eman ta zabal zazu

UPV/EHU, Facultad de Informática

Dpto. de Arquitectura y Tecnología de Computadores

PROCESADO DIGITAL DE SEÑAL

PROYECTO ESPECÍFICO - LABORATORIO 9 (Mue)

MUESTREO Y CUANTIFICACIÓN.

Componentes del grupo:
- Alex Beltrán
- Daniel Cañadillas
- Ainhoa Serrano

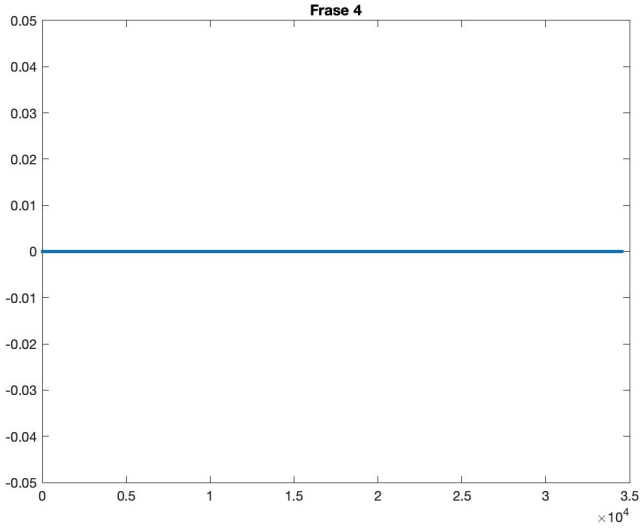
Nota: Enviar este documento "Lab9_Mue_resultados.doc" completado con las tareas solicitadas, el código generado y los comentarios y aclaraciones que consideréis oportunos, junto con los correspondientes ficheros .m, en un fichero .zip vía eGela.

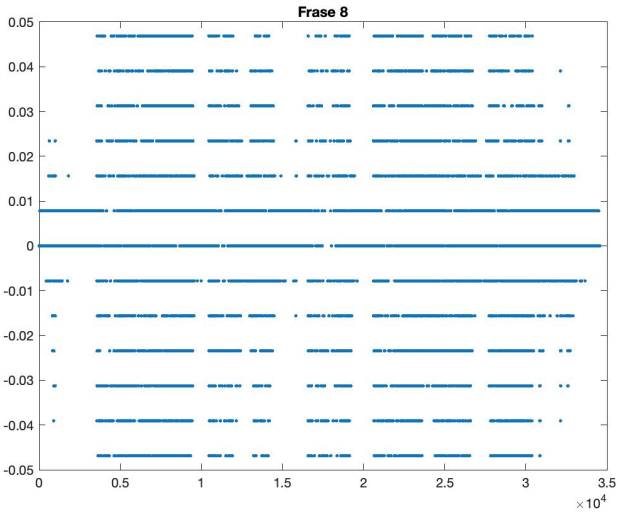
CUANTIFICACIÓN UNIFORME

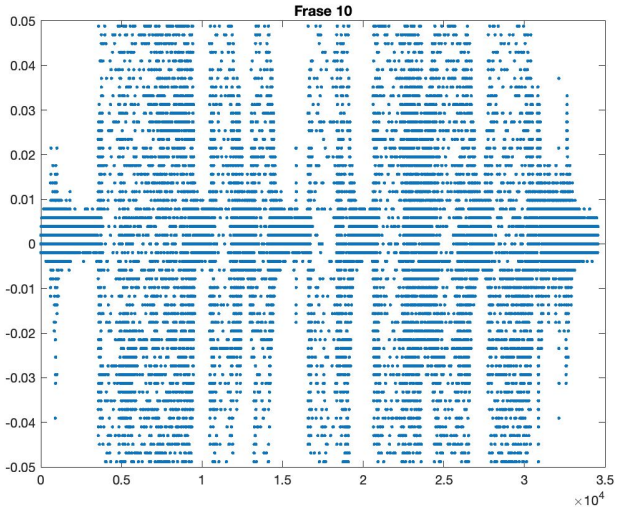
Ejercicio 2 (P9_1_cuan_uniforme.m)

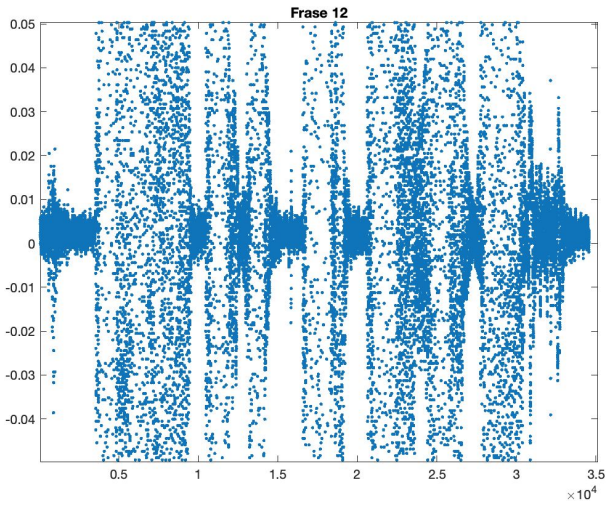
- Calcula a partir de la señal contenida en frase12, otras 3 señales que utilicen 10, 8 y 4 bits respectivamente en la cuantificación. Visualiza las tres señales generadas y compáralas con la original a 12 bits. Podéis usar el zoom utilizado en el ejemplo anterior (-0.05 – 0.05) para ver las diferencias. Comentar los resultados.

```
frase10=round(frase12*512)/512; %2^10/2=512
frase8=round(frase12*128)/128;  %2^8 /2=128
frase4=round(frase12*8)/8;      %2^4/2=8

%Escuchar las señales
soundsc(frase12,fs);
soundsc(frase10,fs);
soundsc(frase8,fs);
soundsc(frase4,fs);
```







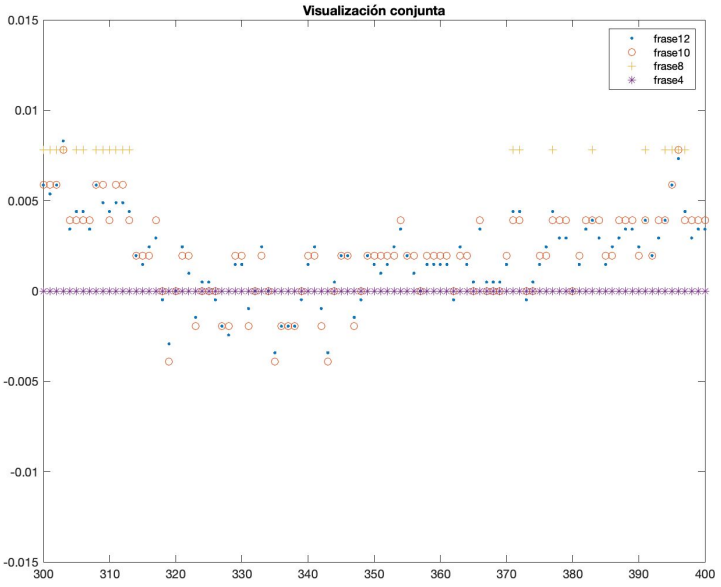


Como se puede ver, a medida que se reducen los bits con los que se representa la señal, también se reduce su representación en su forma gráfica. La gráfica de la señal de 12 bits tiene muchos más puntos que la de 4 bits. Esto en la reproducción del sonido se representará como una adición de ruido de fondo.

- Escúchalas y trata de percibir las diferencias. ¿Suenan igual? Visualiza el segmento comprendido entre 300 y 400 muestras de las cuatro señales en una misma figura (`hold on`) con diferentes símbolos para ver la diferencia de precisión de cada opción. Utiliza una extensión para el eje vertical de 0.015 y 0.015 con la opción `axis`. Explicar lo que se ve en la figura.

Al escuchar las señales, se puede percibir un aumento de ruido conforme se van reduciendo el número de bits usados para representar la señal. Entre las señales 10 y 12 no se nota demasiada diferencia, pero las de 4 y 8 tienen un notable ruido. Por tanto la cuantificación crea un error notable.

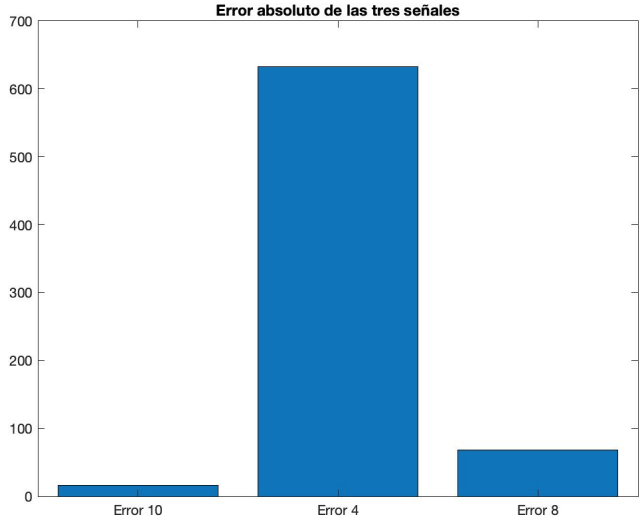
```
%Visualiza el segmento comprendido entre 300 y 400
figure, plot(frase12,'.');
hold on;
plot(frase10,'o');
plot(frase8,'+');
plot(frase4,'*');
hold off;
title("Visualización conjunta"); axis([300 400 -0.015 0.015]);
legend('frase12','frase10', 'frase8', 'frase4')
```



Como se puede ver, los puntos de la frase12 tiene representacion en varios valores del eje y. Cuanto menos bits, esa representación baja, y al final se puede comprobar que los valores de la frase4 solo se encuentran en el valor 0 de y.

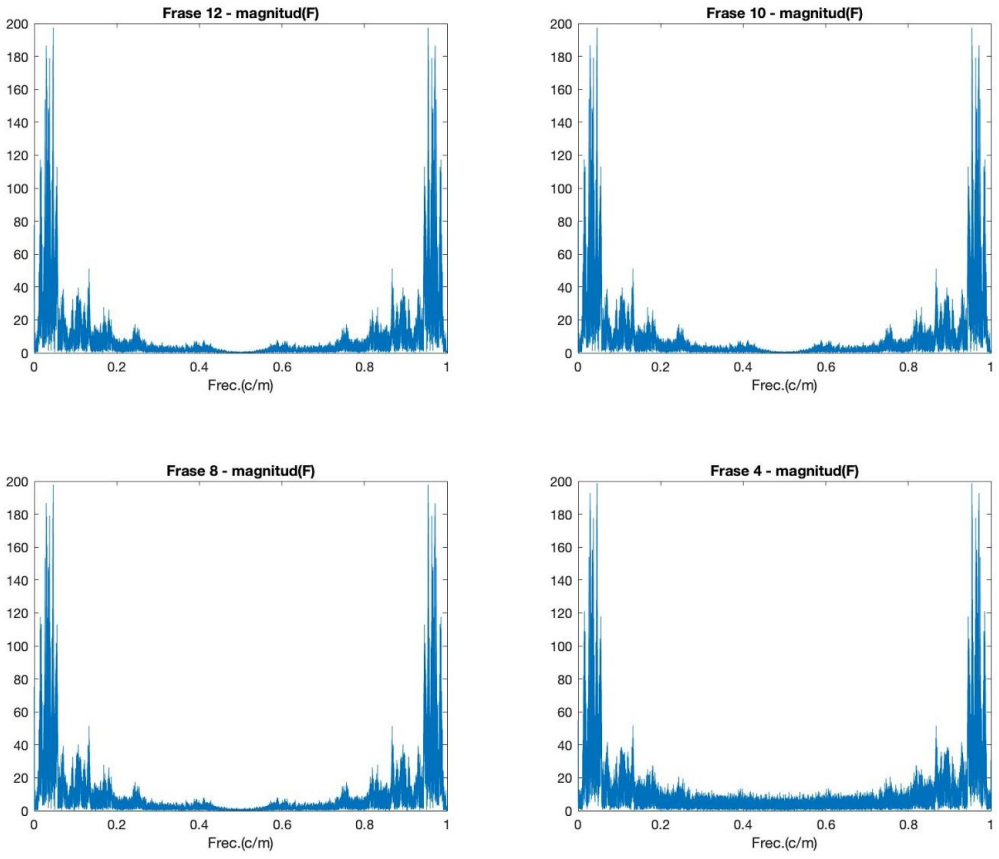
- Calcula el error absoluto de las tres señales (10, 8 y 4 bits) respecto de la señal original en el tiempo. Genera un gráfico de barras (`bar(…)`) con los tres errores.

```
%Cálculo de errores absolutos
error10=sum(abs(frase12-frase10));  %error10=16.5103
error8=sum(abs(frase12-frase8));    %error8=67.77
error4=sum(abs(frase12-frase4));    %error4=632.8140

%Grafico de barras - error absoluto de las tres señales (error por cuantificación)
c = categorical({'Error 10','Error 8','Error 4'});
bar(c, [error10, error8, error4]); title("Error absoluto de las tres señales");
```



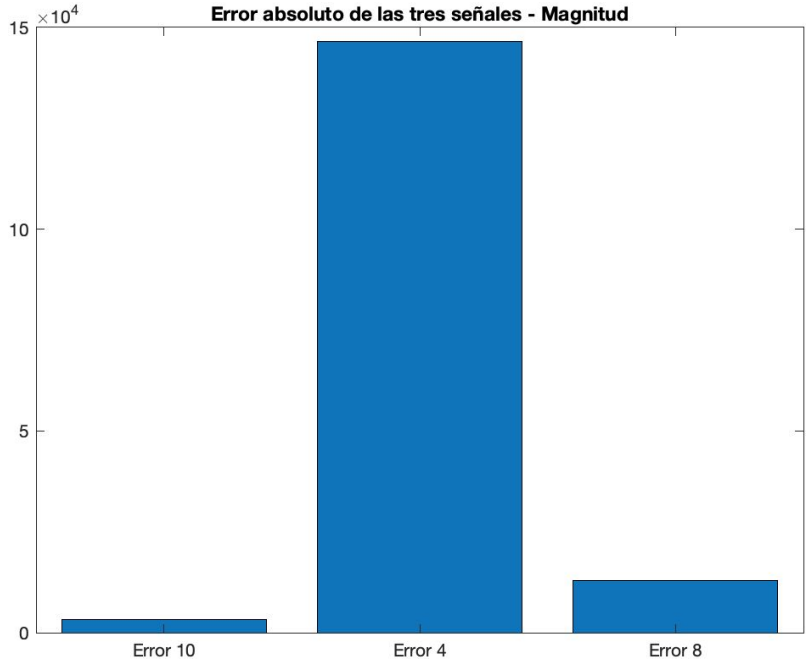
- Visualiza en una figura con 4 subplots (`subplot(2,2,1-4)`), los espectros de magnitud de las cuatro señales.

```
%Visualización de los espectros de magnitud
F=(0:34999)/35000;
f12=fft(frase12,35000);
f10=fft(frase10,35000);
f8=fft(frase8,35000);
f4=fft(frase4,35000);
figure,
subplot(2,2,1), plot(F,abs(f12)), title('Frase 12 - magnitud(F)'), xlabel('Frec.(c/m)');
subplot(2,2,2), plot(F,abs(f10)), title('Frase 10 - magnitud(F)'), xlabel('Frec.(c/m)');
subplot(2,2,3), plot(F,abs(f8)), title('Frase 8 - magnitud(F)'), xlabel('Frec.(c/m)');
subplot(2,2,4), plot(F,abs(f4)), title('Frase 4 - magnitud(F)'), xlabel('Frec.(c/m)');
```



- Calcula el error absoluto de los espectros de las tres señales (10, 8 y 4 bits) respecto de la señal original. Genera un gráfico de barras (`bar(…)`) con los tres errores.

```
%Grafico de barras - error absoluto de los espectros de las tres señales (error por cuantificación)
error10m=sum(abs(f12-f10));  %error10m=3.336e+03
error8m=sum(abs(f12-f8));    %error8m=1.3e+04
error4m=sum(abs(f12-f4));    %error4m=1.464e+05

c = categorical({'Error 4','Error 8','Error 10'});
bar(c, [error4m, error8m, error10m]); title("Error absoluto de las tres señales - Magnitud");
```

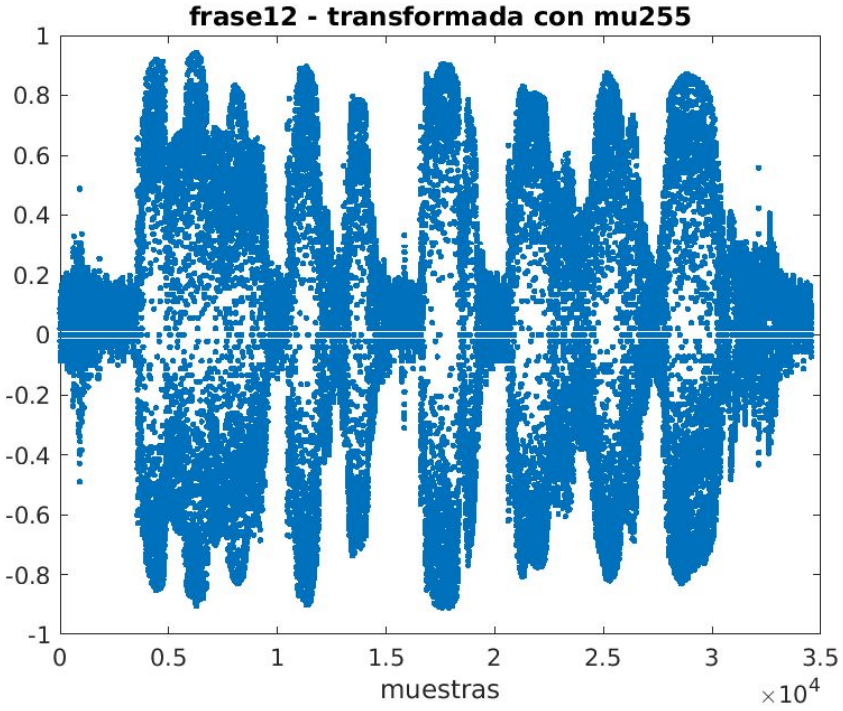


CUANTIFICACIÓN NO UNIFORME – Ley μ255

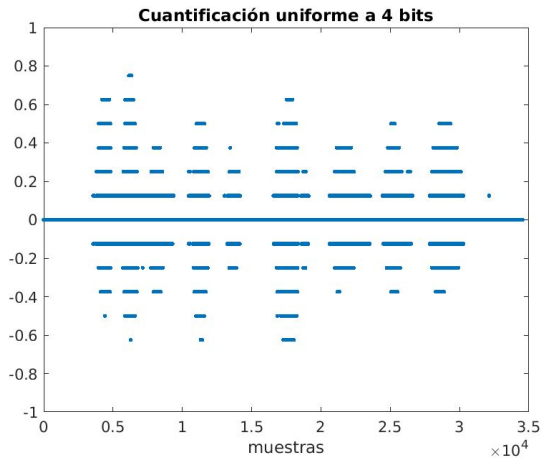
Ejercicio 3 (P9_2_cuan_no_uniforme.m)

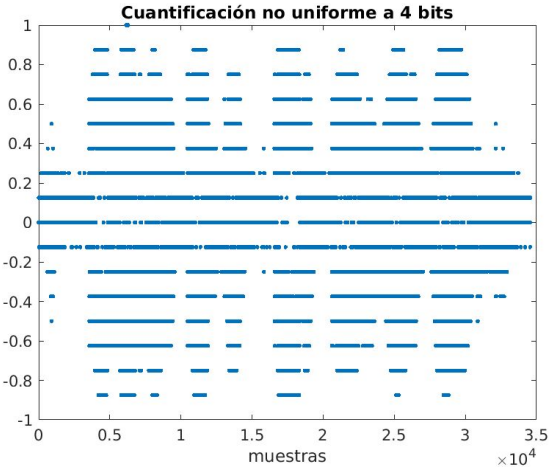
```
load('frase12.mat');
figure, plot(frase12,'.');
axis([0 35000 -1 1]);
title('frase12'),xlabel('muestras');
frase = mu255(frase12);
figure, plot(frase,'.');
title('frase12 - transformada con mu255'),xlabel('muestras');
axis([0 35000 -1 1]);


soundsc(frase12,fs);

% Cuantificación no uniforme a 4 bits
frase04cuan = round(frase*8)/8; %2^4 / 2 = 8 -> 4 bits
soundsc(frase04cuan,fs);
figure,
plot(frase04cuan,'.');
title('Cuantificación no uniforme a 4 bits'), xlabel('muestras');
axis([0 35000 -1 1]);


% Cuantificación a 4 bits
frase12_4_cuan = round(frase12*8)/8;
soundsc(frase12_4_cuan,fs);
figure,
plot(frase12_4_cuan,'.');
title('Cuantificación uniforme a 4 bits'), xlabel('muestras');
axis([0 35000 -1 1]);

% Inversa mu255 de la cuantificación a la señal cuantificada no uniformemente
inv = mu255inv(frase04cuan);
soundsc(inv,fs);
figure,
plot(inv,'.');
title('Inversa de cuantificación no uniforme a 4 bits'), xlabel('muestras');
axis([0 35000 -1 1]);
```

- Aplica la función mu255 a la señal frase12. Visualiza el resultado.



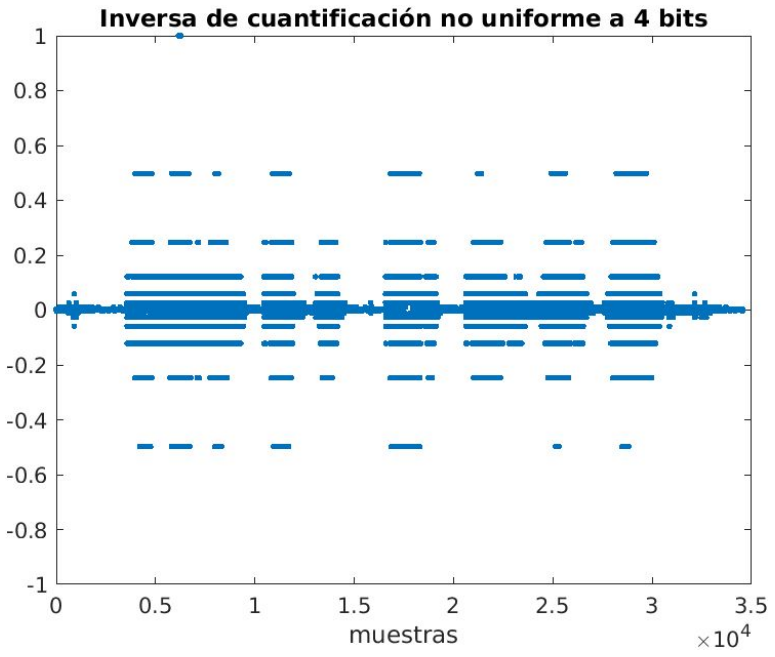
- A continuación, cuantifica a 4 bits la señal original (cuantificación uniforme) y la señal cuantificada mediante mu255 (cuantificación no uniforme) y visualiza el resultado. Compara las señales con cuantificación uniforme y no uniforme, verás que el efecto de la ley mu es ampliar el detalle de las amplitudes bajas.





Podemos apreciar como se pierde mucha más información en la cuantificación uniforme a 4 bits que la no uniforme, esto es debido al realce que hace la cuantificación no uniforme de los valores cercanos al cero.

- Ahora aplica la función inversa a la señal cuantificada no uniformemente y escúchala, comparándola con la original y con la obtenida con cuantificación uniforme de 4 bits. Compara resultados.

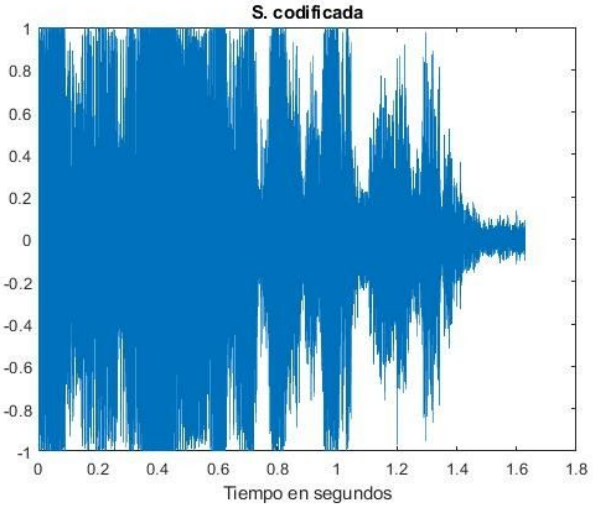


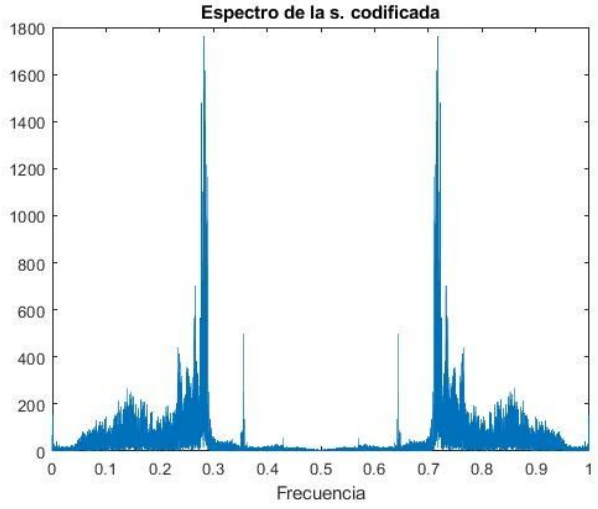
Podemos ver como comparando la inversa con las demás, debido a la aplicación de la inversa ahora los valores cercanos al cero se ven amplificados

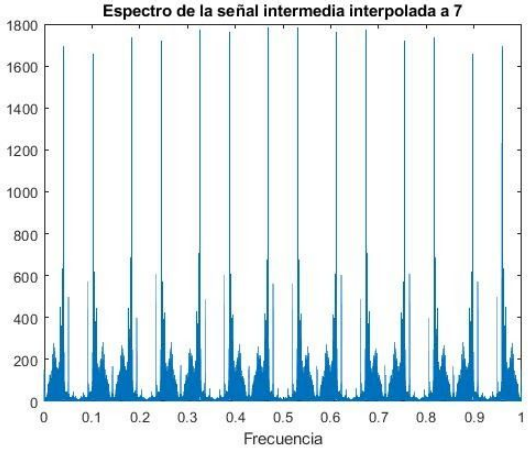
CAMBIOS EN LA FRECUENCIA DE MUESTREO.

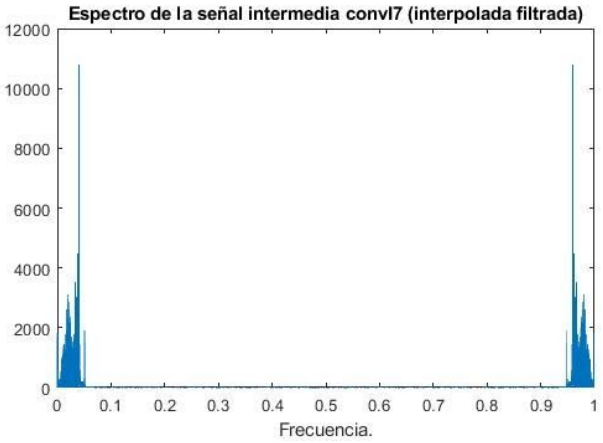
Ejercicio 4 – Canal+ (P9_3_canalplus.m)

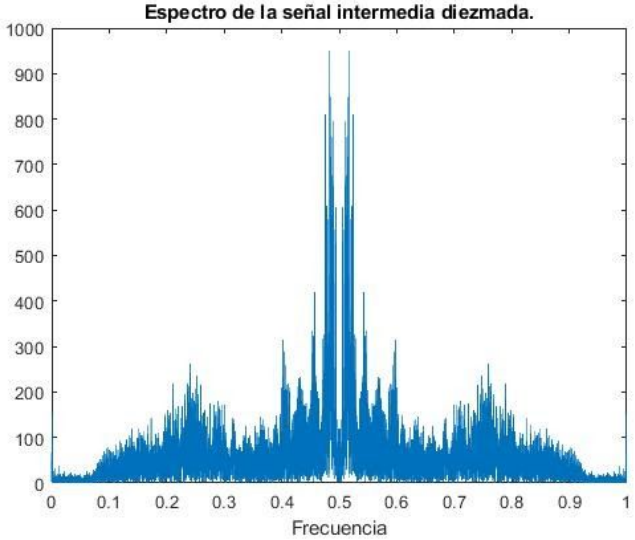
El ejercicio que se propone consiste en escribir un programa MATLAB (`P9_3_canalplus.m`) que modifique la velocidad de muestreo de la señal del fichero **cplus44.wav** aproximándola lo más posible a 25.6kHz y luego invierta el espectro cambiando de signo las muestras impares. La relación entre las dos frecuencias es aproximadamente 7/12, es decir, interpolar por 7 y diezmar por 12. Aplicarlo a la señal cplus44.wav. Dibujar el espectro de la señal original codificada, y de las señales que se van generando en cada paso indicado en el documento guion del proyecto: señal interpolada, interpolada filtrada, diezmada, descodificada (al cambiar de signo), y la señal final descodificada a 44100Hz

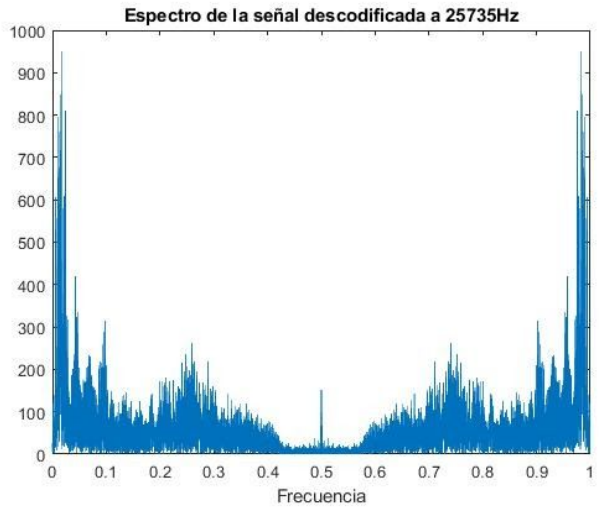


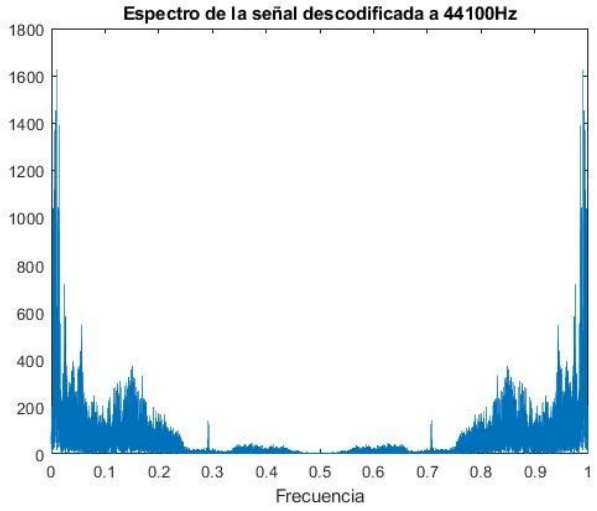













```
clear all, close all, clc;
K = 7;  % factor de interpolado
L = 12;  % factor de diezmado
Nfir = 50;   % orden del filtro FIR (par)

%% 1:
% Leer del fichero cplus44.wav el sonido a descodificar y escucharlo.
% Dibujar la señal en el tiempo y el espectro de la misma.

[x fs] = audioread("cplus44.wav");
n = length(x); t = (1:n)/fs;
X = fft(x); Frec = (0:n-1)/n;

% Plot:
figure; plot(t,x); title("S. codificada"); xlabel("Tiempo en segundos");
% soundsc(x,fs);  % Descomentar para escuchar el sonido.

% Plot:
figure, plot(Frec,abs(X)); title("Espectro de la s. codificada"); xlabel("Frecuencia");

%% 2:
% Diseñar un filtro IIR paso bajo con frecuencia de corte 1/12 -> min(0.5/7,0.5/12)*2
h = fir1(Nfir, min(0.5/K,0.5/L)*2);

%% 3:
% Cambiar la velocidad de muestreo de la señal (interpolar por 7 y diezmar por 12 utilizando el filtro anterior).
% Cuidado, hay que compensar el desplazamiento que introduce el filtro fir1 (mitad del orden).

interpol7(1:K:K*length(x)) = x; % interpolar por 7

n = length(interpol7); fftInt7 = fft(interpol7); Frec = (0:n-1)/n;

% Plot:
figure, plot(Frec, abs(fftInt7)),
title("Espectro de la señal intermedia interpolada a 7"),
xlabel("Frecuencia");

convi7 = conv(h, interpol7)*L;
n = length(convi7); fftConvI7 = fft(convi7); Frec = (0:n-1)/n;

% Plot:
figure, plot(Frec, abs(fftConvI7)),
title("Espectro de la señal intermedia convI7 (interpolada filtrada)"),
xlabel("Frecuencia.");

% compensar el desplazamiento introducido por el filtro firl (mitad del orden)
% 50 -> 25 --> >25
convi7 = convi7(26:end);
% diezmar por 12
convi7diezmada = convi7(1:L:end);
```

```
n = length(convi7diezmada);
fftconvi7diezmada = fft(convi7diezmada); Frec = (0:n-1)/n;

% Plot:
figure, plot(Frec, abs(fftconvi7diezmada)),
title("Espectro de la señal intermedia diezmada."),
xlabel("Frecuencia");

%% 4:
% Cambiar el signo de las muestras impares para invertir el espectro.

convi7diezmada(1:2:end) = -convi7diezmada(1:2:end);

fftconvi7diezmada = fft(convi7diezmada);
n = length(fftconvi7diezmada); Frec = (0:n-1)/n;

% Plot.
figure, plot(Frec, abs(fftconvi7diezmada)),
title("Espectro de la señal descodificada a 25735Hz"),
xlabel("Frecuencia");

%% 5:
% Escuchar el resultado (con velocidad de muestreo 25725 Hz).
soundsc(convi7diezmada, 25725);

%% 6:
% Aumentar la frecuencia de la señal a 44100 Hz.
% Es el proceso simétrico al realizado en los pasos anteriores.

auxInterpol(1:L:L*length(convi7diezmada)) = convi7diezmada;
auxInterpol = conv(h, auxInterpol)*L;
auxInterpol = auxInterpol(26:end);
sfinal = auxInterpol(1:K:end); % diezmar por 7
fftFinal = fft(sfinal); n = length(fftFinal); Frec = (0:n-1)/n;

% Plot:
figure, plot(Frec, abs(fftFinal)),
title("Espectro de la señal descodificada a 44100Hz"),
xlabel("Frecuencia");

% Resultado:
soundsc(sfinal, fs);
```