

TRANSMISIÓN DE DATOS 2006/2007		
Examen Prácticas Extraordinario		15 de Enero de 2007
		Calificación
Apellidos, Nombre		
DNI		

## Normas

- Se dispone de un máximo de tres horas para la realización del examen.
- El examen es individual.
- **NO** se podrá hacer uso del código de las prácticas realizadas durante el curso.
- Se puntuará la corrección de los resultados, pero también la “calidad” del código (legibilidad, comentarios, uso de MATLAB, etc.).
- Se entregará esta memoria, así como un fichero .zip (por el método de entrega de prácticas habitual indicando el grupo A) con el siguiente nombre:  
*“examenExtraordinarioTxDatosEnero06\_PrimerApellido\_SegundoApellido\_Nombre.zip”*.
- Todos los ficheros desarrollados deberán incluir un comentario en la cabecera con el nombre y los apellidos de su autor.
- Se evaluarán las funciones y ficheros implementados con los nombres, entradas y salidas indicados en el enunciado del examen.
- No se evaluarán ficheros que produzcan errores en MATLAB al ejecutarlos
- El fichero .zip entregado deberá contener todos los ficheros MATLAB necesarios para ejecutar el código escrito en el examen (cualquier fichero o función auxiliar que se haya utilizado).
- La hora límite para entrega de los ficheros será exactamente tres horas después del comienzo del examen. A partir de dichas hora se penalizará con un punto cada minuto de retraso.
- Este examen consta de dos ejercicios. Es **imprescindible obtener al menos 2 puntos en cada ejercicio** para que se evalúe el resto del examen.

## 1. Ejercicio 1: Codificación PCM no uniforme (4 puntos)

En este ejercicio se habrá de implementar un codificador G.711 modificado.

La Recomendación G.711, divide el rango (simétrico respecto al origen) en 13 segmentos rectilíneos. Si nos centramos en la zona positiva, se generan 7 segmentos (realmente 6+1+1). El segmento 7 ocupa la mitad del rango normalizado a 1 (Unidad de Tensión Normalizada –UTN–), esto es, de 1 a 0.5. El segmento 6 de 0.5 a 0.25, y así hasta llegar al segmento 1. El segmento 1 se divide en dos segmentos de igual tamaño (el 1 y el 0) por lo que suele hablar de 6+1+1 segmentos. Dentro de cada segmento no uniforme, se cuantifica el rango mediante 16 intervalos de cuantificación uniformes. La codificación **estándar** consiste en asignar un bit al signo (1 si positivo), 3 bits al segmento, y 4 bits al intervalo de cuantificación.

Las pruebas de la implementación se realizarán sobre el fichero de audio ‘cuantif\_A.mat’ y sobre la secuencia ‘cuantif\_B.mat’.

### PCM Uniforme: (2 puntos)

En el desarrollo del G.711 modificado será necesaria la codificación uniforme dentro de cada segmento. Se pide una función con el siguiente interfaz:

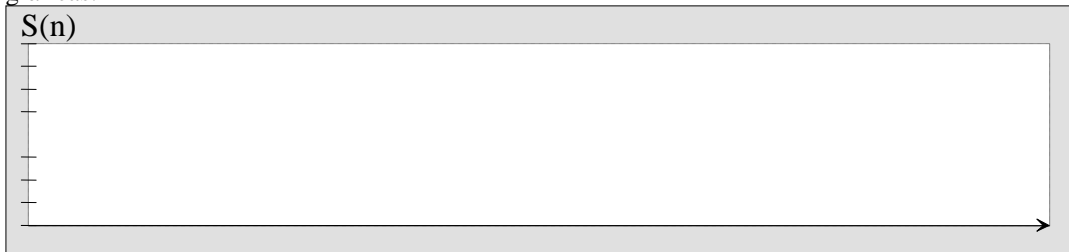
```
function [Scuan Srecon]=PCMUniforme(S,nbits,v)
%S:          secuencia de audio grabada
%nbits:      número de bits que usaremos para la codificación
%v:         valor de sobrecarga
%Scuan:     índice de los intervalos para la señal de entrada
%Srecon:    señal de entrada reconstruida
```

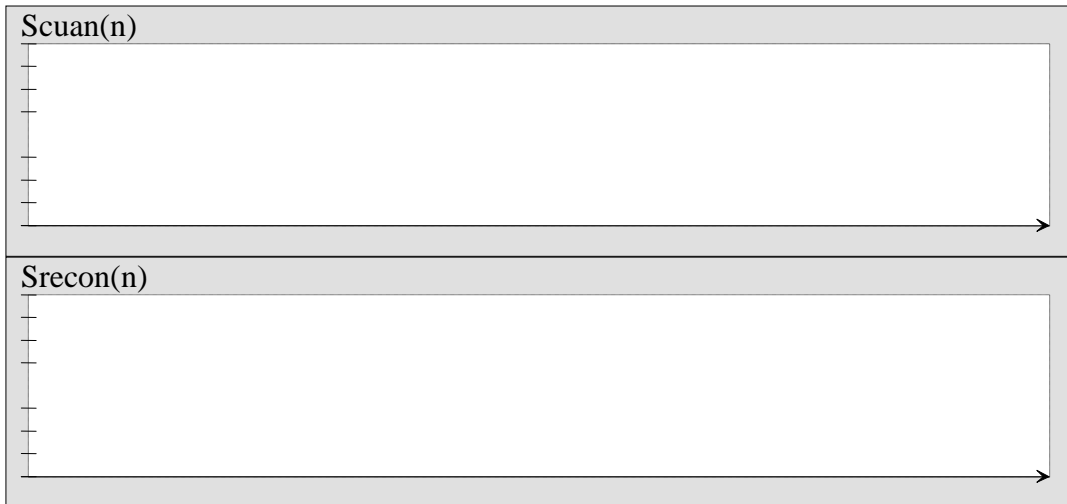
Además de realizar la codificación, habrá de presentar por pantalla una figura con tres gráficas (S, Scod y Srecon).

Ejecutar la función para la secuencia ‘cuantif\_A.mat’.  
¿Cuánto vale la potencia de error?

**Perror =**

Dibujar las señales indicadas en las siguientes gráficas:





### Implementación modificada de G711: (2 puntos)

En esta modificación del G711, asignaremos 1 bit al signo, 2 (en vez de 3) a cada segmento y 5 (en vez de 4) a cada intervalo.

<b>G711:</b>	Signo	Segmento			Intervalo			
<b>ModG711:</b>	Signo	Segmento		Intervalo				

¿Cuántos segmentos tendremos?

¿Cuántos intervalos por segmento?

¿Cuáles serán los segmentos en el tramo positivo (su límite inferior y superior)?

Indice	Lim. Inf	Lim. Sup

El interfaz de la función a implementar será el siguiente:

```
function [Scod Srecon]= ModG711 (S)
%S:      secuencia de entrada
%Scod:   secuencia de valores 0-255 [signo(1bit:+ 0,- 1)
%        segmento(2bits) %intervalo(5bits)]
%Srecon: señal de entrada reconstruida (normalizada)
```

La función devolverá en Scod la codificación descrita, y en ese Srecon la señal reconstruida. Además habrá de presentar por pantalla una figura con dos gráficas (S y Srecon). Se recomienda una implementación dividida en las siguientes partes:

- 1) Normalización de la señal: que ningún valor sea menor que -1 o mayor que 1.
- 2) Definición de una matriz con los valores inferior y superior de cada segmento
- 3) Obtención del signo para cada una de las muestras.
- 4) Obtención de los índices de los segmentos a los que pertenece cada una de las muestras.
- 5) Realización de la cuantificación uniforme para cada segmento, utilizando la función implementada en el apartado anterior (la entrada deberá ser normalizada). Esto nos dará los índices de los intervalos a los que pertenece cada muestra.
- 6) Codificación de los resultados,  $S_{cod}$ .
- 7) Reconstrucción de la señal ( $S_{recon}$ ) usando: signos, segmentos e intervalos de cada muestra.

Ejecutar la función para la secuencia de entrada 'cuantif\_B.mat'.

Rellenar la siguiente tabla con los resultados:

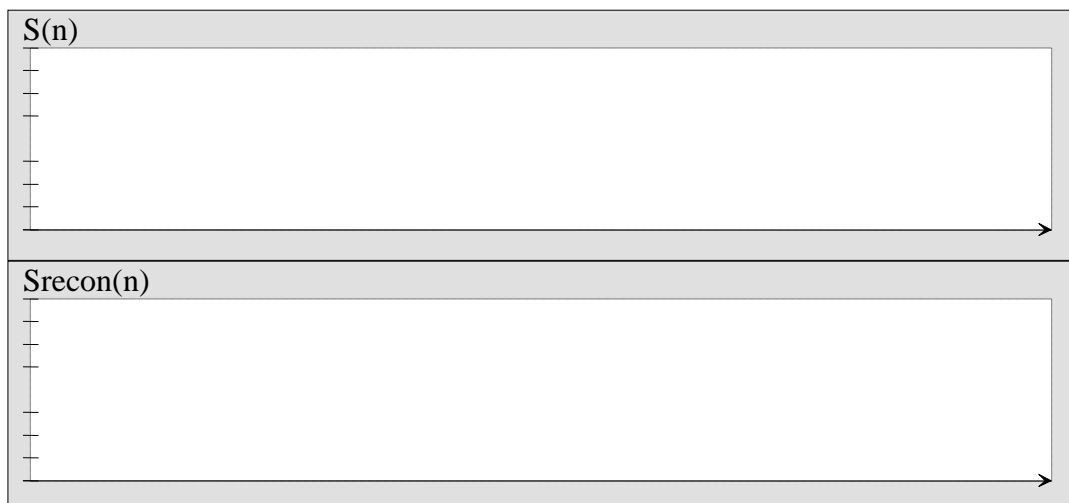
Valor de entrada	Scuan

Ejecutar la función para la secuencia 'cuantif\_A.mat'.

¿Cuánto vale la potencia de error?

**Perror =**

Dibujar las siguientes gráficas:





### Detección de errores: (3 puntos)

Se piden las funciones (las mismas que las implementadas en la práctica 5):

```
function [H] = matrizChequeoParidad(G, n, k)
%H:      matriz de chequeo de paridad
%G:      matriz generatriz
%k:      longitud de la palabra código

function [sindromes, errores] = calculoSindrome(c,H)
%sindromes:  matriz cuyas filas son los síndromes para
%            cada palabra código de la matriz c
%errores:    vector columna que indica la ocurrencia o no
%            de error en cada una de las palabras código
%c:          Palabras código sobre las que calcular el
%            síndrome
%H:          matriz de chequeo de paridad
```

**Nota:**  $H = [P^T \mid I_{(n-k)}]$  y  $s = c H^T$

Genere (con la función `SecAleatoria`) un patrón de errores de longitud 48 con probabilidad de error de bit de 0.3.

**error =**


Aplique el patrón de error al código generado en el apartado anterior:

**codigo63Errores = codigo63+ error**


Rellene la siguiente tabla con las palabras código resultado de codificar todas las posibles palabras mensaje:

Palabras Mensaje:	Palabras código:	Síndromes:

Rellene la siguiente tabla con las palabras código fruto de la codificación realizada con la aplicación del patrón de error (**codigo63Errores**):

Palabras mensaje:	Palabras código:	Síndromes:

Interprete los resultados: