Práctica 4

Table of Contents

Teresa González y Miguel Oleo LE2 G4

Introducción

El objetivo de la práctica es familiarizarnos con el uso de las funciones conv y filt, para aplicar un filtro, y conocer sus diferencias. En esta práctica también tenemos el primer acercamiento al diseño de filtros y aplicación de los mismos.

Filtrado de señales

Apartado a)

Cargamos el fichero y calculamos fs, tal y como hemos hecho en prácticas anteriores.

```
load('PDS_P4_LE2_G4.mat');
fs = (max(t)/size(x,1))^-1;
```

Apartado b)

En este apartado hemos implementado la función convolución proporcionada por el profesor.

El tamaño de la señal de salida es de m(length x) + n (length b) - 1. Este tamaño se puede justificar fácilmente haciéndolo manualmente. El vector tiene tamaño m, pero como "se va moviendo" el vector b, para convolucionar, al final tendremos n - 1 muestras extra, con lo que nos queda un tamaño de m + n - 1. El -1 se debe a que la ultima muestra, es la del vector n - 1, ya que el n ya se multiplicaría por 0.

```
Yn = convolucion(b,x);
```

Apartado c)

Aquí volvemos hacer una convolución, pero esta vez de con la función propia de Matlab. El tamaño de salida sigue siendo m + n - 1. Más tarde se comprueba como la salida de esta función es la misma que la del apartado b)

```
Gn = conv(b,x);
```

Apartado d)

Con la función filter la señal de salida se queda acotada a la señal de entrada, por lo que la longitud=m

```
Hn = filter(b,1,x);
```

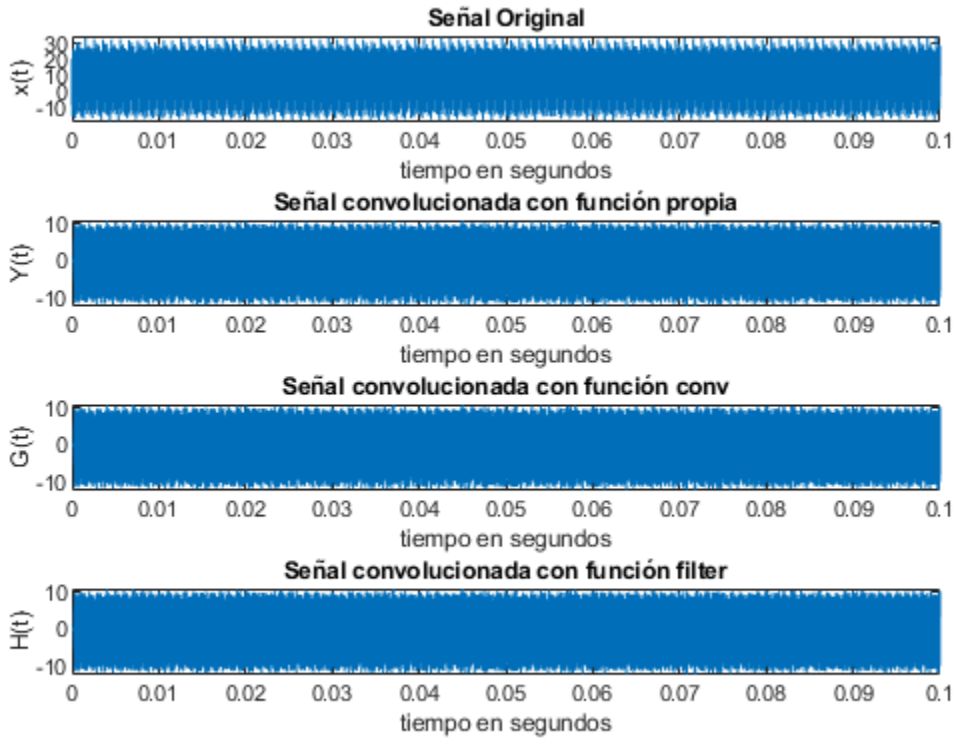
Apartado e)

Representamos en un subplot tanto la señal original, como la señal filtrada con los tres métodos anteriores.

En la gráfica no se puede apreciar diferencias entre las 3 formas de filtrar empleadas. En la gráfica de Hn (señal filtrada con función filter) se ha realizado un ajuste, ya que el tamaño del vector es menor que el de las otras dos funciones (como ya hemos explicado en el apartado d)

```
t_y = 0:1/fs:(length(Yn)-1)/fs;
t_g = t_y;
t_h = 0:1/fs:(length(Hn)-1)/fs;

figure()
subplot(4,1,1)
plot(t,x)
title('Señal Original')
xlabel('tiempo en segundos')
ylabel('x(t)')


subplot(4,1,2)
plot(t_y,Yn)
title('Señal convolucionada con función propia')
xlabel('tiempo en segundos')
ylabel('Y(t)')
xlim([0 0.1])

subplot(4,1,3)
plot(t_g,Gn)
title('Señal convolucionada con función conv')
xlabel('tiempo en segundos')
```

```
ylabel('G(t)')
xlim([0 0.1])

subplot(4,1,4)
plot(t_h,Hn)
title('Señal convolucionada con función filter')
xlabel('tiempo en segundos')
ylabel('H(t)')
```



Apartado f)

Transitorio: ms que tarda en llenarse la memoria del filtro con todas las muestras de la señal de entrada. L-1 muestras.

Retardo de grupo: retardo de la información de la señal de salida respecto a la original. L-1/2 muestras.

```
disp('Transitorio de Yn en ms')
tranYn = ((length(Yn)-1)/fs)*1000

disp('Transitorio de Gn en ms')
tranGn = ((length(Gn)-1)/fs)*1000

disp('Transitorio de Hn en ms')
tranHn = ((length(Hn)-1)/fs)*1000

disp('Retardo de grupo de Yn en ms')
```

```
RgYn = ((length(Yn)-1)/(2*fs))*1000

disp('Retardo de grupo de Gn en ms')
RgGn = ((length(Gn)-1)/(2*fs))*1000

disp('Retardo de grupo de Hn en ms')
RgHn = ((length(Hn)-1)/(2*fs))*1000

Transitorio de Yn en ms

tranYn =

  100.5351

Transitorio de Gn en ms

tranGn =

  100.5351

Transitorio de Hn en ms

tranHn =

   99.9946

Retardo de grupo de Yn en ms

RgYn =

   50.2676

Retardo de grupo de Gn en ms

RgGn =

   50.2676

Retardo de grupo de Hn en ms

RgHn =

   49.9973
```

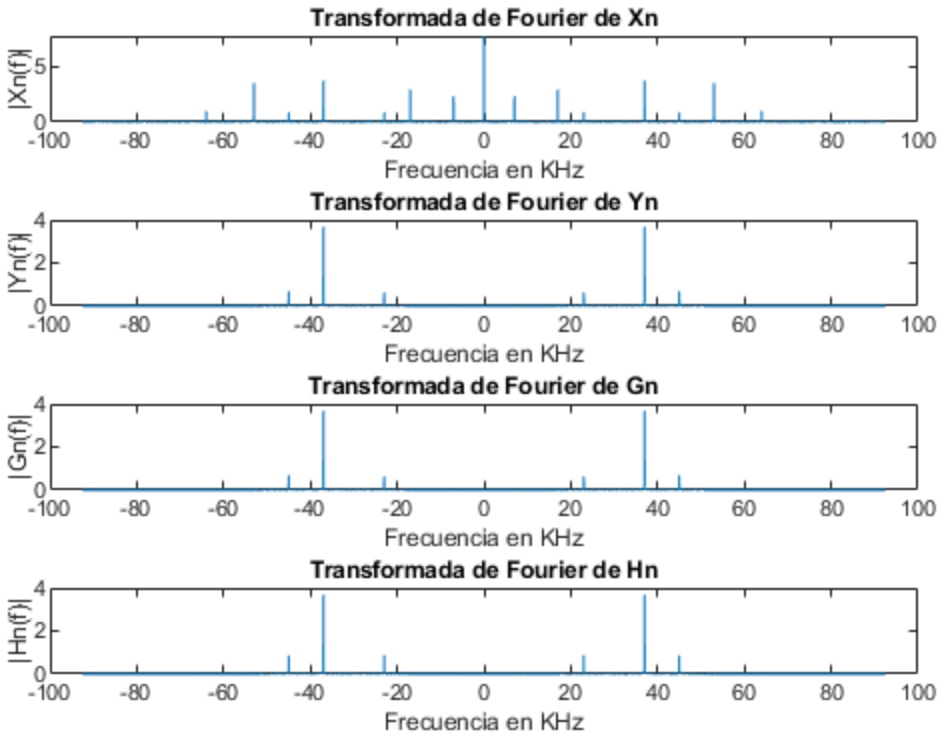
Apartado g)

En este apartado se calcula la transformada de Fourier de las señales originales y las filtradas con los tres métodos distintos. Con estás gráficas se puede apreciar mejor que en el apartado e) que el funcionamiento del dichas funciones es correcto y es igual para las tres.

Al ver las originales y las filtradas, podemos extraer información de que tipo de filtro se trata. En este caso se observa claramente que se trata de un filtro paso banda, ya que elimina las frecuencias bajas y las altas,

pero no unas intermedias. Podríamos decir que la frecuencia de corte por abajo sería de aproximadamente 20KHz y la superior estaría aproximadamente sobre los 50KHz.

```
figure()
subplot(4,1,1)
Xf = (fft(x,length(x)))/length(x);
f_x = linspace(-fs/2,fs/2,length(x));
plot(f_x/1000,fftshift(abs(Xf)));
title('Transformada de Fourier de Xn')
xlabel('Frecuencia en KHz')
ylabel('|Xn(f)|')

subplot(4,1,2)
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 Yn')
xlabel('Frecuencia en KHz')
ylabel('|Yn(f)|')

subplot(4,1,3)
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 Gn')
xlabel('Frecuencia en KHz')
ylabel('|Gn(f)|')


subplot(4,1,4)
Hf = (fft(Hn,length(Hn)))/length(Hn);
f_h = linspace(-fs/2,fs/2,length(Hn));
plot(f_h/1000,fftshift(abs(Hf)));
title('Transformada de Fourier de Hn')
xlabel('Frecuencia en KHz')
ylabel('|Hn(f)|')
```



Diseño de filtros

Apartado a y b)

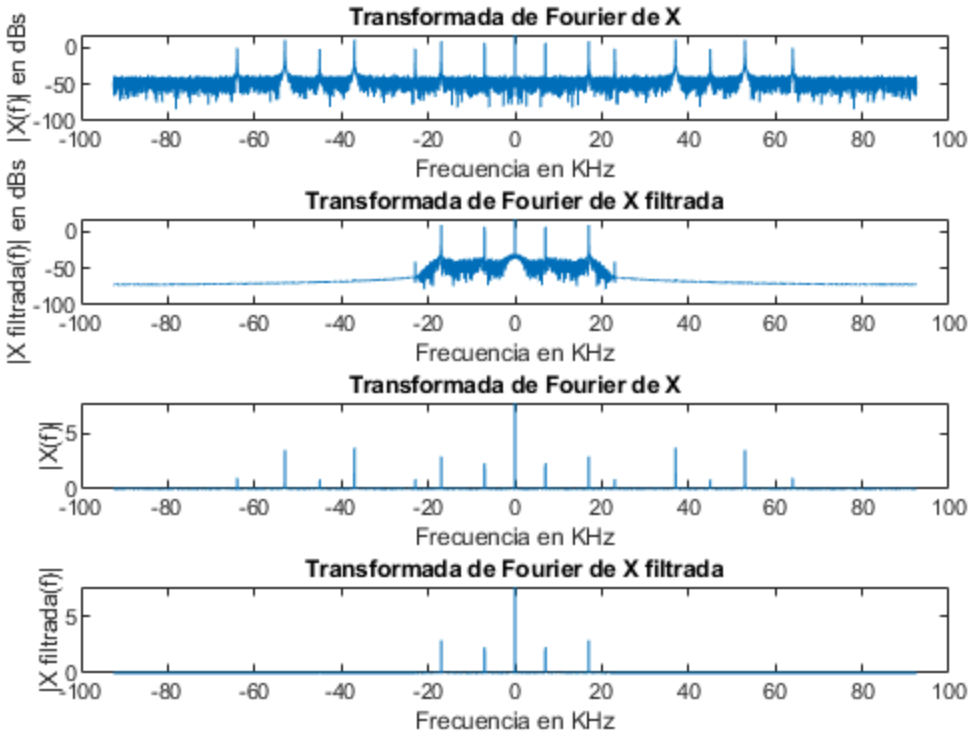
Se procede a diseñar un filtro paso bajo con las especificaciones del enunciado. Dicho filtro lo exportamos en un .mat, para poder leer dicho vector sin tener que diseñar el filtro todas las veces. Debido a esto, la primera instruccion es la del load del fichero.

Para comprobar que el diseño es correcto, hemos decidido realizar una serie de subplots de los espectros. La primera gráfica es la se corresponde a la señal original en dBs y la seguna a la filtrada también en dBs. Esta representación en decibelos la hemos escogido por la facilidad de comprobar si el filtro cumple con las especificaciones indicadas. Las ultimas dos gráficas son iguales que las primeras, pero en unidades naturales. En estas últimas dos, son menos expresivos los valores de amplitud, pero se ve más claro el proceso de filtrado, ya que no se ve tanto pico en la señal.

Es importante indicar que, cuando (por ejemplo en este filtro) te indican que la banda de paso tiene atenuación de 80dBs, se refiere a que la señal filtrada, esas frecuencias deben estar a 80dBs por debajo de la original, no a -80dBs respecto del 0. Por ello, hemos ploteado también la señal original en dBs.

```
load('lpf1.mat');
X_filt = filter(lpf1,1,x);

Xf_filt = (fft(X_filt,length(X_filt)))/length(X_filt);
f_xfilt = linspace(-fs/2,fs/2,length(X_filt));
```

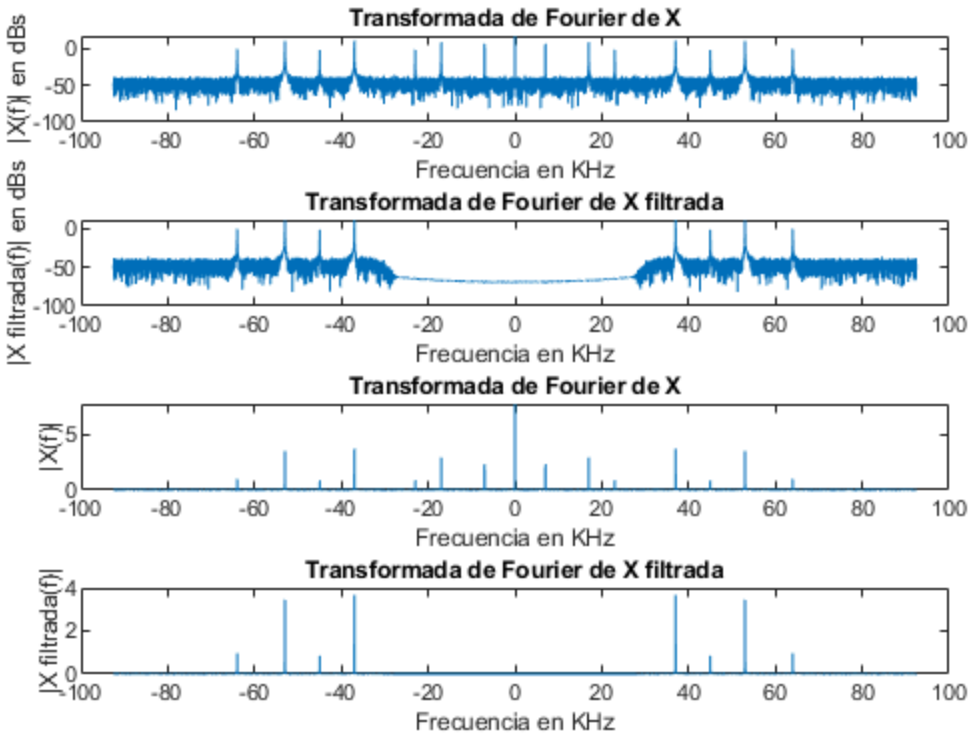
```
figure()
subplot(4,1,1)
plot(f_x/1000,mag2db(fftshift(abs(Xf))));
title('Transformada de Fourier de X')
xlabel('Frecuencia en KHz')
ylabel('|X(f)| en dBs')

subplot(4,1,2)
plot(f_xfilt/1000,mag2db(fftshift(abs(Xf_filt))));
title('Transformada de Fourier de X filtrada')
xlabel('Frecuencia en KHz')
ylabel('|X filtrada(f)| en dBs')

subplot (4,1,3)
plot(f_x/1000,fftshift(abs(Xf)));
title('Transformada de Fourier de X')
xlabel('Frecuencia en KHz')
ylabel('|X(f)|')

subplot(4,1,4)
plot(f_xfilt/1000,fftshift(abs(Xf_filt)));
title('Transformada de Fourier de X filtrada')
xlabel('Frecuencia en KHz')
ylabel('|X filtrada(f)|')
```



Apartado c y d)

Ahora, repetimos todo lo anterior pero con un nuevo filtro a diseñar, esta vez paso alto. En la gráficas se puede observar que se cumplen con las especificaciones, siguiendo el mismo procedimiento que el apartado anterior y con las mismas gráficas.

```
load('hpf1.mat');
X_filt = filter(hpf1,1,x);

Xf_filt = (fft(X_filt,length(X_filt)))/length(X_filt);
f_xfilt = linspace(-fs/2,fs/2,length(Xf_filt));

figure()
subplot(4,1,1)
plot(f_x/1000,mag2db(fftshift(abs(Xf))));
title('Transformada de Fourier de X')
xlabel('Frecuencia en KHz')
ylabel('|X(f)| en dBs')

subplot(4,1,2)
plot(f_xfilt/1000,mag2db(fftshift(abs(Xf_filt))));
title('Transformada de Fourier de X filtrada')
xlabel('Frecuencia en KHz')
ylabel('|X filtrada(f)| en dBs')

subplot(4,1,3)
plot(f_x/1000,fftshift(abs(Xf)));
title('Transformada de Fourier de X')
xlabel('Frecuencia en KHz')
ylabel('|X(f)|')

subplot(4,1,4)
plot(f_xfilt/1000,fftshift(abs(Xf_filt)));
title('Transformada de Fourier de X filtrada')
xlabel('Frecuencia en KHz')
ylabel('|X filtrada(f)|')
```



Análisis de Filtros

Apartado a)

Se pide filtrar con el filtro paso bajo diseñado la señal original. Nosotros hemos optado por usar la función proporcionada por Matlab filter().

```
Yn1 = filter(lpf1,1,x);
```

Apartado b)

Ahora procedemos a filtrar la señal original con el filtro paso alto diseñado anteriormente con la función filter().

```
Gn1 = filter(hpf1,1,x);
```

Apartado c)

Diseñamos un filtrom paso banda llamado bpf1 con las especificaciones requeridas y lo exportamos en un .mat por las razones indicadas anteriormente.

Apartado d)

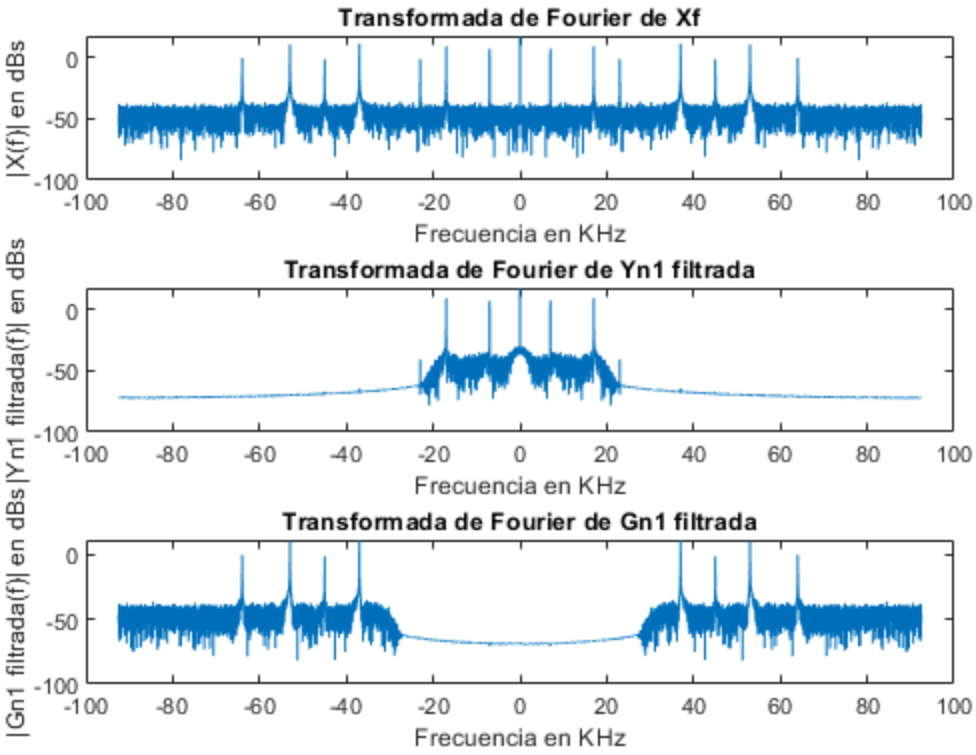
Cargamos el filtro paso banda y filtramos la señal original con filter().

```
load('bpf1.mat');
Hn1 = filter(bpf1,1,x);
```

Apartado e)

Representamos los espectros de la señal original, la original filtrada por un filtro paso bajo y por un paso alto. Los plots los volvemos a representar en dBs para poder asegurar que los requisitos del enunciado se cumplen.

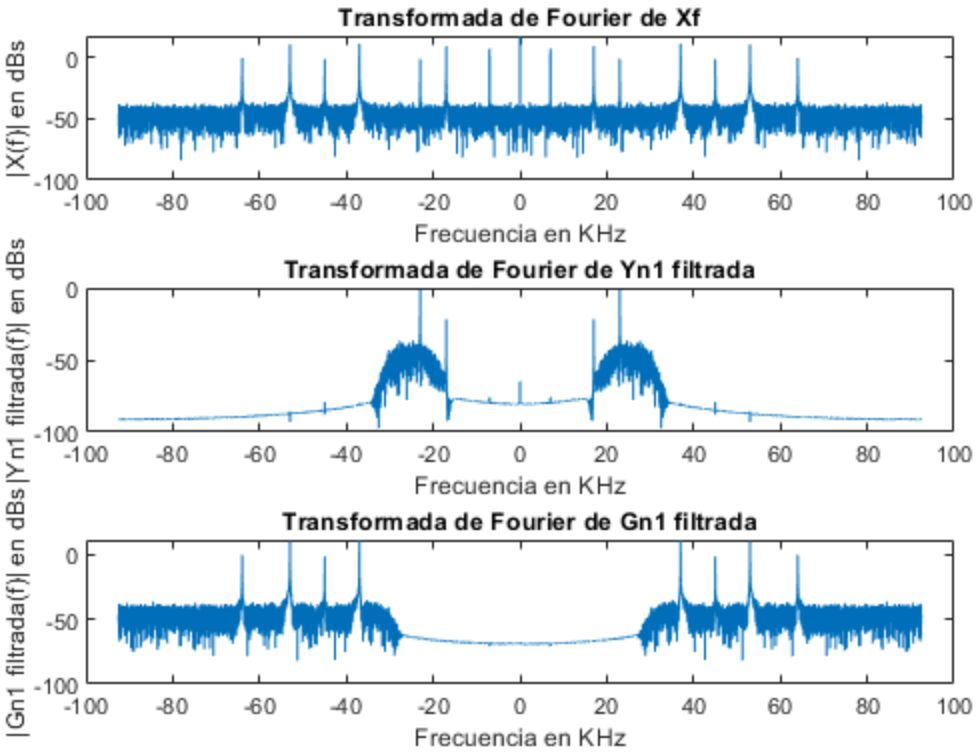
```
Yn1f = (fft(Yn1,length(Yn1)))/length(Yn1);
f_hfilt = linspace(-fs/2,fs/2,length(Yn1f));

Gn1f = (fft(Gn1,length(Gn1)))/length(Gn1);

figure()
subplot(3,1,1)
plot(f_h/1000,mag2db(fftshift(abs(Xf))));
title('Transformada de Fourier de Xf')
xlabel('Frecuencia en KHz')
ylabel('|X(f)| en dBs')

subplot(3,1,2)
plot(f_hfilt/1000,mag2db(fftshift(abs(Yn1f))));
title('Transformada de Fourier de Yn1 filtrada')
xlabel('Frecuencia en KHz')
ylabel('|Yn1 filtrada(f)| en dBs')

subplot(3,1,3)
plot(f_hfilt/1000,mag2db(fftshift(abs(Gn1f))));
title('Transformada de Fourier de Gn1 filtrada')
xlabel('Frecuencia en KHz')
ylabel('|Gn1 filtrada(f)| en dBs')
```



Apartado f)

Repetimos el apartado anterior pero esta vez representanando la señal original, la filtrada con un paso alto y la filtrada con un paso banda. Volvemos a representaro en dBs para ver que el proceso es correcto.

```
Hn1f = (fft(Hn1,length(Hn1)))/length(Hn1);
figure()
subplot(3,1,1)
plot(f_h/1000,mag2db(fftshift(abs(Xf))));
title('Transformada de Fourier de Xf')
xlabel('Frecuencia en KHz')
ylabel('|X(f)| en dBs')

subplot(3,1,2)
plot(f_hfilt/1000,mag2db(fftshift(abs(Hn1f))));
title('Transformada de Fourier de Yn1 filtrada')
xlabel('Frecuencia en KHz')
ylabel('|Yn1 filtrada(f)| en dBs')

subplot(3,1,3)
plot(f_hfilt/1000,mag2db(fftshift(abs(Gn1f))));
title('Transformada de Fourier de Gn1 filtrada')
xlabel('Frecuencia en KHz')
ylabel('|Gn1 filtrada(f)| en dBs')
```



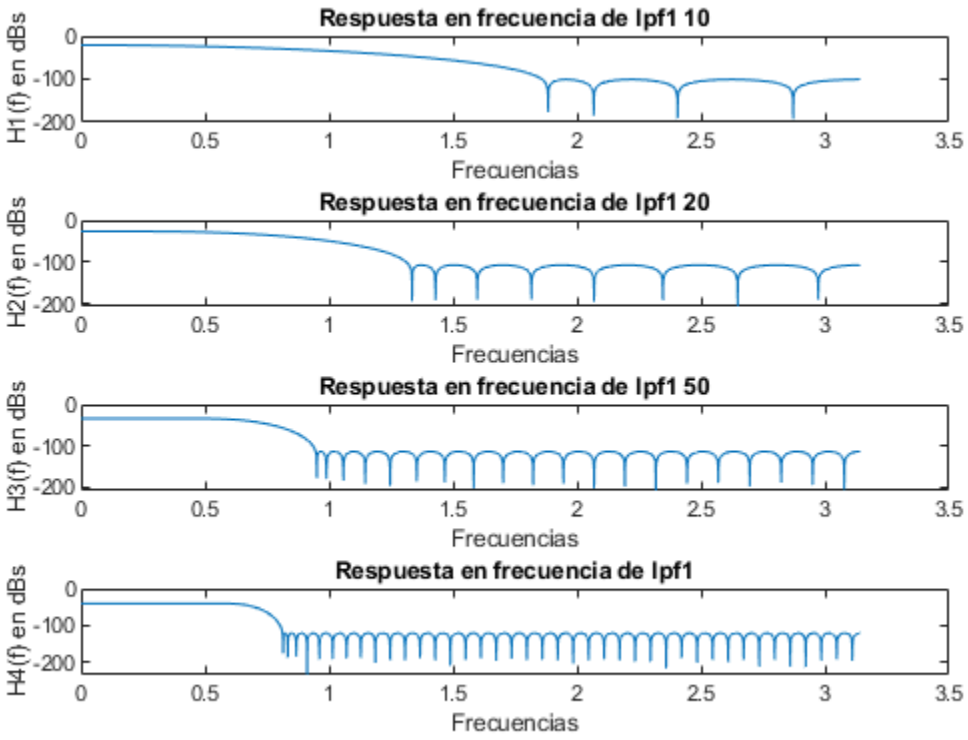
Orden del filtro

Apartado a y b)

Creamos los filtros paso bajo con 10, 20 y 50 coeficientes y lo exportamos en un .mat. Para representar gráficamente los espectros de los filtros paso bajo, empleamos la función freqz proporcionada por Matlab. En la documentación se observa que los imputs de dicha función son muy diversos, según lo que queramos a la salida. Nosotros hemos optado por imputs: freqz(filtro,Tam_Filtro,Fs), ya que nos proporciona a la salida, tanto la transformada de Fourier como el vector de frecuencias, así no tenemos que calcularlo nosotros.

```
load('lpf1_10.mat');
load('lpf1_20.mat');
load('lpf1_50.mat');
[h1,f1]=freqz(lpf1_10,length(lpf1_10), 185010);
[h2,f2]=freqz(lpf1_20,length(lpf1_20), 185010);
[h3,f3]=freqz(lpf1_50,length(lpf1_50), 185010);
[h4,f4]=freqz(lpf1,length(lpf1), 185010);
figure();
subplot (4,1,1)
plot (f1,mag2db(abs(h1)))
title('Respuesta en frecuencia de lpf1 10')
xlabel('Frecuencias')
ylabel('H1(f) en dBs')
subplot (4,1,2)
```

```
plot (f2,mag2db(abs(h2)))
title('Respuesta en frecuencia de lpf1 20')
xlabel('Frecuencias')
ylabel('H2(f) en dBs')
subplot (4,1,3)
plot (f3,mag2db(abs(h3)))
title('Respuesta en frecuencia de lpf1 50')
xlabel('Frecuencias')
ylabel('H3(f) en dBs')
subplot (4,1,4)
plot (f4,mag2db(abs(h4)))
title('Respuesta en frecuencia de lpf1')
xlabel('Frecuencias')
ylabel('H4(f) en dBs')
```



Apartado c)

Se ha calculado de la misma forma que en la primera parte.

```
disp('Retardo de grupo de H1 en ms')
RgH1 = ((length(h1)-1)/(2*fs))*1000

disp('Retardo de grupo de H2 en ms')
RgH2 = ((length(h2)-1)/(2*fs))*1000

disp('Retardo de grupo de H3 en ms')
```

```
RgH3 = ((length(h3)-1)/(2*fs))*1000

disp('Retardo de grupo de H4 en ms')
RgH4 = ((length(h4)-1)/(2*fs))*1000
```

```
Retardo de grupo de H1 en ms

RgH1 =

  499.9973

Retardo de grupo de H2 en ms

RgH2 =

  499.9973

Retardo de grupo de H3 en ms

RgH3 =

  499.9973

Retardo de grupo de H4 en ms

RgH4 =

  499.9973
```

Published with MATLAB® R2019b