# Variación de la performance de decodificadores LDPC de distancia Euclideana con la base logarítmica utilizada

Nicolás Wassinger[1], Mónica Liberatori[2] y Jorge Castiñeira Moreira[2]

[1] Laboratorio de Instrumentación y Control, Facultad de Ingeniería, Universidad Nacional de Mar del Plata, Juan B. Justo 4302, Mar del Plata, Argentina

[2] Laboratorio de Comunicaciones, Facultad de Ingeniería, Universidad Nacional de Mar del Plata, Juan B. Justo 4302, Mar del Plata, Argentina

**Email:**nwassinger, mlibera, casti}@fi.mdp.edu.ar

**Abstract.** En este trabajo se estudia el efecto que tiene en el rendimiento de tasa de error de la transmisión, la elección de la base logarítmica en las operaciones de cálculo de un algoritmo de decodificación de códigos LDPC basado en la métrica de la distancia Euclideana. Este algoritmo posee un funcionamiento en tasa de error que se hace dependiente de ese parámetro. El sistema de decodificación es analizado como un sistema realimentado, donde al utilizar un bloque de compensación que es básicamente una ganancia constante, se observa una fuerte reducción en la dependencia del funcionamiento con el parámetro estudiado. Sin embargo, el aumento en complejidad resultante no justifica el uso de la compensación, llegando a la conclusión de que es preferible determinar el valor optimo de la base, que depende de las características del código, y decodificar utilizando dicho valor. El estudio del valor óptimo de la base logarítmica permite llegar a la conclusión de que el mismo no depende de la relación señal ruido con la que opera el sistema.

**Keywords:** Códigos LDPC. Decodificación iterativa

## 1. Introducción

Los códigos LDCP (Low-Density Parity-Check Codes) son utilizados para la transmisión de datos en canales ruidosos permitiendo alcanzar una eficiencia cercana al límite de Shannon [1]. El concepto de código LDPC fue presentado por R. Gallager en los años 60 [2] pero, debido a limitaciones tecnológicas, su aplicabilidad fue posible recién en los años 90. Para aumentar las velocidades de transmisión y reducir los requerimientos de hardware en la decodificación de estos códigos, es deseable que la implementación de los algoritmos utilizados consuma la mínima cantidad de recursos posibles. En [3] se presentó un algoritmo denominado Simplified Soft Distance (SSD), el cual utiliza como métrica la distancia euclidiana permitiendo independizar a la implementación del decodificador de medir el ruido presente en el canal de transmisión. Los recursos necesarios para su implementación son reducidos debido al uso de matemática logarítmica junto con tablas de "Look Up". En este artículo se describe un fenómeno de pérdida de performance observado en la decodificación, el cual se relaciona con la base logarítmica utilizada. En este trabajo se estudia este fenómeno desde el enfoque de los sistemas realimentados mediante la evaluación experimental del decodificador por medio de simulaciones.

## 2. Códigos LDPC

Los códigos LDPC pertenecen al conjunto de los códigos de bloques. Estos se basan en segmentar la información a codificar en bloques de $k$ bits denominados bits de mensaje, que en conjunto constituyen $2^k$ mensajes. El codificador transforma cada bloque de datos en un bloque de $n > k$ bits denominados bits de la palabra de código. En este esquema el codificador agrega a la palabra original $n$-$k$ bits de redundancia. Este tipo de código se denomina código de bloques $C^b(n, k)$ y se define

una tasa de código asociada como la relación $k/n$ la cual mide el nivel de redundancia empleada en la codificación.

En los códigos de bloques, la información a codificar puede ser organizada en grupos de $k$ bits que constituyen un vector de mensaje $m = (m_o m_1 ...m_{k-1})$. El codificador toma este mensaje y crea un vector de código $c = (c_o\ c_1 ... c_{n-1})$ .

Existe una asignación biyectiva entre los $2^k$ vectores de mensaje y los $2^n$ vectores de código. Esta relación entre ambos conjuntos puede ser descrita por medio de una matriz generadora $G$ tal que $c = mG$. Cuando el código se encuentra en su forma sistemática la matriz $G$ tiene la forma $G\ [P\ I_k]$ y por lo tanto el vector código queda constituido por $n - k$ bits de paridad más los $m$ bits correspondientes al vector mensaje.

Se define una matriz de paridad $H$ tal que cada vector del espacio fila de la matriz $G$ es ortogonal a las filas de la matriz $H$ . En su forma sistemática la matriz H puede ser escrita como $H = [I_{n-k}\ P^T]$

Dadas las propiedades que relacionan a las matrices $G$ y $H$ se cumple que $G.H^T =$y por lo tanto $C.H^T = mG.H^T = 0$.

El vector $e$ representa la componente de ruido añadida al vector $c$ al ser afectado por el canal de transmisión. El vector recibido $r=(ro\ r1 ...rn-1)$ es entonces igual a $r=c \oplus e$ . La detección de este error se puede realizar por medio del vector síndrome $S$ el cual se define como $S = r.H^T$ Cuando el vector $r$ pertenece al código el vector $S$ resulta nulo, lo que indica que se recibió la palabra correcta o que esta contiene más errores que los que el código es capaz de detectar. Cuando el vector $S$ resulta no nulo se desprende la existencia de errores en el vector r y dependiendo de la capacidad de corrección del código se puede obtener e a partir de $S$ para luego recuperar el mensaje original. Las capacidades de detección y corrección de errores, $l$ y $t$ , de cada código representan la cantidad de errores que el código es capaz de detectar y corregir respectivamente. Estas se calculan como $l = d_{min} - 1$ , donde $t = d_{min}$ es la distancia mínima del código y se establece como la distancia mínima existente entre / todos los vectores del mismo.

Los códigos LDPC se construyen sobre la base de una matriz de paridad que tiene como característica ser de baja densidad, es decir, que la mayoría de sus elementos son nulos. Cuando el numero de '1's es fijo en filas y columnas se los denomina códigos LDPC regulares. De acuerdo a la definición de Gallager, un código LDPC se define como $C_{LDPC}\ (n,s,v)$ donde $n$ es la longitud de las palabras de bloque, de forma que la matriz de paridad correspondiente posee s '1's por columna y $v$ '1's por fila siendo normalmente $s=3$.

La construcción de la matriz $H$ se puede realizar por dos métodos: construcción aleatoria [4][5] o estructurada [6] [7]. Cuando la matriz $H$ resulta no sistemática es conveniente generar una matriz equivalente $H' = [I_{n-k}\ P^T]$ por medio de operaciones sobre filas aplicando el método de Gauss. Partiendo de ésta se puede obtener la matriz generadora como $G = [\ P\ I_k\ ]$ . Un método equivalente resulta de escribir a la matriz de paridad como $H = [\ A\ B\ ]$, siendo $A$ una matriz cuadrada de dimensión( ) ( )n k $( n - k )' (n - k )$. Luego, si existe $A^{-1}$, se puede escribir de donde $H' = A^1.H=[\ I_{n-k}\ A^{-1}\ B]$ se desprende $P^T = A^{-1}B$.

## Decodificación de códigos LDPC

Para la decodificación de los códigos LDPC se utiliza un método distinto al tradicional basado en el cálculo del síndrome. La esencia de la decodificación es la estimación del vector $d$ que cumpla la condición $H \cdot d = 0$. Se estima la probabilidad a posteriori de cada símbolo en función de la señal recibida, de las ecuaciones de paridad y de las características del canal. El algoritmo se describe sobre la base de un gráfico bipartito (Figura 1) que se define de acuerdo a la matriz $H$ en el cual se gráfica la relación entre dos tipos de nodos, los nodos representativos de los símbolos de transmisión, y los nodos representativos de las ecuaciones de paridad que vinculan a los símbolos [8]. Las filas de la matriz $H$ describen a los símbolos que se encuentran involucrados en las ecuaciones de paridad. De esta forma la conexión entre el nodo símbolo $d_j$ y la ecuación de paridad $h_i$ existirá siempre y

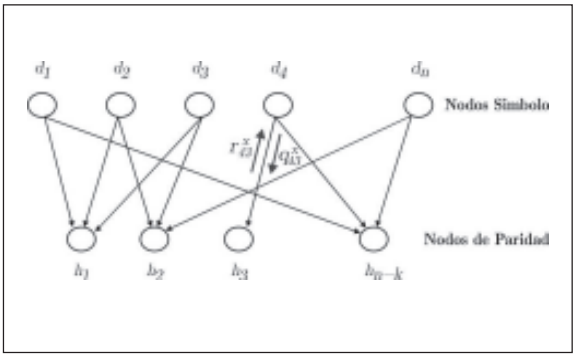



**Figura 1.** Grafo Bipartito

cuando el elemento de la matriz *H* sea uno, $H_{ij}=1$ A los nodos de paridad relacionados con un nodo símbolo $d_j$ se los denomina nodos hijos de $d_j$ mientras que a los nodos símbolo relacionados con un nodo de paridad ih se los denomina nodos padre de $h_i$.

## Algoritmo Suma-Producto

En el algoritmo de suma-producto (SP) cada nodo símbolo $d_j$ envía al nodo de paridad $h_i$ la información probabilística $q^x_{ij}$ basada en la información proporcionada por los otros nodos de paridad relacionados con el nodo símbolo, de que el nodo de paridad se encuentre en el estado *x* . Por otro lado cada nodo de paridad $h_i$ envía la información $r^x_{ij}$ a cada nodo símbolo $d_j$ informando sobre la probabilidad de que la paridad del nodo $h_i$ se satisfaga, suponiendo que el nodo símbolo se encuentra en estado *x*, tomando la información proporcionada por todos los otros nodos símbolo.

Este proceso de intercambio de información entre nodos es iterativo, y se detiene si se cumple que la condición de síndrome es satisfecha, o si el número máximo de iteraciones es alcanzado.

Las ecuaciones para el cálculo de $q^x_{ij}$ y $r^x_{ij}$ resultan ser:

$$r^x_{ij} = \sum_{\mathbf{d}:d_j=x} P(h_i / \mathbf{d}) \prod_{k\in N(i)\setminus j} Q^{dk}_{ik} \qquad (1)$$

$$q^x_{ij} = \alpha_{ij} f^x_j \prod_{k\in M(j)\setminus i} R^x_{kj} \qquad (2)$$

Estas ecuaciones se basan en las siguientes consideraciones:

- $q^x_{ij}$ se inicializa con la probabilidad a priori de los símbolos $f^x_j$ que es la probabilidad de que el j-ésimo símbolo sea x . Para el caso de un canal Gaussiano con transmisión en formato polar la probabilidad a priori puede ser obtenida como:

$$f^0_j = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(y_j+1)^2}{2\sigma^2}} \quad f^1_j = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(y_j-1)^2}{2\sigma^2}} \qquad (3)$$

  donde $y_j$ es el símbolo obtenido a la salida del canal en el instante de tiempo *j*.

- *N ( i )* representa el conjunto de subíndices de todos los nodos padre del nodo $h_i$, mientras que *N (i)* \ *j* ( indica la exclusión del nodo *j* de ese conjunto.
- *M ( j )* representa el conjunto de subíndices de todos los nodos hijo del nodo $d_j$, mientras que *M ( j )* \ *i* indica la exclusión del nodo *i* de ese conjunto.
- La constante de normalización $\alpha_{ij}$ se evalúa de forma que $q^o_{ij}+q^1_{ij}=1$

  La estimación de la decodificación para el valor del símbolo en la posición *j* se calcula como:

$$\hat{d}_j = \arg\max_x f^x_j \prod_{k\in M(j)} R^x_{kj} \qquad (4)$$

Si este vector decodificado cumple con la ecuación de síndrome entonces se lo considera un vector de código valido.

En la Tabla 1 se resumen los pasos necesarios para la ejecución del decodificador.

Estos se dividen en cuatro etapas denominadas Inicialización, Paso Horizontal, Paso Vertical y Estimación de $\hat{r}\ j$. En la etapa de Inicialización se determinan los valores iniciales para los $r^x_{ij}$ y $q^1_{ij}$ . Esta se ejecuta una única vez en la primera iteración del algoritmo. En los pasos horizontal y vertical se actualizan, en cada iteración, las probabilidades xijr y xijq respectivamente. Finalmente se calcula la Estimación de $\hat{r}$ y se evalúa su síndrome resolviendo en función de este si la palabra estimada pertenece al código. En caso contrario se continúa iterando.

**Tabla 1.**
Algoritmo del Decodificador de Suma Producto.

| Métrica |
|---|
| $f^0_j = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(y_j+1)^2}{2\sigma^2}} \quad f^1_j = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(y_j-1)^2}{2\sigma^2}}$ |
| Inicialización |
| $q^x_{ij} = f^x_j$ |
| Paso horizontal |
| $r^x_{ij} = \sum_{\mathbf{d}:d_j=x} P(h_i / \mathbf{d}) \prod_{k\in N(i)\setminus j} Q^{d_k}_{ik}$ |
| Paso vertical |
| $q^x_{ij} = \alpha_{ij} f^x_j \prod_{k\in M(j)\setminus i} R^x_{kj}$ |
| Estimación del símbolo decodificado |
| $\hat{d}_j = \arg\max_x f^x_j \prod_{k\in M(j)} R^x_{kj}$ |

## Algoritmo Suma-Producto Simplificado

En [4] se presentó el algoritmo de Suma-Producto Simplificado el cual permite reducir los requerimientos de cálculo del SP ya que no requiere tener en cuenta todos los casos posibles de la ecuación de paridad en el cálculo de $q^x_{ij}$ . En este algoritmo, la información del canal se obtiene haciendo uso de las expresiones siguientes:

$$f_j^1 = \frac{1}{1+e^{\frac{2Ay_j}{\sigma^2}}},\ f_j^0 = 1 - f_j^1 \qquad (5)$$

La Tabla 2 contiene las ecuaciones correspondientes al algoritmo simplificado.

**Tabla 2.**
Algoritmo del Decodificador de Suma Producto Simplificado.

| Métrica |
|---|
| $f_j^1 = \frac{1}{1+e^{\frac{2Ay_j}{\sigma^2}}}\quad f_j^0 = 1 - f_j^1$ |
| Inicialización |
| $q_{ij}^x = f_j^x$ |
| Paso horizontal |
| $dq_{ij} = q_{ij}^0 - q_{ij}^1$ |
| $dr_{ij} = \prod_{j' \in N(i)\setminus j} dq_{ij'}$ |
| $r_{ij}^0 = \frac{1+dr_{ij}}{2}$ |
| $r_{ij}^1 = \frac{1-dr_{ij}}{2}$ |
| Paso vertical |
| $q_{ij}^x = \alpha_{ij} f_j^x \prod_{i' \in M(j)\setminus i} r_{i'j}^x$ |
| Estimación del símbolo decodificado $\hat{r}_j$ |
| $q_j^x = \alpha_j f_j^x \prod_{i \in M(j)} r_{ij}^x$ |

## Algoritmo Suma-Resta

La complejidad en la implementación del algoritmo de suma producto simplificado puede ser reducida mediante el uso del cálculo logarítmico. Éste permite convertir productos y cocientes en sumas y restas. La probabilidad a priori se obtiene haciendo uso de las expresiones siguientes:

$$F_j^1 = e^{-f_j^1} = \frac{1}{1+e^{\frac{2y}{\sigma^2}}},\ F_j^0 = e^{-f_j^0} = 1 - F_j^1 \qquad (6)$$

El algoritmo se basa en el hecho de que, para una probabilidad $F_j^x$ menor a 1 , se cumple que, si $F_j^x = e^{-|f_j^x|}$ entonces $|f_j^x| = |\ln(F_j^x)|$ . Luego $f_j^-$ y $f_j^+$ pueden ser calculadas como (ver apéndice):

$$f_j^1 = \ln(1+e^{-2y_j/\sigma^2}) = f_+(2y_j/\sigma^2,0),\ f_j^0 = 2y_j/\sigma^2 + f_+(2y_j/\sigma^2,0) \qquad (7)$$

Aplicando esta simplificación surgen las ecuaciones del decodificador Suma-Resta (SR), las cuales se resumen en la Tabla 3. Las funciones $f_+(\cdot)$ y $f_-(\cdot)$ pueden ser implementadas mediante tablas de búsqueda o "Look Up".

**Tabla 3.**
Algoritmo del Decodificador de Suma-Resta.

| Métrica |
|---|
| $f_j^1 = \ln(1+e^{-2y_j/\sigma^2}) = f_+(2y_j/\sigma^2,0)$<br>$f_j^0 = 2y_j/\sigma^2 + f_+(2y_j/\sigma^2,0)$ |
| Inicialización |
| $q_{ij}^0 = f_j^0\quad q_{ij}^1 = f_j^1$ |
| Paso horizontal 1 |
| $db_{ij} = \max(-q_{ij}^0,-q_{ij}^1) + f_-(q_{ij}^0,q_{ij}^1)$<br>$s_{ij} = 0$ si $-q_{ij}^0 > -q_{ij}^1$, sino $s_{ij} = 1$ |
| Paso horizontal 2 |
| $db_{ij} = \sum_{j' \in N(i)\setminus j} dq_{ij'}$<br>$sdb_{ij} = \sum_{j' \in N(i)\setminus j} s_{ij'}$ |
| Paso horizontal 3 |
| si $sdb_{ij}$ es par $\quad r_{ij}^0 = \log(2) - f_+(db_{ij},0)$<br>$r_{ij}^1 = \log(2) + f_-(db_{ij},0)$ |
| si $sdb_{ij}$ es impar $\quad r_{ij}^0 = \log(2) - f_+(db_{ij},0)$<br>$r_{ij}^1 = \log(2) - f_+(db_{ij},0)$ |
| Paso vertical |
| $c_{ij} = -f_j^1 + \sum_{i' \in M(j)\setminus i} r_{i'j}^1$<br>$q_{ij}^0 = -c_{ij}^0 - \max(c_{ij}^0,c_{ij}^1) + f_-(c_{ij}^0,c_{ij}^1)$<br>$q_{ij}^1 = -c_{ij}^1 - \max(c_{ij}^0,c_{ij}^1) + f_-(c_{ij}^0,c_{ij}^1)$ |
| Estimación del símbolo decodificado $\hat{r}_j$ |
| $c_{ij} = -f_j^1 + \sum_{i' \in M(j)\setminus i} r_{i'j}^1$<br>$\hat{r}_j = 0$ si $c_j^0 > c_j^1$, sino $\hat{r}_j = 1$ |

## Algoritmo de Distancia Suave

En el algoritmo de distancia suave o "Soft Distance", (SD) se utiliza como métrica la distancia euclidiana en lugar de recurrir a cálculos probabilísticos. Esta métrica no requiere del conocimiento desvalor de dispersión de ruido σ, lo que permite independizar a la implementación del método de las condiciones del canal de transmisión. Las ecuaciones del algoritmo resultan las mostradas en la Tabla 4.

**Tabla 4.**
Algoritmo del Decodificador de Distancia Suave.

| Métrica |
|---|
| $d_j^1 = (y_j - 1)^2 \quad d_j^0 = (y_j + 1)^2$ |
| Inicialización |
| $q_{ij}^0 = d_j^0 \quad q_{ij}^1 = d_j^1$ |
| Paso horizontal |
| $r_{ji}^x = -\log_2 \sum_{\forall i' \neq i} \left[ 2^{-\left( \sum_{i' \in M(j) \setminus i} q_{i'j}^{x_{i'}} \right)} \right]$ |
| Paso vertical |
| $q_{ij}^0 = d_j^x + \sum_{i \in M(j)} r_{ij}^0$ |
| Estimación del símbolo decodificado $\hat{r}_j$ |
| $\hat{r}_j = \arg\min_x \left[ d_j^x + \sum_{i \in M(j)} r_{ij}^x \right]$ |

**Tabla 5.**
Algoritmo del Decodificador de Distancia Suave Simplificado.

| Métrica |
|---|
| $d_j^1 = (y_j - 1)^2 \quad d_j^0 = (y_j + 1)^2$ |
| Inicialización |
| $q_{ij}^0 = d_j^0 \quad q_{ij}^1 = d_j^1$ |
| Paso horizontal 1 |
| $sq_{ij} = +f_+(q_{ij}^0, q_{ij}^1)$ |
| $dq_{ij} = -f_-(q_{ij}^0, q_{ij}^1)$ |
| $s_{ij} = 0$ si $-q_{ij}^0 > -q_{ij}^1$, sino $s_{ij} = 1$ |
| Paso horizontal 2 |
| $sr_{ji} = \sum_{i' \in M(j) \setminus i} sq_{ji'}$ |
| $dr_{ji} = \sum_{i' \in M(j) \setminus i} dq_{ji'}$ |
| $sdr_{ji} = \sum_{i' \in M(j) \setminus i} s_{ji'}$ |
| Paso horizontal 3 |
| si $sdr_{ij}$ es par $\quad r_{ij}^0 = -f_+(sr_{ij}, dr_{ij})$ |
| $r_{ij}^1 = +f_-(sr_{ij}, dr_{ij})$ |
| si $sdr_{ij}$ es impar $\quad r_{ij}^1 = +f_-(sr_{ij}, dr_{ij})$ |
| $r_{ij}^0 = -f_+(sr_{ij}, dr_{ij})$ |
| Paso vertical |
| $q_{ji}^x = d_j^x + \sum_{i' \in M(j) \setminus i} sr_{ji'}^x$ |
| Estimación del símbolo decodificado $\hat{r}_j$ |
| $r_j^0 = q_{ij}^0 + r_{ij}^0$ |
| $r_j^1 = q_{ij}^1 + r_{ij}^1$ |
| $\hat{r}_j = 1$ si $r_j^1 < r_j^0$, sino $\hat{r}_j = 0$ |

## Algoritmo de Distancia Suave Simplificado

De forma equivalente a la simplificación que permite alcanzar el algoritmo SR a partir del de SP, aplicando los conceptos presentados en [4] y haciendo uso del cálculo logarítmico, el algoritmo SD puede ser simplificado para obtener el algoritmo de Distancia Suave Simplificado, o "Simplified Soft Distance" (SSD). Este algoritmo fue presentado en [3], donde se indicó la existencia de un fenómeno de pérdida de performance al variar la base logarítmica $b$ . Este fenómeno se estudia en la próxima sección desde el enfoque de los sistemas realimentados. Las ecuaciones correspondientes al decodificador SSD se indican en la Tabla 5.

## 4. Análisis mediante simulaciones del decodificador DSS

En esta sección se realiza un estudio mediante simulaciones del comportamiento del decodificador DSS en función a la variación del parámetro $b$ . En todos los casos presentados se ensaya la decodificación de un conjunto de 1000 palabras de código transmitidas sobre un canal Gaussiano con una *Eb / No* asociada, acotando a 16 la cantidad de iteraciones máximas ejecutadas por el decodificador.

En la Figura 2 se muestran un conjunto de curvas de la probabilidad binaria de error *Pbe vs*. La relación energía promedio de bit a densidad de ruido, *Eb /No* correspondientes a un código $C_{LDPC}$(273,171) para diferentes valores de la base logarítmica *b*. Como se puede observar, para valores de *Eb / No* elevados existe una variación significativa en la performance del código dependiente del *b* utilizado.

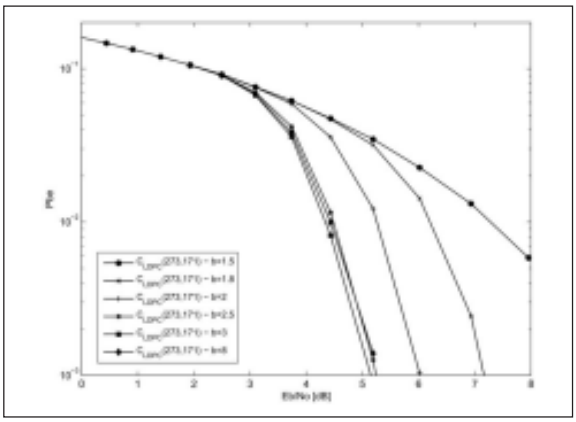
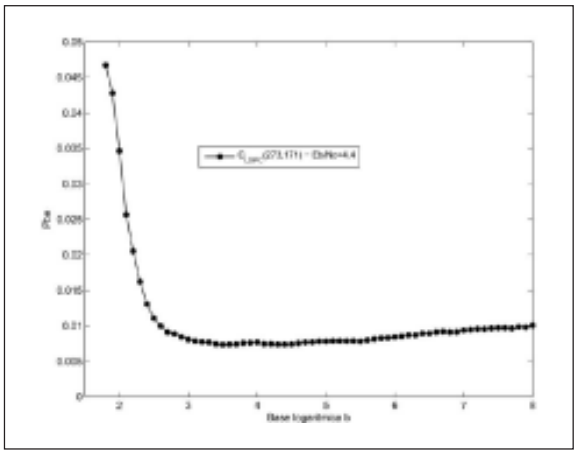


**Figura 2.**
Performance en la decodificación de un código $C_{LDPC}$(273,171 para diferentes *b*.

Se puede notar que la performance del código aumenta en forma continua al aumentar el valor de *b* utilizado desde 1.5 a 3 pero que luego para

*8b =* esta se reduce nuevamente. Este efecto se analiza con más detalle en la Figura 3 donde se grafica Pbe vs. *b* para una *Eb / No* fija igual a 4.4 .

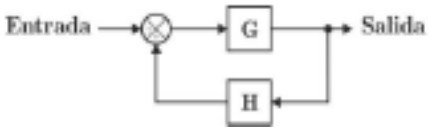
**Figura 3.**
*Pbe* en la decodificación de un código $C_{LDPC}(273,171)$ frente a la variación de b para una *Eb / No* fija.



En la Figura 3 se observa que la *Pbe* disminuye al aumentar b hasta alcanzar un valor mínimo para $b \approx 3.5$ y luego comienza a aumentar lentamente al continuar aumentando*b* . En esta figura, *Pbe* se grafica en forma lineal ya que permite observar este efecto con más claridad.

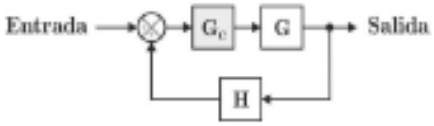
Para estudiar el fenómeno descripto, se analiza el comportamiento del decodificador SSD visto como un sistema realimentado. En la Figura 4 se muestra la estructura de un sistema realimentado básico. En esta estructura existen dos bloques: el bloque *G* representa una transferencia hacia adelante y el bloque *H* la transferencia de la realimentación.

**Figura 4.**
Diagrama en bloques de un sistema realimentado básico.



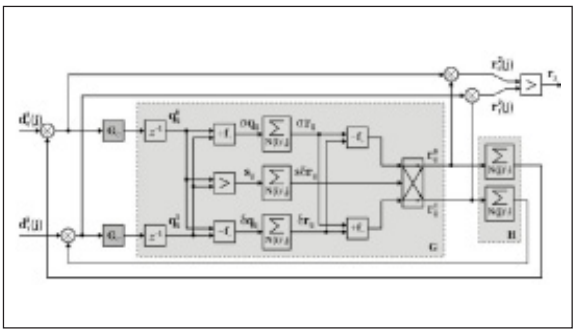

El diseño de los sistemas realimentados se basa en el cumplimiento de especificaciones tales como dinámica, ganancia, rechazo a perturbaciones, etc. En los casos donde la transferencia del sistema realimentado no es la deseada, es posible agregar un bloque de compensación *GC* . Con este bloque se busca modificar el comportamiento original del sistema para que este se aproxime, tanto como sea factible, al comportamiento deseado. En la Figura 5 se muestra el diagrama en bloques del sistema compensado.

**Figura 5.**
Diagrama en bloques de un sistema realimentado básico con compensador



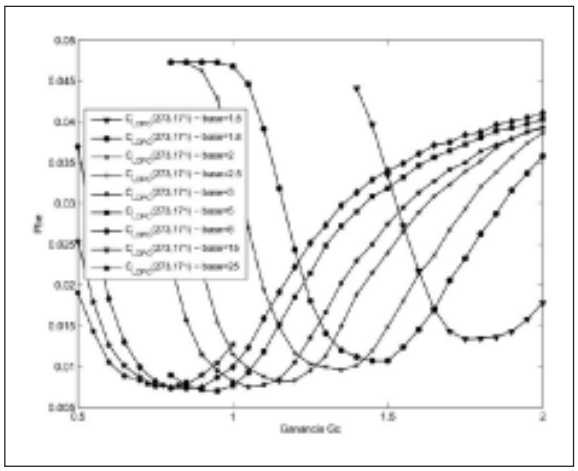

En la Figura 6 se muestra un diagrama en bloques del decodificador *SSD*. En este diagrama se considera que las iteraciones se ejecutan en forma periódica con un periodo $T_i$, lo que permite tratar al decodificador como un sistema discreto realimentado. El bloque $z^{-1}$representa un retardo de $T_i$ y el bloque $G_C$representa una transferencia que depende de la estrategia de compensación utilizada. El bloque de transferencia hacia adelante *G* está integrado por los pasos horizontales y el bloque de realimentación *H* se compone parcialmente por el paso vertical.

**Figura 6.**
Diagrama en bloques del decodificador *SSD* ($z^{-1}$ representa un retardo de una iteración).



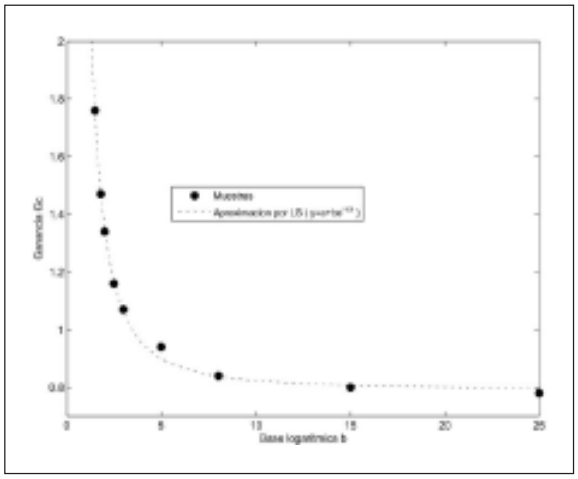
Dada la multiplicidad de no-linealidades existentes en el modelo de la Figura 7 (comparaciones, sumatorias selectivas, logaritmos) no resulta factible el estudio del sistema por medio de métodos convencionales. Otra forma de estudiar el decodificador es en forma indirecta, es decir, aplicando un compensador determinado y analizando el efecto que éste produce sobre el comportamiento del sistema. En este trabajo se propone utilizar un compensador proporcional definido como una ganancia constante *GC=cte.*

**Figura 7.**
Pbe en la decodificación de un código $C_{LDPC}(273,171)$ frente a la variación de la ganancia del compensador *GC* para diferentes *b.*



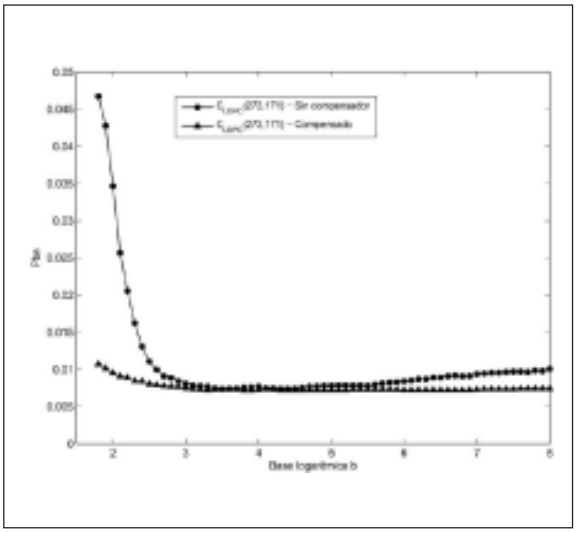

En la Figura 8 se muestra un conjunto de curvas para diferentes b donde se analiza el efecto de la variación del valor de *GC* sobre la *Pbe*. Como se puede ver, en todas las curvas existe un *GC* que minimiza la *Pbe* . Además, este valor de *GC* óptimo varía según la base utilizada.

**Figura 8.**
*GC* óptimas en relación a *b* en la decodificación de un código $C_{LDPC}(273,171)$ junto con una curva aproximada por mínimos cuadrados.




Para analizar la relación existente entre el *b* utilizado y la *GC* óptima correspondiente, se grafican en la Figura 8 el conjunto de puntos *GC* óptima vs. *b* obtenidos de la Figura 7. Aplicando el algoritmo de mínimos cuadrados se aproxima esta relación a una curva de la forma $y=a+be^{-cx}$ la cual se grafica en líneapunteada en la misma figura. Los parámetros obtenidos son: $a$=0.79266, $b$=1.9881 y $c$=1.8132. Para validar el funcionamiento del compensador se simula el algoritmo SSD compensado. En la Figura 9 se comparan las curvas *Pbe vs. b* correspondientes al sistema original y al compensado. Como se puede ver, el sistema compensado mejora en forma significativa el efecto de pérdida de performance para *b* chicos y lo cancela para valores de *b* superiores al óptimo.

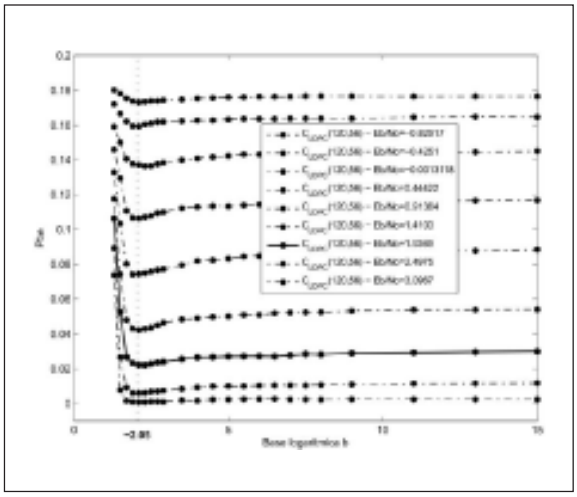
**Figura 9.**
*Pbe* en la decodificación de un código $C_{LDPC}(273,171)$frente a la variación de *b* para una *Eb /No* fija. Decodificador original y compensado.




En la Figura 9 se ve que el uso de un compensador proporcional resulta en mejoras significativas en la curva *Pbe vs b*. El inconveniente de esta estrategia de corrección es que requiere de la implementación de un número elevado de multiplicadores lo que interfiere con el objetivo de simplicidad del algoritmo DSS. Por otra parte, el uso del compensador ecualiza la curva *Pbe vs b* pero no logra mejorar el valor de *Pbe* mínimo, por lo que la optimización del algoritmo se puede realizar por medio de la elección adecuada del b utilizado.

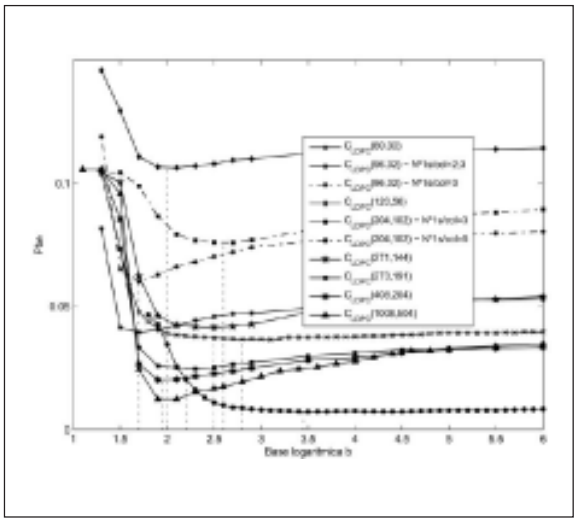
A continuación, se estudia la ubicación del *b* óptimo en relación al código y a las características del canal de transmisión. En la Figura 10 se muestran un conjunto de curvas *Pbe vs. b* para diferentes *Eb / No*. Se puede observar que la posición de la base óptima no varía con la *Eb / No*. Esta independencia permite asociar a cada código un *b* óptimo fijo e independiente del canal de transmisión.

**Figura 10.**
*Pbe* en la decodificación de un código $C_{LDPC}$(120, 56) frente a la variación de *b* para diferentes *Eb / No.*



En la Figura 11 se muestran las curvas *Pbe vs. b* para un conjunto de códigos con diferentes características. A partir de ésta se obtienen los *b* óptimos para cada uno de los códigos. En la Tabla 6 se muestran las características de los códigos evaluados junto con el *b* óptimo correspondiente. Como se puede observar, el valor de *b* óptimo resulta dependiente tanto de la longitud como de la tasa y de la cantidad de unos por columna en la matriz H del código.

**Figura 11.**
*Pbe* frente a la variación de b para un conjunto de códigos LDPC con diferentes características.



## 5. Conclusiones

En este trabajo se realizó un estudio del comportamiento del decodificador SSD para códigos LDPC mediante simulaciones en el entorno MATLAB. Se ensayó el efecto sobre la performance en la decodificación de la base logarítmica *b* utilizada. Se observó la existencia de un b óptimo y un deterioro de la performance para valores diferentes siendo más notable cuando el *b* elegido es inferior al óptimo.Se analizó el decodificador desde el enfoque de los sistemas realimentados y se propuso el agregado de un compensador del tipo proporcional con ganancia *GC*. Se realizaron ensayos de la variación de la performance en función a la ganancia *GC* para diferentes *b*. Se observó la existencia de valores de *GC* que maximizan la performance en la decodificación siendo estos valores diferentes según la b utilizada.

Mediante el método de mínimos cuadrados se obtuvo una función que aproxima a los valores de *GC* óptimos en función de*b* .

Aplicando la estrategia de compensación propuesta se evaluó nuevamente el comportamiento del compensador y se obtuvieron mejoras significativas en la performance frente al sistema sin compensar. Como el decodificador SSD se caracteriza por su simplicidad no resulta deseable el agregado del calculo que requiere esta estrategia por lo que se propuso variar la *b* en función al código utilizado manteniendo el valor de *GC* constante e igual a 1.

Se estudió la variación de la *b* óptima con los parámetros del código y las características del canal de transmisión. Se concluyó que el valor de la *b* óptima no presenta variaciones significativas frente a la variación de la relación señal a ruido en el canal. Esto es de gran

**Tabla 6.**
Base logarítmica óptima *b* para un conjunto de códigos LDPC con diferentes características.

| Longitud | Tasa | Nºls/columna | Base Óptica |
|---|---|---|---|
| 60-30 | 0.5 | 3 | 2.8 |
| 96-32 | 0.3 | 2;3 (2.67) | 1.7 |
| 96-32 | 0.3 | 3 | 1.7 |
| 120-56 | 0.46 | 3 | 2 |
| 204-02 | 0.5 | 3 | 2.2 |
| 204-102 | 0.5 | 5 | 2.6 |
| 271-144 | 0.53 | 3 | 2.5 |
| 273-191 | 0.69 | 3 | 3.45 |
| 408-204 | 0.5 | 3 | 1.95 |
| 1008-504 | 0.5 | 3 | 2 |

importancia ya que independiza a la elección de la b de las características del canal de transmisión resultando un b fijo para cada código. Se generó una tabla con las b óptimas para un conjunto de códigos con características diversas observándose dependencia de este valor tanto con la longitud del código como con su tasa y con la cantidad de unos por columna en la matriz H .

## Apéndice: Algebra Logarítmica

Si se define $X=b^x$, $y = b^y$ y $Z = b^z$, entonces para $Z = X + Y=$ se cumple que: $z = (x, y) + \log_b (1 + b^{-|x - y|})$

Además, para Z = X - Y se cumple que: z = max (x, y) - $|\log_b (1-b^{-|x-y|})|$

El cálculo de los logaritmos se realiza mediante tablas de Look Up $f_+(x,y)$ y $f_-(x,y)$.

## Bibliografia


1. Castiñeira Moreira, J., Farrell, P.: Essentials of error-control coding. John Wiley & Sons, (2006).

2. Gallager, R. Low-density parity-check codes, IRE Transactions on Information Theory, (1962) 8 21-28.

3. Farrell, P. G., Arnone, L. J., Castiñeira Moreira, J.: Euclidean Distance Soft-Input Soft-Output Decoding for LDPC Codes, IET Communications, (2011).

4. MacKay, D., Neal, R.: Near Shannon limit performance of low density parity check codes, Electronics Letters, (1997), 33, 457 -458.

5. MacKay, D.: Good error-correcting codes based on very sparse matrices IEEE Transactions on Information Theory, (1999), 45, 399-431.

6. Kou, Y., Lin, S., Fossorier, M.: Low-density parity-check codes based on finite geometries: a rediscovery and new results, IEEE Transactions on Information Theory, (2001) 47 2711-2736.

7.Ammar, B., Honary, B., Kou, Y., Xu, J., Lin, S.: Construction of low-density parity-check codes based on balanced incomplete block designs, IEEE Transactions on Information Theory (2004) 50 1257–1269.

8. Tanner, R.: A recursive approach to low complexity codes, IEEE Transactions on Information Theory, (1981) 27 533–547.