IEE239 - Procesamiento de Señales e Imágenes Digitales
Laboratorio 3 - Ejercicios Propuestos
Primer Semestre 2018

1. Tratando de salir de sus deudas, el señor Jürgen acepta un trabajo para estudiar señales de audio propias de la fauna de los bosques de la ciudad de Los Alamos. Durante su trabajo, se le pide diseñar filtros para detectar la presencia de tres tipos de animales salvajes propios de la región, cuyos sonidos ocupan las siguientes bandas de frecuencias:

   - Puma: 100 – 1500 Hz
   - Coyote: 1700 – 2300 Hz
   - Ardilla: 2400 – 3000 Hz.

   Durante su última expedición al bosque, el señor Jürgen está listo para probar su sistema de identificación de animales en tiempo real. Para ello, graba la señal $x[n]$ proporcionada en el archivo `grabacion.mat` que contiene la señal grabada y la frecuencia de muestreo `fs`. Para esta pregunta, se pide:

   a. Diseñar un filtro pasabanda para los sonidos de pumas. Usar el método de invarianza del impulso (comando `impinvar`), con la misma frecuencia de muestreo de las señales grabadas, para convertir el filtro Chebyshev analógico provisto en el archivo `analog_cheby.mat`. En este archivo los coeficientes del denominador y numerador de la función de transferencia del filtro analógico se encuentran en las variables `at` y `bt`, respectivamente. Usar `freqz` con 512 puntos para graficar y comprobar la respuesta del filtro diseñado. Indicar las frecuencias de corte del filtro y el rizado en la banda de paso y rechazo.

   b. Diseñar un filtro FIR pasabanda de orden 100 para los sonidos de coyote en la banda especificada usando el método de enventanado (función `fir1`). Emplear una ventana Hamming. Graficar la respuesta en magnitud en escala logarítmica del filtro diseñado usando las funciones `fft`, `fftshift`, `abs` y `log10`. El gráfico debe estar en el intervalo de frecuencia angular normalizada ($-\pi$ a π). Adicionalmente, utilizar `angle`, `fftshift` y `unwrap` para graficar la respuesta en fase del filtro (`fftshift` reorderna un vector de respuestas en frecuencia de longitud N para que en lugar de ir de 0 a N-1 vaya de -N/2+1 a N/2. `unwrap` suma múltiplos de 2π a una respuesta en fase para eliminar las discontinuidades de salto, de manera que esta se pueda mostrar de manera continua).

   c. Diseñar un filtro butterworth digital de orden 4 para los sonidos de ardilla en la banda especificada. Para ello, usar el comando `butter`. Usar `freqz` con 512 puntos para graficar y comprobar la respuesta del filtro diseñado. Indicar el ancho de la banda de transición y el valor de la frecuecia de corte ¿Es la respuesta monótona en las bandas de paso y rechazo?

   d. Usar los filtros diseñados para cada uno de los animales y filtrar la señal grabada $x[n]$ usando `conv` para los filtros FIR y `filter` para los filtros IIR. Usar `stem` y `subplot` para mostrar todas las respuestas en una misma ventana. Graficar y rotular los espectros de

magnitud en escala logarítmica de cada señal filtrada usando `fft`, `fftshift`, `abs`, `log10` y `linspace`. Comprobar que se está filtrando en las bandas de frecuencia adecuadas.

e. Sea $y_i[n]$ la señal filtrada para el i-ésimo animal, se define la energía normalizada como

$$\epsilon_i = \frac{\sum_n y_i^2[n]}{\sum_n x^2[n]}.$$

Si se considera que un ϵ_i mayor a 0.07 implica que el i-ésimo animal está presente en la grabación, entonces ¿Qué animales están cerca al señor Jürgen?

2. Un filtro *notch* es un filtro rechazabanda con una banda de rechazo muy angosta. Estos filtros normalmente son útiles para rechazar interfencias de línea eléctrica a 60 Hz (o 50 Hz en ciertos países) y son especialmente usados en pre-procesamiento de señales bioeléctricas (*e.g.* Señales mioeléctricas, señales electroencefalográficas superficiales), cuyas bajas amplitudes las hacen especialmente sensibles a este tipo de interfencias. Considerando que se va a trabajar a una frecuencia de muestreo de 1 KHz, se pide:

a. Diseñar un filtro *notch* a $\frac{2\pi 60}{1000}$ rad/muestra con una estructura en paralelo de un filtro pasa-bajos y un filtro pasa-altos. Primero, diseñar un filtro FIR pasabajos de orden 32 con frecuencia de corte de $\frac{2\pi 40}{1000}$ rad/muestra (se toma una frecuencia ligeramente menor a $\frac{2\pi 60}{1000}$ rad/muestra) utilizando el método de enventanado (función `fir1`). Utilizar una ventana Hamming. Graficar la respuesta del filtro diseñado en escala logarítmica (dB) usando las funciones `fft`, `fftshift`, `abs` y `log10`. El gráfico debe estar en el intervalo de frecuencia angular normalizada ($-\pi$ a π).

Solución: Recordar que para sus funciones de diseño de filtros, MATLAB utiliza un valor de frecuencia normalizada de 0 a 1, donde 1 corresponde a `Fs/2`, donde `Fs` es la frecuencia de muestreo a ser utilizada. Por tanto, para convertir cualquier valor en Hz a este valor de frecuencia, basta con dividir por `Fs/2`. La respuesta del filtro se muestra en la Figura 1.

```
Fs = 1000; %frecuencia de muestreo
Fc = 40/(Fs/2); % frecuencia de corte normalizada
orden = 32;
NFFT = 512;
b_bajos = fir1(orden,Fc,hamming(orden+1));
H = abs(fftshift(fft(b_bajos,NFFT)));
w_axis = 2*pi*(-NFFT/2+1:NFFT/2)/NFFT; % eje de frecuencia normalizada
figure, plot(w_axis,20*log10(H));
xlabel('Frecuencia normalizada (rad/muestra)');
ylabel('Magnitud')
title('Filtro pasa-bajos usando fir1')
```

b. Ahora diseñar un filtro pasaaltos de orden 32 con una frecuencia de corte de $\frac{2\pi 80}{1000}$ rad/muestra (se toma una frecuencia ligeramente mayor a 60 Hz) utilizando el método de muestreo en frecuencia (función `fir2`). Utilizar diez muestras uniformemente espaciadas. Graficar la respuesta del filtro diseñado en escala logarítmica usando las funciones `fft`, `fftshift`, `abs` y `log10`. El gráfico debe estar en el intervalo de frecuencia angular normalizada ($-\pi$ a π).

Solución: La respuesta del filtro se muestra en la Figura 2.

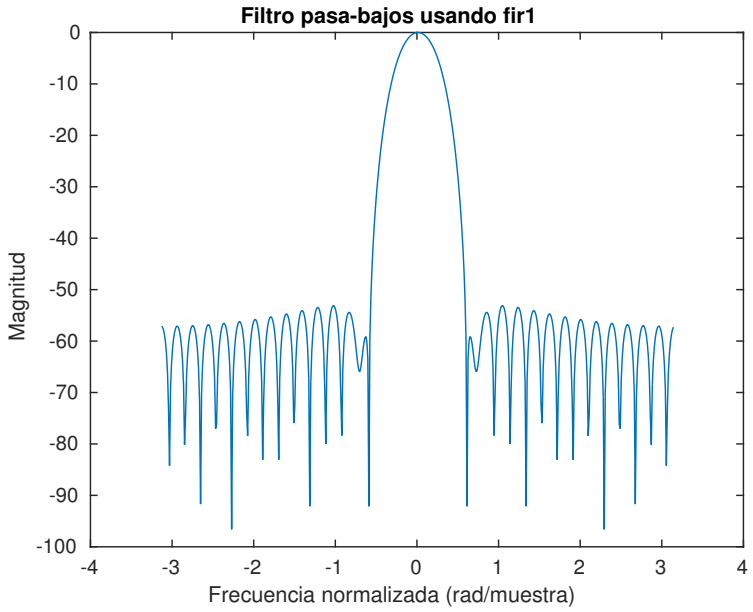


Figura 1: Respuesta del filtro pasa-bajos diseñado con `fir1`.

```
Fs = 1000; %frecuencia de muestreo
Fc = 80/(Fs/2); % frecuencia de corte normalizada
long_filtro = 10;
frec_vector = linspace(0,1,long_filtro)'; %valores de frecuencia
amp_vector = zeros(long_filtro,1); amp_vector(frec_vector>Fc)=1; ...
    %definir valores deseados de respuesta
orden = 32;
NFFT = 512;
b_altos = fir2(orden,frec_vector,amp_vector);
H = abs(fftshift(fft(b_altos,NFFT)));
w_axis = 2*pi*(-NFFT/2+1:NFFT/2)/NFFT; % eje de frecuencia normalizada
figure, plot(w_axis,20*log10(H));
xlabel('Frecuencia normalizada (rad/muestra)');
ylabel('Magnitud')
title('Filtro pasa-altos usando fir2')
```

c. Sean $x[n]$ e $y[n]$ las señales de entrada y filtrada, respectivamente, y sean $h_{LP}[n]$ y $h_{HP}[n]$, obtener un filtro resultante de una estructura en paralelo con los filtros detallados en los incisos 2a y 2b tal como se muestra en el diagrama de bloques de la Figura 3. Usar `freqz` con 512 puntos para graficar y comprobar la respuesta del filtro diseñado.

Solución:

Sean $x[n]$ e $y[n]$ las señales de entrada y filtrada, respectivamente, y sean $h_{LP}[n]$ y $h_{HP}[n]$ los filtros pasa-bajos y pasa-altos diseñados anteriormente, por la estructura en paralelo se tiene:

$$y[n] = x[n] * h_{LP}[n] + x[n] * h_{HP}[n] = (h_{LP}[n] + h_{HP}[n]) * x[n],$$

por lo que el filtro resultante es la suma de los dos filtros diseñados. La respuesta del filtro resultante se muestra en la Figura 4.

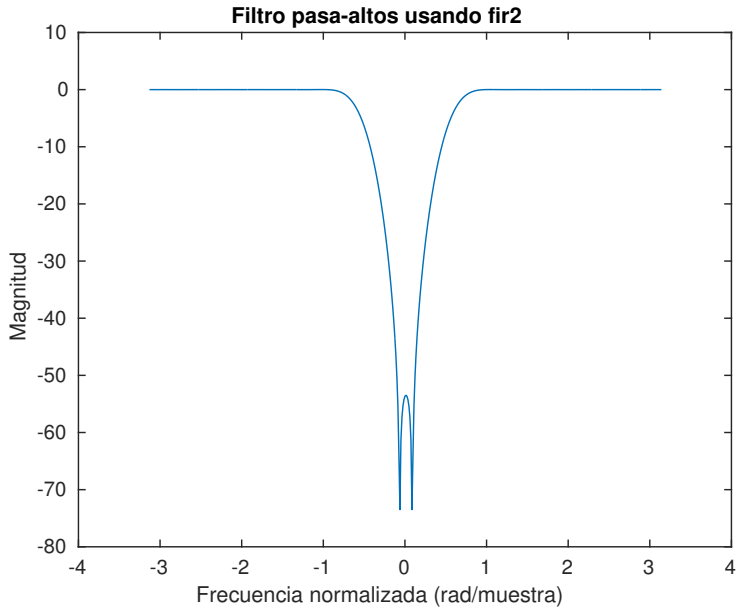


Figura 2: Respuesta del filtro pasa-altos diseñado con `fir2`.

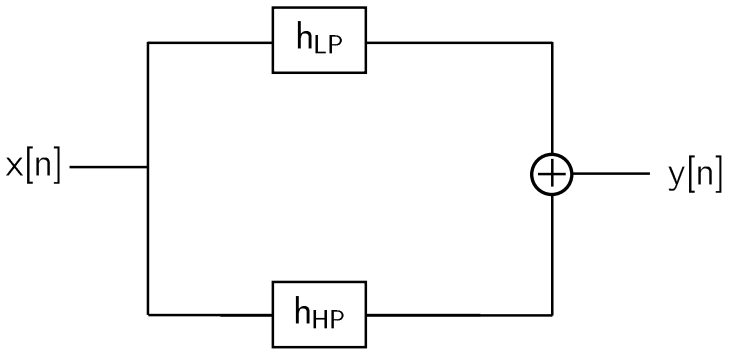


Figura 3: Estructura paralela para el filtro *notch*.

```
b_paralelo = b_bajos + b_altos; % respuesta de filtros en paralelo
NFFT=512;
H = abs(fftshift(fft(b_paralelo,NFFT)));
w_axis = 2*pi*(-NFFT/2+1:NFFT/2)/NFFT; % eje de frecuencia normalizada
figure, plot(w_axis,20*log10(H));
xlabel('Frecuencia normalizada (rad/muestra)');
ylabel('Magnitud');
title('Filtro notch en estructura paralela');
```

d. Leer la señal `emg.mat` que contiene una señal de electromiografía contaminada con interferencia de 60 Hz. Filtrar la señal utilizando la estructura en paralelo del inciso anterior. Graficar el espectro de magnitud normalizado de la señal resultante y hallar la atenuación (en dB) a aproximadamente $\frac{2\pi 60}{1000}$ rad/muestra.

Solución:

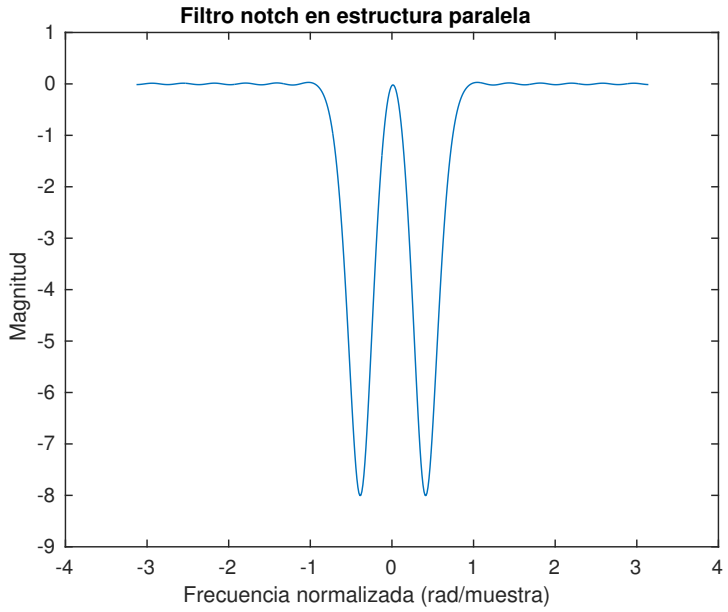


Figura 4: Filtro notch resultante mediante la estructura en paralelo.

Los espectros se muestran en la Figura 5. Los valores de magnitud a $\frac{2\pi 60}{1000}$ rad/muestra se hallan buscando con `min` el índice correspondiente al valor del eje más cercano a $\frac{2\pi 60}{1000}$ y luego tomando este mismo índice en los vectores de magnitud (líneas 17 y 18 del código mostrado a continuación). La atenuación es $20\log(\frac{47.7}{118.9}) \approx -7.92$ dB.

```
1  load 'emg.mat'
2  Fs = 1000; %frecuencia de muestreo
3  filtrada1 = filter(b_bajos,1,emg) + filter(b_altos,1,emg);
4  NFFT = 2^nextpow2(length(emg)); % nextpow proporciona el exponentes de ...
       la siguiente potencia de dos. La ejecucion de FFT es mas rapida si ...
       se realiza en una secuencia cuya longitud es una potencia de dos.
5  Mag_entrada = abs(fftshift(fft(emg,NFFT)));
6  Mag_salida = abs(fftshift(fft(filtrada1,NFFT)));
7  w_axis = 2*pi*(-NFFT/2+1:NFFT/2)/NFFT; % eje de frecuencia normalizada
8  figure
9  subplot(1,2,1), plot(w_axis,(Mag_entrada));
10 title('Espectro de Magnitud - Entrada');
11 xlabel('Frecuencia normalizada (rad/muestra)');
12 ylabel('Amplitud');
13 subplot(1,2,2), plot(w_axis,(Mag_salida));
14 title('Espectro de Magnitud - Filtrada');
15 xlabel('Frecuencia normalizada (rad/muestra)');
16 ylabel('Amplitud');
17 [minimo,pos] = min(abs(w_axis-2*pi*60/Fs));  % buscar la frecuencia mas ...
       cercana a 2*pi*60/1000
18 atenuacion = 20*log10(Mag_salida(pos)/Mag_entrada(pos))
```

e. Considerar la siguiente función de transferencia de un filtro *notch* analógico:

$$H(s) = \frac{s^2 + \omega_c^2}{s^2 + \sigma s + \omega_c^2}, \tag{1}$$

donde ω_c es la frecuencia angular de corte y σ es un factor que controla la caída del filtro

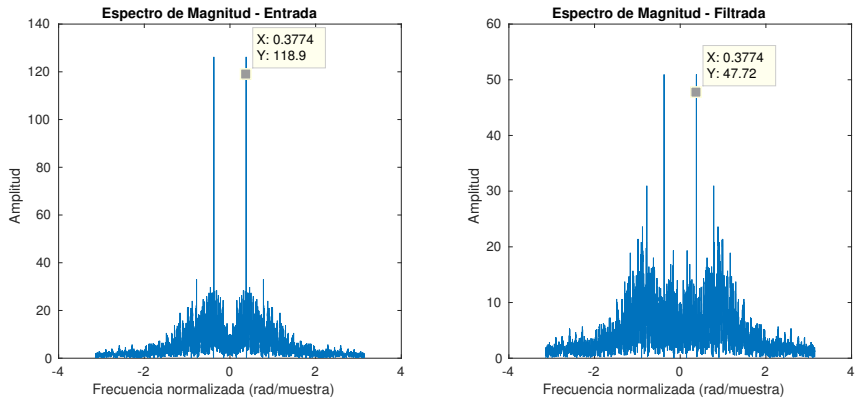


Figura 5: Espectros de magnitud de la señal de entrada y la señal filtrada.

(Mientras menor sea ω_c, la caída es más rápida). Tomar $\sigma = 20$ y diseñar un filtro *notch* digital a 60 Hz (la cual debe ser $\omega_c/2\pi$ según el criterio de -3dB) utilizando el método de la transformación bilineal (función `bilinear`). Usar `freqz` con 512 puntos para graficar y comprobar la respuesta del filtro diseñado.

Solución:

La respuesta del filtro se muestra en la Figura 6.

```
wc = 2*pi*60;
sigma = 20;
NFFT = 512;
num_analog = [1 0 wc^2];
den_analog = [1 sigma wc^2];
[numd,dend] = bilinear(num_analog,den_analog,Fs);
figure, freqz(numd,dend,NFFT);
title('Filtro notch con transformada bilineal');
```

f. Filtrar la señal contenida utilizando el filtro diseñado en el inciso anterior. Graficar el espectro de magnitud normalizado de la señal original y la señal resultante. Hallar la atenuación (en dB) a aproximadamente 60 Hz.

Solución:

Los espectros se muestran en la Figura 7. Los valores de magnitud a $\frac{2\pi 60}{1000}$ rad/muestra se hallan buscando con `min` el índice correspondiente al valor del eje más cercano a $\frac{2\pi 60}{1000}$ y luego tomando este mismo índice en los vectores de magnitud (líneas 17 y 18 del código mostrado a continuación). La atenuación es $20\log(\frac{47.1}{118.9}) \approx -8.03$ dB.

```
      load 'emg.mat'
Fs = 1000; %frecuencia de muestreo
filtrada2 = filter(numd,dend,emg);
NFFT = 2^nextpow2(length(emg)); % nextpow proporciona el exponentes de ...
    la siguiente potencia de dos. La ejecucion de FFT es mas rapida si ...
    se realiza en una secuencia cuya longitud es una potencia de dos.
Mag_entrada = abs(fftshift(fft(emg,NFFT)));
Mag_salida = abs(fftshift(fft(filtrada2,NFFT)));
w_axis = 2*pi*(-NFFT/2+1:NFFT/2)/NFFT; % eje de frecuencia normalizada
```

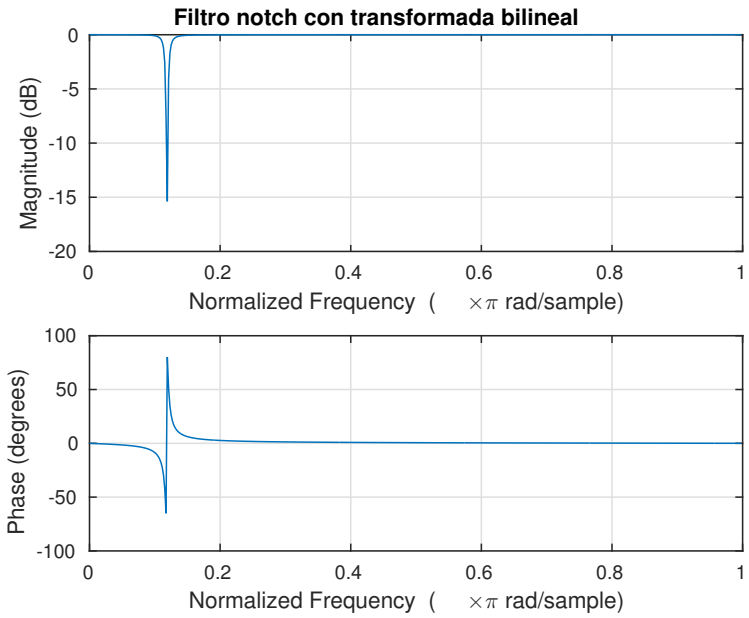


Figura 6: Respuesta del filtro *notch* diseñado con transformación bilineal

```
figure;
subplot(1,2,1), plot(w_axis,(Mag_entrada));
title('Espectro de Magnitud - Entrada');
xlabel('Frecuencia normalizada');
ylabel('Amplitud');
subplot(1,2,2), plot(w_axis,(Mag_salida));
title('Espectro de Magnitud - Filtrada');
xlabel('Frecuencia normalizada (rad/muestra)');
ylabel('Amplitud');
[minimo,pos] = min(abs(w_axis-2*pi*60/Fs));  % buscar la frecuencia mas ...
    cercana a  2*pi*60/1000
atenuacion = 20*log10(Mag_salida(pos)/Mag_entrada(pos))
```

g. Discutir las ventajas y desventajas de las implementaciones de los incisos 2c y 2e. Considerar las atenuaciones que se obtienen en la frecuencia de rechazo y el orden de las implementaciones. Usar además los comandos `tic` y `toc` para comparar los tiempos necesarios para realizar el filtrado con ambas implementaciones.

Solución:

```
    tic
    filtrada1 = filter(b_bajos,1,emg) + filter(b_altos,1,emg);
    toc
    tic
    filtrada2 = filter(numd,dend,emg);
    toc
```

Ambas implementaciones logran atenuaciones bastante similares de alrededor de 8 dB, sin embargo, la implementación paralela 2c, al ser FIR, requiere un orden mucho mayor que la implementación IIR de 2e. En cuestión de tiempos la primera toma alrededor de 0.558 ms pero la implementación IIR toma 0.119 ms (Note que esto puede variar dependiendo

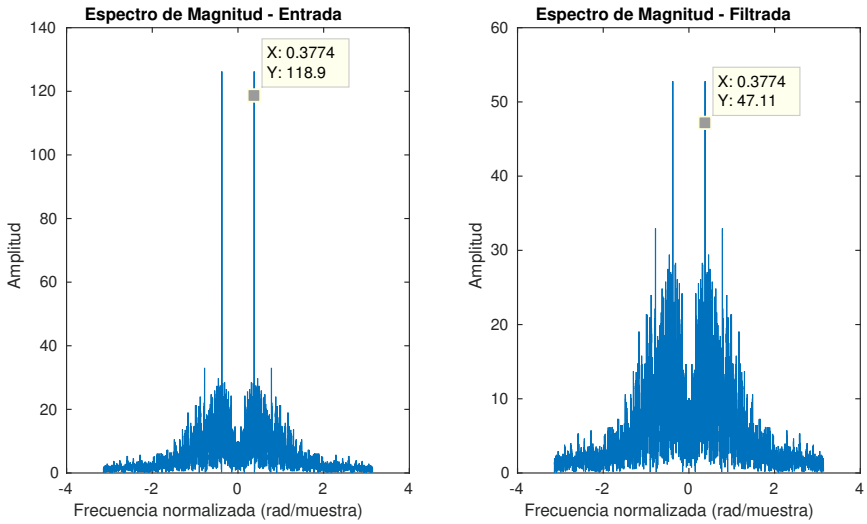


Figura 7: Espectros de magnitud de la señal de entrada y la señal filtrada.

de la plataforma en donde se ejecuta el código).

Por otro lado, se observa en la Figura 6 que la respuesta en fase alrededor de la frecuencia de rechazo es altamente no lineal, lo cual podría ser un problema dependiendo de la aplicación. Por el contrario, la Figura 4 comprueba que la implementación del inciso 2c tiene una respuesta en fase lineal.

3. El objetivo principal de los filtros analógicos es eliminar las componentes de frecuencia no deseadas. Se desea eliminar las componentes de la secuencia de entrada por encima de una determinada frecuencia normalizada. Por ello, se propone diseñar dos filtros pasabajos con frecuencia de paso y rechazo en determinadas posiciones en frecuencia normalizada.

   La secuencia $x[n]$ es la señal de entrada de un sistema con dos filtros en paralelo, definida como:

$$x[n] = \begin{cases} 1 & |n| \leq N_1 \\ 0 & N_1 < |n| < x \end{cases}$$

   El filtro FIR se diseña en base a un filtro pasabajos ideal y el filtro IIR se diseña en base a un pasabajos Butterworth. Las salidas de ambos son dos señales $y_1[n]$ y $y_2[n]$ (ver Figura 8). Ambos filtros deben tener las siguientes características:

   - $\omega_p = 0.35\pi$ rad/muestras. $H(j\omega_p) = 0.5$ dB.
   - $\omega_s = 0.50\pi$ rad/muestras. $H(j\omega_s) = -50$ dB.

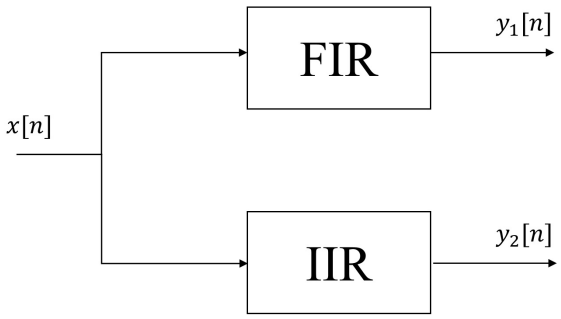


Figura 8: Sistema con dos filtros en paralelo.

a. Calcular de forma analítica la transformada de Fourier de la secuencia rectangular $x[n]$. Calcular su DFT usando MATLAB y graficar la señal en espacio de muestras, así como su espectro de magnitud y espectro de fase. Usar las funciones `abs()`, `angle()` y `unwrap()`. Considerar $N = 20$ y $N_1 = 5$.

Solución: Se define la secuencia rectangular como:
Aplicando la transformada discreta de Fourier:

$$X(e^{jw}) = \sum_{-N_1}^{N_1} (1) \cdot e^{jwN_1} = e^{jwN_1} \cdot \sum_{m=0}^{2N_1} (1) \cdot e^{-jwm}$$

$$X(e^{jw}) = \begin{cases} e^{jwN_1} \cdot (\frac{1-e^{-jw(2N_1+1)}}{1-e^{-jw}}) = \frac{\sin(\frac{w}{2}(2N_1+1))}{\sin(\frac{w}{2})} & \text{w no es múltiplo de } 2\pi \\ (2N_1+1) & \text{w es múltiplo de } 2\pi \end{cases}$$

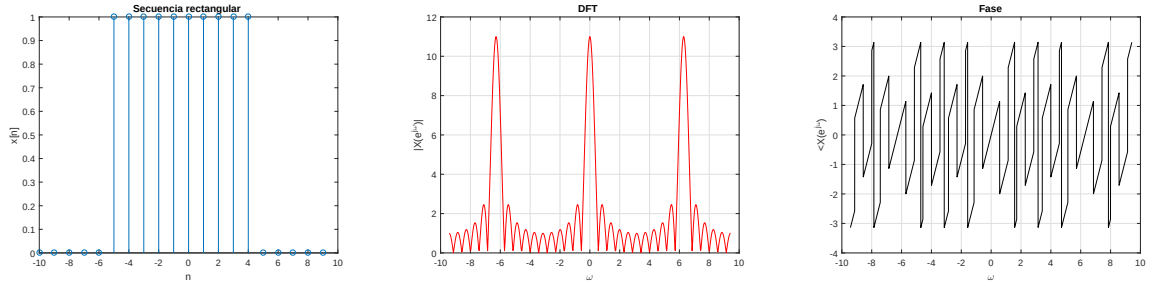
Para implementar en MATLAB, es necesario definir el número de muestras de totales de la secuencia $n = [-N/2, \cdots, N/2-1]$. Luego, se procede a definir las amplitudes de la secuencia para los espacios de muestras de 0 a $N/2$ con las funciones `ones()` y `zeros()`. La secuencia final se obtiene duplicando invertida la primera secuencia (ver Figura 9 (a)). La ecuación obtenida por la forma analítica es la que se debe implementar. Para obtener el espectro, se usa la función `abs()` (ver Figura 9 (b)). Para graficar la fase, se utilizan las funciones `angle()` y `unwrap()` (ver Figura 9 (c)). El código de MATLAB se muestra a continuación.

```
N1 = 5;
N = 20;
n = -N/2:N/2-1;                    % vector de muestras discretas
w = 2* pi*(-30: 0.1: 30)/N;

x = [ones(1,N1+1) zeros(1,N/2-N1)];
x_n = [x(end:-1:1+1) x(2:end)];    % secuencia discreta
% Del calculo analitico:
X = exp(1i*2*w).*sin((2*N1+1)*w/2)./sin(w/2);
X(mod(w,2*pi)==0)= 11;

figure(01);
subplot(311);
stem(n,x_n);xlabel('n');
ylabel('x[n]'); title('Secuencia rectangular');
```

```
16      subplot(312),
17      plot(w, abs(X),'r');
18      ylabel('|X(e^{j\omega})|'); xlabel('\omega');
19      title('DFT '); grid on;
20      subplot(313),
21      plot(w,(angle(X)),'k');
22      ylabel('<X(e^{j\omega})');
23      title('Fase'); grid on;
```



(a) (b) (c)

Figura 9: (a) Secuencia de entrada. (b) DFT de la secuencia de entrada. (c) Fase.

b. Determinar el orden y frecuencia de corte del filtro Butterworth pasabajos de forma analítica. Luego, validar el resultado usando `buttord()`.

Solución: Se sabe que el espectro de magnitud del filtro Butterworth es,

$$|H(j\omega)|^2 = \frac{1}{1+(\frac{\omega}{\omega_c})^{2N}} \tag{2}$$

Se reemplazan la frecuencia de paso y de rechazo en la ecuación (2):

$$|H(j\omega)|^2 = \frac{1}{1+(\frac{\omega_s}{\omega_c})^{2N}} = \delta_2^2 \tag{3}$$

$$|H(j\omega)|^2 = \frac{1}{1+(\frac{\omega_p}{\omega_c})^{2N}} = \delta_1^2 \tag{4}$$

Dividiendo (3)/(4), despejando N y reemplazando:

$$N = \frac{1}{2}\cdot\frac{\log(\frac{10^{0.1\delta_2}-1}{10^{0.1\delta_1}-1})}{\log(\omega_s/\omega_p)} = \frac{1}{2}\cdot\frac{\log(819539.9)}{\log(1.43)} = \frac{1}{2}\cdot\frac{5.91}{0.15} = 19.7 \approx 20 \tag{5}$$

El código de MATLAB se muestra a continuación.

```
1       Wp = 0.35;
2       Ws = 0.50;
3
4       Rp = 0.5;              %rizado en banda de paso
5       Rs = 50;               %rizado en banda de rechazo
6       %Orden del filtro Butterworth
7       [n_butt,Wn_b] = buttord(Wp,Ws,Rp,Rs);
```

c. Identificar las frecuencias de paso y rechazo del filtro analógico en radianes por segundo. Graficar la magnitud y verificar que la magnitud en dichas posiciones son coherentes con el diseño planteado en el inciso anterior. Considerar una frecuencia de muestreo de 100 Hz. Usar las funciones `butter()` y `freqs()`. Verificar si el filtro tiene una fase no lineal.

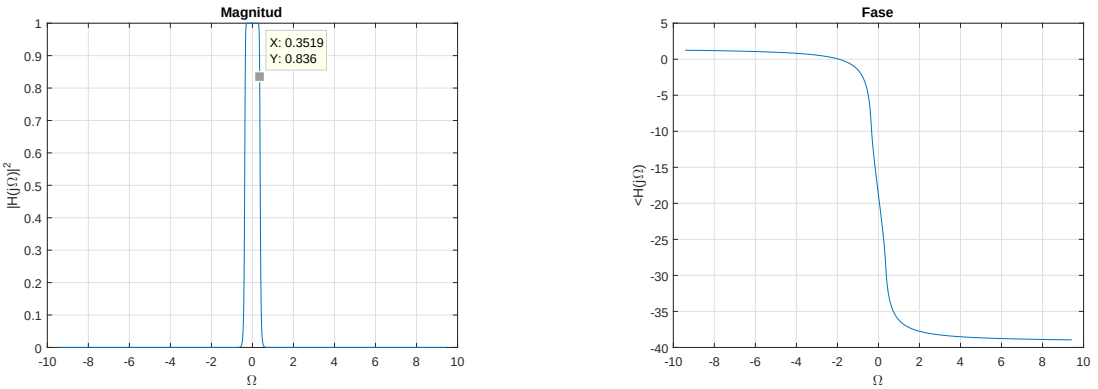
Solución: Considerando la frecuencia de muestreo de 100 Hz,la frecuencia de paso y la frecuencia de rechazo son 35 rad/segundos y 50 rad/segundos. Para graficar la magnitud del filtro, es necesario diseñar el filtro en MATLAB con la función `butter()` para obtener los coeficientes de la ecuación diferencial que caracteriza al filtro analógico. Posteriormente, se usar la función `freqs()` para obtener el filtro. A continuación, se muestra el código.

```
% Diseno del filtro Butterworth
[b1, a1] = butter(n_butt, Wn_b,'s');

N = 20;
w = 2* pi*(-30: 0.01: 30)/N;
[h1,w1] = freqs(b1,a1,w);

figure(1),
subplot(211),
plot(w1,(abs(h1))),grid on,
ylabel('|H(j\Omega)|^{2}'); xlabel('\Omega');
title('Magnitud');

subplot(212),
plot(w1,unwrap(angle(h1))),grid on,
ylabel('<H(j\Omega)'); xlabel('\Omega');
title('Fase');
```



(a) (b)

Figura 10: (a) Espectro de magnitud y (b) Fase.

d. Diseñar un filtro pasabajos FIR en base a un filtro ideal con las características mencionadas. Usar el método de enventanado. Definir el tipo de ventana, el orden del filtro y el modelo del filtro ideal $h_{ideal}[n]$.

Solución: Dado que el requerimiento del filtro es de $\delta_p = -50$ dB, los filtros Hamming y Blackman cumplen. Sin embargo, Backman obliga a usar un M mayor en comparación con Hamming.

Para el caso del diseño FIR, con w_s y w_p se puede elegir la ventana y el orden.

$$\Delta\omega = \omega_s - \omega_p = 0.15\text{rad/muestra}$$

Luego, se calcula el orden:

$$0.15\pi = \frac{8\pi}{M}$$

$M = 53.3 \approx 54$. Entonces $N = 53$. La ventana Hamming está definida como:

$$w[n] = \begin{cases} 0.54 - 0.46\cos(\frac{2n\pi}{M}) & 0 \leq n \leq 53 \\ 0 & \text{otro casos.} \end{cases}$$

Por teoría, se sabe que la frecuenca de corte $\omega_c = \frac{\omega_p + \omega_s}{2} = 0.425\pi$. Entonces, la respuesta del filtro pasabajos ideal está descrita por:

$$h_{ideal}[n] = 0.425\text{sinc}(0.425n)$$

e. Con la información del orden del filtro, modelo del filtro ideal h_{ideal} y ventana hamming, crear el vector $h[n]$. Validar esta secuencia con rutinas MATLAB. Graficar el filtro diseñado y verificar que las frecuencias de paso y rechazo cumplen con los requerimientos de diseño. Usar `fir1()`, `hamming()` y `freqz()`.

Solución: Para tener una respuesta lineal y causal, es necesario aplicar un desplazamiento de $\frac{M}{2}$ en la respuesta ideal $h_{ideal}[n]$ y multiplicarla por la ventana Hamming. Entonces, la respuesta al impulso del filtro ideal es

$$h[n] = \left[0.54 - 0.46\cos\left(\frac{2n\pi}{M}\right)\right][0.425 \cdot \text{sinc}(0.425(n-27))]$$

Para implementar el filtro en MATLAB, se presenta el siguiente código:

```
N = 53;
h1 = fir1(N, [0.425]);     % filtro FIR
w1 = hamming(length(h1));      % por metodo de enventanado Hamming

figure;
freqz(h1'.*w1, 1); title('Filtro a partir de enventanado Hamming');
```

f. Aplicar ambos filtros a la señal de entrada. Usar la función `filter()` para aplicar el filtro IIR y la función `conv()` para aplicar el filtro FIR. Truncar la señal resultante de la convolución al número de la señal de entrada. Graficar el espectro de magnitud y fase de las respuesta de ambos filtros ante la secuencia de entrada $x[n]$. Usar `fft()`, `fftshift()`, `abs()` y `angle()`. ¿Se podría implemetar el filtro IIR por convolución?. Verificar el efecto de ambos filtros en la fase de secuencia resultante y que el filtro IIR cumpla con las condiciones de computacionalmente más eficiente (menor número de muestras). Usar `tic toc` para medir el tiempo e imrpimirlo.

Solución: Se usa la función `filter()` para aplicar al filtro IIR. Para el filtro FIR, se usa `conv()`. No se podría usar `conv()` para implementar el filtro IIR. Dado que este es recursico, solo `filter()` permite incluir esta características. El tamaño de la señal es

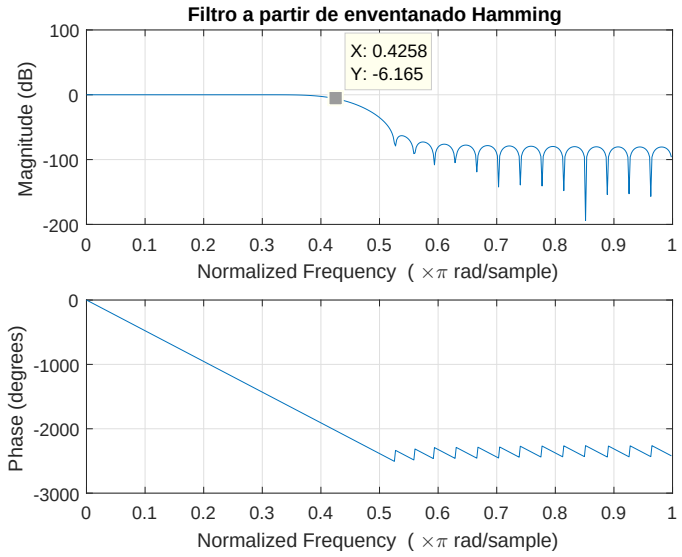


Figura 11: Gráfica del filtro.

$2L-1$ y se debe truncar a L. La función `tic toc` mide el tiempo transcurrido del cálculo y lo muestra en la línea de comandos. Se obtiene el menor tiempo computacional con el filtro IIR. El código en MATLAB es el siguiente:

```
s1 = x_n;
wd = linspace(-1, 1, length(s1));
tic
y_butt = filter(b1,a1,s1);       %salida de IIR
toc
tic
y_win = conv(s1,h1'.*w1);        %salida de FIR
toc
%Calculo del espectro de magnitudes
mag_ybut = fftshift(abs(fft(y_butt)));
mag_ywin = fftshift(abs(fft(y_win(1:length(s1))))); %trunca senal
w_but = 2*pi*(0:(length(mag_ybut)-1))/length(mag_ybut);  % ...
    frecuencia discreta (de 0 a 2pi)
w_but = unwrap( fftshift(w_but)-2*pi);    % frecuencia discreta(de ...
    -pi a pi)
w_win = 2*pi*(0:(length(mag_ywin)-1))/length(mag_ywin);  % ...
    frecuencia discreta (de 0 a 2pi)
w_win = unwrap( fftshift(w_win)-2*pi);    % frecuencia discreta(de ...
    -pi a pi)
% Graficas
figure,
subplot(121),
plot(w_win,mag_ywin,'r');
xlabel('Frecuencia (\omega)'), ylabel('amplitud'),title('Espectro ...
    de magnitd de y1'),grid on,
subplot(122),
plot(w_but,mag_ybut);
xlabel('Frecuencia (\omega)'), ylabel('amplitud'),title('Espectro ...
    de magnitud de y2'),grid on,
figure,
subplot(121),
```

```
26      plot(w_win,angle(fft(y_win(1:length(s1)))),'r');
27      xlabel('Frecuencia (\omega)'), ylabel('fase'),title('Fase de ...
            y1'),grid on,
28      subplot(122),
29      plot(w_but,angle(fft(y_butt)));
30      xlabel('Frecuencia (\omega)'), ylabel('fase'),title('Fase de ...
            y2'),grid on,
```

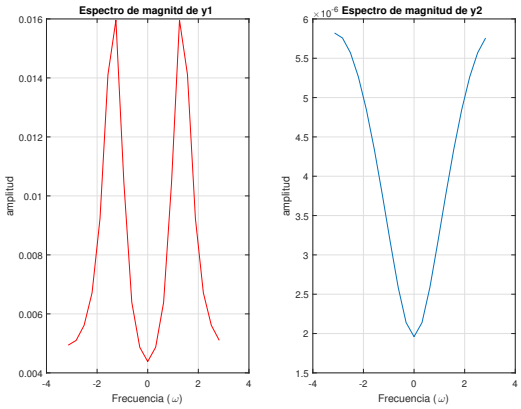


Figura 12: Magnitudes de y_1 y y_2.

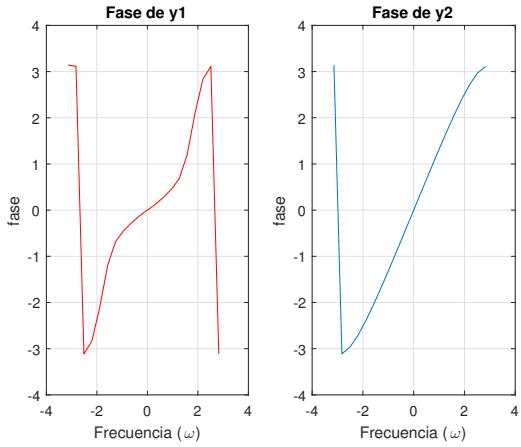


Figura 13: Fases de y_1 y y_2.

4. Se realizan pruebas mecánicas para detectar fallos en los acoples de un conjunto de resortes para la construcción de un automóvil. Este cuenta con tres resortes: dos en paralelo (k_1 y k_2), los cuales están en serie con el restante (k_3). Adicionalmente, se acopla un sensor de movimiento situado debajo del resorte en serie k_3. Debido a la fricción de las uniones, el sensor detectó una señal de 60 Hz. Asimismo, dado que la prueba no fue realizada en el vacío, se detectó ruido del ambiente. Las señales adquiridas por este sensor se describen como:

$$y_c(t) = x_c(t) + \cos(120\pi t) + \eta;$$

donde η es ruido aleatorio con distribución Gaussiana de media 0.5 y varianza 0.1^2 ($\eta \sim \mathcal{N}(0.5, 0.1^2)$) y $x_c(t)$ es la señal deseada definida como:

$$x_c(t) = \cos(100\pi t) - 2 \cdot \cos(50\pi t);$$

La señal resultante $y_c(t)$ es de 1.25 segundos de duración.

a. Graficar ambas señales en función del tiempo $t \in [0, 1.25]$ con pasos de 0.01 segundos. Rotular adecuadamente. Calcular de forma analítica la frecuencia mínima de muestreo para evitar el efecto aliasing. Usar la función `plot()`, `ylabel`, `xlabel()` y `hold on()`.

b. Digitalizar las dos señales $y_c(t)$ y $x_c(t)$ con una frecuencia de muestreo de 500 Hz. Considerar $n \in \{0, \cdots, 120\}$ Graficar ambas versiones: continua y discreta en una misma ventana con sus respectivos rótulos. Usar las funciones: `plot()`, `stem()`, `ylabel()`, `xlabel()` y `title()`. Asimismo, calcular los respectivos espectros de potencia con la siguiente ecuación. Indicar de qué tipo de señal se trata.

$$P_x = \sum_{n=-\infty}^{+\infty} |c_k|^2$$

$$c_k = \frac{1}{T_p} X\left(j\frac{2\pi k}{T_p}\right)$$

c. Calcular y graficar el espectro de la señal e identificar de la señal original. Usar `fft()`, `fftshift()`, `abs()`, `unwrap()`. Diseñar un filtro rechaza banda Butterworth que elimine la componente en frecuencia no deseada y atenúe el ruido. Considerar $H(j\omega_p) = 3$ dB y $H(j\omega_s) = -20$ dB. Usar las funciones `buttord()` y `butter()`. Indicar el orden del filtro. Graficar la señal de salida en el espacio de muestras, debidamente rotulada, al aplicar el filtro. Asimismo, graficar la señal en frecuencia, magnitud y fase con sus respectivos rótulos. Usar `filter()`, `fft()`, `fftshift()`, `abs()`, `plot()`, `ylabel()` y `xlabel()`.

d. Se cuenta con el sistema de la Figura 14, el cual filtra la señal $f_c(t)$ que corresponde al coseno de 60 Hz de frecuencia y la resta con la señal grabada. Diseñar un filtro pasa altos para obtener $f_c(t)$. Considerar $H(j\omega_p) = 3$ dB y $H(j\omega_s) = -20$ dB. Graficar la señal resultante del filtro, la señal resultante del sistema en espacio de muestras y sus espectros de magnitud y fase. Usar las funciones `buttord()`, `butter()`, `filter()`, `fft()`, `abs()`, `angle()`, `unwrap()`.

e. Diseñar un filtro Wiener pedrictivo basado en la Figura 15. La señal de entrada:

$$y_c(t) = x_c(t);$$

es contaminada por una señal $w[n]$, que es la suma de ruido blanco y una señal de 60 Hz de frecuencia. Las señales $y[n]$ y $w[n]$ son no correlacionadas y son WSS. Asimismo, $d[n]$ y $w[n]$ son no correlacionadas y son WSS. Escribir las ecuaciones de Wiener-hops para un orden de 10 y retardo de 10. Tener en cuenta lo siguiente para el código en MATLAB,

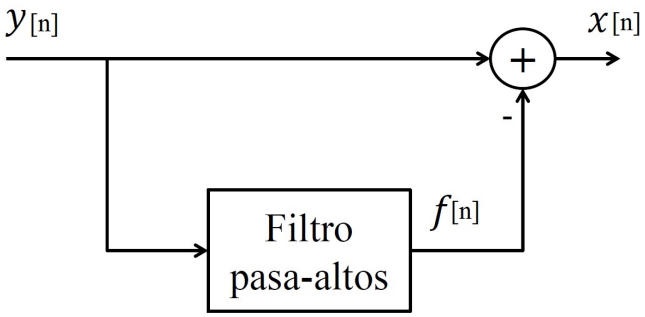


Figura 14: Diagrama de bloques del sistema.

i. Generar la matriz de autocorrelación de x[n]. Usar `xcorr()` y `toeplitz()`.

ii. Generar el vector de correlación cruzada de $x[n]$ y $d[n]$. Usar `xcorr()`.

iii. Usar `filter()` para obtener la señal de salida del filtro.

Graficar la secuencia de entrada y la secuencia sin ruido en una misma ventana, rotulando adecuadamente. Usar `hold on()`, `xlabel`, `ylabel` y `title`.

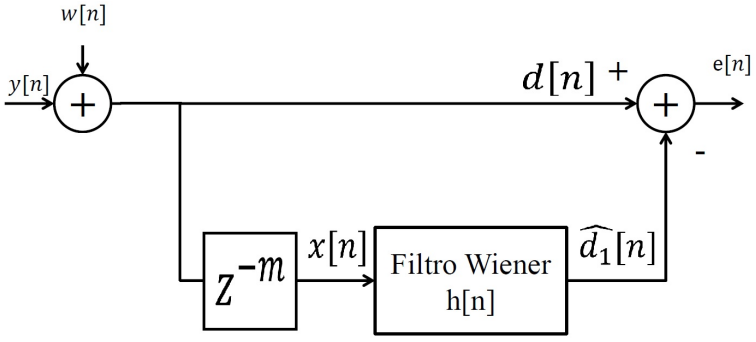


Figura 15: Eliminación de ruido blanco usando filtro Wiener.

f. Analizar la precisión de filtro. Para ello, considerar los siguientes órdenes: 5, 10 y 20. ¿Se obtiene mejor respuesta con algún orden en particular?. En comparación con los filtros implementados en los incisos previos, ¿qué diferencias encuentra?, ¿con cuál de ellos se obtiene un mejor resultado?.