Guía Laboratorio Filtros
Procesamiento Digital de Señales

Felipe Ayala Valencia

2023-1

NOTAS:

- Enviar el informe del laboratorio con el siguiente nombre: *Lab filtros PDS Apellido Nombre.ipynb*
- Enviar junto con el informe los archivos adicionales generados y descargados. Todo esto debe ir en un archivocomprimido con el siguiente nombre: *Lab8 PDS Apellido Nombre.zip*
- ¡OJO! Recuerde tener cuidado con la indentación y caracteres como el guión bajo y las llaves cuando copie y pegue el código entregado en esta guía.
- Las preguntas deberán ser resueltas en el notebook indicando sus respectivos numerales.

1. Introducción

En esta práctica se evaluará el uso de filtros FIR en distintas aplicaciones, puntualmente para este laboratorio se proveerá de un audio, el mismo contiene cantos de 3 especies, grillos, ranas y sapos, la practica consiste en el diseño de filtros que sean capaces de atenuar los cantos para poder escuchar solo una de las especies

Filtros FIR:

- Los filtros FIR son filtros de respuesta finita al impulso, lo que significa que su respuesta al impulso tiene una duración finita.
- Son filtros lineales y de fase lineal, lo que significa que no introducen distorsión de fase en la señal de entrada.
- Se pueden diseñar utilizando ventanas o métodos de diseño como el filtro de Parks-McClellan o el algoritmo de la ventana óptima.
- Tienen una respuesta en frecuencia de tipo "caja", con una respuesta de amplitud plana en la banda de paso y una atenuación rápida en la banda de rechazo.
- Son adecuados para aplicaciones que requieren una respuesta de amplitud precisa, como ecualización de audio, filtrado de ruido, filtrado de señales digitales y procesamiento de imágenes.

Filtros IIR:

- Los filtros IIR son filtros de respuesta infinita al impulso, lo que significa que su respuesta al impulso se extiende indefinidamente en el tiempo.
- Son filtros recursivos y pueden tener realimentación, lo que les permite tener una respuesta en frecuencia más compleja que los filtros FIR.
- Pueden ser de fase lineal o no lineal, lo que significa que pueden introducir distorsión de fase en la señal de entrada.
- Se pueden diseñar utilizando métodos como la transformación bilineal, el método de aproximación de Butterworth o el método de Chebyshev.
- Tienen una respuesta en frecuencia más flexible que los filtros FIR, lo que los hace adecuados para aplicaciones que requieren una respuesta en frecuencia más compleja, como filtrado de señales de audio, diseño de ecualizadores gráficos, procesamiento de señales biomédicas y análisis de señales de comunicación.

1.1 Cargue el audio provisto y escuche la versión original, asegúrese de normalizar el audio, grafique (“Para otro tipo de audios que se puedan encontrar en repositorios que sean esteros usar el siguiente código para poder trabajar con ellos”).

```
#--------------------------------Codigo Necesario por si el audio cargado es estereo-----------------------
import soundfile as sf

# Ruta del archivo de audio estéreo
archivo_estereo = '10466.wav'

# Ruta de destino para el archivo de audio mono
archivo_mono = 'test2.wav'

    # Cargar el archivo de audio estéreo
    data, samplerate = sf.read(archivo_estereo)

    # Extraer una sola columna (canal) del archivo de audio estéreo
    data_mono = data[:, 0]  # Seleccionar el primer canal (izquierdo)

    # Escribir el archivo de audio mono
    sf.write(archivo_mono, data_mono, samplerate)
```

2. Espectro

Grafique el espectro del audio y explique, ¿Desde el espectro se pueden apreciar las frecuencias de cada especie mencionada en la guía?

3. Diseño de Filtro, Canto de grillos

3.1 para extraer 2 audios, el primer filtro será para obtener el sonido de los grillos, consulte en internet el rango de frecuencia que producen los grillos y en base a eso realice la parametrización de un filtro pasa-banda, diseñe el filtro pasa-banda y aplíquele una ventana que usted considere (No se evaluara cual ventana es la correcta, asuma que tiene recursos suficientes para usar cualquier ventana), luego grafique la respuesta de ese filtro con y sin ventana. Función para crear la ventana.

```
def choose_windows(name='Hamming', N=20):
    # Rect/Hanning/Hamming
    if name == 'Hamming':
        window = np.array([0.54 - 0.46 * np.cos(2 * np.pi * n / (N - 1)) for n in range(N)])
    elif name == 'Hanning':
        window = np.array([0.5 - 0.5 * np.cos(2 * np.pi * n / (N - 1)) for n in range(N)])
    elif name == 'Rect':
        window = np.ones(N)
    elif name == 'black':
        window = np.array([0.42-0.5*np.cos((2*np.pi*n)/(N-1))+0.08*np.cos((4*np.pi*n)/(N-1))for n in range(N)])
```

```
        return window
```

Código Para crear el filtro, definir los parámetros que sean necesarios.

```
h1 = ((wc2)/np.pi) *                np.sinc((wc2*n)/np.pi)-((wc1)/np.pi)*
np.sinc((wc1*n)/np.pi) # Respuesta del filtro ideal

h1[n==0]=(wc2-wc1)/np.pi # cuando es pasabanda

w1,Hh1 = signal.freqz(h1,1,whole=True, worN=N) # Respuesta en frecuencia
del filtro ideal

win2 = choose_windows(name='Hamming', N=len(n))

h2=h1*win2 # Multiplico la respuesta ideal por la ventana

    A=np.sqrt(10**(0.1*Adb))
    h2=h2*A # Ganancia del filtro
    w2,Hh2 = signal.freqz(h2,1,whole=True, worN=N) #    Respuesta    en
frecuencia del filtro enventanado
```

Código Para graficar Las repuestas del Filtro diseñado.

```
fig1,axs1 = subplots(4,1)
fig1.set_size_inches((8,12))
subplots_adjust(hspace=0.4)

ax=axs1[0]
#ax.axis(xmax=2000, xmin=-2000)  # Límite en el eje x
ax.stem(n+M,h1,basefmt='b-')
ax.set_xlabel("$n$",fontsize=24)
ax.set_ylabel("$h_1$",fontsize=24)


    ax=axs1[1]
    #ax.axis(xmax=2000, xmin=-2000)  # Límite en el eje x
    ax.stem(n+M,h2,basefmt='b-')
    ax.set_xlabel("$n$",fontsize=24)
    ax.set_ylabel("$h_2$",fontsize=24)



    ax=axs1[2]
    ax.plot((w1-np.pi)*fs/(2*np.pi),np.abs(np.fft.fftshift(Hh1)))
    ax.axis(xmax=fs/2,xmin=-fs/2)
    #ax.axis(xmax=2000, xmin=-2000)  # Límite en el eje x
    #ax.vlines([-fc,fc],0,1,color='g',lw=2.,linestyle='--')
    #ax.hlines(1,-fc,fc,color='g',lw=2.,linestyle='--')
    ax.set_xlabel(r"$f (Hz)$",fontsize=18)
    ax.set_ylabel(r"$|H1(\omega)| $",fontsize=18)

```

```
    ax=axs1[3]
    ax.plot((w2-np.pi)*fs/(2*np.pi),np.abs(np.fft.fftshift(Hh2)))
    ax.axis(xmax=fs/2,xmin=-fs/2)
    #ax.axis(xmax=2000, xmin=-2000)  # Límite en el eje x
    #ax.vlines([-fc,fc],0,1,color='g',lw=2.,linestyle='--')
    #ax.hlines(1,-fc,fc,color='g',lw=2.,linestyle='--')
    ax.set_xlabel(r"$f (Hz)$",fontsize=18)
    ax.set_ylabel(r"$|H2(\omega)| $",fontsize=18)
```

3.2 aplique el filtro al audio original y grafique en sub-plots cada resultado, escuche el resultado obtenido y responda, ¿Se logra separar el canto de los grillos? y si no, ¿Por que? (Nota: el estudiante usando parámetros correctos debe ser capaz de obtener un resultado satisfactorio)

3.3 Grafique nuevamente el espectro del resultado anterior, responda: ¿El resultado es el esperado según su respuesta en el numeral #2?

4. Canto de ranas y sapos

Repita el punto 3 para separar el canto de las ranas y los sapos, consulte en internet el rango de frecuencias, y finalmente grafique en un mismo plot el espectro obtenido por los 2 filtros, utilice colores para identificar correctamente cada canto.

5. Ventanas Parametrizables (Kaiser)

Realice una consulta sobre la ventana de Kaiser, sus aplicaciones y si es viable para la extracción de cantos en animales, realice un código que varie el parámetro Beta en los valores "[1, 5, 20, 150, 350]", explique que cambia con la variación de este parámetro. Ejemplo para graficar las ventanas Kaiser.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import windows, freqz

# Parámetros de la ventana de Kaiser
window_length = 101  # Longitud de la ventana
beta_values = [1, 5, 20, 150, 350]  # Valores de beta

    # Graficar la ventana de Kaiser
    window = windows.kaiser(window_length, beta)
    plt.plot(window, label=f'Beta = {beta}')

    # Graficar la respuesta en frecuencia
    window = windows.kaiser(window_length, beta)
    w, h = freqz(window)
    frequencies = w / (2 * np.pi)
    amplitude = np.abs(h)
    plt.plot(frequencies, amplitude, label=f'Beta = {beta}')
```

6. Conclusiones

Realice conclusiones generales sobre la práctica. Recuerde que las conclusiones son parte fundamental de su evaluación en el laboratorio, tómese el tiempo de pensar las conclusiones.