

# Práctica 4

## COMPRESIÓN AUDIO MPEG 1 NIVEL III



**Fecha de Entrega : 13/11/2020**

# Índice

- Descripción
- Objetivo
- Parte A
- Parte B

## Descripción

La práctica trata del segundo tema de la asignatura, “voz y audio”.

Nos adentraremos en la compresión de audio mediante **MPEG**, mecanismo que aprovecha las ventajas de los modelos psicoacústicos para construir una gran tabla de consulta para transmitir componentes de frecuencia enmascarados con muchos menos bits

En este informe se reflejan los comentarios sobre el código llevado a cabo en los scripts de matlab y se adjuntan imágenes para comprender y analizar los resultados.

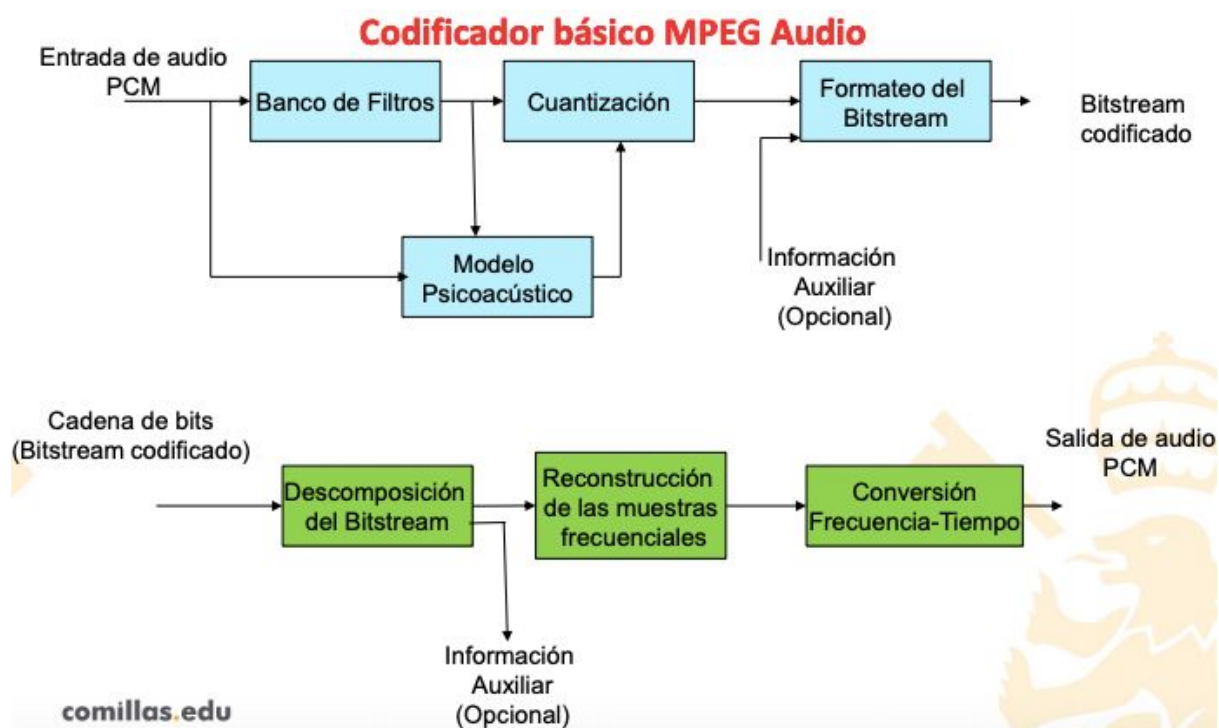


Fig I. Funcionamiento del codificador/decodificador MPEG Audio

## Objetivo

Analizar un fichero de audio (no voz) mediante Mpeg 1 nivel I y nivel III y entender cómo funcionan estos mecanismos de compresión.

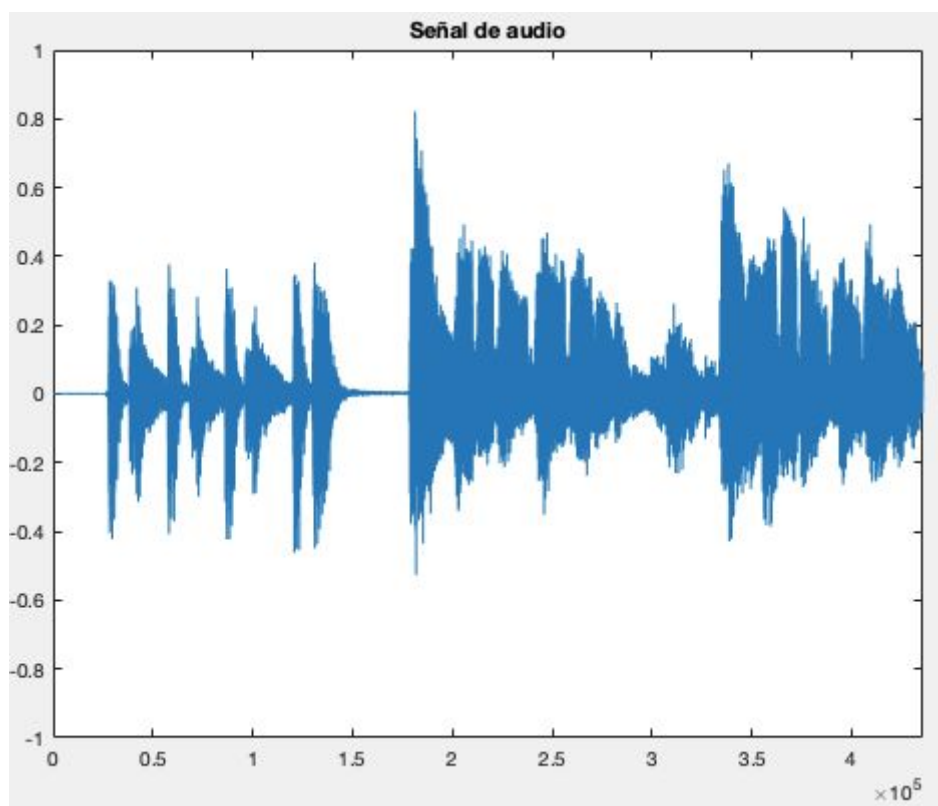
## Parte A

Esta primera parte de la práctica consiste en el análisis de un fichero de audio utilizando MPEG 1 audio en nivel 1.

El objetivo de cada nivel de MPEG es lograr un compromiso entre la calidad y velocidad de transmisión de bits. El **nivel 1** ofrece una calidad muy buena comparada con la alta velocidad de transmisión que facilita. Es el más simple y está pensado para bitrates superiores a 128 kbps.

- a) Elegimos fichero de audio (no voz)

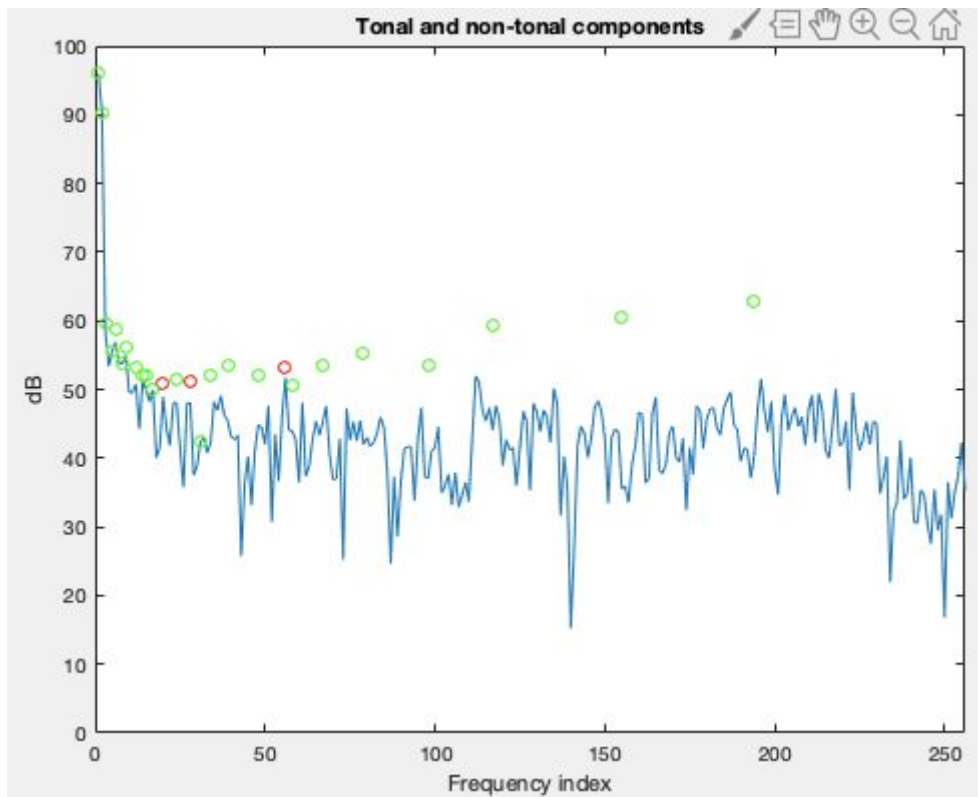
Hemos elegido el fichero '**Jazz.wav**'



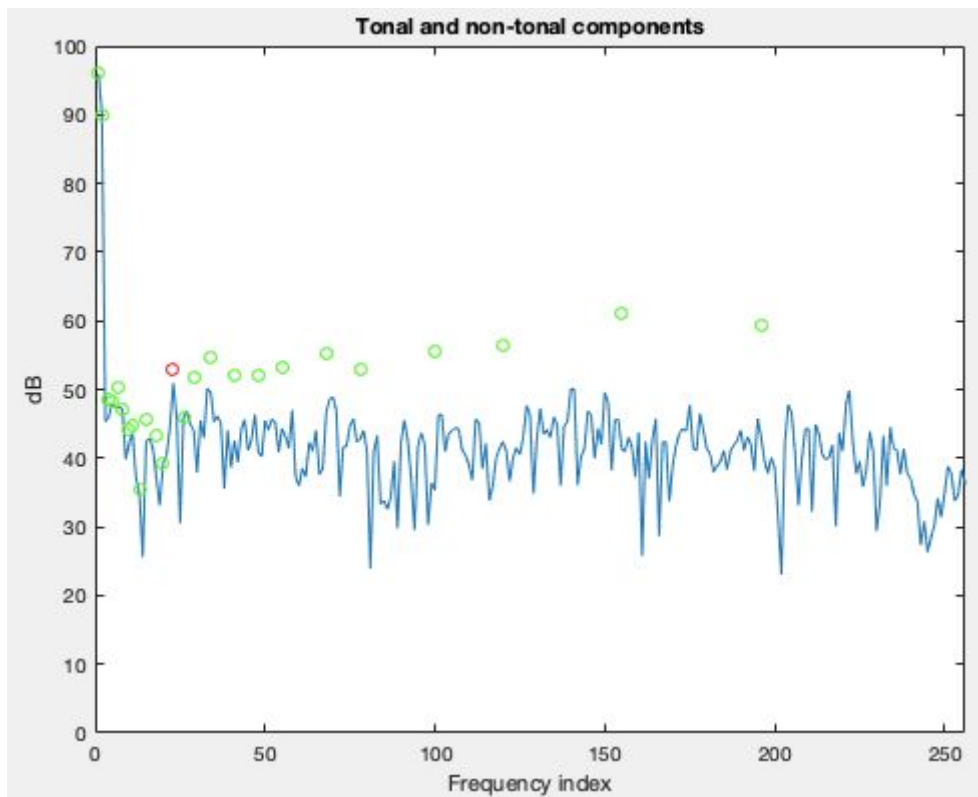
A continuación vamos a utilizar la función **Test\_MPEG.m** para realizar el análisis de dos fragmentos del fichero de audio seleccionado. Hemos escogido analizar el primer fragmento y el octavo.

- b) Encontramos las componentes tonales y no tonales de los dos pequeño fragmentos del fichero

### Fragmento #1

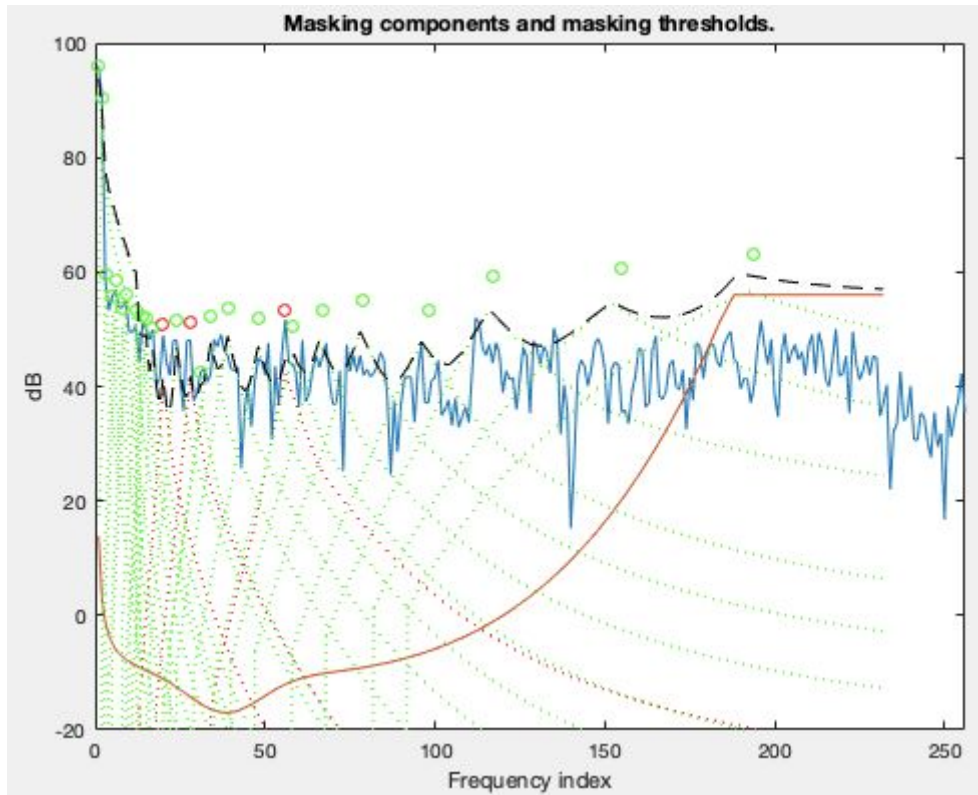


### Fragmento #8

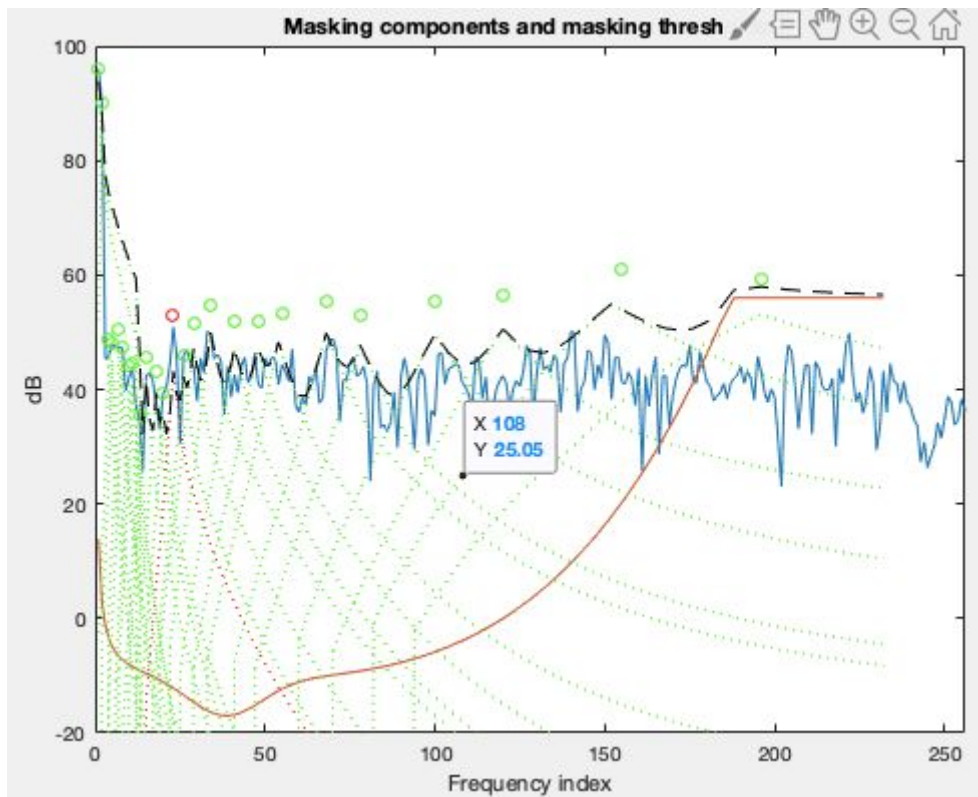


- c) Componentes de enmascaramiento y sus respectivos umbrales para cada fragmento

### Fragmento #1

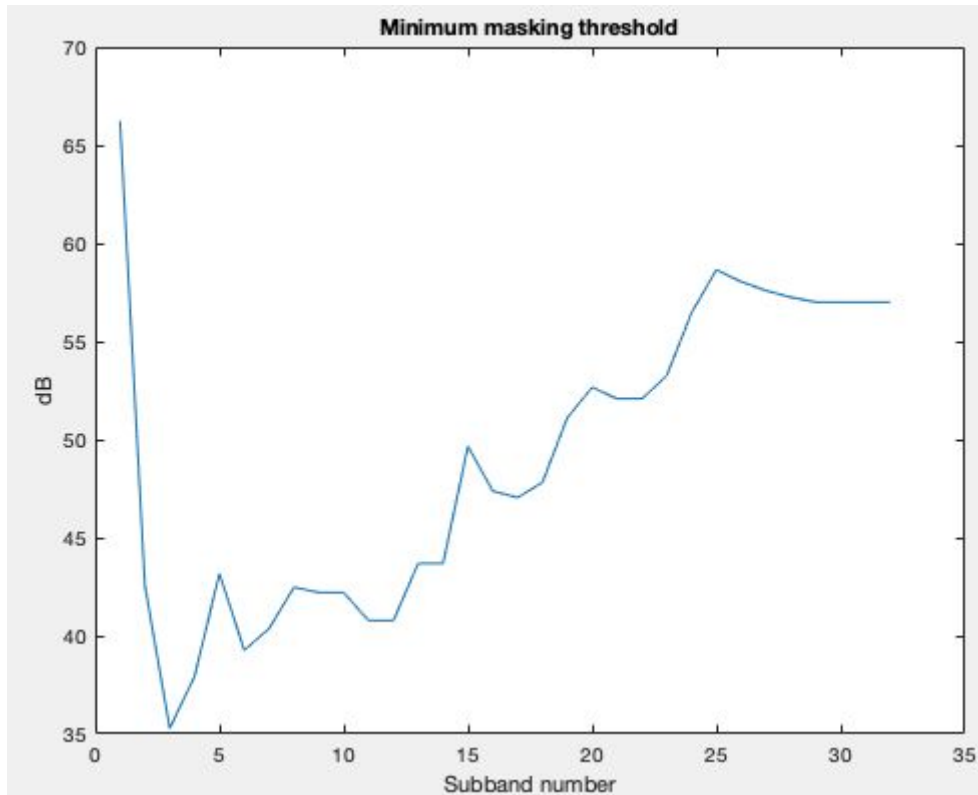


### Fragmento #8

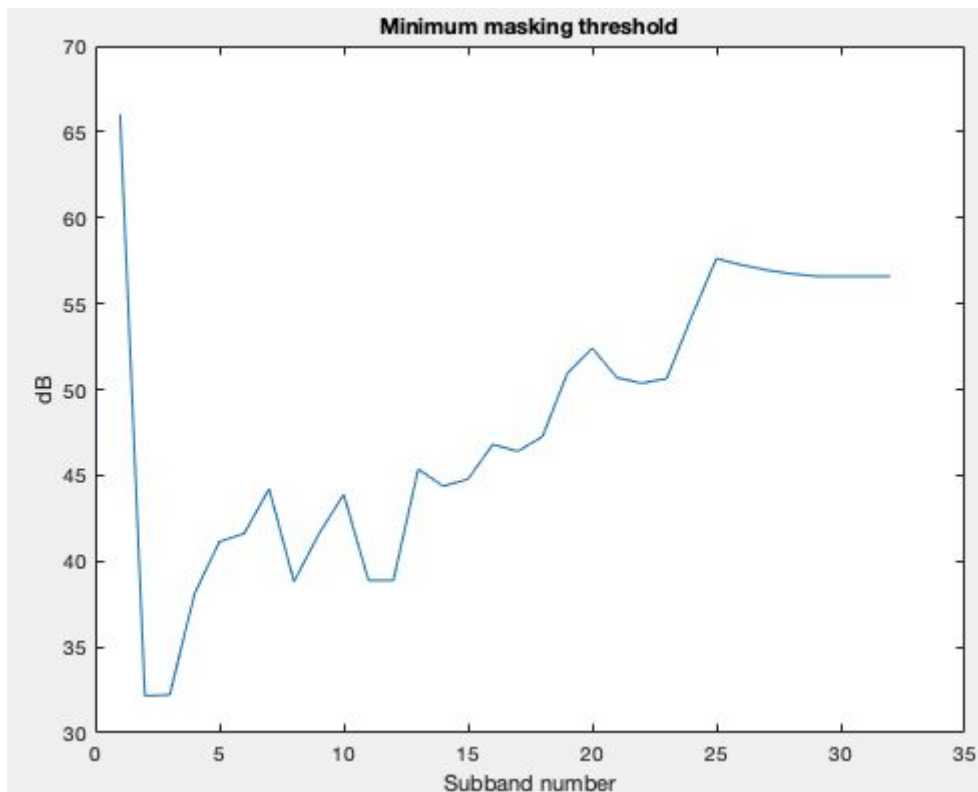


d) Umbral de enmascaramiento mínimo en cada sub-banda para cada fragmento

### Fragmento #1

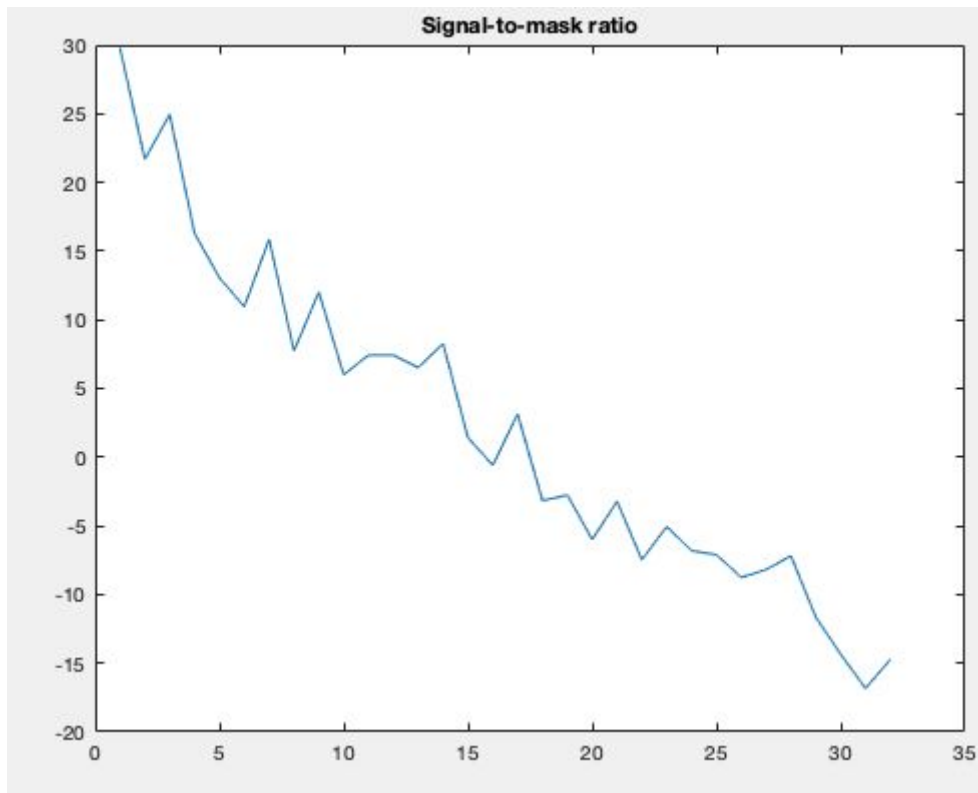


### Fragmento #8

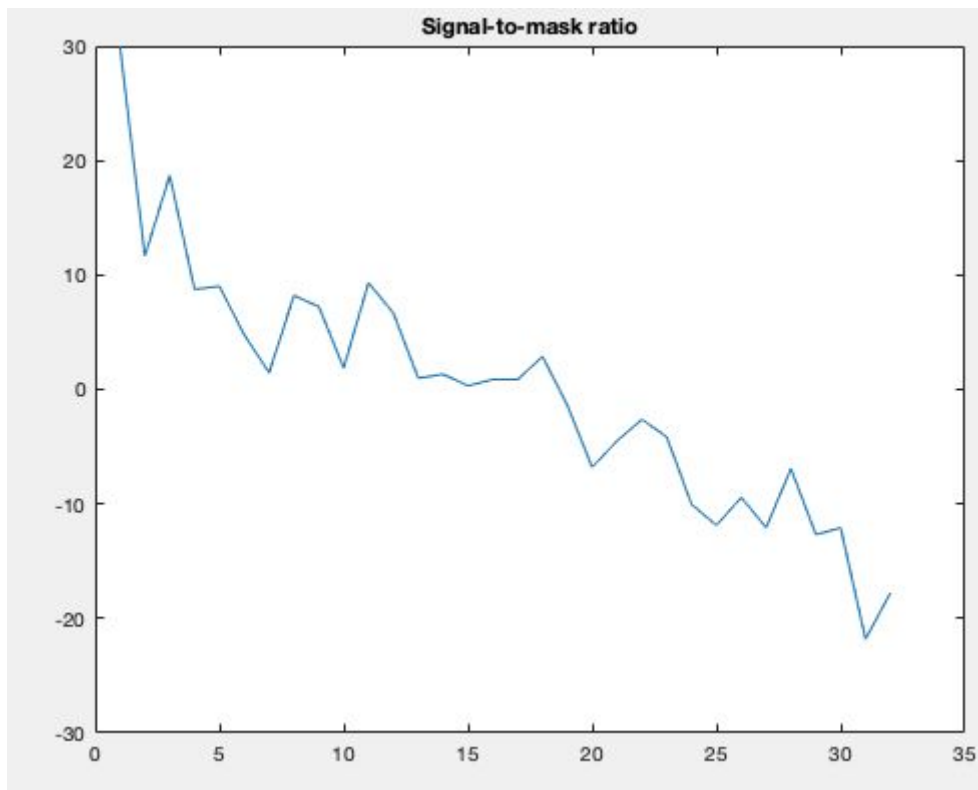


e) Ratio señal/máscara para cada fragmento

**Fragmento #1**



**Fragmento #8**





Comentemos las gráficas. Para empezar, hay que saber que el algoritmo de MPEG divide la señal de entrada en 32 subbandas de frecuencia con el banco de filtros.

A partir de ahí, cada frame de audio consta de 12 muestras por subbanda ya que estamos en el nivel I.

A continuación, separamos los valores espectrales de las bandas críticas de cada fragmento en componentes tonales (senoidales) y componentes no-tonales (ruido). Si hubiésemos escogido dos fragmentos consecutivos o cercanos, no habiéramos apreciado diferencias entre estas componentes.

En el siguiente paso, el modelo aplica una función de enmascaramiento a las señales en las diferentes bandas críticas para más tarde conseguir el umbral de enmascaramiento para cada subbanda.

Recordemos que el enmascaramiento de frecuencia usa un modelo psicoacústico para estimar el nivel de ruido audible. El codificador equilibra el comportamiento de enmascaramiento y el número de bits eliminando las frecuencias inaudibles y se escala la cuantización de acuerdo al nivel del sonido que queda sobre los niveles de enmascaramiento.

Por último, obtenemos el umbral mínimo para cada subbanda, utilizado para eliminar más componentes frecuenciales y poder transmitir menos datos, y el ratio señal/máscara (MNR) para cada sub-banda que representan la energía de la señal en la sub-banda dividida por el mínimo umbral de enmascaramiento en la sub-banda. Se obtienen 32 MNR que es la salida del modelo (una por subbanda).

## Parte B

La segunda parte de la práctica consiste en el análisis de un fichero de audio utilizando MPEG 1 audio en nivel 3.

**El nivel 3 (MP3)** es más complejo y originalmente tuvo como objetivo la transmisión de audio sobre líneas ISDN (Integrated Services Digital Network). Pensado para bitrates cercanos a 64 Kbps por canal

- a) Usar el fichero de audio anterior y el código para convertir el fichero wav a mp3 usando al menos dos velocidades (tasas de bits diferentes)

Hemos utilizado la función **Wav2mp3.m** para convertir nuestro fichero a formato mp3 utilizando 3 velocidades distintas (96, 160 y 256 kbps).

- b) Indicar el tiempo transcurrido en el proceso de paso a mp3 en ambas ocasiones

He usado tic+toc para calcular el tiempo de cada proceso.

```
% b) Indicar el tiempo transcurrido en el proceso de paso a mp3 en ambas
% ocasiones

% V=96bps --> t = 14.456788 seconds
% V=160 bps --> t = 18.865840 seconds
% V=256 bps --> t = 26.083397 seconds
```

- c) Comparar tamaños de fichero wav y los dos obtenidos al pasarlos por el compresor (ratios de compresión)

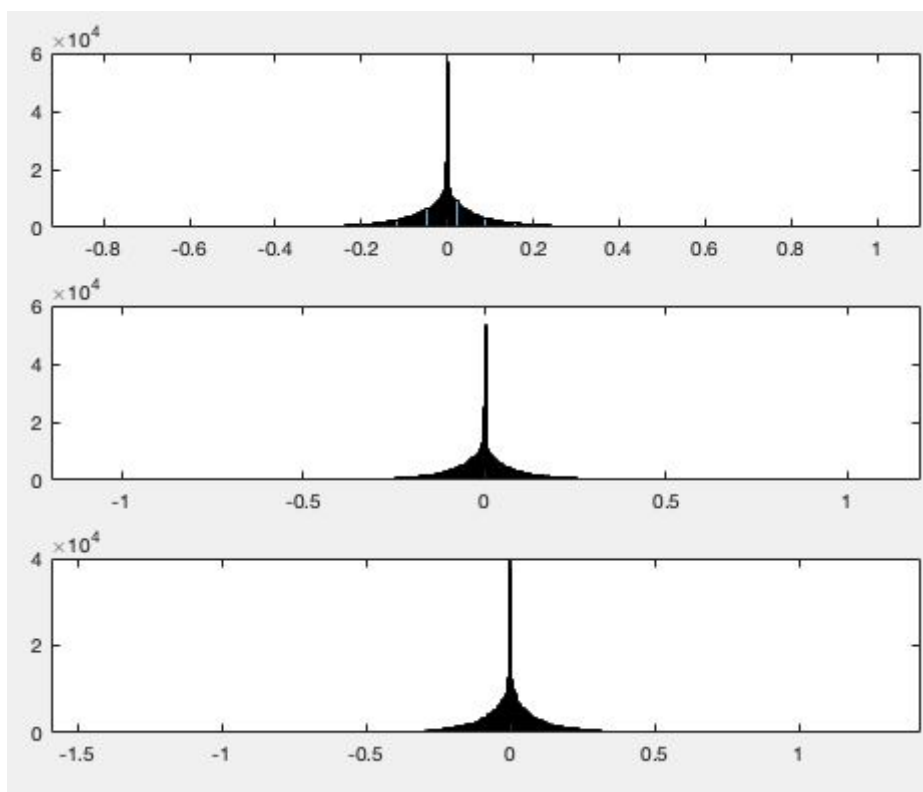
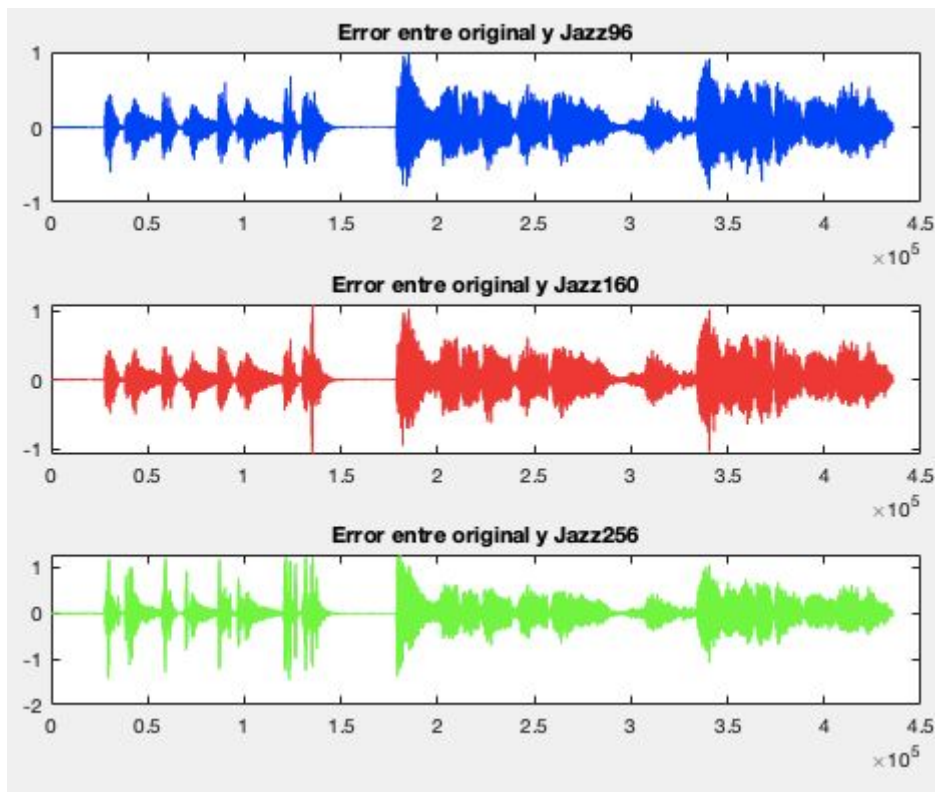
He comprobado el tamaño de cada fichero en Kb y he calculado los ratios. Como podemos ver cuanta mas velocidad se utilice en el proceso, mayo resulta el tamaño del archivo mp3, y por lo tanto el ratio es menor.

```
Tam_wav = 871;
Tam_Jazz96 = 118;
Tam_Jazz160 = 197;
Tam_Jazz256 = 316;

Cr96 = Tam_wav / Tam_Jazz96;
Cr160 = Tam_wav / Tam_Jazz160;
Cr256 = Tam_wav / Tam_Jazz256;
```

 Cr160	4.4213
 Cr256	2.7563
 Cr96	7.3814

- d) Comentar la calidad obtenida en todos los casos. Evaluar diferencia o errores de manera cuantitativa



Podemos ver como el error es mayor cuando la velocidad también lo es. Así, el fichero Jazz256.mp3 tiene mayor media y mayor desviación típica que los otros dos.

```
>> mean(error1)
      mean(error2)
      mean(error3)
```

```
ans =
```

```
4.0823e-04
```

```
ans =
```

```
3.2094e-04
```

```
ans =
```

```
2.9788e-04
```

```
>> std(error1)
      std(error2)
      std(error3)
```

```
ans =
```

```
0.1112
```

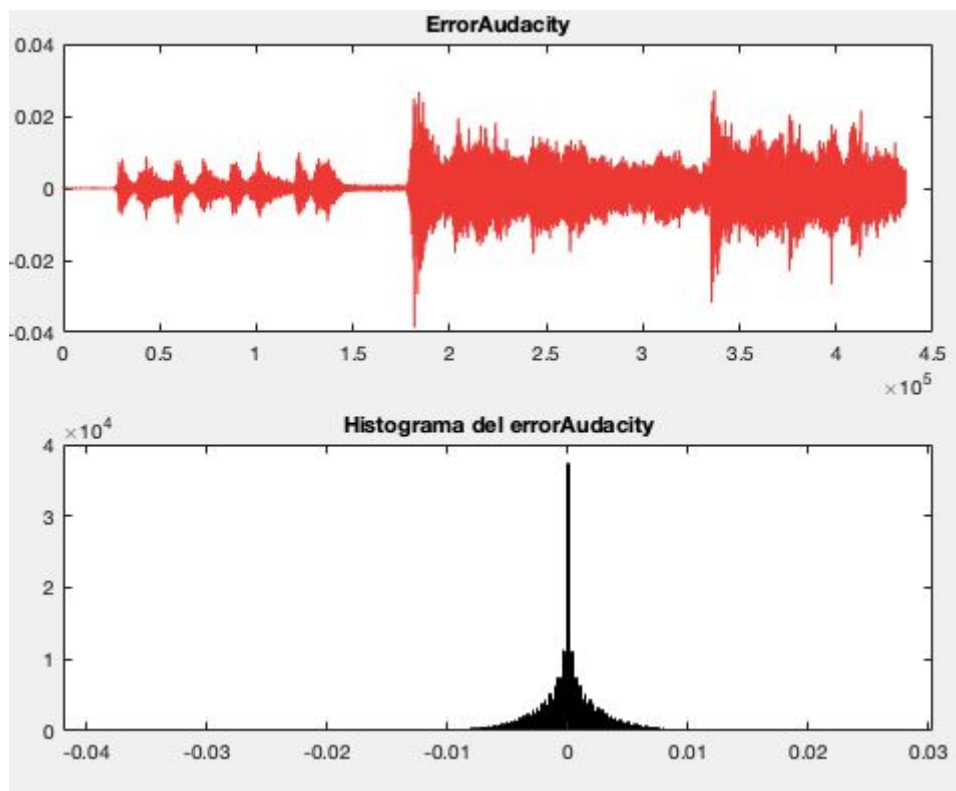
```
ans =
```

```
0.1190
```

```
ans =
```

```
0.1560
```

- e) Usar el fichero de audio en formato wav para cargarlo en 'Audacity' y desde esa aplicación exportarlo con formato mp3. Evaluar tamaños origen y final y su ratio de compresión. Cargar ambos ficheros en Matlab, tanto el original wav, como el exportado en mp3, y analizar sus diferencia o error de manera cuantitativa y de forma gráfica con gráficos e histograma.



```
>> mean(errorAu)
```

```
ans =
```

```
1.3781e-05
```


```
>> std(errorAu)
```

```
ans =
```

```
0.0030
```

- f) Comentar la calidad obtenida respecto a los casos analizados antes, Evaluar las diferencias encontradas en ambos método de paso a mp3.

El método Audacity es mucho mejor que el primero, ya que el fichero se comprime mucho más (114Kb) y el error tiene menor media y menor desviación típica.

 Jazz.mp3	ayer 17:54	316 KB
 Jazz.wav	28 may 2001 16:43	871 KB
 Jazz96.mp3	ayer 16:00	118 KB
 Jazz160.mp3	ayer 15:59	197 KB
 Jazz256.mp3	ayer 16:01	316 KB
 JazzAudacity.mp3	hoy 13:21	114 KB