Table of Contents

Practica 7

Miguel Oleo y Teresa González

```
close all
clc
clear
```

Introducción

El objetivo de esta práctica es aplicar los conocimientos dados en teoría de la DFT y el filtrado de bloques por streaming usando la misma. Inicialmente diseñamos un filtro FIR causal y estable y posteriormente filtramos una señal de audio dada utilizando las DFT mediante el método de overlap-save.

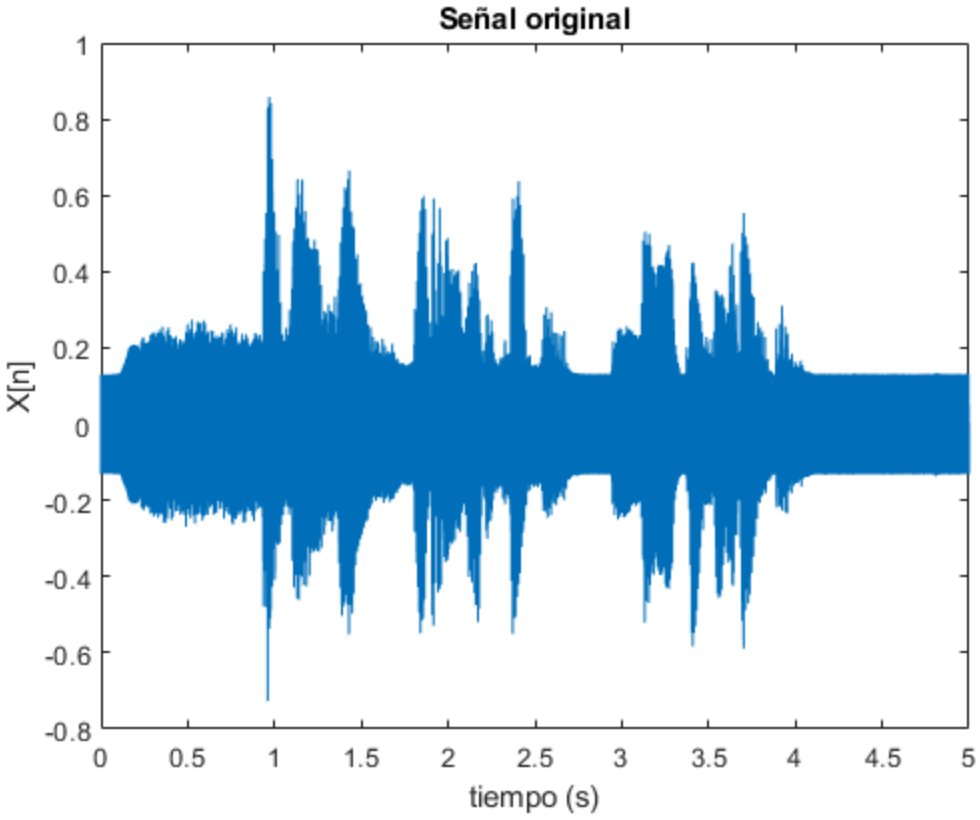
Diseño del un filtro FIR

Apartado a)

Cargamos el filtro, escuchamos la señal y ploteamos en el tiempo dicha señal. Es importante destacar que se escucha un tono aguado bastante desagradable.

```
load('PDS_P7_LE2_G4');
[Xn, fs] = audioread('PDS_P7_LE2_G4.wav');

dt = 1/fs;
t= 0:dt:((length(Xn)-1)/fs);

figure()
plot(t,Xn)
xlim([0 5])
```

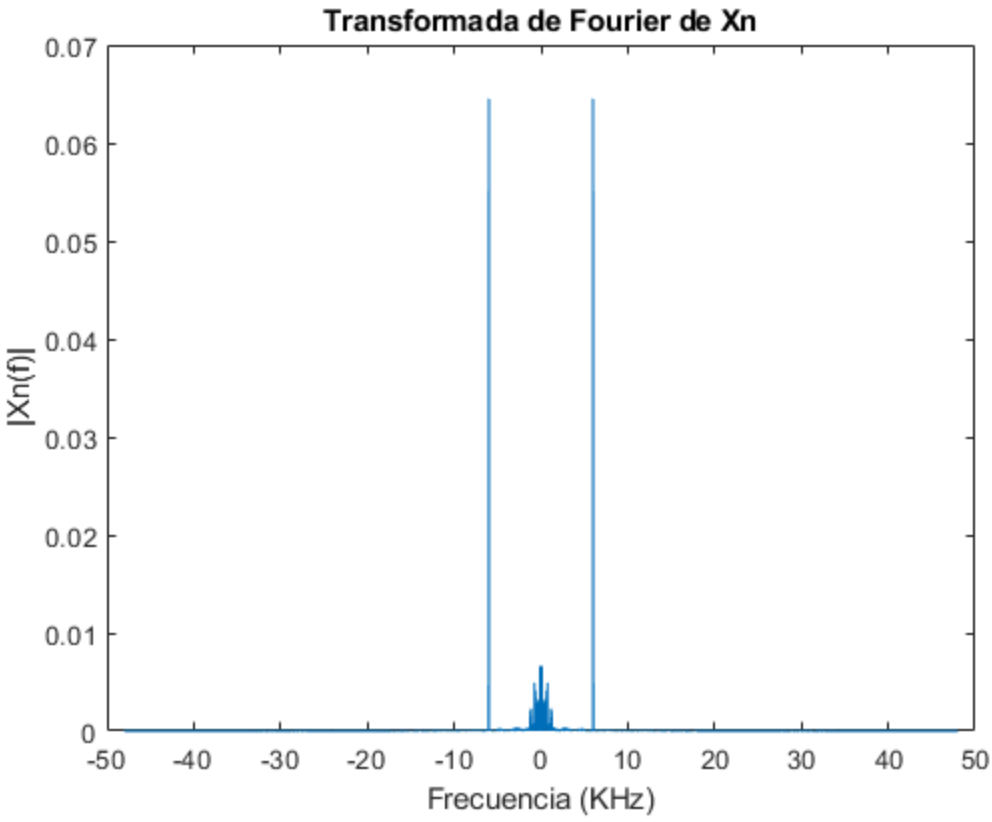
```
xlabel('tiempo (s)')
ylabel('X[n]')
title('Señal original')


sound(Xn,fs);
```



Apartado b)

Para ver en que frecuencia esta dicho tono, ploteamos la transformada de Fourier. Se pueden ver claramente las dos deltas a 6 KHz correspondientes a este tono agudo. Para ver que frecuencia de corte debemos escoger para el filtro, debemos de tener en cuenta el transitorio del mismo. Despues de unas pruebas, con 5KHz para la frecuencia de corte ya se eliminan las deltas. La frecuencia de muestreo se corresponde a la misma que nos devuelve el comando audioread.

```
Xf = (fft(Xn,length(Xn)))/length(Xn);
f_x = linspace(-fs/2,fs/2,length(Xn));
figure();
plot(f_x/1000,fftshift(abs(Xf)));
title('Transformada de Fourier de Xn')
xlabel('Frecuencia (KHz)')
ylabel('|Xn(f)|')
```



Apartado c)

Diseñamos el filtro con los parámetros especificados en el archivo .mat y con la frecuencia de corte de 5 KHz.

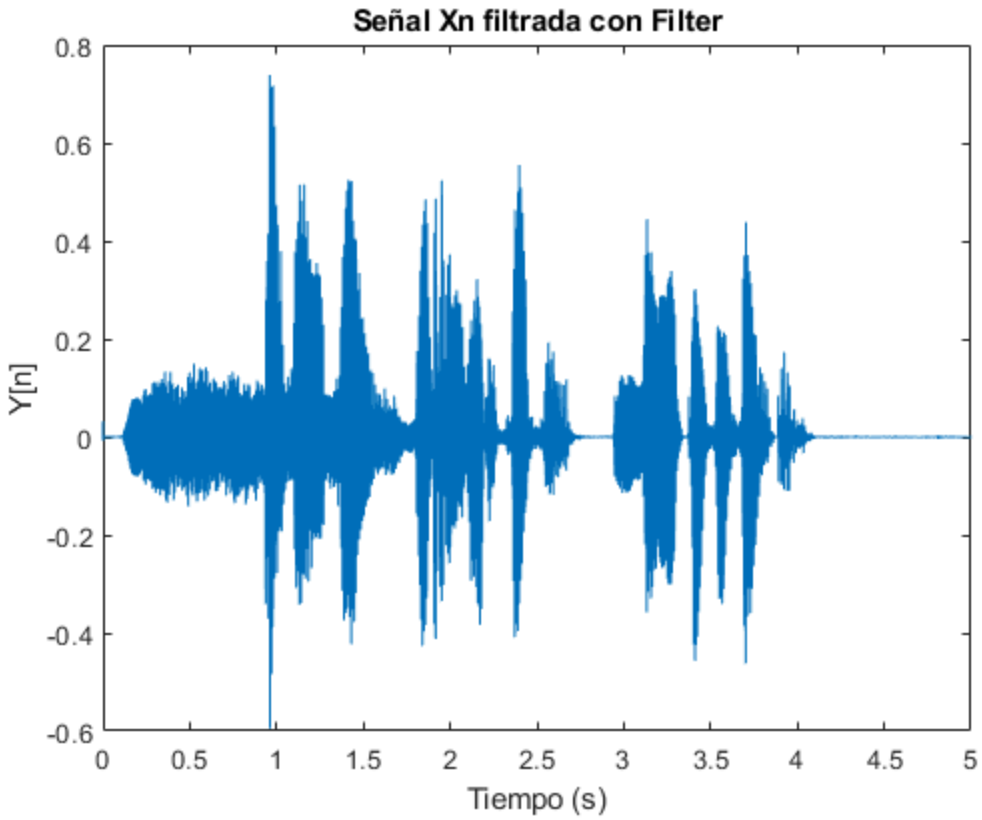
```
load('LPF_1.mat')
```

Apartado d)

Filtramos la señal original utilizando la función filter de Matlab. Es importante recordar que al usar esta función, el tamaño de la señal de salida es el mismo que el de entrada.

Hacemos un plot de la señal filtrada. Se puede apreciar que ya no hay tantos agudos en la señal, ya que no hay señal de rápida variación en el tiempo.

```
Yn = filter(LPF_1,1,Xn);
figure()
plot(t,Yn)
title('Señal Xn filtrada con Filter')
xlabel('Tiempo (s)')
ylabel('Y[n]')
xlim([0 5])
```

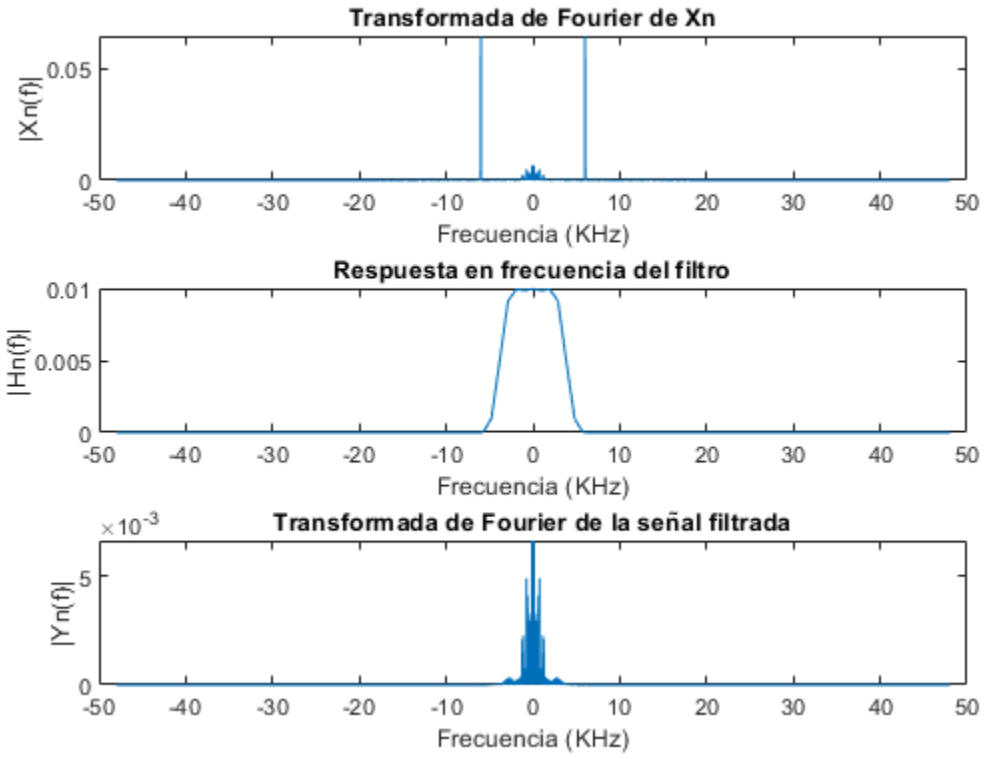


Apartado e)

Para ver mejor el correcto funcionamiento del filtrado, lo representamos en frecuencia y compramos la señal original con la filtrada. Se puede observar en la segunda imagen que el proceso de filtrado es correcto. También hemos escuchado la señal resultante del filtro y se escucha sin el pitido.

```
figure()
subplot(3,1,1)
plot(f_x/1000,fftshift(abs(Xf)));
title('Transformada de Fourier de Xn')
ylabel('|Xn(f)|')
xlabel('Frecuencia (KHz)')

subplot(3,1,2)
Yfilt = (fft(LPF_1,length(LPF_1)))/length(LPF_1);
f_filt = linspace(-fs/2,fs/2,length(LPF_1));
plot(f_filt/1000,fftshift(abs(Yfilt)));
title('Respuesta en frecuencia del filtro')
xlabel('Frecuencia (KHz)')
ylabel('|Hn(f)|')

subplot(3,1,3)
Yn_2 = filter(LPF_1,1,Xn);
Yf_nueva = (fft(Yn_2,length(Yn_2)))/length(Yn_2);
f_y = linspace(-fs/2,fs/2,length(Yn_2));
plot(f_y/1000,fftshift(abs(Yf_nueva)));
```

```
title('Transformada de Fourier de la señal filtrada');
xlabel('Frecuencia (KHz)');
ylabel('|Yn(f)|');

sound(Yn,fs);
```



Implementación de un filtro FIR utilizando DFT

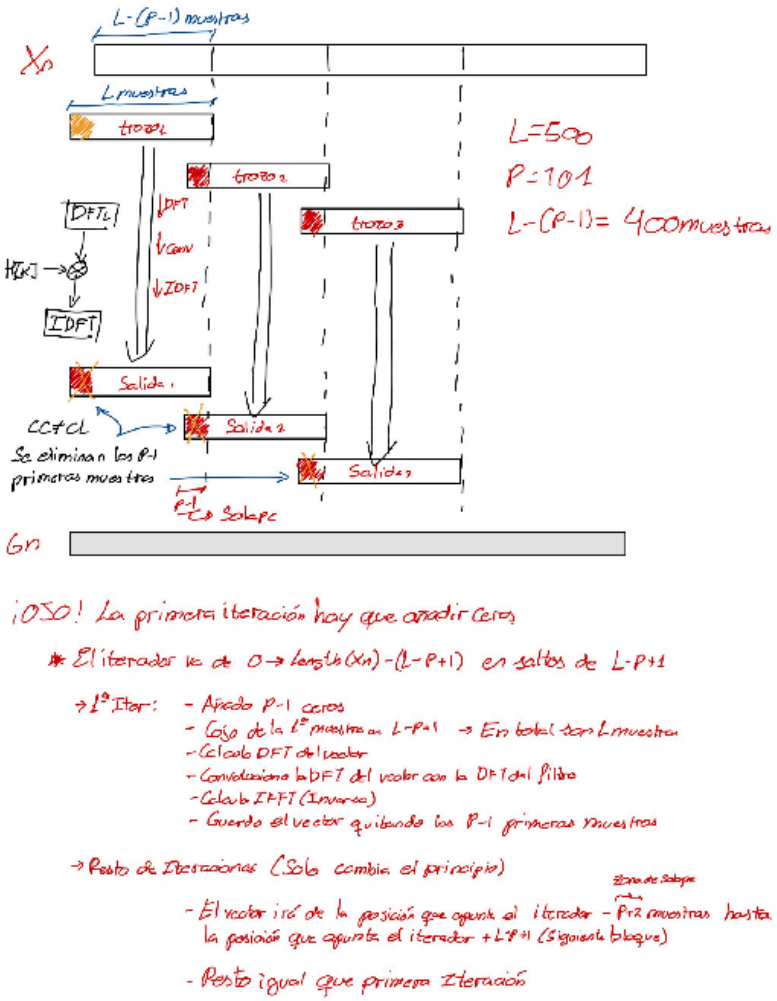
Apartado a b y c

En este apartado procedemos a implementar el algoritmo de overlap-save. Para ello hay que tener en cuenta que P = logitud del filtro FIR = 101 muestras. A continuación se muestra una imagen con la explicación de este algoritmo.

```
L = 500;
disp('P = longitud del filtro')
P = length(LPF_1)
ceros_filt = zeros(1,L-P);

LPF_2 = [LPF_1 ceros_filt];

Yfilt = fft(LPF_2,length(LPF_2));
f_filt = linspace(-fs/2,fs/2,length(LPF_2));

Gn = zeros(length(Xn),1);
```

```
for i=0:L-P+1:length(Xn)-(L-P+1)
    %i/(L-P+1)
    if(i==0)
        trozo = [zeros(P-1,1); Xn(1:L-P+1,1)];
        bloque = fft(trozo,length(trozo));
        convolucion = bloque.*Yfilt.';
        salida=ifft(convolucion,length(convolucion));
        Gn(1:L-P+1,1) = salida(P:end,1);
    else
        trozo = Xn(i-P+2:i+L-P+1,1);
        bloque = fft(trozo,length(trozo));
        convolucion = bloque.*Yfilt.';
        salida=ifft(convolucion,length(convolucion));
        Gn(i+1:i+L-P+1,1) = salida(P:end,1);

    end

end

% Como la foto no se ve del todo bien, la adjuntamos en la entrega

figure()
img = imread('imagen.jpg');
imshow(img);
title('Explicación Algoritmo');

sound(Gn,fs)

P = longitud del filtro

P =

   101
```

Explicación Algoritmo



Análisis de resultados

Apartado a)

Hemos calculado el ECM con todas las muestras de las señales. Esto se debe a que Yn es Xn filtrada utilizando filter, por lo que el tamaño de Yn es igual al de Xn e igual al de Gn.

Se puede observar como el error cuadrático medio entre Yn y Gn es despreciable (del orden de 10^-10).

```
disp('ECM entre Yn y Gn')
ECM(Yn,Gn)
```

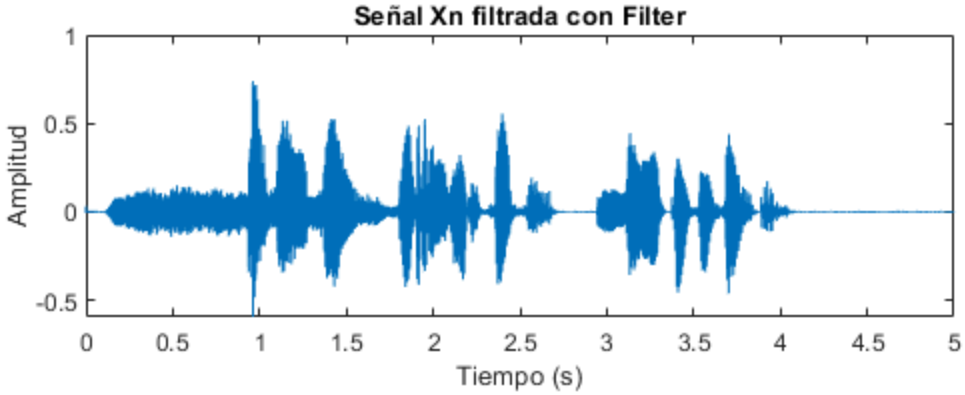
```
ECM entre Yn y Gn
```

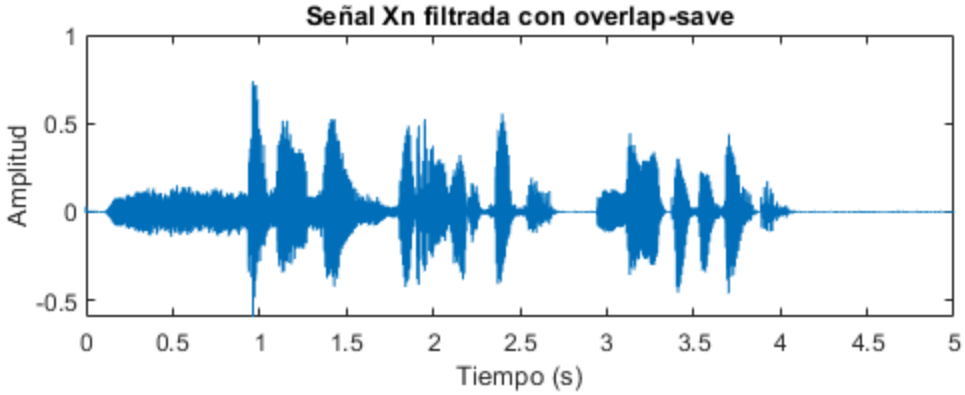
```
ans =

   4.1780e-10
```

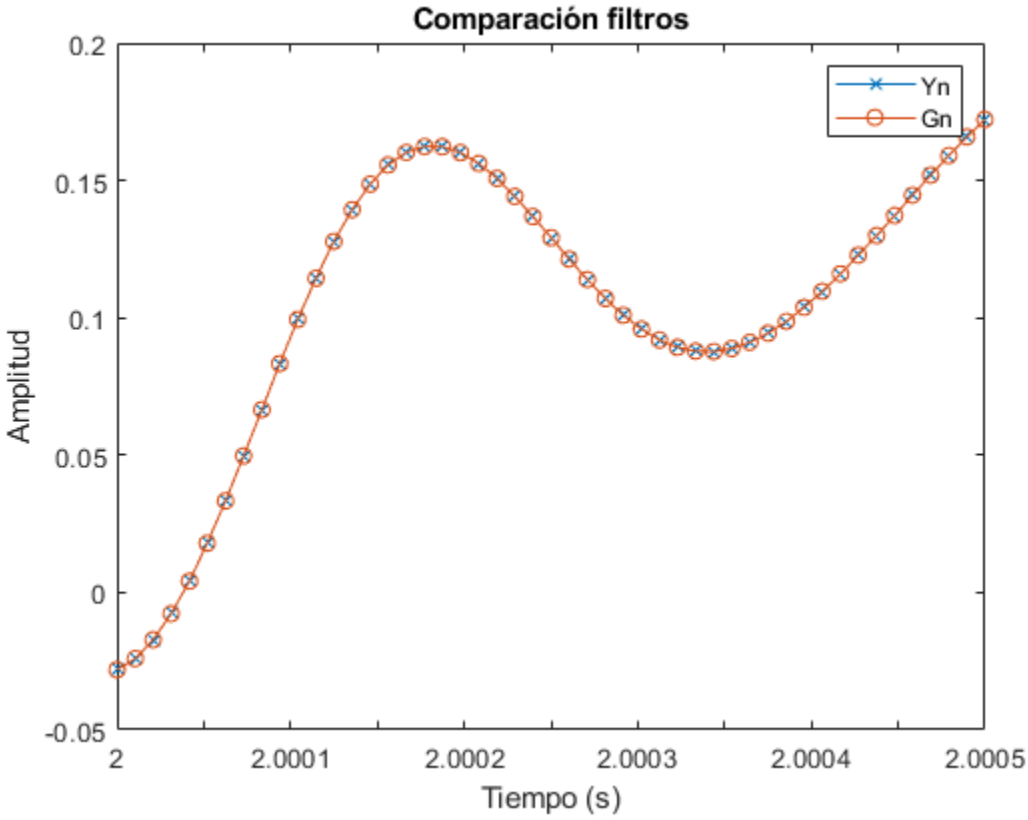
Apartado b)

Se puede observar que son prácticamente idénticas las salidas de los dos métodos empleados para filtrar Xn

```
figure()
subplot(2,1,1)
title('Señal filtrada con filter');
plot(t,Yn)
xlabel('Tiempo (s)')
ylabel('Amplitud')
title('Señal Xn filtrada con Filter')
xlim([0 5])

subplot(2,1,2)
plot(t,Gn)
xlabel('Tiempo (s)')
ylabel('Amplitud')
title('Señal Xn filtrada con overlap-save')
xlim([0 5])

figure()
plot(t,Yn,'-x')
hold on
plot(t,Gn,'-o')
xlabel('Tiempo (s)')
ylabel('Amplitud')
title('Comparación filtros')
legend('Yn','Gn')
xlim([2 2.0005])
```


Señal Xn filtrada con Filter
Amplitud
Tiempo (s)

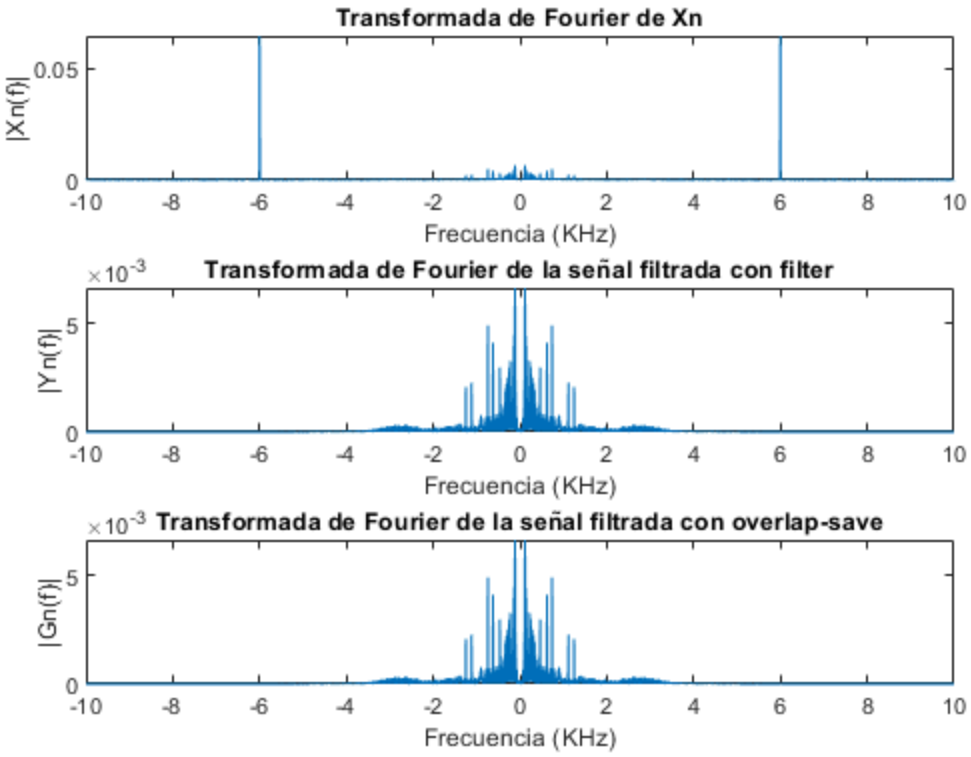

Señal Xn filtrada con overlap-save
Amplitud
Tiempo (s)


Comparación filtros
Yn
Gn
Amplitud
Tiempo (s)

Apartado c)

En el dominio de la frecuencia se reafirma que los dos métodos son prácticamente identicos.

```
figure()
subplot(3,1,1)
plot(f_x/1000,fftshift(abs(Xf)));
title('Transformada de Fourier de Xn')
xlabel('Frecuencia (KHz)')
ylabel('|Xn(f)|')
xlim([-10 10])

subplot(3,1,2)
Yf = (fft(Yn,length(Yn)))/length(Yn);
f_y = linspace(-fs/2,fs/2,length(Yn));
plot(f_y/1000,fftshift(abs(Yf)));
title('Transformada de Fourier de la señal filtrada con filter')
xlabel('Frecuencia (KHz)')
ylabel('|Yn(f)|')
xlim([-10 10])

subplot(3,1,3)
Gf = (fft(Gn,length(Gn)))/length(Gn);
f_g = linspace(-fs/2,fs/2,length(Gn));
plot(f_g/1000,fftshift(abs(Gf)));
title('Transformada de Fourier de la señal filtrada con overlap-save')
xlabel('Frecuencia (KHz)')
ylabel('|Gn(f)|')
xlim([-10 10])
```



Published with MATLAB® R2020a