Guion 7: Codificación Diferencial

Información sobre la entrega de la práctica

Las prácticas se entregarán en un único fichero comprimido Practica07ApellidoNombre.zip. El fichero contendrá:

- Las funciones de Matlab a realizar en ficheros .m con los nombres de las funciones que se indiquen en el guion.
- Los trozos de código a realizar, que se entregarán todos en los pasos correspondientes de un único fichero .m llamado Practica07ApellidoNombre.m . Este fichero lo crearás modificando el fichero .m Practica07MolinaRafael.m en el servidor.
- Las discusiones y respuestas solicitadas en el guion se entregarán en un único fichero pdf. El nombre del fichero será Practica07ApellidoNombre.pdf. Lo construirás editando Practica07MolinaRafael.doc y salvándolo en formato pdf.

Paso 1

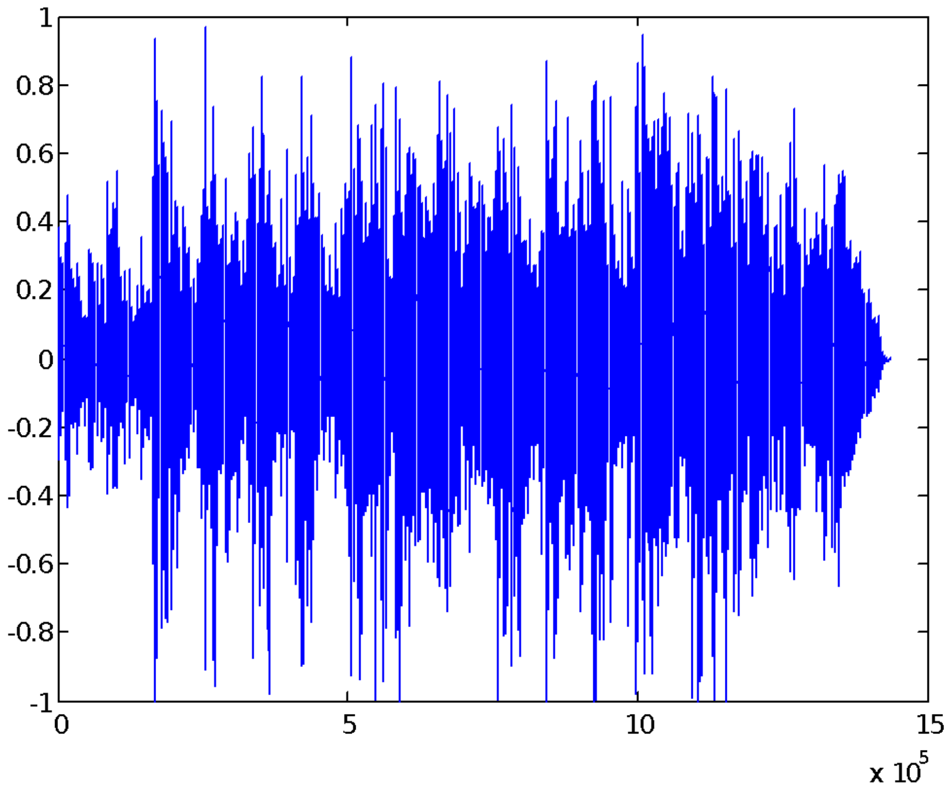
Lee el fichero de audio lazyrn16bits.wav y entiende que se almacena en las variables y y fs

```
[y,fs]=audioread('lazyrn16bits.wav');
```

Paso 2

Dibuja la señal de audio y reproduce el sonido

```
plot(y);
sound(y,fs);
```



Paso 3

A continuación vamos a calcular los coeficientes del mejor predictor de orden uno para la señal y. Es decir, vamos a suponer que la señal se puede predecir bien utilizando *y(i) = a + by(i-1)* y encontraremos los valores a y b que minimizan el error cuadrático

$$\sum_i (y(i) - a - by(i-1))^2$$

Para ello utilizamos

```
predictor=dpcmopt(y,1)
```

cuya salida (predictor) es el vector fila [a b] con a= 0 y b= 0.9863

Paso 4

Vamos ahora a construir un predictor óptimo de orden 1 que tenga cuatro particiones, es decir, los errores de cuantificación van a estar en uno de cuatro posibles intervalos. Las particiones aparecen en partition (tres números, entiende que significan estos números) y los cuatro valores que representan a las particiones estarán en codebook. Examina el contenido de todas las variables

```
[predictor,codebook,partition]=dpcmopt(y,1,4);
```

Paso 5

A continuación, para cada elemento de y, calculamos el índice de la partición (intervalo de cuantificación), comenzando en 0, donde está el error de predicción, es decir, la diferencia entre y(i) y su predicción basada en la reconstrucción de y(i-1)

```
indx=dpcmenco(y,codebook,partition,predictor);
```

Paso 6

Ahora decodificamos la señal comprimida y oímos la música reconstruida mediante

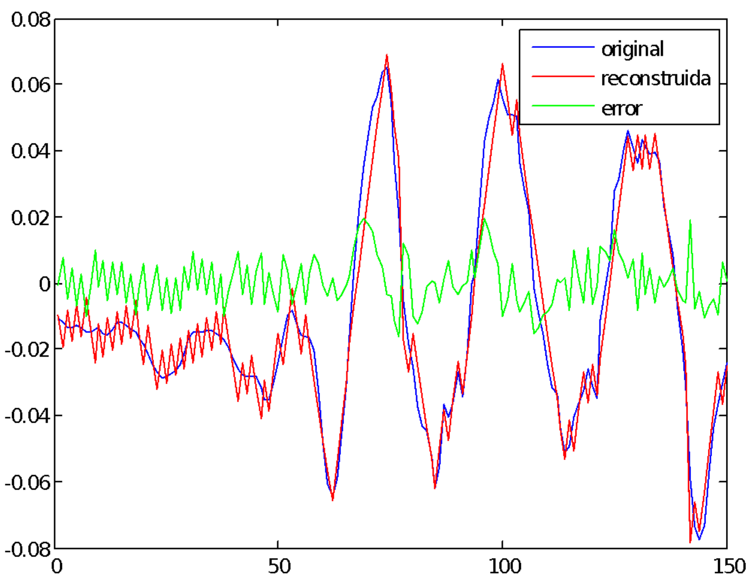
```
y2= dpcmdeco(indx,codebook,predictor);
sound(y2,fs);
```

Si pruebas con un mayor número de particiones y un mayor orden de predicción y2 sonará más similar todavía a la señal original y.

Paso 7

Analiza también la reconstrucción y el error en algún segmento de la señal de audio. Por ejemplo ejecuta

```
plot(y(1:150),'b')
hold on;
plot(y2(1:150),'r');
error = y(:)-y2(:);
plot(error(1:150),'g');
legend('original', 'reconstruida', 'error')
sound(error,fs)
```



Paso 8

Responde razonadamente en el paso 8 de Practica07ApellidoNombre.pdf a las siguientes preguntas:

1. ¿Cuál es el tamaño de la señal original sin la primera muestra?//mirar cuanto bytes
2. ¿Cuántos bits por muestra utiliza la señal original?
3. ¿Cuántos bits por muestra como mucho necesitaríamos para almacenar los elementos de indx?
4. ¿Cuál sería la razón de compresión sin tener en cuenta el espacio ocupado por codebook, partition y predictor?

Respuestas:

1. el tamaño es 11496320 bytes
2. 11496320/1437040=8
3. el valor maximo es 3 por tanto como mucho necesitaremos 2bits
4. 11496320*8/11496320*2=4

Paso 9

Escribe la función **entropíatusiniciales** que dado un histograma calcule la entropía de una fuente. Si te gusta la que escribiste para el primer guión simplemente cópiala.

```
histogramaNZ=histograma;

histogramaNZ(find(histogramaNZ==0.0))=[];

suma=sum(histogramaNZ);

prob= histogramaNZ./suma;

resultado=-sum(prob.*log2(prob));
```

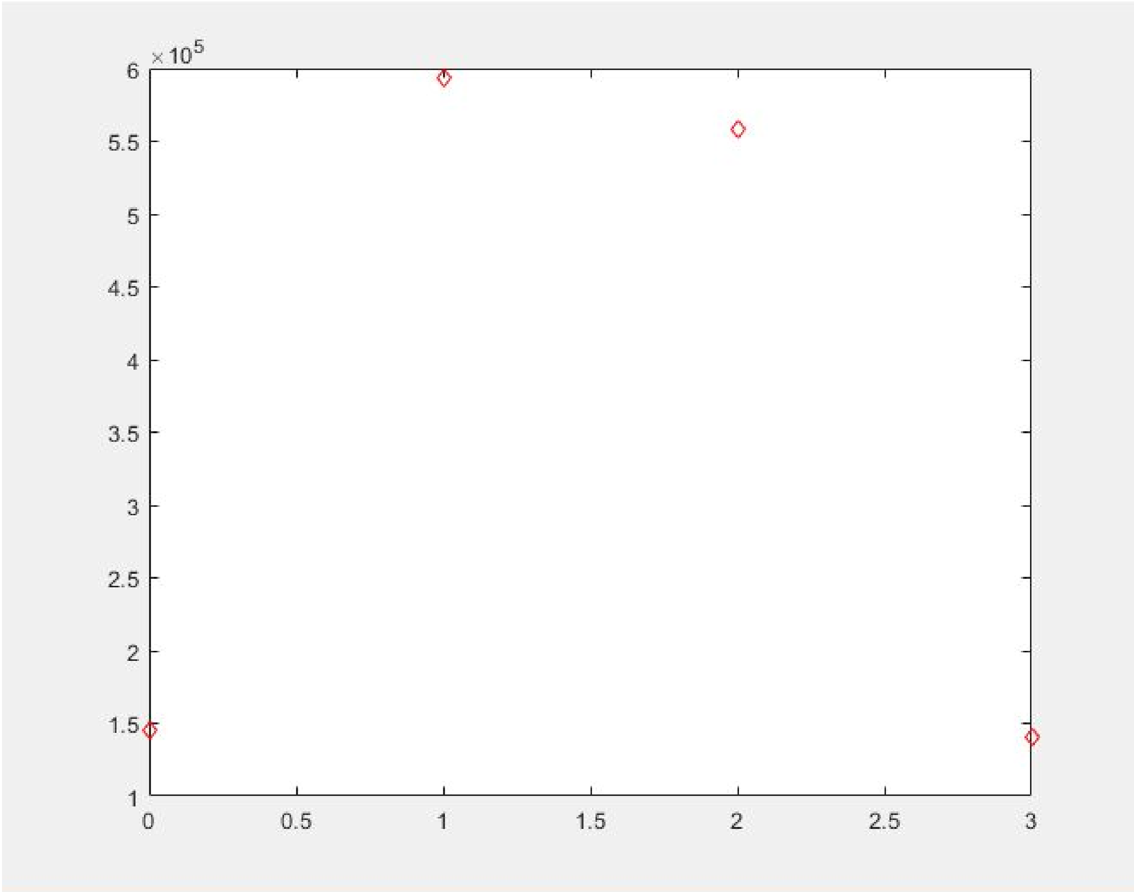
Paso 10

Incluye en el paso 10 de Practica07ApellidoNombre.m código en Matlab que calcule el histograma de indx, lo represente, calcule su entropía y el error cuadrático medio entre y e y2.

Paso 11

Incluye en el paso 11 de Practica07ApellidoNombre.pdf una gráfica del histograma que has calculado, la entropía y el error cuadrático medio. Discute, teniendo en cuenta el histograma y la entropía, formas de comprimir sin pérdida los elementos de indx.

Respuestas:



1. La entropía resulta 1.7189 y el error cuadrático medio es 6.7745e-04. Teniendo en cuenta los datos una buena solución de comprimirlo sin pérdida seria de longitud variable, ya que hay muchos valores en un rango. Dentro de estos algoritmos, la codificación aritmética, me resulta mejor que Huffman ya que hay muchos valores y el árbol resultante a codificar puede ser muy grande. Otro factor a tomar en cuenta es que la gráfica denota una gran dependencia de datos y tampoco sería mala idea aplicar antes las diferencias.

Paso 12

Incluye en el paso 12 de Practica07ApellidoNombre.pdf las respuestas a las siguientes preguntas:

1. ¿Qué ocurre si cambiamos los valores de codebook pero manteniéndolos en las mismas particiones, cambia indx, el ecm o ambos?. Razona la respuesta.
2. ¿Y si movemos las particiones de forma que codebook siga siendo válido?.

Respuestas

1. puede resultar que el error medio sea mayor debido a que el valor que obtiene codebook está situado donde la mayoría de los datos, si se cambia ese valor puede ser que esté en otro punto menos denso de datos.

2. Al igual que ocurría en el paso anterior, el error aumenta pues habrá datos que estén más lejos dentro del intervalo que el otro codebook.

   Siendo P1,P2,...,Pn puntos y K1,K2,..,Kn los puntos de los intervalos tal que ∀i∃j|Pi -Kj| sea mínimo

   Ahora con P1,P2,...Pn puntos y M1,M2,...Mn nuevos puntos de los intervalos tal que ∀i∃j|Pi -Mj| sea mínimo.

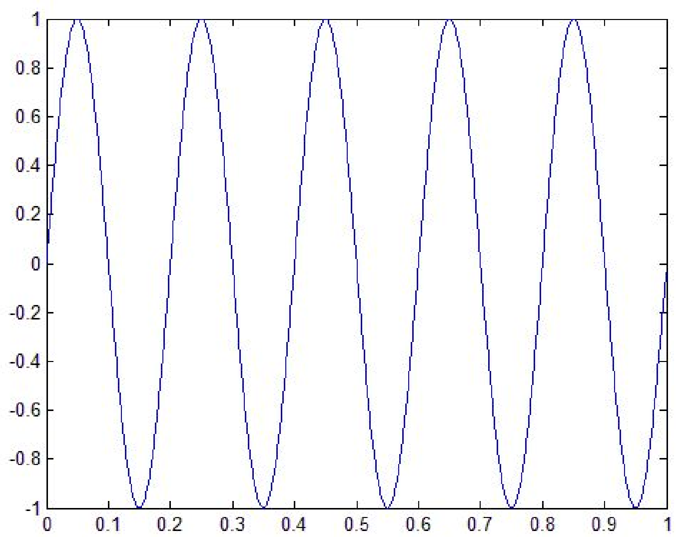
   entonces Kj == Mj o ∀i∃j|Pi -Mj|< ∀i∃j|Pi -Kj| y entonces ∀i∃j|Pi -Kj| no es mínimo o

   ∀i∃j|Pi -Mj|>∀i∃j|Pi -Kj| y por tanto ∀i∃j|Pi -Mj| no es mínimo. Por tanto al cambiar las particiones aumenta el error.

Paso 13

Considera la señal siguiente

```
close all; clear all;
x=[0:999]/1000;
y=sin(10*pi*x);
plot(x,y);
```



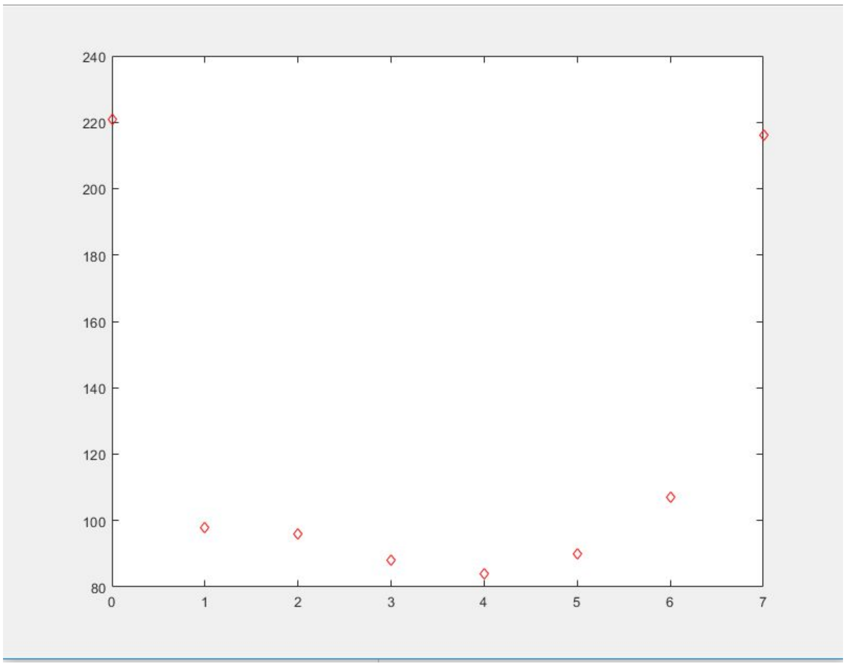
Paso 14

Incluye en el paso 14 de Practica07ApellidoNombre.m el código que realiza con esta señal los mismos pasos que con la señal de sonido usando una predicción de orden 1 con 8 particiones

Paso 15

Incluye en el paso 15 de Practica07ApellidoNombre.pdf una gráfica del nuevo histograma que has calculado, la entropía y el error cuadrático medio. Discute, teniendo en cuenta el histograma y la entropía, formas de comprimir sin pérdida los elementos de indx.

Respuestas:

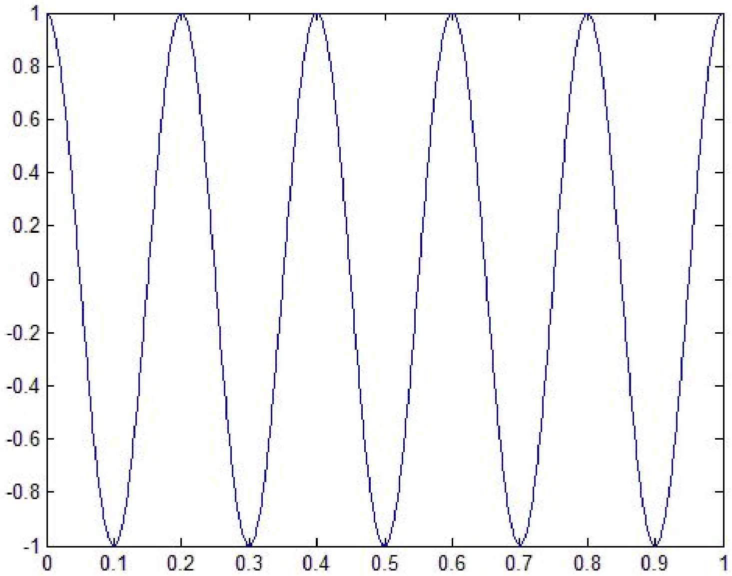


1. La entropía es 2.8782 y el error es de 1.89259e-04. Viendo el histograma al igual que en el anterior histograma el conjunto de datos aumentaba en el 0, en este son en el 1 y el -1, análogamente que pasaba con el anterior, una buena idea es codificarlo con longitud variable y la codificación aritmética resulta una buena solución. Aunque al aplicar las diferencias puede resultar no tan buena idea, pues los datos tienen pinta de ser uniformes salvo en valores cercanos al 0.

Paso 16

Considera ahora la señal

```
close all; clear all;
x=[0:999]/1000;
y=cos(10*pi*x);
plot(x,y);
```



Paso 17

Como ves la señal original no está próxima a cero en la primera observación. Si hacemos su predicción suponiendo que el valor y(0) es cero nos podemos equivocar bastante. ¿Cómo solucionarías este problema?. Incluye tu respuesta razonada en el paso 17 de Practica09ApellidoNombre.pdf

Respuestas:

1. La solución se basaría en aplicar una función y para convertirla en a la original aplicar la inversa de esa función. En este caso al tener el cos(x) su función inversa es la sec(x) que obtendría valores muy próximos al 0 y por tanto y(0) es cero y no hay tanto error. Para volver al original solo haría falta volver a aplicar la inversa que produce cos(x)

Paso 18

Escribe en el paso 18 de Practica07ApellidoNombre.m código en Matlab, tan general como sea posible, que calcule los valores de a y b que minimicen

$$\sum_{i=2}^{\mathrm{numel(y)}} (y(i) - a - by(i-1))^2$$

Muestra dichos valores. Obviamente no puedes usar dpcmopt.