tp1_ejercicio_3

June 13, 2024

0.1 Ejercicio 3

La ecuación en diferencias es

$$y[n] = \frac{1}{M_1 + M_2 + 1} \sum_{k=-M_1}^{k=M_2} x[n-k]$$

Con $M_1 = 0$ y $M_2 = 4$.

0.1.1 Ejercicio 3 parte a

Obtenemos la respuesta al impulso aplicando una delta a la entrada, es decir $x[n] = \delta[n]$.

Para resolverlo, tenemos en cuenta que

$$\sum_{k=-M_1}^{k=M_2} \delta[n-k] = \begin{cases} 1 & \text{si } -M_1 \leq n \leq M_2 \\ 0 & \text{demás casos} \end{cases}$$

Es una función cuadrada entre $-M_1$ y M_2. Podemos expresarla como

$$\sum_{k=-M_1}^{k=M_2} \delta[n-k] = u[n + M_1] - u[n - M_2 - 1]$$

siendo $u[n]$ la función escalón.

Entonces, la respuesta al impulso es

$$y[n] = \frac{1}{M_1 + M_2 + 1} u[n + M_1] - u[n - M_2 - 1]$$

Para los valores dados de $M_1 = 0$ y $M_2 = 4$, es una cuadrada entre 0 y 4 con altura $\frac{1}{M_1+M_2+1} = \frac{1}{0+4+1} = 0.2$

0.1.2 Ejercicio 3 parte b

Para obtener la respuesta en frecuencia, consultamos la tabla de transformadas 2.3 en la página 62 de Oppehnheim Schafer. Vemos que se establece el par

$$\begin{cases} 1 & \text{si } 0 \leq n \leq M \\ 0 & \text{demás casos} \end{cases} \iff \frac{\sin(\omega(M+1)/2)}{\sin(\omega/2)} e^{-j\omega M/2}$$

Considerando en nuestro caso que $M_1 = 0$ Podemos aplicar directamente la expresión. Sino, deberíamos aplicar la propiedad de desfasaje. Dejando genérico M_2 con valor M, llegamos a que nuestra respuesta en frecuencia es:

$$\frac{1}{M+1} \frac{\sin(\omega(M+1)/2)}{\sin(\omega/2)} e^{-j\omega M/2}$$

En nuestro caso es $M = 4$ y tenemos un factor de escala de 0.2. Entonces la respuesta en frecuencia es

$$0.2 \frac{\sin(2.5\omega)}{\sin(\omega/2)} e^{-j\omega 2}$$

0.1.3 Ejercicio 3 parte c

[1]:
```python
import numpy as np
import plotly.graph_objs as go
import plotly.subplots as sp

# Definir la función lambda para la respuesta en frecuencia
H = lambda M, omega: (1/(M+1)) * (np.sin(omega * (M+1) / 2) / np.sin(omega /2)) * np.exp(-1j * omega * M / 2)

M = 4
omega = np.linspace(-np.pi, np.pi, 1000)
H_M = H(M, omega)

magnitud = np.abs(H_M)
fase = np.angle(H_M)

fig = sp.make_subplots(rows=2, cols=1, subplot_titles=('Magnitud', 'Fase'))

fig.add_trace(go.Scatter(x=omega, y=magnitud, name='Magnitud'), row=1, col=1)
fig.add_trace(go.Scatter(x=omega, y=fase, name='Fase'), row=2, col=1)

# Configurar los ticks del eje X en unidades de pi
pi_ticks = [-np.pi, -np.pi/2, 0, np.pi/2, np.pi]
pi_tick_labels = ['- ', '- /2', '0', ' /2', ' ']
```

```
fig.update_xaxes(
    tickvals=pi_ticks,
    ticktext=pi_tick_labels,
    title_text='  (radianes)',
    row=1, col=1
)
fig.update_xaxes(
    tickvals=pi_ticks,
    ticktext=pi_tick_labels,
    title_text='  (radianes)',
    row=2, col=1
)

fig.update_layout(height=600, width=800, title_text="Respuesta en Frecuencia")
fig.show()
```

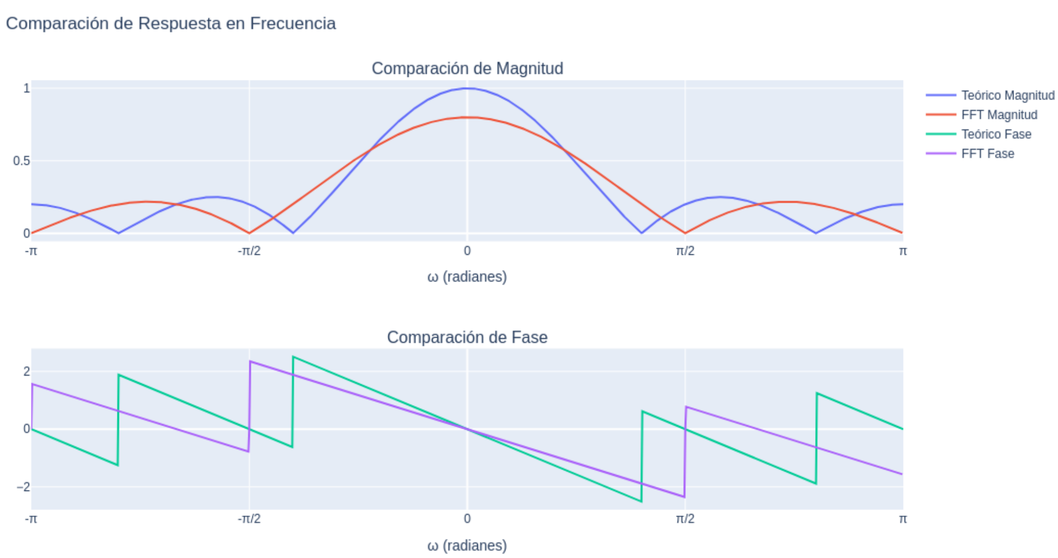
El filtro obtenido es pasabajos, ya que tiene una banda de paso alrededor de frecuencia 0, y atenuación en frecuencias superiores.

0.1.4 Ejercicio 3 parte d

[2]:

```
# Generar la señal cuadrada
N_fft = 1024
signal = np.zeros(N_fft)
signal[:M] = 1 / (M + 1)

# Calcular la FFT de la señal
fft_signal = np.fft.fftshift(np.fft.fft(signal))
omega_fft = np.fft.fftshift(np.fft.fftfreq(len(signal))) * 2 * np.pi

magnitud_fft = np.abs(fft_signal)
fase_fft = np.angle(fft_signal)

# Crear la figura de comparación
fig = sp.make_subplots(rows=2, cols=1, subplot_titles=('Comparación de␣
 ↪Magnitud', 'Comparación de Fase'))

fig.add_trace(go.Scatter(x=omega, y=magnitud, name='Teórico Magnitud'), row=1,␣
 ↪col=1)
fig.add_trace(go.Scatter(x=omega_fft, y=magnitud_fft, name='FFT Magnitud',␣
 ↪mode='lines'), row=1, col=1)
fig.add_trace(go.Scatter(x=omega, y=fase, name='Teórico Fase'), row=2, col=1)
fig.add_trace(go.Scatter(x=omega_fft, y=fase_fft, name='FFT Fase',␣
 ↪mode='lines'), row=2, col=1)

# Configurar los ticks del eje X en unidades de pi
pi_ticks = [-np.pi, -np.pi/2, 0, np.pi/2, np.pi]
```

```
pi_tick_labels = ['- ', '- /2', '0', ' /2', ' ']

fig.update_xaxes(
    tickvals=pi_ticks,
    ticktext=pi_tick_labels,
    title_text='  (radianes)',
    row=1, col=1
)
fig.update_xaxes(
    tickvals=pi_ticks,
    ticktext=pi_tick_labels,
    title_text='  (radianes)',
    row=2, col=1
)

fig.update_layout(height=600, width=800, title_text="Comparación de Respuesta␣
 ↪en Frecuencia")
fig.show()
```



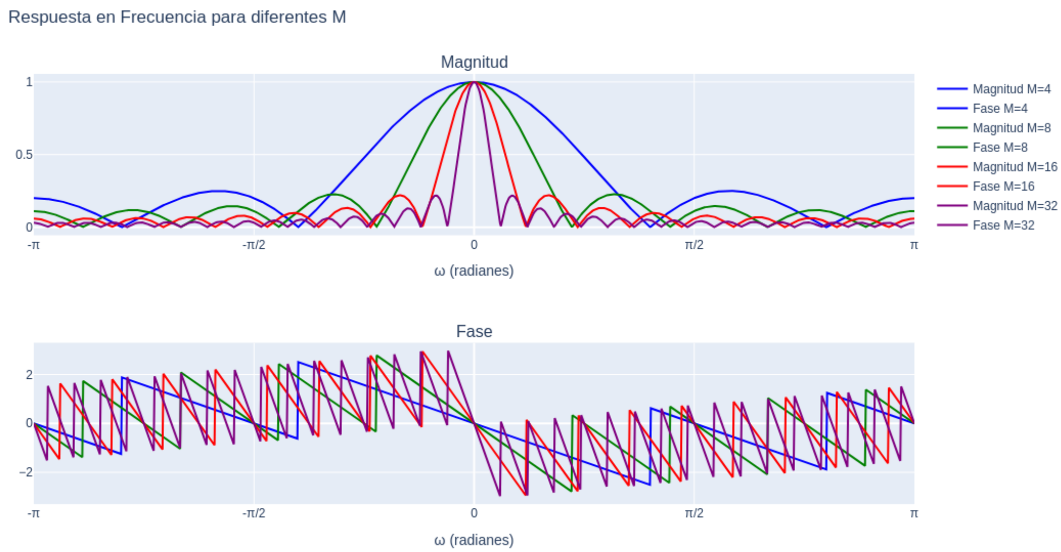
0.1.5 Ejercicio 3 parte e

Sí, la salida depende de entradas anteriores, ya que es el promedio de las últimas M muestras.

Este tipo de filtros en los que la salida depende de salidas anteriores se denominan **con memoria**.

0.1.6 Ejercicio 3 parte f

[3]:
```
# Definir la función lambda para la respuesta en frecuencia
H = lambda M, omega: (1/(M+1)) * (np.sin(omega * (M+1) / 2) / np.sin(omega /␣
↪2)) * np.exp(-1j * omega * M / 2)

omega = np.linspace(-np.pi, np.pi, 1000)
M_values = [4, 8, 16, 32]

fig = sp.make_subplots(rows=2, cols=1, subplot_titles=('Magnitud', 'Fase'))

# Configurar los ticks del eje X en unidades de pi
pi_ticks = [-np.pi, -np.pi/2, 0, np.pi/2, np.pi]
pi_tick_labels = ['- ', '- /2', '0', ' /2', ' ']

colors = ['blue', 'green', 'red', 'purple']
for i, M in enumerate(M_values):
    H_M = H(M, omega)
    magnitud = np.abs(H_M)
    fase = np.angle(H_M)

    fig.add_trace(go.Scatter(x=omega, y=magnitud, mode='lines', name=f'Magnitud␣
↪M={M}', line=dict(color=colors[i])), row=1, col=1)
    fig.add_trace(go.Scatter(x=omega, y=fase, mode='lines', name=f'Fase M={M}',␣
↪line=dict(color=colors[i])), row=2, col=1)

fig.update_xaxes(
    tickvals=pi_ticks,
    ticktext=pi_tick_labels,
    title_text='  (radianes)',
    row=1, col=1
)
fig.update_xaxes(
    tickvals=pi_ticks,
    ticktext=pi_tick_labels,
    title_text='  (radianes)',
    row=2, col=1
)

fig.update_layout(height=600, width=800, title_text="Respuesta en Frecuencia␣
↪para diferentes M")
fig.show()
```



A medida que M_2 aumenta, observamos como se angosta la banda de paso, es decir, se reduce el ancho de banda del sistema.