

## Trabajo práctico: Filtrado digital IIR

### 1) Filtro Leaking Integrator (LI) con señales senoidales en MATLAB

- Genere una señal senoidal con frecuencia fundamental de 100Hz.
- Agregue ruido a la señal senoidal tal que la relación señal a ruido entre la señal senoidal y la señal con ruido sea de 15 dB.
- Diseñe un filtro *leaking integrator* (LI) con  $\lambda$  igual a 0.7.
- Grafique la respuesta en frecuencia y fase del filtro LI. Use la función `freqz()`. Determine la frecuencia de corte  $f_{co}$  con:
$$f_{co} = -\ln(\lambda) \cdot f_s / \pi$$
- Determine el cero y el polo del filtro con la función `zplane()`. ¿Es el filtro estable?.
- Aplique el filtro LI a la señal con ruido. Utilice la función `filter()`.
- Grafique la respuesta en el tiempo de las señales original y filtrada y compare.
- Grafique la respuesta en frecuencia de las señales original y filtrada y compare. Utilice la función provista `my_dft()`.
- Repita los puntos c) a h) para  $\lambda$  igual a 0.9 y 0.98. Analice el comportamiento de la  $f_{co}$ .

### 2) Filtro IIR en el dominio de la frecuencia con señales de audio en MATLAB

- Cargue el archivo de audio provisto llamado `Tchaikovsky.mat`.

```
> load Tchaikovsky.mat
```

Se cargarán dos variables, la matriz `signal` con dos canales (estéreo) y la variable `Fs`. Elija 1 de los 2 canales disponibles.

- Use la herramienta `filterDesigner` para diseñar un filtro IIR:

- Pasa-banda.
- Filtro Elliptic, orden 6.
- Frecuencias de corte de 300 Hz y 3400 Hz (canal telefónico).
- Frecuencia de muestreo 44100 Hz.
- Astop: 60 dB
- Apass: 0.5 dB

b) Aumente el orden del filtro a 12. ¿Se modifica la respuesta en frecuencia del filtro?.

c) Exporte el filtro del punto b) haciendo `File > Generate MATLAB Code > Filter Design Function`. Nombre la función como `iir_elliptic_300_3400.m`.

d) Utilice como señal de entrada el archivo `Tchaikovsky.mat`.

e) Aplique a la señal de interés el filtro diseñado en el punto b) haciendo:

```
SOS = Hd.sosMatrix;  
G    = Hd.ScaleValues;  
  
iir_output = filtfilt(SOS, G, signal);
```

f) Grafique los espectros de la señal original (`signal`) y filtrada (`iir_output`) con la función `my_dft()`.

g) Examine ambas gráficas. ¿Qué diferencia observa entre ambas señales?.

### 3) Dimensiones de filtros FIR e IIR para un mismo tipo de filtro

a) Ejecute la función `iir_vs_fir.m.mlx`.

b) Analice las funciones `fir_kaiser_3400_44100.m` e `iir_elliptic_3400_44100.m` ¿Qué tipos de filtros implementan ambas funciones?.

c) Observe ambas respuestas en frecuencia. ¿Qué diferencias hay entre ambas respuestas?

d) Grafique las respuestas en fase y compárelas. ¿Qué diferencias hay entre ambas respuestas?

e) ¿Cuál es la dimensión del numerador del filtro FIR y cuántos coeficientes presenta la matriz SOS del filtro IIR? ¿A qué conclusión puede abordar?

### 4) Diseño de un filtro IIR con transformada bilineal

Diseñe un filtro digital tipo IIR con la transformada bilineal a partir del diseño de un filtro analógico, utilizando funciones provistas por MATLAB.

a) Diseñe un filtro IIR:

- Chebyshev Tipo I
- Pasa-banda entre 300 y 3.400 Hz con
- 1 dB de ripple en la banda pasante.

b) Normalice las frecuencias de corte en radianes según la frecuencia de muestreo:

```
wc1_n = 2*pi*fc1/fs;  
wc2_n = 2*pi*fc2/fs;
```

c) Aplique precombado (pre-warping) a las frecuencias analógicas de interés.

```
wc1_p = (2/dt)*tan(wc1_n/2);  
wc2_p = (2/dt)*tan(wc2_n/2);
```

d) Diseñe el filtro analógico con las funciones cheb1ap() y lp2bp():

```
[Z, P, K] = cheb1ap(orden, ripple);  
[A, B] = zp2tf(Z, P, K);  
freqs(B, A)  
w_central = sqrt(wc1_p*wc2_p);  
bw = wc2_p - wc1_p;  
[num_a, den_a] = lp2bp(A, B, w_central, bw);
```

e) Discretice el filtro analógico:

```
H_analog = tf(num_a, den_a);  
  
H_digital = c2d(H_analog, dt, 'tustin');  
num_d = H_digital.numerator{1, 1};  
den_d = H_digital.denominator{1, 1};
```

f) Grafique la respuesta del filtro digital en el plano Z:

```
zplane(num_d, den_d, 500000)  
grid on
```

g) Determine la respuesta en frecuencia de ambos filtros:

```
[h_a, w_a] = freqs(num_a, den_a, 1000);  
[h_d, w_d] = freqz(num_d, den_d, 1000);
```

h) Grafique la respuesta en frecuencia y fase del filtro digital:

```
mag_a = abs(h_a);  
phi_a = phase(h_a);  
f_a = w_a/pi/2;  
  
mag_d = abs(h_d);  
phi_d = phase(h_d);  
f_d = w_d/pi/2 * fs;  
  
figure  
plot(f_a, mag_a, 'b');  
hold on  
plot(f_d, mag_d, 'g');  
title('RESPUESTA EN FRECUENCIA')  
grid on  
legend('Filtro analogico', 'Filtro digital')  
  
figure  
plot(f_a, phi_a, 'b');  
hold on  
plot(f_d, phi_d, 'g');  
title('RESPUESTA EN FASE')  
grid on
```

```
legend('Filtro analogico', 'Filtro digital')
```

## 5) Filtros IIR en C, de 2do orden tipo Direct I y Direct II, formato punto flotante

Se pretende ejecutar desde MATLAB una función descrita en C que implementa un filtro IIR (versión online) en el dominio de la frecuencia. Se propone el siguiente ejemplo.

Se cuenta con una señal de entrada compuesta por:

- Tono de 300 Hz, más tono de 600 Hz, más tono de 50 Hz (frecuencia de línea eléctrica).

Se desea diseñar un filtro pasa-banda que rechace la señal de 50 Hz y que deje pasar los dos tonos.

a) Se diseña un filtro con `filterDesigner`:

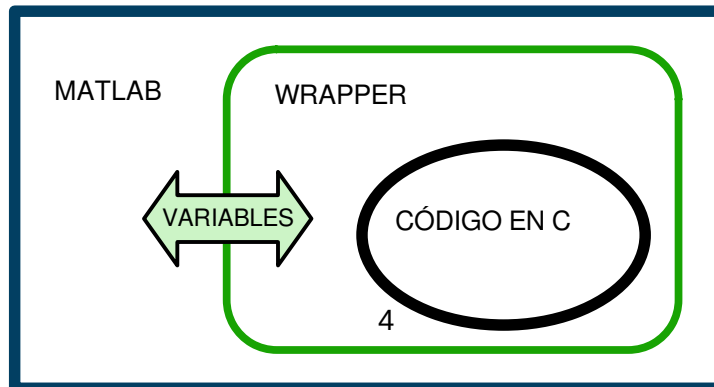
- Filtro IIR
- Pasa banda.
- Elliptic, orden 2.
- Frecuencias de corte 200 Hz y 800 Hz.
- Frecuencia de muestreo de 10 kHz.
- Apass 1, Astop 80.

b) Exporte el archivo `fdacoeffs.h`. Los coeficientes del filtro IIR se exportan haciendo `Targets > Generate C Header`, al archivo `fdacoeffs.h` en formato punto flotante, precisión simple.

c) Compile en consola las funciones `iir_wrapper.c` y `iir_filter.c`, en este específico orden, con el comando:

```
>> mex iir_wrapper.c iir_filter.c
```

`iir_wrapper.c` construye la interfaz entre las variables del Wokrspace de MATLAB y los argumentos de entrada/salida de las funciones en C.



`iir_filter.c` contiene la función `iir_2nd_df1_float()`, la cual integra un filtro IIR Forma Directa 1 de segundo orden y una señal de entrada, todo en formato punto flotante, precisión simple (float).

d) Analice el código de la función `iir_2nd_df1_float()`.

e) Analice el código de la función `iir_online.m` y ejecútela. ¿Qué observa?

f) Repita todo el ejercicio pero utilizando la función `iir_2nd_df2_float()`.

## **6) Filtros IIR en C, de 6to orden tipo Direct I y Direct II, formato punto flotante**

Repita el ejercicio 5) pero para un filtro IIR de 6to orden. Utilice las funciones `iir_nth_df1_float()` e `iir_nth_df2_float()`.

### **NOTA**

Si se ejecuta MATLAB bajo Windows, se recomienda usar el compilador MinGW [1] y configurarlo cómo se indica en [2].

[1] <https://www.mingw-w64.org/>

[2] <https://www.mathworks.com/matlabcentral/answers/313298-i-already-have-mingw-on-my-computer-how-do-i-configure-it-to-work-with-matlab>