PROCESAMIENTO DIGITAL DE SEÑALES

PROFESOR: HUMBERTO LOAIZA C., Ph.D.

LABORATORIO No. 6 (Software)

Filtros Digitales

Filtros con Respuesta Impulsional Finita - Filtros con Respuesta Impulsional Infinita

I. Objetivos

- *Diseñar filtros digitales FIR utilizando las técnicas de Ventanas, Muestreo en Frecuencia y el algoritmo Min-Max, Remez.*
- *Diseñar filtros digitales IIR utilizando las técnicas de transformación bilineal, invarianza impulsional y el método de Prony (mínimos cuadrados + Padé)*
- *Implementar a partir de filtros paso bajo, filtros paso altos digitales.*
- *Efectuar un análisis comparativo de las respuestas en magnitud y fase de filtros digitales FIR e IIR.*

II. Introducción

1. Diseño de filtros

Los filtros digitales se utilizan ampliamente en el tratamiento digital de señales provenientes de diversas aplicaciones, que incluyen tratamiento y transmisión de señales de voz, datos, vídeo e imágenes, sonar, radar, sísmicas y de exploración petrolera, entre otras. Los filtros LTI son los que se utilizan frecuentemente debido a la simplicidad para analizar, diseñar e implementar.

Un filtro digital LTI puede identificarse de forma única en el dominio tiempo/espacio por su respuesta impulsional $h(n)$, (donde, n es un índice entero que se utiliza como variable independiente). De forma alterna, los filtros digitales pueden caracterizarse de forma única en el dominio frecuencial por su respuesta en frecuencia $H(w)$ (donde, w variable de frecuencia de valor real en radianes), la cual también corresponde a la Transformada de Fourier en Tiempo Discreto de la secuencia $h(n)$. Los filtros digitales LTI son de dos tipos: Filtros FIR (Finite-duration Impulse Response, para los cuales la respuesta impulsional h(n) es diferente de cero para un número finito de muestras; y filtros IIR (Infinite-duration Impulse Response) cuya respuesta impulsional presenta un número infinito de muestras diferentes de cero. En el caso FIR las muestras de la secuencia $h(n)$ corresponden directamente a los coeficientes de los filtros; para el caso IIR, los parámetros encontrados corresponden a los coeficientes de una ecuación de diferencia que incluye términos de retroalimentación.

El diseño de filtros digitales ha tomado mucha importancia desde las últimas tres décadas. El diseño y realización de filtros digitales involucra un combinación de teorías, aplicaciones y tecnologías. En muchas aplicaciones, el objetivo es diseñar filtros selectivos en frecuencia que alteran o dejan sin modificación diferentes componentes frecuenciales. En estos casos, las especificaciones de diseño se dan en el dominio de frecuencia especificando una respuesta en frecuencia deseada $H_D(f)$. En general, $H_D(f)$ es de valor complejo, constituida por una respuesta

en magnitud deseada $|H_D(f)|$ y una respuesta en fase deseada $\angle H_D(f)$. Uno de los problemas más importantes es el diseño de un filtro digital altamente selectivo en frecuencia con transiciones muy cortas entre las bandas de paso y de rechazo. Sin embargo, estas características corresponden matemáticamente a discontinuidades y no tienen implementación práctica. Por lo tanto, el problema de diseño de filtros digitales consiste en encontrar un filtro implementable del menor orden posible y cuya respuesta en frecuencia $H(f)$ sea lo más aproximada a la respuesta deseada $H_D(f)$.

El diseño de filtros digitales se realiza típicamente en tres pasos:

i. Convertir las restricciones de diseño deseadas en especificaciones precisas sobre las respuestas de magnitud y fase deseadas, tipo de filtro (FIR o IIR), orden del filtro, error tolerable o criterio de error.
ii. Encontrar un filtro FIR o IIR que se aproxime a las especificaciones de diseño del paso i, tal que la respuesta en frecuencia del filtro obtenido se ajuste mejor a esas especificaciones de diseño de acuerdo con un criterio de error matemático.
iii. Realizar el filtro usando la tecnología digital más adecuada para la aplicación bajo consideración.

El paso *i* es altamente dependiente de la aplicación y de los detalles suministrados por el usuario. El paso *ii* se realiza utilizando métodos de aproximación y optimización matemáticos. El paso *iii* depende de la tecnología hardware y software utilizada para construir el filtro.

2. Normalización de frecuencia en el Toolbox de Procesamiento de Señales

Todas las funciones del Signal Processing Toolbox operan con frecuencias normalizadas por defecto, por lo que estas funciones no requieren de argumentos de entrada adicionales para especificar la frecuencia de muestreo del sistema. En este Toolbox se adopta la frecuencia unidad como la *frecuencia de Nyquist*, definida como la mitad de la frecuencia de muestreo. Por lo tanto, la frecuencia normalizada se expresa en el rango de $0 \leq fa \leq 1$.

Por ejemplo, para un sistema con frecuencia de muestreo de 1000 Hz, su frecuencia Nyquist es 500. Con estos datos, la frecuencia de 300 Hz equivale en frecuencia normalizada a 300/500=0.6.

Para convertir de frecuencia normalizada en frecuencia angular alrededor del círculo unitario solo basta por multiplicarla por π. Para convertir la frecuencia normalizada en Hertz, solo es necesario la multiplicación por la mitad de la frecuencia de muestreo.

III. Funciones prácticas de Matlab

1. Funciones para análisis de filtros

abs: Valor absoluto – magnitud-
angle: Cálculo de la fase
freqs: Respuesta en frecuencia de filtros análogos

freqspace: Genera el espaciamiento en frecuencia(eje horizontal) para respuestas en frecuencias
freqz: Respuesta en frecuencia de filtros digitales
freqzplot: Gráfico de la respuesta en frecuencia de datos
grpdelay: Calcula el retardo promedio del filtro (retardo de grupo)
impz: Calcula la respuesta impulsional de filtros digitales
unwrap: Corrige ángulos de fase adicionando múltiplos de $\pm 2\pi$
zplane: Gráfico de polos y ceros

2. Funciones para la implementación de filtros

conv: Convolución y multiplicación de polinomios
conv2: Convolución de dos dimensiones
deconv: De-convolución y división de polinomios
fftfilt: FFT basada en el filtrado FIR que usa el método overlap-add
filter: Aplicación de filtrado digital con técnicas recursiva (IIR) o no-recursiva (FIR)
filter2: Filtrado digital en dos dimensiones
filtfilt: Filtrado digital de fase cero
filtic: Encuentra condiciones iniciales para implementación de filtros en forma directa II transpuesta
latcfilt: Implementación de filtros en celosías y celosías-escalera (lattice, lattice-ladder)
medfilt1: Filtro mediana unidimensional
sgolayfilt: filtro Savitzky-Golay
sosfilt: Filtro digital IIR de segundo orden (bicuadrático)
upfirdn: Aplica las operaciones de interpolación, filtrado con FIR, y diezmado

3. Diseño de filtros digitales FIR

convmtx: Convolución con matrices
cremez: Diseño de filtros FIR equi-rizado de fase no-lineal y complejo
fir1: Diseña un filtro FIR mediante el método de enventanado
fir2: Diseña un filtro FIR mediante el método de muestreo en frecuencia
fircls: Diseña un filtro FIR por mínimos cuadrados para filtros multibanda
fircls1: Igual que el anterior, pero para paso-bajos y paso-altos de fase lineal.
firls: Diseño de filtros de fase lineal por mínimos cuadrados
firrcos: Diseño de filtros FIR de fase lineal con banda de transición coseno en relieve
intfilt: Diseño por interpolación de filtros FIR de F-L
kaiserord: Estima parámetros de un filtro FIR diseñado con ventana Kaiser
remez: Calcula el filtro óptimo FIR mediante Parks-McClellan
remezord: Estimación del orden óptimo de filtros según Parks-McClellan
sgolay: Diseño de filtros Savitzky-Golay de alisamiento

4. Diseño de filtros IIR

butter: Diseño de filtros Butterworth analogos y digitales
cheby1: Diseño de filtros Chebyshev Tipo I (Banda de paso con rizado)

cheby2: Diseño de filtros Chebyshev Tipo II (Banda de rechazo con rizado)
ellip: Diseño de filtros Elípticos (Cauer)
maxflat: Diseño de filtros digitales Butterworth generalizados
prony: Diseño de filtros IIR por el método de Prony
stmcb: Cálculo de modelo lineal usando la iteración de Steiglitz-McBride
yulewalk: Diseño de filtros digitales recursivos

5. Estimación del orden de filtros IIR

buttord: Calcula el orden y frecuencia de corte de un filtro Butterworth
cheb1ord: Calcula el orden para un filtro Chebyshev Tipo I
cheb2ord: Calcula el orden para un filtro Chebyshev Tipo II
ellipord: Calcula el mínimo orden para filtros elípticos

6. Transformación de filtros analógicos

lp2bp: Transforma filtros análogos paso-bajos a banda de paso
lp2bs: Transforma filtros análogos paso-bajos a banda de rechazo
lp2hp: Transforma filtros análogos paso-bajos a paso-altos
lp2lp: Cambia la frecuencia de corte de un filtro análogo paso-bajo

7. Discretización de filtros

bilinear: Transformación Bilineal para convertir filtros análogos a digital
impinvar: Método de invarianza impulsional para convertir filtros análogos a digital

8. Parámetros para el diseño de filtros por Enventanado

barthannwin: Calcula los parámetros de la ventana Bartlett-Hann
bartlett: Calcula los parámetros de la ventana Bartlett
blackman: Calcula los parámetros de la ventana Blackman
blackmanharris: Calcula los parámetros de la ventana Blackman-Harris
bohmanwin: Calcula los parámetros de la ventana Bohman
chebwin: Calcula los parámetros de la ventana Chebyshev
gausswin: Calcula los parámetros de la ventana Gaussian
hamming: Calcula los parámetros de la ventana Hamming
hann: Calcula los parámetros de la ventana Hann (Hanning)
kaiser: Calcula los parámetros de la ventana Kaiser window
nuttallwin: Calcula los parámetros de la ventana Nuttall-defined /Blackman-Harris
rectwin: Calcula los parámetros de la ventana rectangular
triang: Calcula los parámetros de la ventana triangular
tukeywin: Calcula los parámetros de la ventana Tukey
window: Función pasarela para calcular los parámetros de diferentes ventanas

IV. Procedimiento

Todos los programas deben presentarse funcionando frente al grupo y frente al profesor de la asignatura.

1. Señal de prueba

1.1. Obtenga una señal discreta x(n) como combinación lineal de varias ondas seno de distinta frecuencia de la forma,

$$x(n)= A_0 + A_1 \text{ sen } (W_1\, nT) + A_2 * \cos(W_2\, nT) - A_3 * \text{ sen}(W_3\, n\, T)$$

donde A_i son constantes, $W_i = 2\,\pi\, F_i$ es frecuencia de las señales en el dominio temporal, T es el periodo de muestreo y n es el instante de muestreo.

1.2. Asigne valores a los coeficientes A_i y W_i (altas y bajas frecuencias, tal que $W_1 << W_2 << W_3$) para calcular varias señales $x_i(n)$. Tenga en cuenta el teorema de muestreo para seleccionar T. El tamaño del vector x(n) debe ser tal que contenga por lo menos un periodo del componente de la onda cosenoidal de menor frecuencia.

i. $x_1(n) \Rightarrow A_0= 5,\ A_1= 5,\ A_2= 2.5,\ A_3= 10;\ W_1=2\pi\ 10,\ W_2=2\pi\ 100,\ W_3=2\pi\ 1000$
ii. $x_2(n) \Rightarrow A_0= 10,\ A_1= 15,\ A_2= 30,\ A_3= 15;\ W_1=2\pi\ 100,\ W_2=2\pi\ 2000,\ W_3=2\pi\ 5000$
iii. $x_3(n) \Rightarrow A_0= 100,\ A_1= 30,\ A_2= 30,\ A_3= 150;\ W_1=2\pi\ 2500,\ W_2=2\pi\ 3500,\ W_3=2\pi\ 5000$

1.3. Implemente una función en Matlab para visualizar en una misma ventana la señal x(n), su Transformada de Fourier [*fft()*] y su diagrama de frecuencia [ver función guía No. 5 *analyze()*].Consigne el código del programa.

1.4. Pruebe el programa anterior con las señales de §1.2. Consigne, analice e interprete los resultados visualizados. A qué se deben las diferencias entre los gráficos resultantes de las funciones *fft()* y *analyze()* ?

2. Filtros FIR

2.1. Diseño por ventanas

2.1.1. El siguiente script permite visualizar la respuesta de un filtro FIR pasabanda con $0.35 \le w \le 0.65$ de orden N=48 con una ventana Hamming de longitud N+1.

```
b=fir1(48, [0.35 0.65]);
freqz(b, 1, 512 )
Legend('Respuesta en Frecuencia del Filtro Obtenido');
```

2.1.2. Usando el método de ***diseño por ventanas***, implemente analíticamente ***dos*** filtros ***paso bajo*** para eliminar los componentes de $x_i(n)$ con frecuencias superiores a Fi > 2000Hz. Las longitudes de los dos filtros deben ser diferentes. Recuerde que existe una relación entre las frecuencias en el dominio temporal, $W=2\pi F$, y la frecuencia en el dominio

frecuencial; $w=2\pi f$. Consigne el procedimiento de diseño y las funciones h(n) y H(w) del filtro obtenido. Utilice al menos una ventana diferente a la rectangular.

2.1.3. Con base en el script anterior, realice una función en Matlab para visualizar en una misma ventana la respuesta impulsional, los espectros de tensión y fase de los filtros obtenidos en el numeral anterior. Consigne el código del programa y explique el algoritmo utilizado. Incluya los resultados.
Emplee las funciones ***fir1*** (analice la diferencia entre las opciones 'noscale' y 'scale').

2.1.4. En los espectros de fase y tensión del numeral anterior, mida la magnitud y el ancho de los lóbulos principales en cada banda (banda de paso, transición y de rechazo), y verifique que la frecuencia de corte satisface el requerimiento de §2.1.1. Analice los resultados y consígnelos en una tabla.

2.1.5. Implemente una función en Matlab para aplicar a las señales $x_i(n)$ del numeral §1.2 los filtros paso bajo obtenidos en §2.1.1. El programa debe visualizar en una misma ventana las señales x(n) y X(w) *antes* y *después* del filtrado, junto con sus diagramas de frecuencia (resultado de analyze). Consigne el código del programa y explique el algoritmo utilizado.

2.1.6. Consigne, analice e interprete los resultados obtenidos en §2.1.4. Son los resultados coherentes con la teoría? Justifique.

2.2. Diseño por muestreo en Frecuencia

2.2.1. El siguiente script permite visualizar la respuesta de un filtro FIR pasabajo junto con la respuesta deseada en frecuencia. El orden del filtro es N=30.

```
f=[0 0.6 0.6 1];  m=[1 1 0 0];
N=30
b=fir2(N,f,m);
[h,w]= freqz(b, 1, 128);
plot(f,m,w/pi, abs(h));
legend('Filtro Ideal','Filtro Obtenido');
title('Comparacion de la Magnitud de la Respuesta en Frecuencia');
xlabel(frecuencia); ylabel('amplitud');
```

2.2.2. Utilizando el método de ***diseño por muestreo en frecuencia***, implemente analíticamente ***dos*** filtros ***paso bajo*** para eliminar los componentes de x(n) con frecuencias superiores a Fi > 2000Hz. Las longitudes de los dos filtros deben ser iguales a los seleccionados en §2.1.1. Recuerde que existe una relación entre las frecuencias en el dominio temporal, $W=2\pi F$, y la frecuencia en el dominio frecuencial; $w=2\pi f$. Consigne el procedimiento de diseño y las funciones h(n) y H(w) del filtro obtenido.

2.2.3. Con base en el script anterior, realice una función en Matlab para visualizar en una misma ventana la respuesta impulsional, los espectros de tensión y fase de los filtros obtenidos en el numeral anterior. Consigne el código del programa y explique el algoritmo utilizado. Incluya los resultados.

2.2.4. En los espectros de fase y tensión del numeral anterior, mida la magnitud de los lóbulos principales en cada banda (banda de paso, transición y de rechazo) y verifique que la frecuencia de corte satisface el requerimiento de §2.2.1. Analice los resultados y consígnelos en una tabla.

2.2.5. Implemente una función en Matlab para aplicar a las señales $x_i(n)$ del numeral §1.2 los filtros paso bajo obtenidos en §2.2.1. El programa debe visualizar en una misma ventana las señales x(n) y X(w) *antes* y *después* del filtrado, junto con sus diagramas de frecuencia (resultado de analyze). Consigne el código del programa y explique brevemente el algoritmo utilizado.

2.2.6. Consigne, analice e interprete los resultados obtenidos en §2.2.4. Son los resultados coherentes con la teoría? Justifique.

2.3. Diseño por criterio de min-max

2.3.1. El siguiente script permite calcular la respuesta en frecuencia de un filtro paso-banda de orden N=17. El programa visualiza la respuesta deseada y obtenida según el algoritmo de Remez (Parks-McClellan, Chebyshev).

```
f= [0  0.3  0.4  0.6  0.7 1 ]; a=[0 0 1 1 0 0];
N=17;
b=remez(N, f, a);
[h,w]=freqz(b,1,152);
plot(f,a, w/pi, abs(h));
legend('Filtro Deseado', 'Filtro Obtenido');
xlabel('frecuencia'); ylabel('amplitud');
```

2.3.2. Usando el método de ***diseño por criterio min-max***, implemente analíticamente ***dos*** filtros ***paso bajo*** para eliminar los componentes de x(n) con frecuencias superiores a Fi > 2000Hz. Las longitudes de los dos filtros deben ser iguales a los seleccionados en §2.1.1. Recuerde que existe una relación entre las frecuencias en el dominio temporal, $W=2\pi F$, y la frecuencia en el dominio frecuencial; $w=2\pi f$. Consigne el procedimiento de diseño y las funciones h(n) y H(w) del filtro obtenido.

2.3.3. Con base en el script anterior, realice una función en Matlab para visualizar en una misma ventana la respuesta impulsional, los espectros de tensión y fase de los filtros obtenidos en el numeral anterior. Consigne el código del programa y explique el algoritmo utilizado. Incluya los resultados.

2.3.4. En los espectros de fase y tensión del numeral anterior, mida la magnitud y el ancho de los lóbulos principales en cada banda (banda de paso, transición y de rechazo) y verifique que la frecuencia de corte satisface el requerimiento de §2.3.1. Analice los resultados y consígnelos en una tabla.

2.3.5. Implemente una función en Matlab para aplicar a las señales $x_i(n)$ del numeral §1.2 los filtros paso bajo obtenidos en §2.3.1. El programa debe visualizar en una misma ventana

las señales x(n) y X(w) *antes* y *después* del filtrado, junto con sus diagramas de frecuencia (resultado de "analyze": aplicar la transformada inversa a la señal temporal y graficar su magnitud). Consigne el código del programa y explique el algoritmo utilizado.

2.3.6. Consigne, analice e interprete los resultados obtenidos en §2.3.4. Son los resultados coherentes con la teoría? Justifique.

2.4. Comparación

Realice una tabla comparativa de la calidad de los espectros de tensión y del proceso de filtrado efectuado por los filtros FIR obtenidos con los tres métodos de diseño anteriores. Justifique las respuestas. Tenga presente:

- Complejidad en el cálculo teórico del filtro
- Precisión en la frecuencia de corte
- Oscilaciones en cada banda de los filtros (banda de paso, de rechazo y de transición)
- Ancho de las bandas (de paso de transición y de rechazo)
- Linealidad del espectro de fase
- Dificultad en la implementación computacional
- Tiempo de ejecución

2.5. Filtros Paso Altos FIR

2.5.1. A partir de los filtros obtenidos en los numerales §2.1, §2.2 y §2.3 genere filtros paso alto con frecuencias de corte de 2KHz. Consigne el procedimiento.

2.5.2. Repita los numerales §2.1 al §2.4. pero empleando los filtros paso alto.

3. Filtros IIR

3.1. Diseño por transformación bilineal

3.1.1. Usando el método de ***transformación bilineal***, implemente analíticamente ***dos*** filtros ***paso bajo*** para eliminar los componentes de x(n) con frecuencias superiores a Fi > 2000Hz. Los órdenes de los dos filtros deben ser diferentes. Recuerde que existe una relación entre las frecuencias en el dominio temporal, $W=2\pi F$, y la frecuencia en el dominio frecuencial; $w=2\pi f$. Consigne el procedimiento de diseño y las funciones h(n) y H(w) del filtro obtenido.

3.1.2. Realice una función en Matlab para visualizar en una misma ventana la respuesta impulsional, los espectros de tensión y fase de los filtros obtenidos en el numeral anterior. Utilice la función ***bilinear*** de Matlab. Consigne el código del programa y explique el algoritmo utilizado. Incluya los resultados.

3.1.3. En los espectros de fase y tensión del numeral anterior, mida la magnitud y el ancho de los lóbulos principales en cada banda (banda de paso, transición y de rechazo) y verifique

que la frecuencia de corte satisface el requerimiento de §3.1.1. Analice los resultados y consígnelos en una tabla.

3.1.4. Implemente una función en Matlab para aplicar a las señales $x_i(n)$ del numeral §1.2 los filtros paso bajo obtenidos en §3.1.1. El programa debe visualizar en una misma ventana las señales x(n) y X(w) *antes* y *después* del filtrado, al igual que el diagrama de frecuencia (resultado de analyze). Consigne el código del programa y explique el algoritmo utilizado.

3.1.5. Consigne, analice e interprete los resultados obtenidos en §3.1.4. Son los resultados coherentes con la teoría? Justifique.

3.2. Diseño por el Método de Prony

3.2.1. [**Consulta**] Usando el método de ***Prony***, implemente analíticamente ***dos*** filtros ***paso bajo*** para eliminar los componentes de x(n) con frecuencias superiores a Fi > 2000Hz. Los órdenes de los dos filtros deben ser iguales a los seleccionados en el numeral §3.1.1. Recuerde que existe una relación entre las frecuencias en el dominio temporal, $W=2\pi F$, y la frecuencia en el dominio frecuencial; $w=2\pi f$. Consigne el procedimiento de diseño y las funciones h(n) y H(w) del filtro obtenido.

3.2.2. Realice una función en Matlab para visualizar en una misma ventana la respuesta impulsional, los espectros de tensión y fase de los filtros obtenidos en el numeral anterior. Consigne el código del programa y explique el algoritmo utilizado. Incluya los resultados.

3.2.3. En los espectros de fase y tensión del numeral anterior, mida la magnitud y el ancho de los lóbulos principales en cada banda (banda de paso, transición y de rechazo) y verifique que la frecuencia de corte satisface el requerimiento de §3.2.1. Analice los resultados y consígnelos en una tabla.

3.2.4. Implemente una función en Matlab para aplicar a las señales $x_i(n)$ del numeral §1.2 los filtros paso bajo obtenidos en §3.2.1. El programa debe visualizar en una misma ventana las señales x(n) y X(w) *antes* y *después* del filtrado, al igual que sus diagramas de frecuencia (resultado de analyze). Consigne el código del programa y explique el algoritmo utilizado.

3.2.5. Consigne, analice e interprete los resultados obtenidos en §3.2.4. Son los resultados coherentes con la teoría? Justifique.

3.3. Diseño por Invarianza Impulsional

3.3.1. Usando el método de ***invarianza impulsional*** implemente analíticamente ***dos*** filtros ***paso bajo*** para eliminar los componentes de x(n) con frecuencias superiores a Fi > 2000Hz. Los órdenes de los dos filtros deben ser iguales a los seleccionados en el numeral §3.1.1. Recuerde que existe una relación entre las frecuencias en el dominio temporal, $W=2\pi F$, y

la frecuencia en el dominio frecuencial; $w=2\pi f$. Consigne el procedimiento de diseño y las funciones h(n) y H(w) del filtro obtenido.

3.3.2. Realice una función en Matlab para visualizar en una misma ventana la respuesta impulsional, los espectros de tensión y fase de los filtros obtenidos en el numeral anterior. Consigne el código del programa y explique el algoritmo utilizado. Utilice la función ***impinvar*** de Matlab. Incluya los resultados

3.3.3. En los espectros de fase y tensión del numeral anterior, mida la magnitud y el ancho de los lóbulos principales en cada banda (banda de paso, transición y de rechazo) y verifique que la frecuencia de corte satisface el requerimiento de §3.3.1. Analice los resultados y consígnelos en una tabla.

3.3.4. Implemente una función en Matlab para aplicar a las señales $x_i(n)$ del numeral §1.2 los filtros paso bajo obtenidos en §3.3.1. El programa debe visualizar en una misma ventana las señales x(n) y X(w) *antes* y *después* del filtrado, al igual que sus diagramas de frecuencia (resultado de analyze). Consigne el código del programa y explique el algoritmo utilizado.

3.3.5. Consigne, analice e interprete los resultados obtenidos en §3.3.4. Son los resultados coherentes con la teoría? Justifique.

3.4. Comparación

Realice una tabla comparativa de la calidad de los espectros de tensión y del proceso de filtrado efectuado por los filtros IIR obtenidos con los tres métodos de diseño anteriores. Justifique sus respuestas. Tenga presente:

- Complejidad en el cálculo teórico del filtro
- Precisión en la frecuencia de corte
- Oscilaciones en cada banda de los filtros (banda de paso, de rechazo y de transición)
- Ancho de las bandas (de paso de transición y de rechazo)
- Linealidad del espectro de fase
- Dificultad en la implementación computacional
- Tiempo de ejecución

3.5. Filtros Paso Altos IIR

3.5.1. A partir de los filtros obtenidos en los numerales §3.1 y §3.2 genere filtros paso alto con frecuencias de corte de 2KHz. Consigne el procedimiento.

3.5.2. Repita los numerales §3.1 al §3.3. pero empleando los filtros paso alto.

4. Herramienta de Diseño y Análisis de Filtros: FDATool

La herramienta ***fdatool*** de Matlab permite utilizar las funciones de diseño de filtros en un sólo ambiente integrado con funciones de visualización y alto grado de interacción para usuarios con conocimientos de las técnicas de procesamiento digital de señales. La herramienta se invoca desde la línea de comandos de Matlab y permite analizar en corto tiempo las características temporales y frecuenciales de filtros digitales FIR e IIR.

4.1. Filtros FIR

4.1.1. Utilice la herramienta fdatool y calcule los filtros paso-bajo con los mismos parámetros y características desarrollados en las secciones §2.1 a §2.3. Indique a que se deben las diferencias encontradas.

4.1.2. Utilice la herramienta fdatool y calcule los filtros paso-alto con los mismos parámetros y características desarrollados en la sección §2.5. Indique a que se deben las diferencias encontradas.

4.2. Filtros IIR

4.2.1. Utilice la herramienta fdatool y calcule los filtros paso-bajo con los mismos parámetros y características desarrollados en las secciones §3.1 a §3.3. Indique a que se deben las diferencias encontradas.

4.2.2. Utilice la herramienta fdatool y calcule los filtros paso-alto con los mismos parámetros y características desarrollados en la sección §3.5. Indique a que se deben las diferencias encontradas.

5. Herramienta de Procesamiento de Señales: SPTool

La herramienta ***sptool*** cuenta con una interface gráfica para el procesamiento digital de señales, que permite:

- Analizar señales
- Diseñar filtros
- Analizar visualmente los filtros
- Filtrar señales
- Analizar el espectro de señales

La herramienta, que se invoca desde la línea de comandos de Matlab, cuenta con tres interfaces gráficas:

i- *Signal Browser*: permite visualizar, analizar y escuchar señales creadas en Matlab.

ii- *Filter Designer and Viewer*: permite generar y analizar filtros digitales que se pueden aplicar sobre las señales seleccionadas del Signal Browser.

iii- *Spectrum Viewer*: permite estimar y analizar la densidad espectral de potencia (PSD) de una señal, calculada con diferentes métodos.

5.1. Desde el ambiente de Matlab capture con ayuda del micrófono las voces de una persona masculina y una femenina. Las señales deben ser muestreadas a 8000Hz, 8 bits, durante 3 segundos.

5.2. Invoque la herramienta SPTool desde la línea de comandos de Matlab e importe las señales grabadas en el numeral anterior.

Para ello seleccione la opción **File** del menú principal de SPTool y active la casilla **From Workspace**. Seleccione la variable que contiene la señal y haga click en la flecha para que aparezca en el campo **Data**. Llene adecuadamente los campos **Import As** (Signal) y el campo **Sampling Frecuency** (debe corresponder a la frecuencia de muestreo de la señal de entrada) y asigne un nombre a la señal en el campo **Name**. Repita el procedimiento para las demás señales.

Para verificar si las señales han sido importadas selecciónelas y visualícelas con el botón **View** de la interface **Signals** del SPTool.

5.3. Para cada una de las señales obtenidas en §5.1 obtenga la densidad espectral de potencia PSD. Para ello seleccione la señal en la interface **Signals** y oprima el botón **Create** de la interface **Spectra** del SPTool. Sobre la ventana de **Spetrum Viewer** seleccione sobre *Parameters* dentro del campo **Method** FFT y del campo **Nfft** el número de datos del espectro a visualizar.

5.4. Para cada señal mida el ancho de banda y determine las frecuencias de mayor potencia espectral.

5.5. Regrese al menú principal del SPTool y oprima el botón **New Design** dentro de la interface **Filters**.

5.6. Diseñe un filtro FIR paso bajo y un FIR paso alto con la técnica de enventanado usando Kaiser. La frecuencia de corte debe seleccionarse aproximadamente en la mitad del ancho de banda de las señales de voz. Asígnele nombres a los filtros que relacionen el tipo y método de diseño. Consigne los parámetros y coeficientes de los filtros.

5.7. Usando el SPTool, aplique cada uno de los filtros a las señales de voz. Para las señales filtradas, visualícelas en el dominio del tiempo, escúchelas y obtenga sus espectros. Consigne y analice los resultados antes y después del filtrado.

5.8. Diseñe un filtro IIR paso bajo y un IIR paso alto a partir de un filtro Butterworth. La frecuencia de corte debe seleccionarse aproximadamente en la mitad del ancho de banda de las señales de voz. Asígnele nombres a los filtros que relacionen el tipo y método de diseño. Consigne los parámetros y coeficientes de los filtros.

5.9. Usando el SPTool, aplique cada uno de los filtros a las señales de voz. Para las señales filtradas, visualícelas en el dominio del tiempo, escúchelas y obtenga sus espectros. Consigne y analice los resultados antes y después del filtrado.

6. Informe

6.1. Presente un informe escrito claro en donde se consigne los programas, las figuras, las respuestas, las justificaciones y los resultados obtenidos. También incluya el análisis e interpretación de los resultados. Igualmente consigne conclusiones, observaciones, y la literatura consultada. Utilice en el informe la *misma numeración* de la guía de laboratorio.

6.2. Elabore información de ayuda para cada una de las funciones implementadas. Estas ayudas deberán consultarse con el comando **help** de Matlab.

Nota:

a) La omisión de alguno de los ítems en el informe representa una disminución de la nota.

b) Para facilitar la sustentación ante el profesor, realice scripts donde se definan los datos y se invoquen las funciones desarrolladas.