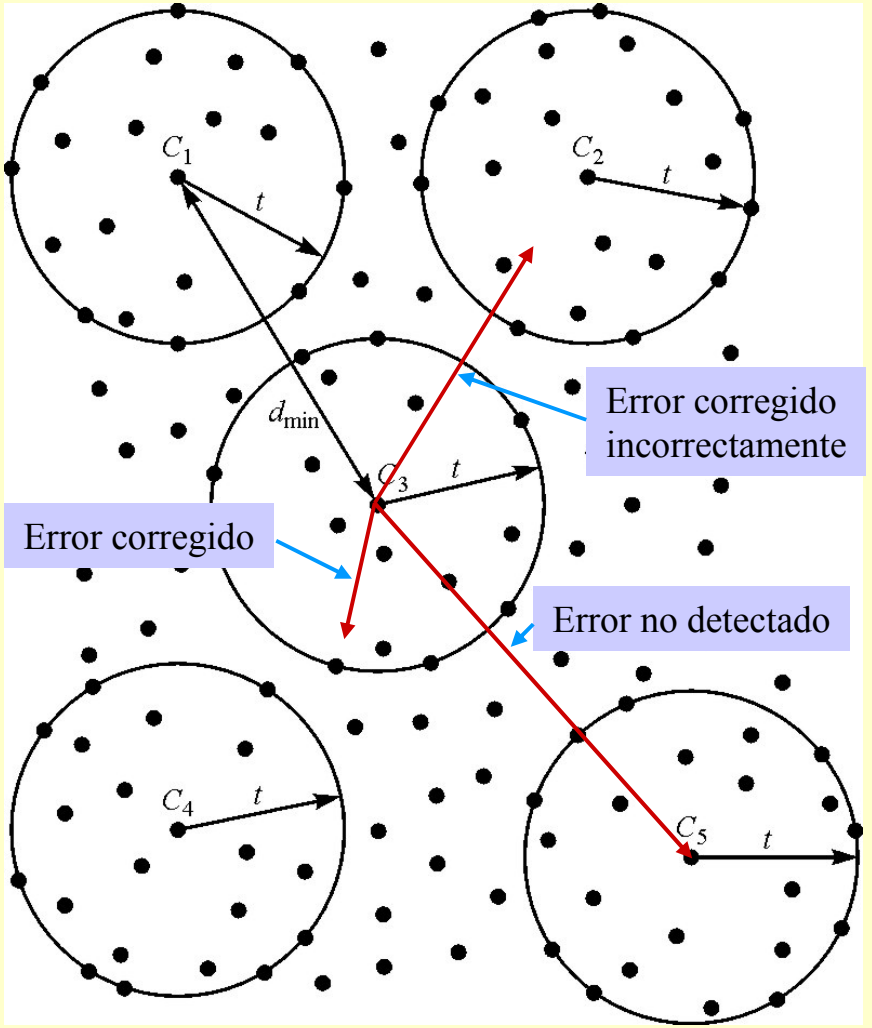
s⇒{**e**}, mín. **e**⇒**v**=**r**+**e** (FEC)

s⇒ error (ARQ)

Ejemplo $p_{ebc}=10^{-2}$

FEC⇒P(error, detec.+corr.) ≈2×10^{-3}

ARQ⇒P(error, x no detec.) ≈7×10^{-6}



Codificación de Canal

Códigos Bloque Lineales Especiales

CODIGOS HAMMING

Un código Hamming (n,k) se caracteriza por una matriz **H** cuyas columnas son todas las posibles secuencias de n-k dígitos binarios excepto el vector **0**. Para todo m=3, 4, 5... existe un código Hamming (n,k)=(2^m-1,2^m-1-m), esto da m bit de paridad. Su distancia mínima es 3, por lo que corrige todos los errores en un bit (t=1), y detecta 2 errores ($d_{mín}$-1).

Tiene la propiedad de que un vector error con un error simple genera un síndrome que determina directamente la posición del error, ya que el síndrome corresponde a una de las filas de $\mathbf{H}^T$.

Codificación de Canal

Códigos Bloque Lineales Cíclicos

Los cód. cíclicos son una subclase de los códigos bloque lineales.
- Son fáciles de implementar en base a registros de desplazamiento.
- Tienen gran capacidad de detección y corrección de errores.
- Son lineales: la suma módulo-2 de dos palabras del código es otra palabra del código.

Principio de operación: el mensaje de información de *k* bits es multiplicado por un operador adecuado en el extremo transmisor, generando el código de *n* bits (>*k*). En el extremo receptor el código recibido, con posibles errores, se divide por el operador para obtener el mensaje, si el resto es cero se valida, según el resto se corrige (FEC) o se rechaza y pide retransmisión (ARQ).

Como ejemplo en base 10, los mensajes podrían ser 1, 2, 3, 4 y el operador 3, lo que da 3, 6, 9, 12 como códigos distanciados para reducir el efecto de los errores del canal, y al dividirlos por el operador en el receptor se obtiene una parte entera (mensaje) y un resto como una medida de cuán cerca se está del mensaje.

La operación de multiplicación en base a un cierto operador genera un desplazamientos cíclico en el código transmitido.

Codificación de Canal

Códigos Bloque Lineales Cíclicos

El mensaje **x**, operador multiplicativo **g** y código **c**, $[x_{k-1}\ x_{k-2} \ldots\ x_1\ x_0]$, $[1\ g_{n-k-1} \ldots g_1\ 1]$ y $[c_{n-1}\ c_{n-2} \ldots\ c_1\ c_0]$, respectivamente, se representan como polinomios:

$x(p)=x_{k-1}p^{k-1}+x_{k-2}p^{k-2}x+\ldots+x_1p+x_0$

$g(p)=p^{n-k}+g_{n-k-1}p^{n-k-1}+\ldots+g_1p+1$

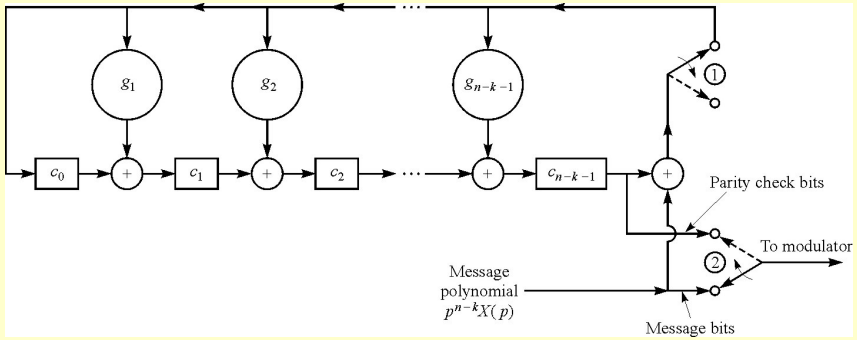
$c(p)=c_{n-1}p^{n-1}+c_{n-2}p^{n-2}+\ldots+c_1p+c_0$

Donde p es la base (2 caso binario). Como ejemplo en base 10, el número 3057 es $3x10^3+0x10^2+5x10^2+7$. Así las multiplicaciones o divisiones por términos p^i equivalen a desplazamientos de los dígitos o coeficiente en i posiciones en el número asociado.

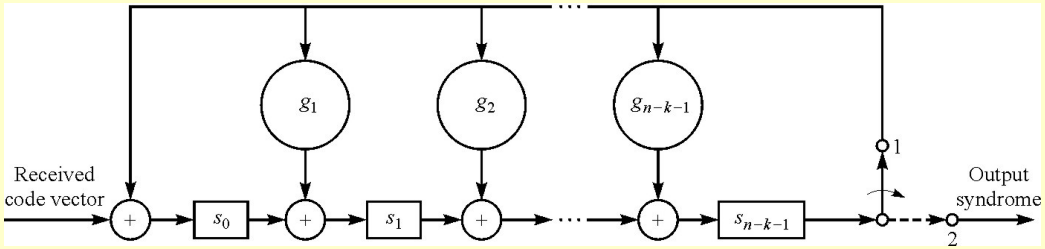
En binario las multiplicaciones y divisiones se pueden implementar con registros de desplazamiento para codificar y decodificar, y ocupar el resto en la división para validar y corregir los mensajes.

Codificación de Canal
Códigos Bloque Lineales Cíclicos

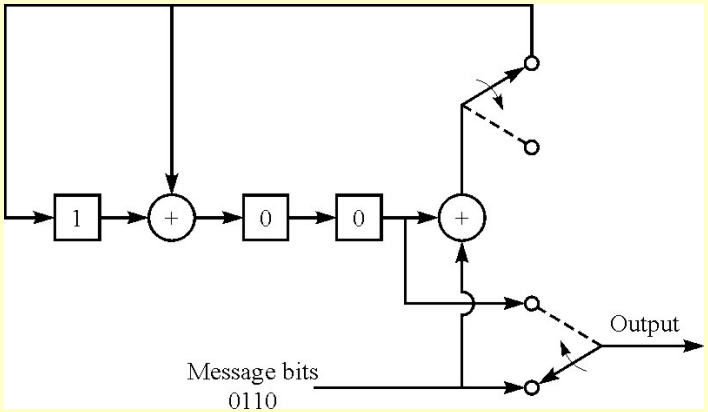
Codificador (multiplicador)

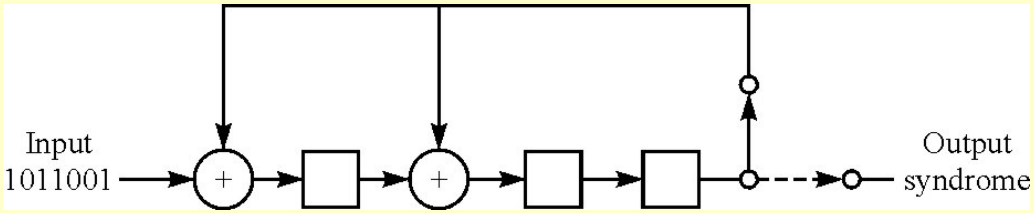


Decodificador (divisor)



Ejemplo código (n,k)=(7,4), n-k=3 bit de paridad, g(p)= p^3+p+1





1000110

Codificación de Canal

Códigos Cíclicos de máxima longitud

Códigos del tipo (n,k)=(2^m-1,m), tasa de código relativa baja r_c=m/(2^m-1)
m es el largo del registro de desplazamiento y son los bits de mensaje, cargados secuencialmente en el registro.

- Se carga el registro con los m bits, y se hace circular n=2^m-1 desplazamientos para generar los n bits de código de salida. Se excluye palabra de m 0's

- $d_{mín}$=2^m-1,
  detecta 2^m-2 errores
  corrige t=[($d_{mín}$-1)/2] = 2^{m-1}-1 errores

Códigos de redundancia cíclica (CRC)

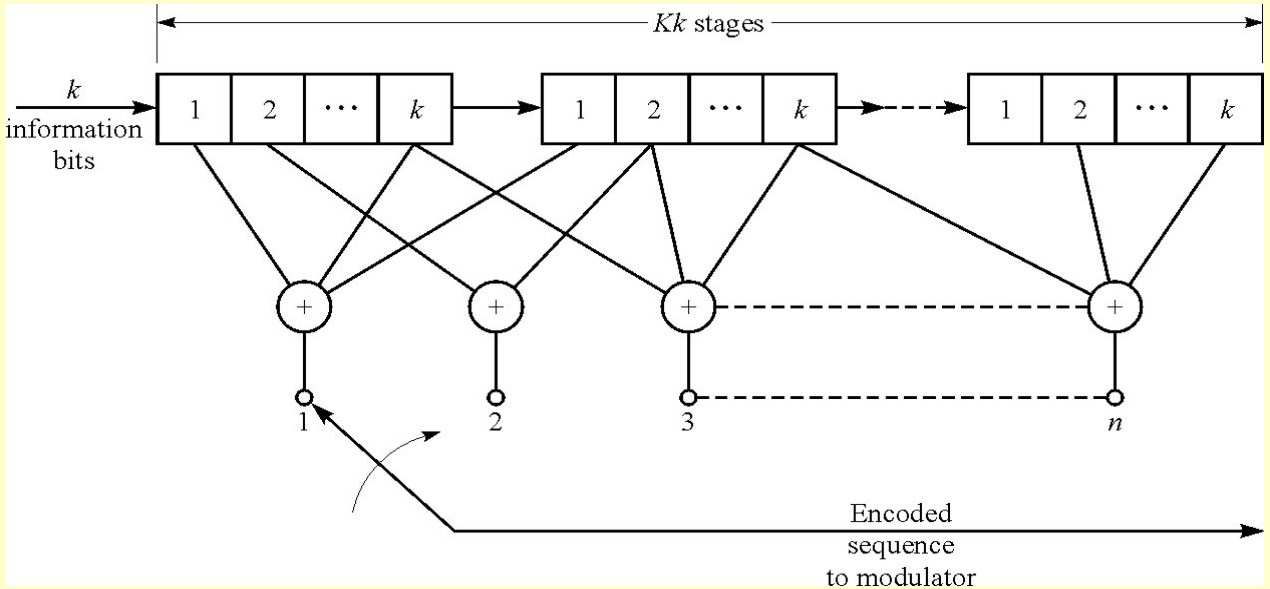
Códigos cíclicos eficientes para la detección de errores, de uso estándar. Son los que se usan en la práctica.

CRC-12	$g(x)= x^{12} + x^{11} + x^3 + x^2 + x + 1$ para redundancia n–k=12
CRC-16	$g(x)= x^{16} + x^{15} + x^2 + 1$
CRC-ITU o CRC-CCITT:	$g(x)= x^{16} + x^{12} + x^5 + 1$
UMTS 3G:	$g(x)= x^{24} + x^{23} + x^6 + x^5 + x + 1$
	$g(x)= x^{16} + x^{12} + x^5 + 1$
	$g(x)= x^{12} + x^{11} + x^3 + x^2 + x + 1$
	$g(x)= x^8 + x^7 + x^4 + x^2 + x + 1$

Codificación de Canal
Códigos Convolucionales

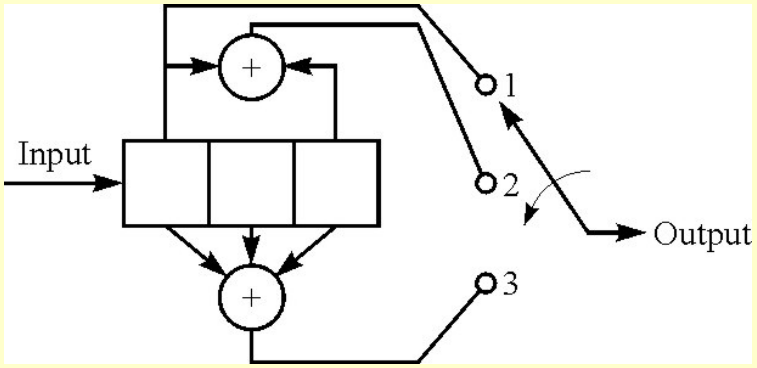
A diferencia de los códigos bloques, los CC no están restringidos a codificar los mensajes en bloque, es decir puede ser una codificación continua de largo indefinido. Están diseñados para corrección (no detección).

La codificación se basa en registros de desplazamiento de K etapas de k bits (K se denomina *constraint length*) y n funciones algebraicas generadoras. Los bits de mensaje entran al registro en paquetes de k bits, y salen codificados en paquetes de n bits, la tasa de código es, r_c=k/n. K da origen a una memoria de Kxk-1 bits que generan la secuencia codificada (lo que le da el nombre de convolucional)

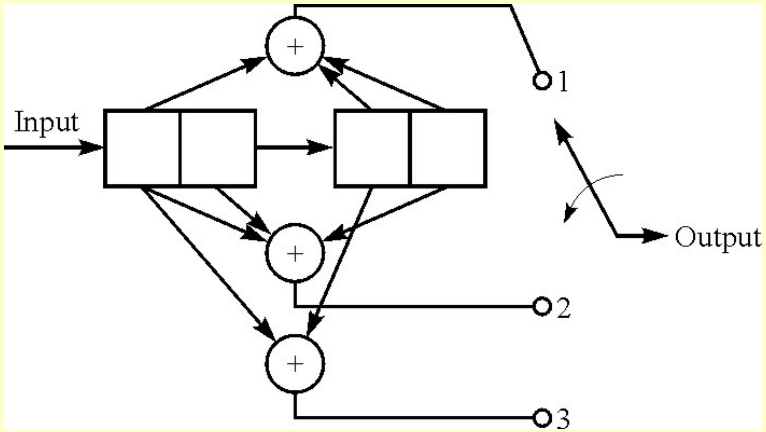


Codificación de Canal
Códigos Convolucionales

Ejemplos
K=3, k=1, n=3
g1=[100]
g2=[101]
g3=[111]
(4,5,7) en octal



K=2, k=2, n=3
g1=[1011]
g2=[1101]
g3=[1010]
(13,15,12) en octal



Codificación de Canal
Códigos Convolucionales

Ejemplo
K=3, k=1, n=3
g1=[100]
g2=[101]
g3=[111]
(4,5,7) en octal

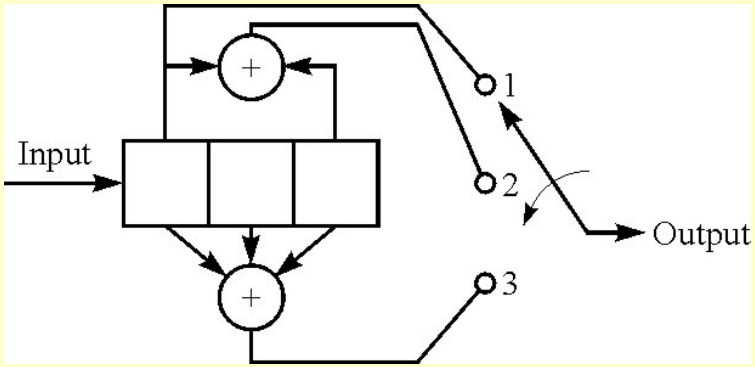
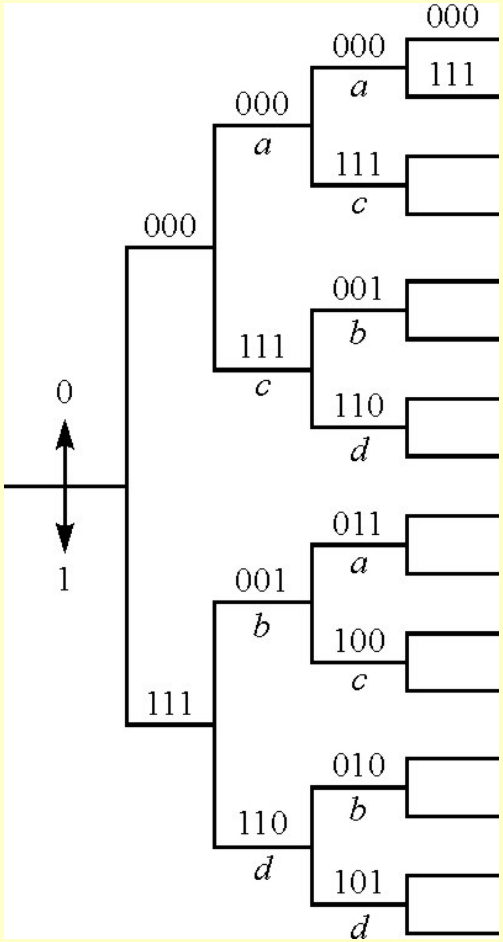


Diagrama de árbol asociado



Codificación de Canal
Códigos Convolucionales

Ejemplo
K=3, k=1, n=3
g1=[100]
g2=[101]
g3=[111]
(4,5,7) en octal

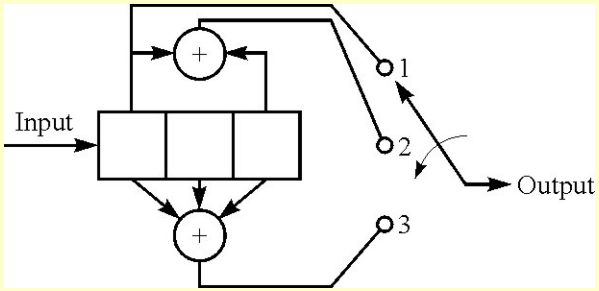
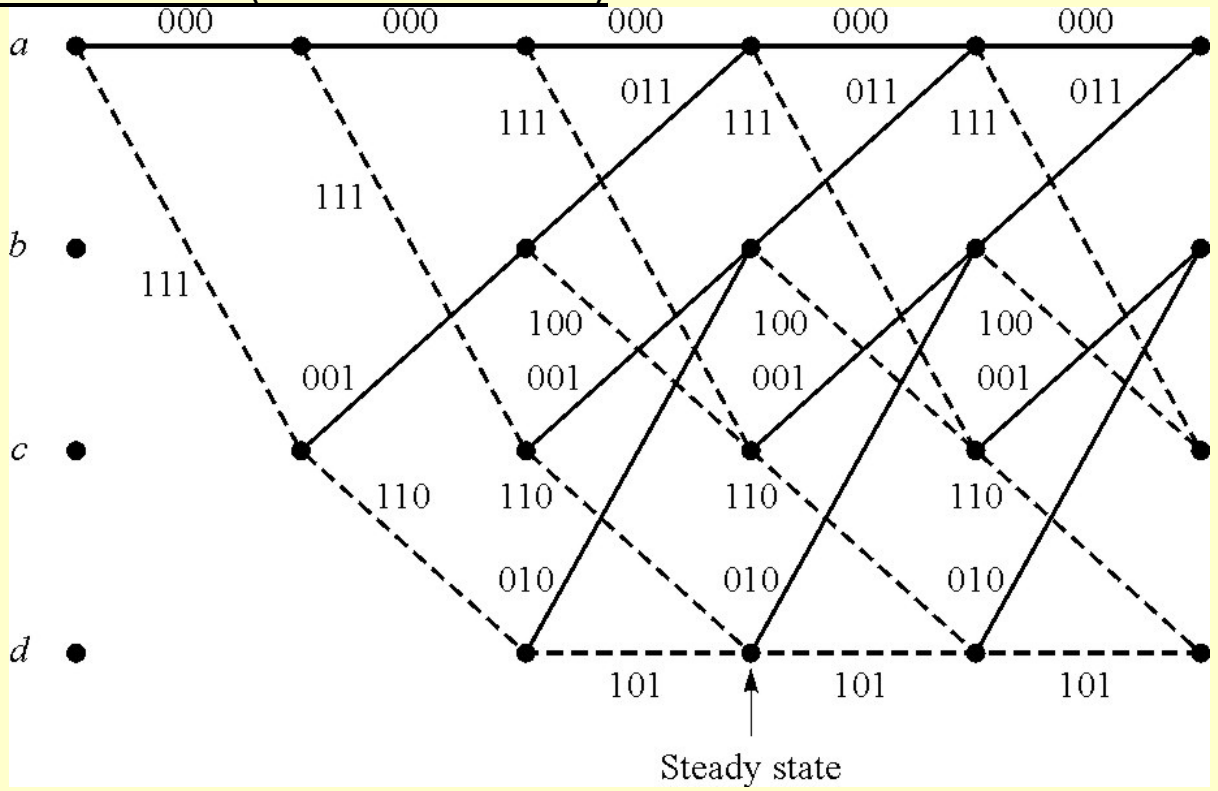


Diagrama de enrejado (trellis) asociado

2^{K-1}=4 estados (memoria K-1=2)



Codificación de Canal
Códigos Convolucionales

Ejemplo
K=2, k=2, n=3
g1=[1011]
g2=[1101]
g3=[1010]
(13,15,12) en octal

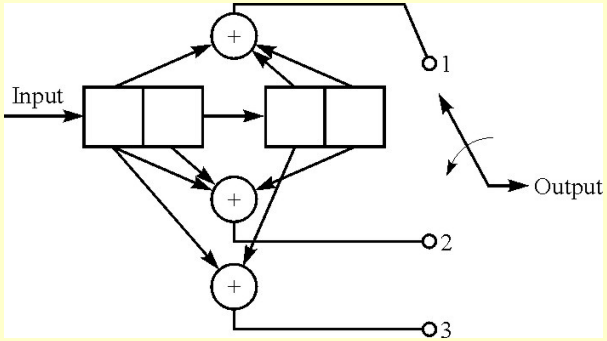
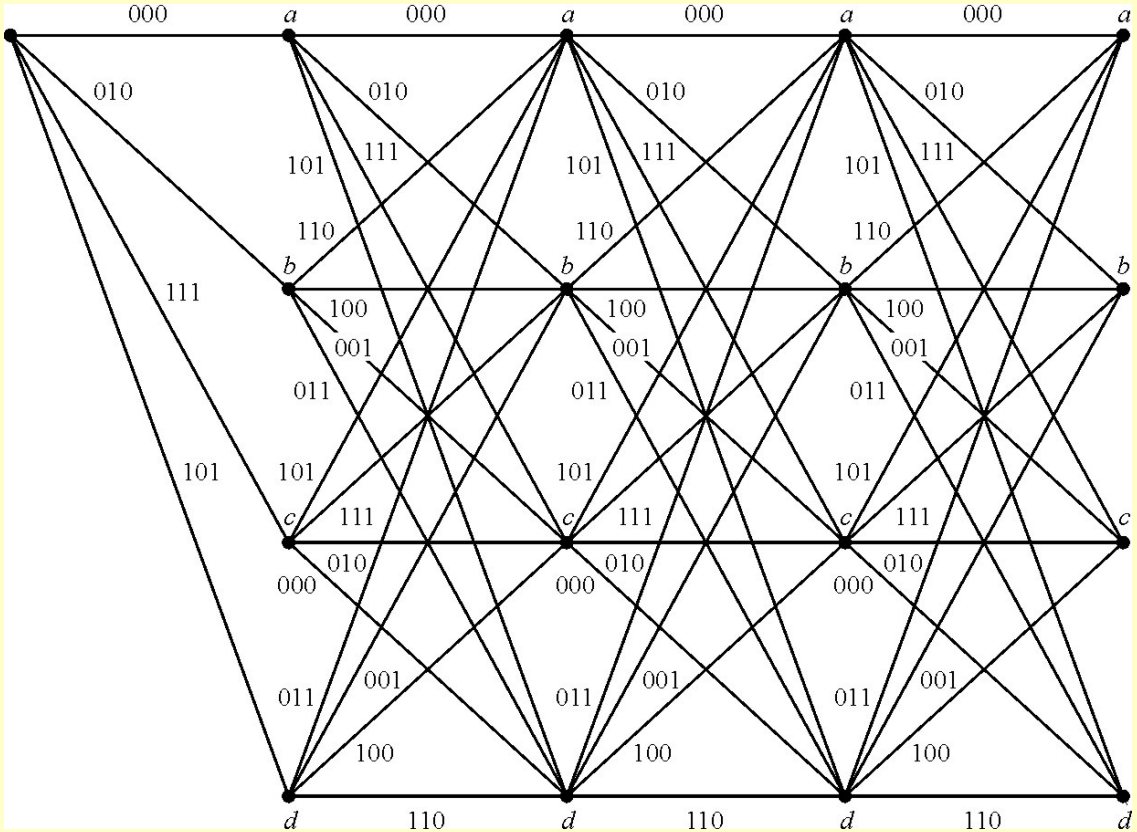


Diagrama de enrejado (trellis) asociado

$2^{(K-1)xk}$=4 estados



Codificación de Canal

Códigos Convolucionales

Decodificación

Si la secuencia de bits de entrada (mensaje) tiene largo L bits, se generarán 2^L trayectorias distintas posibles en el diagrama de trellis. La decodificación óptima consiste en "proyectar" las secuencia de bits de código recibida sobre todas las posible, y elegir la que tiene mayor métrica. Sin embargo pata L grande este cálculo no es práctico y se opta por un algoritmo que en forma secuencial va calculando (con una memoria o profundidad limitada) y eliminando las métricas con menor valor acumulado, llamado algoritmo de *Viterbi*.

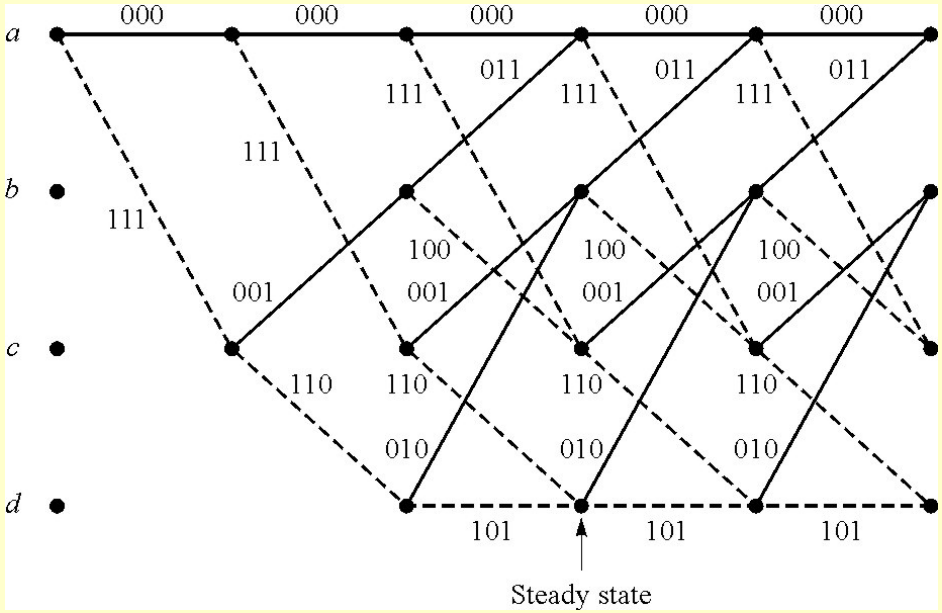
El método de proyectar las secuencias recibidas con las posibles recibidas libres de errores, semejante al esquema de recepción óptima de bit o símbolo, tiene la mínima probabilidad de error de secuencia, y por tanto la mínima tasa media de error binaria de información.

La proyección puede ser binaria (decodificación dura o *hard*) en base a los bits de código recibidos, o en base a la señal obtenida del correlador, bit a bit de código (decodificación blanda o *soft*). La decodificación *soft* es más eficiente (menor Pe) porque da más "peso" en el calcula de la proyección a los bits de canal que son "más cercaos" como señal, a los 1's o a los 0's.

Codificación de Canal
Códigos Convolucionales
Decodificación *soft*

El cálculo de la métrica para selección de máxima proyección asume que los bits de código transmitidos tienen asociados señales antipodales, de manera que las señales de código obtenidas a la salida del correlador receptor r_{jm} tienen polaridad +/-V con ruido aditivo (gaussiano).
Los bits de mensaje y de código esperado se asumen binarios.
- Se calcula la métrica CM para cada una de las i secuencias o trayectos posibles.
- Para cada secuencia se calcula en las B etapas o bit de mensaje la métrica o proyección de r_{jm} sobre el código c_{jm} asociado en la secuencia i.
- Se elige la máxima CM.

$$CM^{(i)} = \sum_{j=1}^{B} \sum_{m=1}^{n} r_{jm}(2c_{jm}^{(i)} - 1)$$



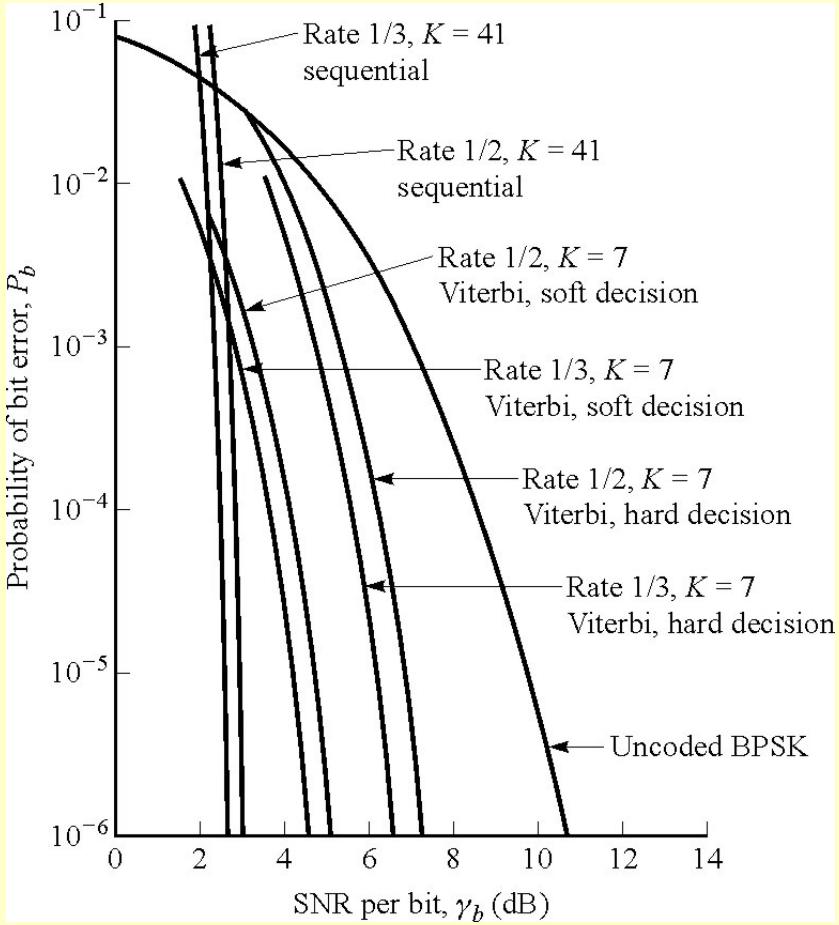
Viterbi no calcula todo, elimina métricas acumuladas que compiten en cada etapa.

Codificación de Canal

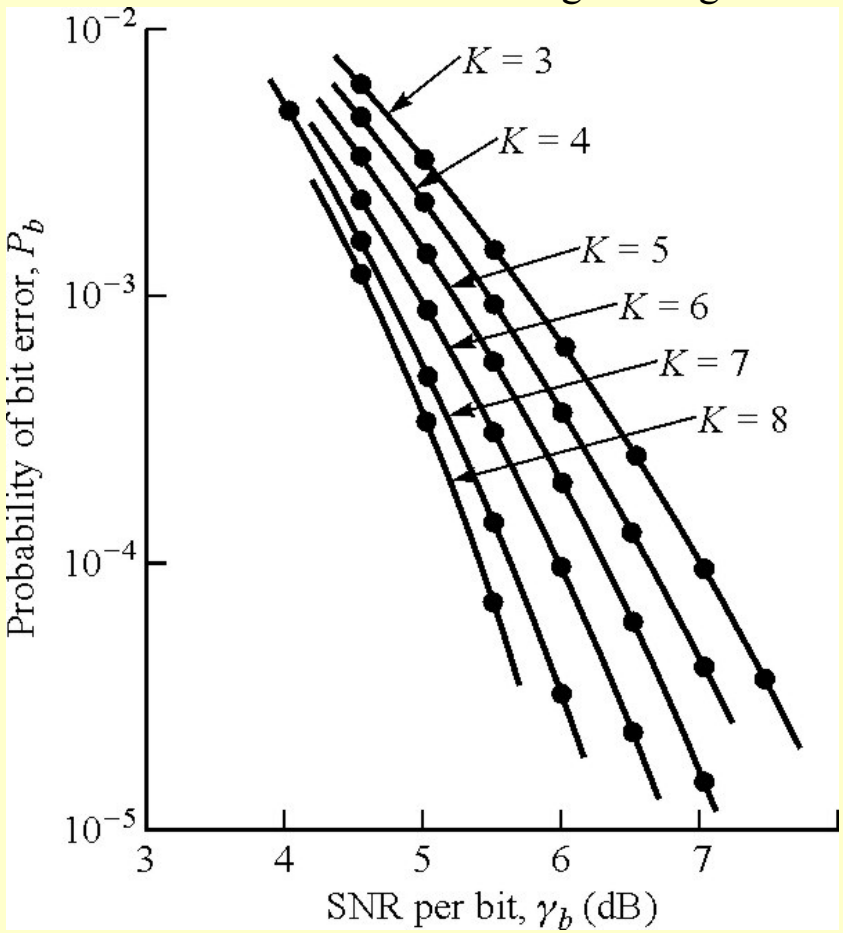
Códigos Convolucionales

Ejemplos de rendimiento en BER

CC de tasa ½ y 1/3 decod. Viterbi hard y soft



CC tasa ½ . Viterbi hard
32 bit de memoria para cálculo de métricas
Para distintos valores de largo de registro.

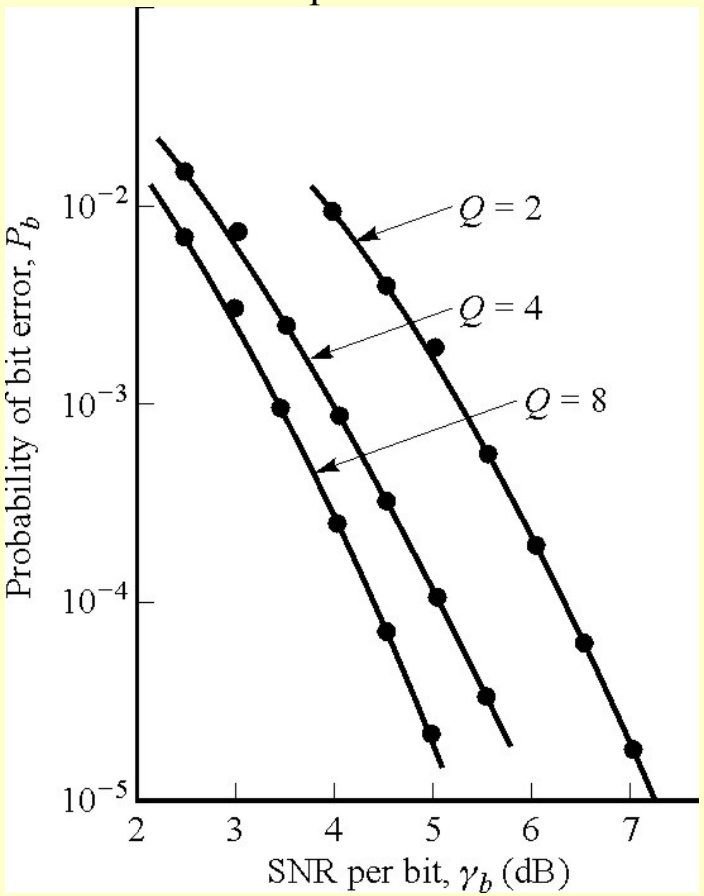


Codificación de Canal

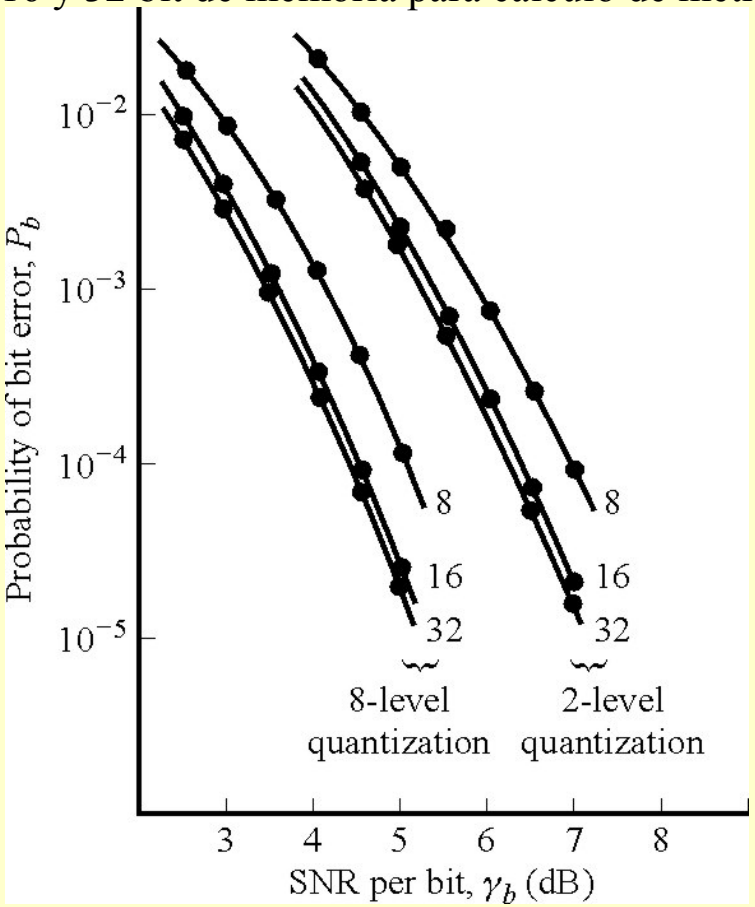
Códigos Convolucionales

Ejemplos de rendimiento en BER

CC tasa ½ K=5 decod. Viterbi hard, y 4 y 8-niveles soft, 32 bit de memoria para cálculo de métricas



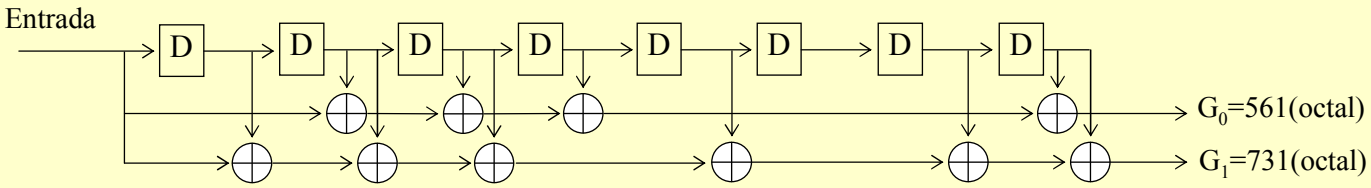
CC tasa ½ decod. Viterbi hard y 8-niveles soft. 8, 16 y 32 bit de memoria para cálculo de métricas



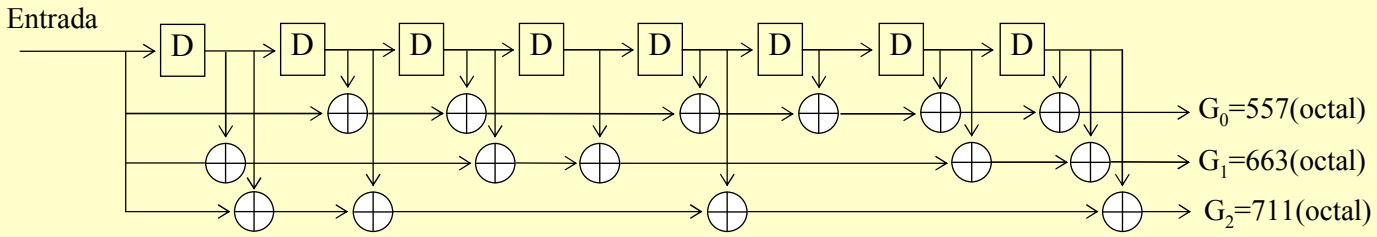
Codificación de Canal

Códigos Convolucionales

Estándar 3G

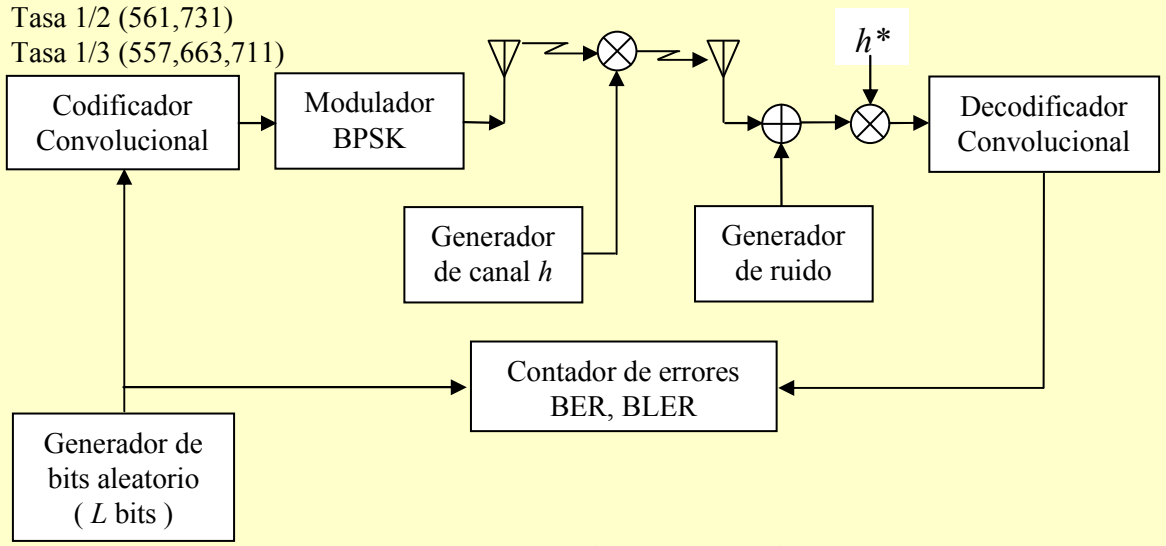


(a) Tasa de codificación 1/2



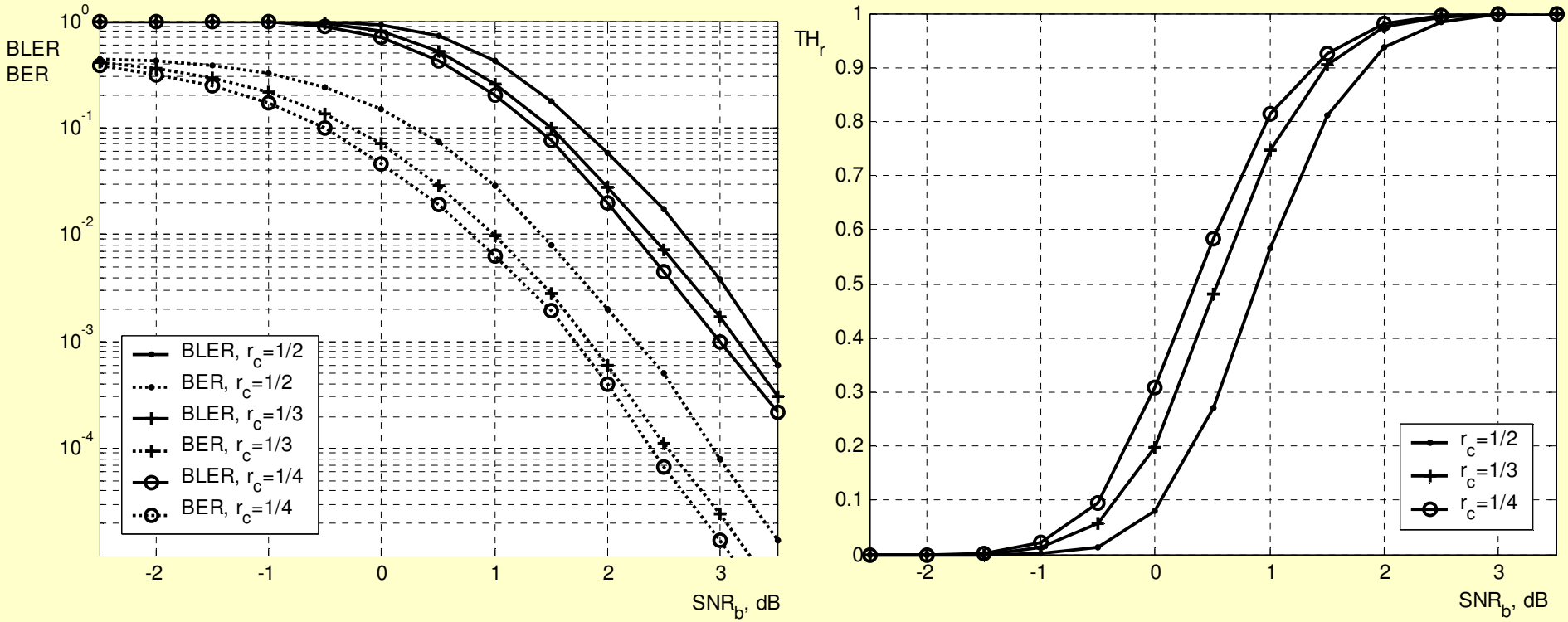
(b) Tasa de codificación 1/3

Codificador Convolucional 3G
Simulación BER, BLER y *Throughput.*
Esquema simplificado de transmisión.

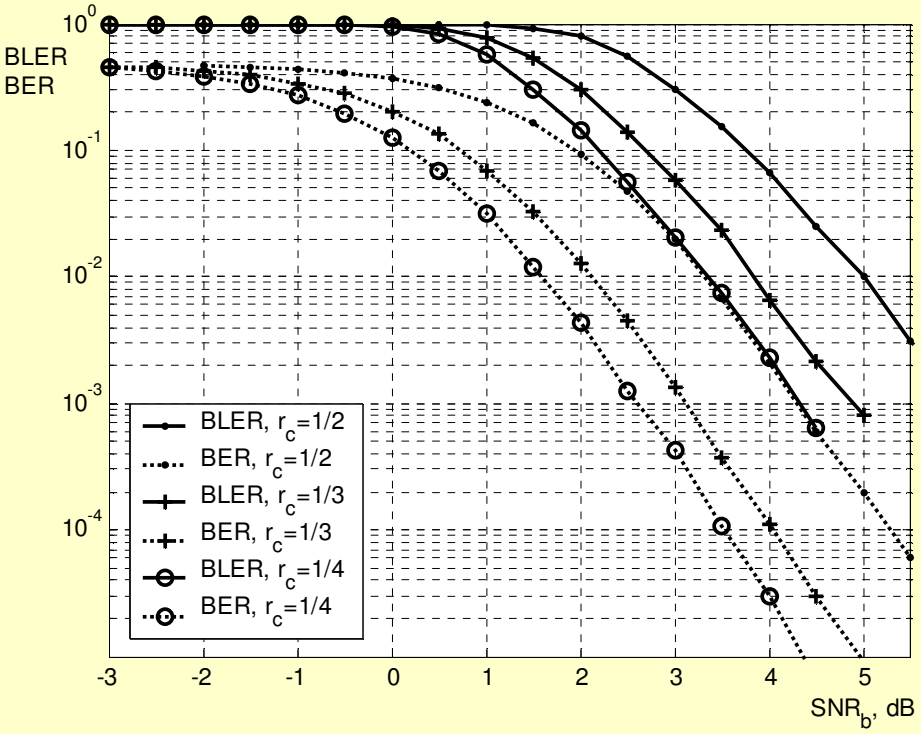


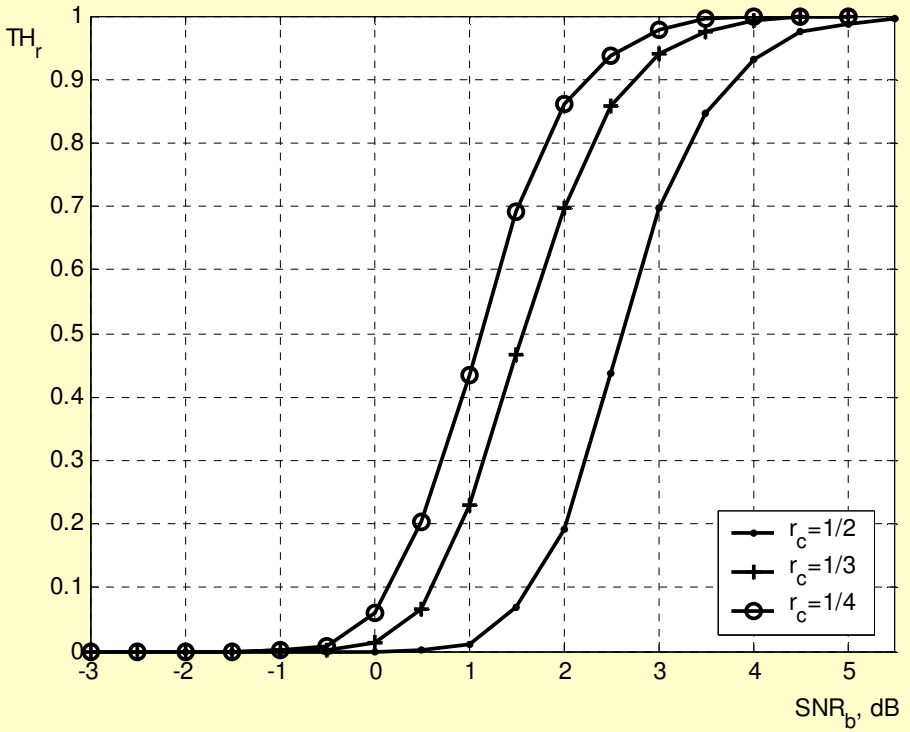
$$TH = R_s r_c B(1 - BLER)$$

Throughput relativo $TH_r = 1 - BLER$

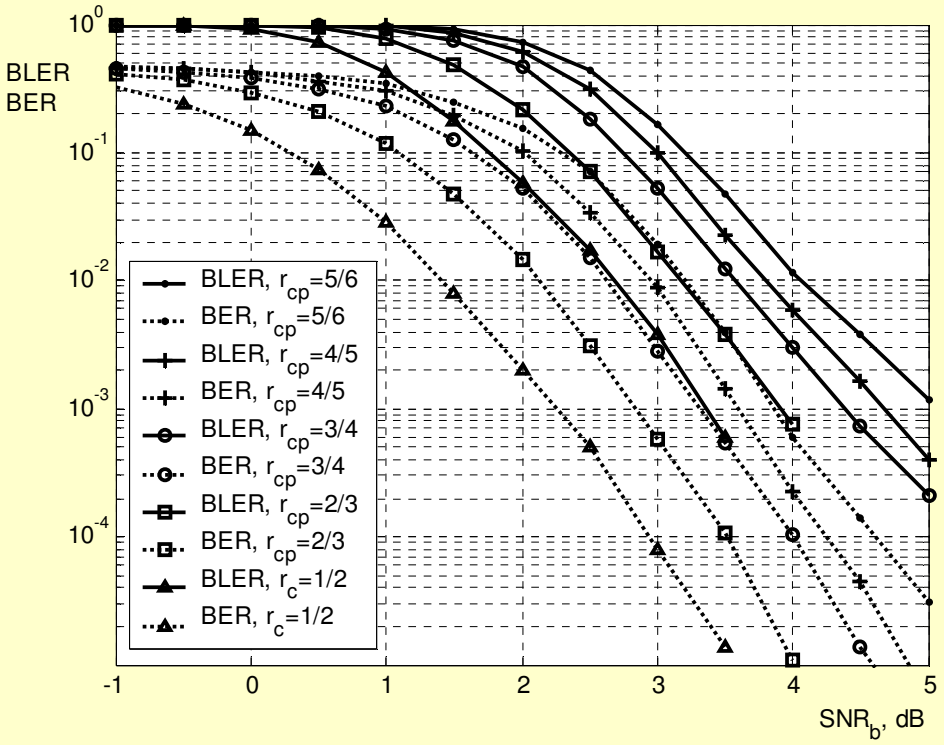


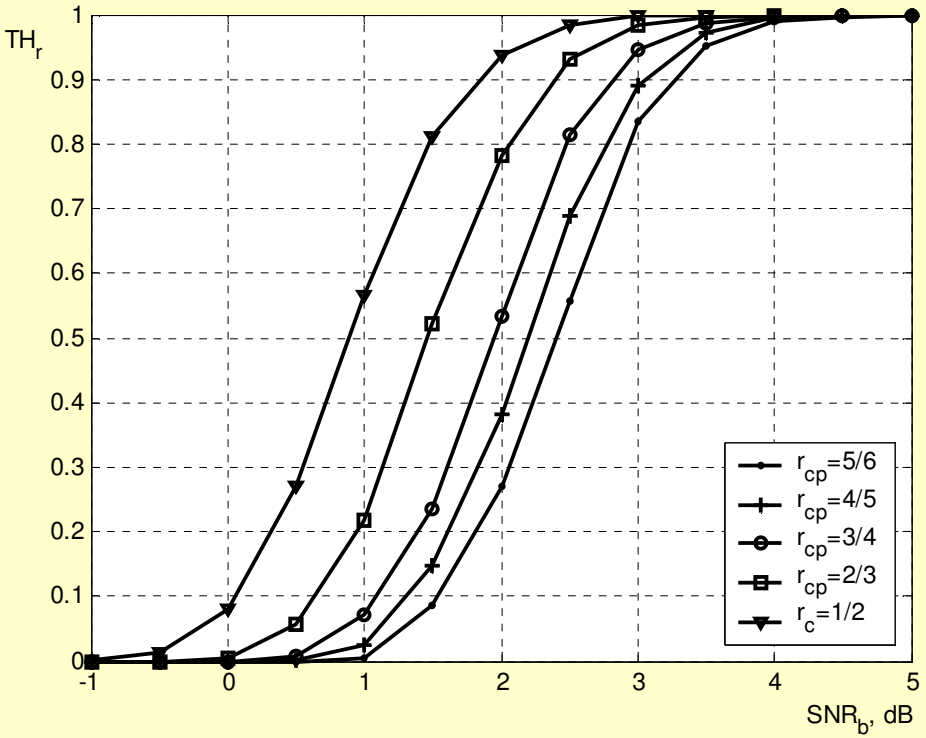
BLER, BER y *throughput* relativo THr versus SNR_b
CC, tasas r_c = 1/2, 1/3 y 1/4, canal Gaussiano.
Tamaño de bloque L=300 bits.



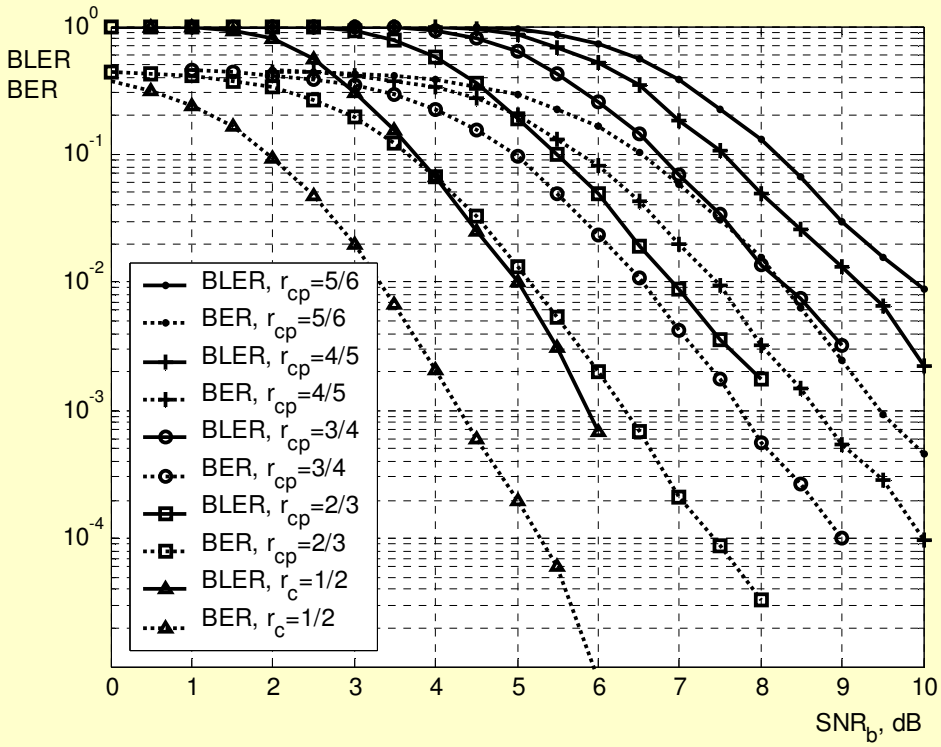


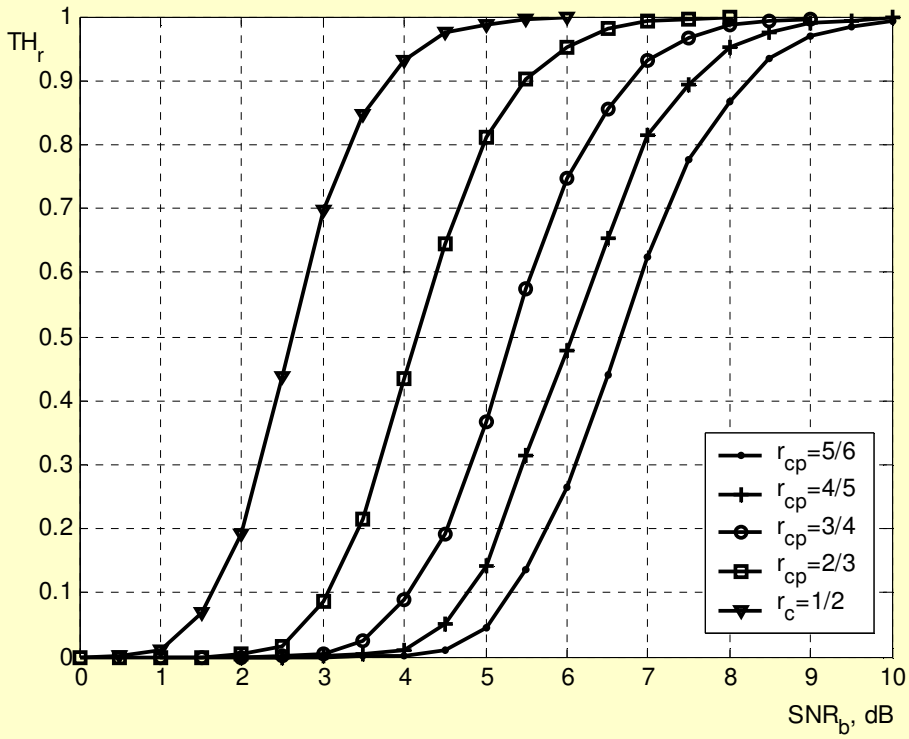
BLER, BER y *throughput* relativo THr versus SNR_b
CC, tasas r_c = 1/2, 1/3 y 1/4, canal Rayleigh independiente
Tamaño de bloque L=300 bits.





BLER, BER y *throughput* relativo THr versus SNR_b
CC, tasas r_c = 1/2, y perforado para
r_{cp}=2/3, 3/4, 4/5 y 5/6, canal Gaussiano
Tamaño de bloque *L*=300 bits.





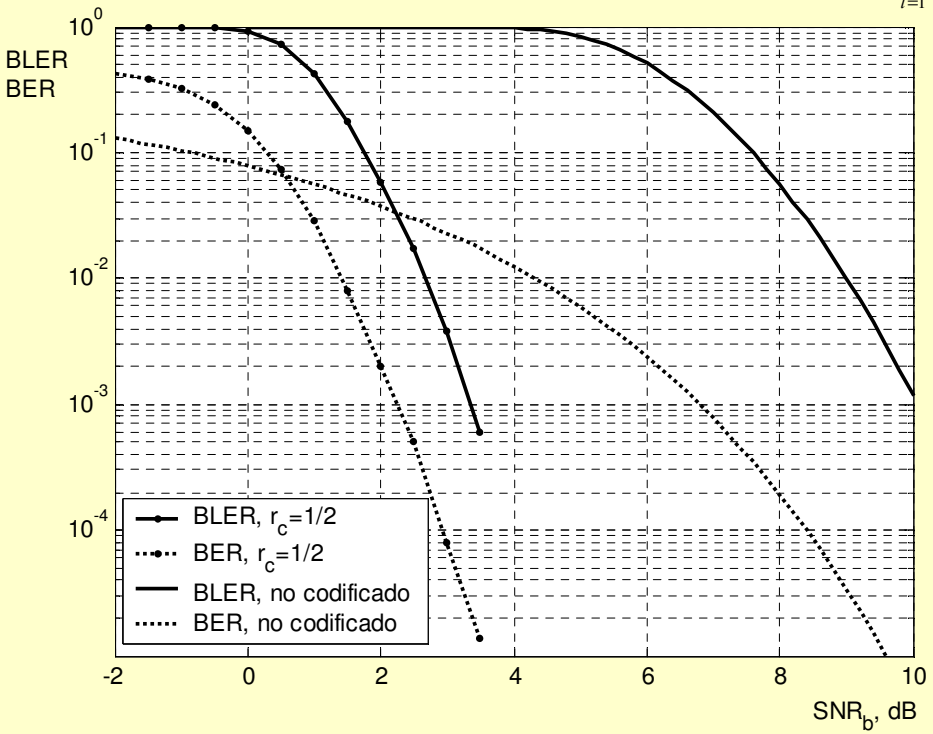
BLER, BER y *throughput* relativo THr versus SNR_b
CC, tasas r_c = 1/2, y perforado para
r_{cp}=2/3, 3/4, 4/5 y 5/6, canal Rayleigh independiente
Tamaño de bloque L=300 bits.

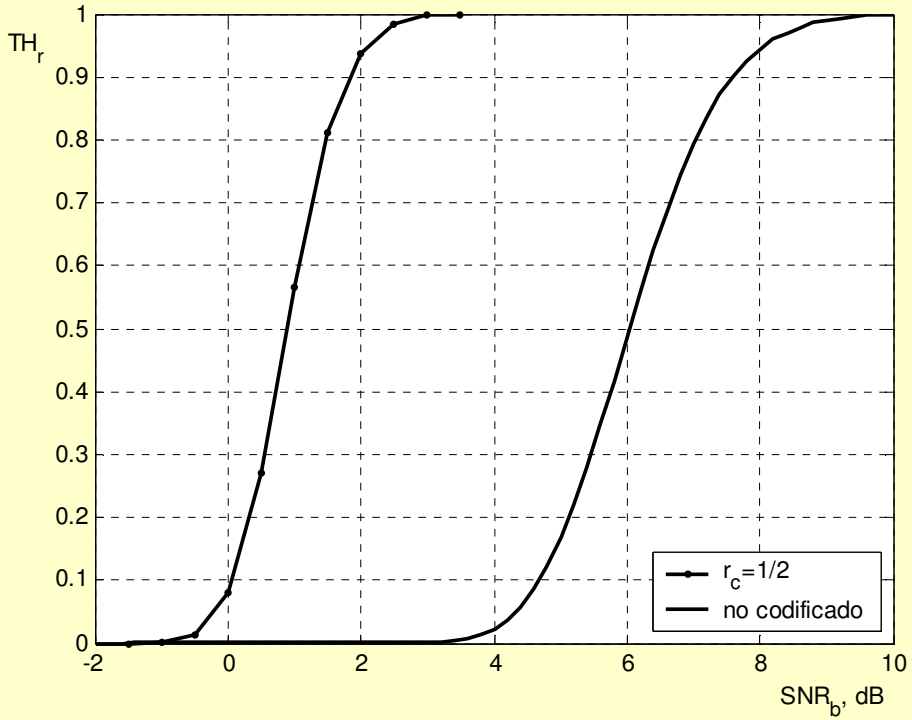
Evaluación del C.C. 3G versus no codificado.

$$BER = E[0{,}5\,erfc(\sqrt{|h|^2\, SNR_b}\,)]$$

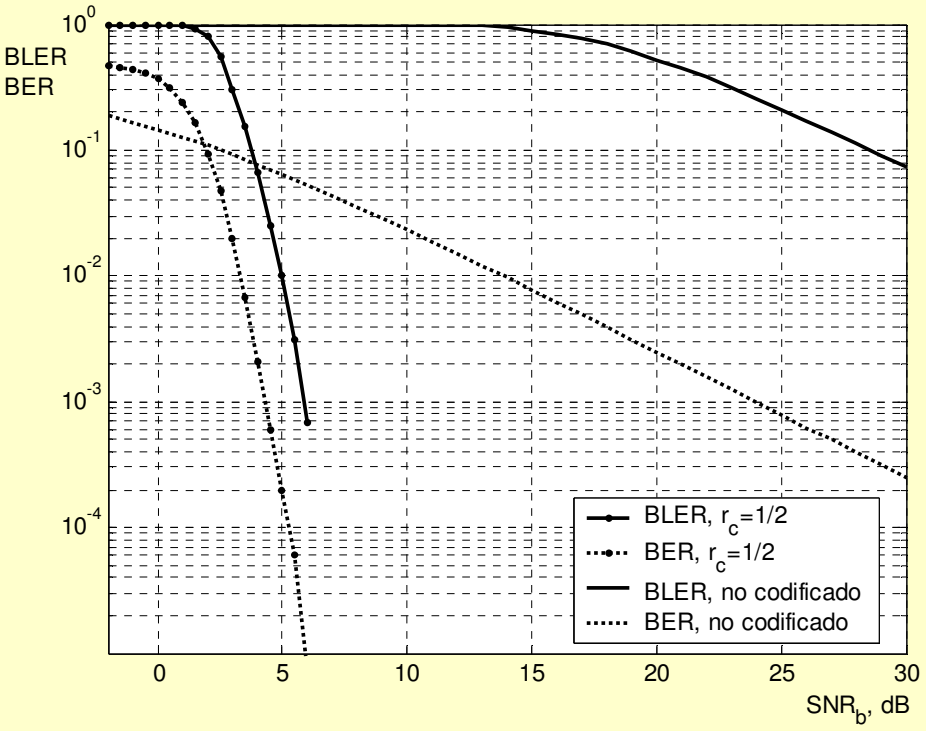
Canal Gaussiano $BLER = 1-(1-BER)^L = 1-(1-0{,}5\,erfc(\sqrt{SNR_b}\,))^L]$

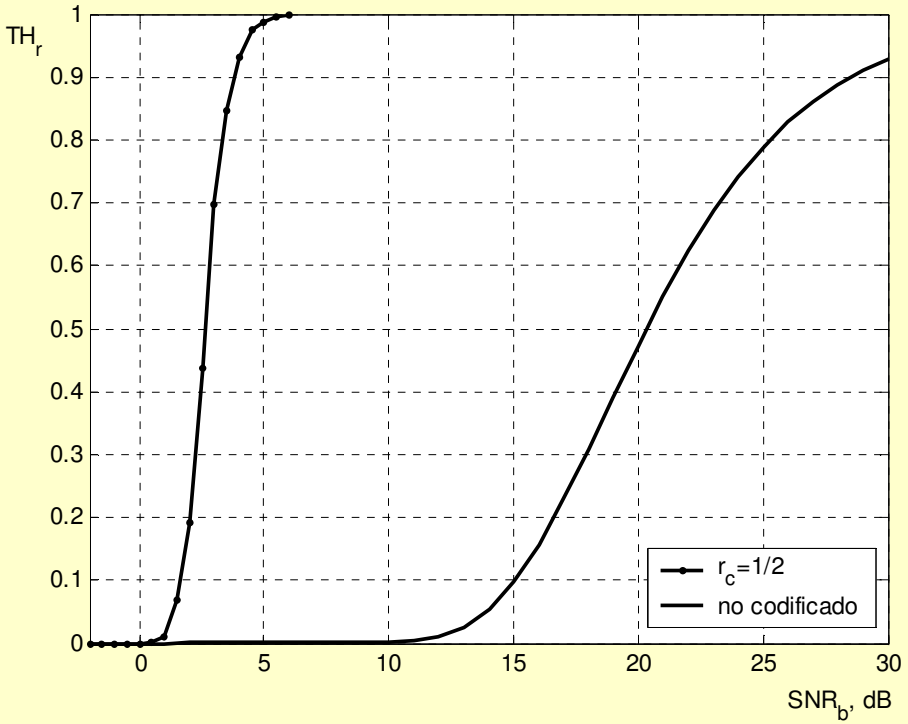
Canal Rayleigh $BLER = E[1-\prod_{l=1}^{l=L}(1-BER_l)] = E[1-\prod_{l=1}^{l=L}(1-0{,}5\,erfc(\sqrt{|h_l|^2\, SNR_b}\,)]$





BLER, BER y *throughput* versus SNR_b
CC tasa 1/2 y sin codificar
Canal Gaussiano, tamaño de bloque L=300 bits





BLER, BER y *throughput* versus SNR_b

CC tasa 1/2 y sin codificar

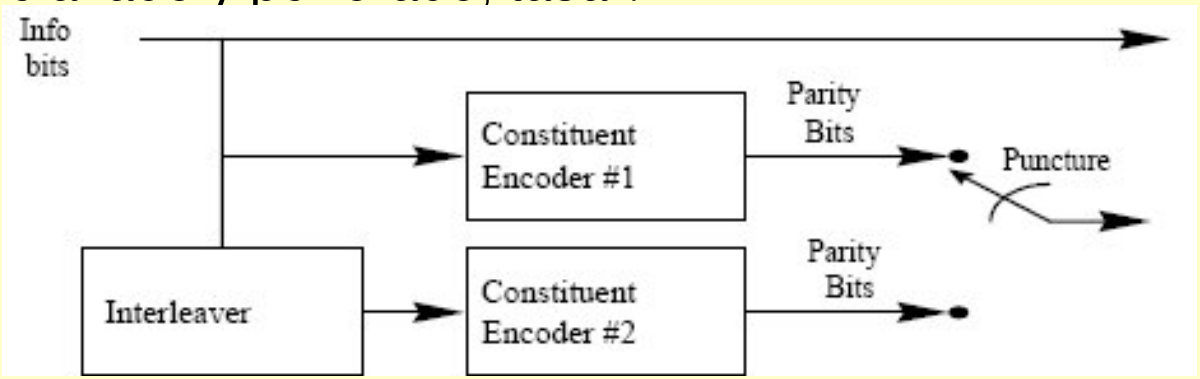
Canal Rayleigh independiente, tamaño de bloque L=300 bits

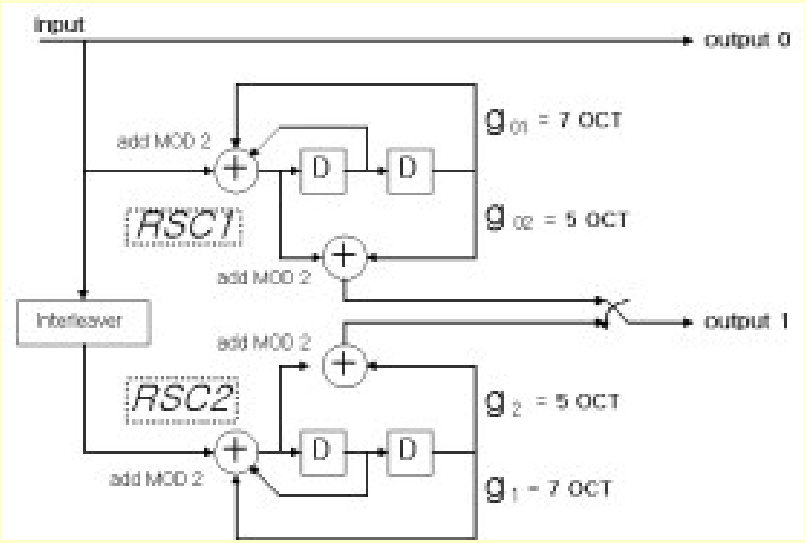
Codificación de Canal

Turbo Códigos

Estándar 3G

Concatenación paralela de 2 CC recursivos iguales, más entrelazado y perforado, tasa ½



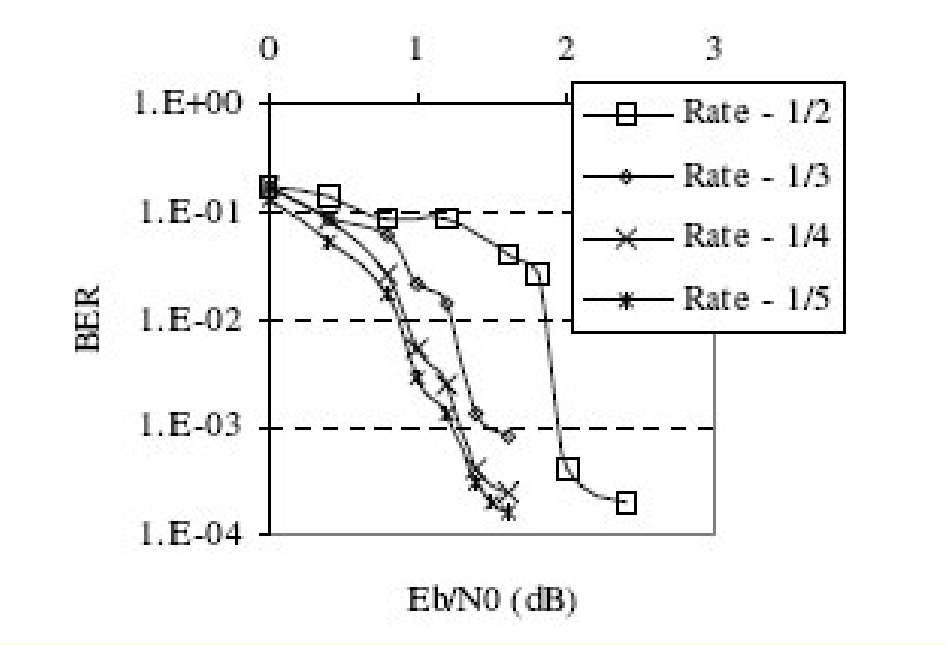


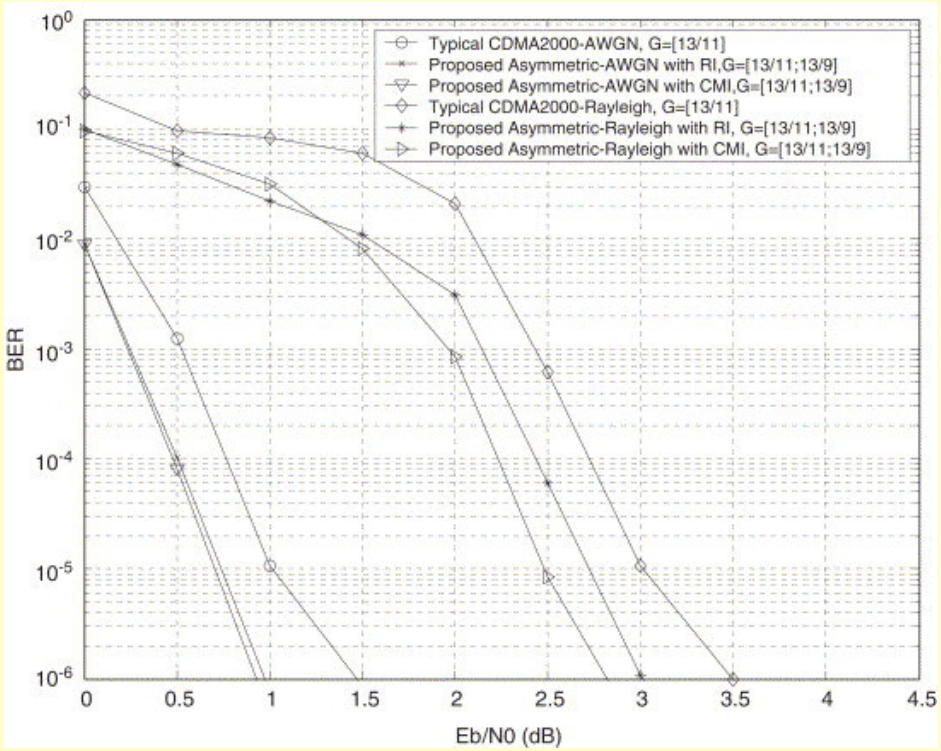
Codificación de Canal

Turbo Códigos

Estándar 3G

Ejemplos de rendimiento





Codificación de Canal
Trellis Code Modulation TCM

Combina codificación (y perforado) con modulaciones altas (M símbolos) para reducir la expansión de ancho de banda debido a una baja tasa de código.