Final Exam

Vito Torina

April 21, 2024

Cover page below
↓

Western Michigan University
Electrical and Computer Engineering Dep
ECE4550/ECE5550-Digital Signal Proce

SPRING 2024

Final Exam

Show all your work! Failure to show your work steps will result in a major loss of your

Due 11 pm on Monday 4/22/2024

Answer all problems as best as you can. Write clearly!

Submit one pdf file for all your work. Number your pages and upload to eLearning dropbox. Image files must be converted to pdf before uploading

Name (last name 1st): Torina Vito

This cover sheet must be scanned and used as a cover sheet with your submission

PROBLEM 1 (25)	
PROBLEM 2 (25)	
PROBLEM 3 (30)	
TOTAL (100)	

Make sure to clearly label all axes of all your graph

Problem #1 (25 Points)

Given

Design a filter that completely blocks the frequency of $\omega_0 = \frac{\pi}{3}$, by placing its poles and zeros in the z-plane. The designed filter should yield real output, given the input is real. Clearly and show all your work, answer the following:

1. Draw poles and zeros of the filter in the Z-plane, clearly showing magnitude and phase.

2. Write the expression for the transfer function, $H(z)$ associated with the filter you designed per your pole-zero plot. You can use $K = 1$ for a scaling constant.

3. Write an expression for $H(\omega)$, and *give its magnitude and phase.*

4. Plot the magnitude of $H(\omega)$ (not the Bode plot), that is $|H(\omega)|$ versus frequency ω. You can use any software.

5. Discuss this filter stability in sight of your choice of its poles and zeros.

6. Determine the output $y(n)$, if the input signal to the filter is given by

$$x(n) = 6 + 3\cos\left(\frac{\pi}{3}n\right) + \sin\left(\frac{\pi}{6}n + \frac{\pi}{2}\right) + 2\cos\left(\frac{\pi}{2}n\right) \quad -\infty < n < \infty$$

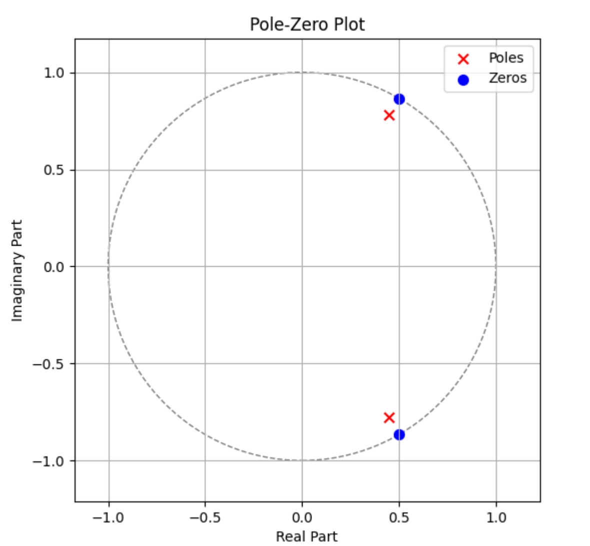
7. What type of a filter can this be used for, and why?

8. Based on the filter's type, explain the relation between $y(n)$ and $x(n)$ in part 6.

Solution

1) Plot Poles and Zeros on Z-plane

```
import numpy as np
import matplotlib.pyplot as plt

poles = [0.9*(np.exp(1j * np.pi/3)), 0.9*(np.exp(-1j * np.pi/3))]
zeros = [np.exp(1j * np.pi/3), np.exp(-1j * np.pi/3)]

fig, ax = plt.subplots(figsize=(6, 6))

unit_circle = plt.Circle((0, 0), 1, fill=False, linestyle='--', color='gray')
ax.add_artist(unit_circle)

ax.scatter(np.real(poles), np.imag(poles), marker='x', color='r', s=50, label='Poles')
ax.scatter(np.real(zeros), np.imag(zeros), marker='o', color='b', s=50, label='Zeros')

ax.set_xlabel('Real Part')
ax.set_ylabel('Imaginary Part')
ax.set_title('Pole-Zero Plot')
ax.legend()
ax.axis('equal')
ax.grid()
plt.show()
```



2) Find H(z)

Zeros are $\omega_0 = \frac{\pi}{3}$ Poles are $0.9 \times e^{j\frac{\pi}{3}}$
so...
$H(z) = K * (z - z1) * (z - z2)/((z - p1) * (z - p2))$
sub in...

$$H(z) = K \cdot \frac{(z - e^{j\pi/3})(z - e^{-j\pi/3})}{(z - 0.9e^{j\pi/3})(z - 0.9e^{-j\pi/3})}$$

simplify:

$$H(z) = \frac{z^2 - 1}{z^2 - z + \frac{1}{2}}$$

3) Find $H(\omega)$

I wrote a MATLAB script to find $H(\omega) and |H(\omega)| and \angle H(\omega)$

```
zeros = [exp(1j*pi/3), exp(-1j*pi/3)];
poles = [0.9*exp(1j*pi/3), 0.9*exp(-1j*pi/3)];

num = poly(zeros);
den = poly(poles);

syms z w

Hz = poly2sym(num, z) / poly2sym(den, z);

Hw = subs(Hz, z, exp(1j*w));

Hw_mag = abs(Hw);
Hw_phase = angle(Hw);

disp('H(z):');
pretty(Hz_sym);

disp('H(w):');
pretty(Hw);

disp('Magnitude of H(w):');
pretty(Hw_mag);

disp('Phase of H(w) (in radians):');
pretty(Hw_phase);
```

Here is the output when running my code.

```
 p1p3
H(z)
   2
  z  - 1
----------
 2      1
z  - z + -
        2

H(w)
  1 + exp(w 2i) - exp(w 1i)
------------------------------
 81                9 exp(w 1i)
--- + exp(w 2i) - -----------
100                    10

Magnitude of H(w)
    1 + exp(w 2i) - exp(w 1i)
-----------------------------------
   81                9 exp(w 1i)
  --- + exp(w 2i) - -----------
  100                    10

Phase of H(w) (in radians)
        1 + exp(w 2i) - exp(w 1i)
angle ------------------------------
       81                9 exp(w 1i)
      --- + exp(w 2i) - -----------
      100                    10
```

Frequency Response $H(\omega)$

$$H(\omega) = \frac{1 + e^{2j\omega} - e^{j\omega}}{\frac{81}{100} + e^{2j\omega} - \frac{9}{10}e^{j\omega}}$$

$$H(\omega) = \frac{1 + \cos(2\omega) - \cos(\omega) + j \cdot (\sin(2\omega) - \sin(\omega))}{\frac{81}{100} + \cos(2\omega) - \frac{9}{10} \cdot \cos(\omega) + j \cdot (\sin(2\omega) - \frac{9}{10} \cdot \sin(\omega))}$$

Magnitude of $H(\omega)$

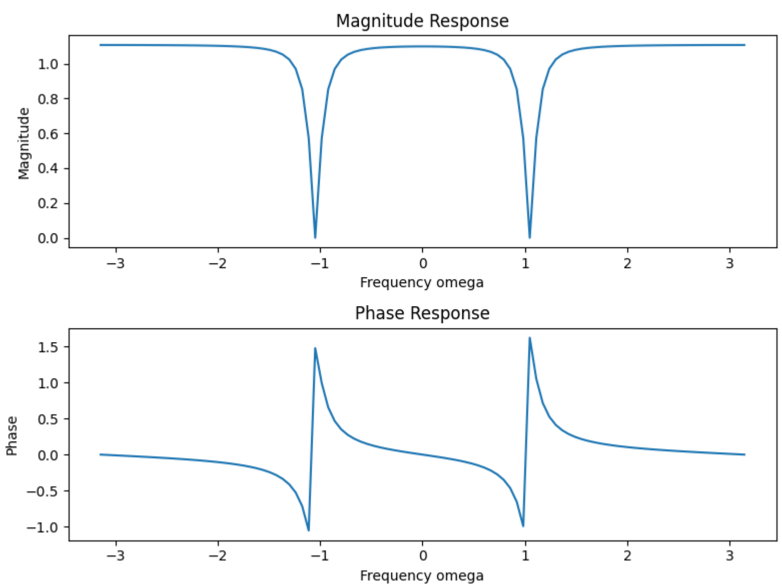
$$|H(\omega)| = \frac{|1 + e^{2j\omega} - e^{j\omega}|}{|\frac{81}{100} + e^{2j\omega} - \frac{9}{10}e^{j\omega}|}$$

Phase of $H(\omega)$ (in radians)

$$\angle H(\omega) = \angle(\frac{1 + e^{2j\omega} - e^{j\omega}}{\frac{81}{100} + e^{2j\omega} - \frac{9}{10}e^{j\omega}})$$

4) Plot $|H(\omega)|$ the magnitude and phase

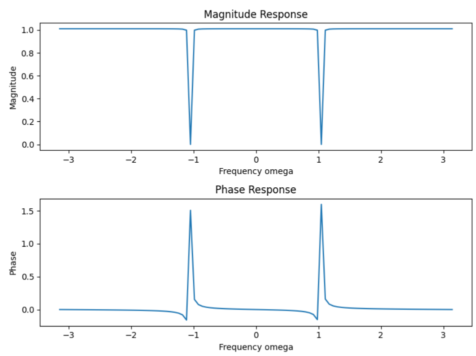
```python
import numpy as np
import matplotlib.pyplot as plt

def H_omega(w, K=1):
    z = np.exp(1j * w)
    numerator = (z - np.exp(1j * np.pi/3)) * (z - np.exp(-1j * np.pi/3))
    denominator = (z - 0.9 * np.exp(1j * np.pi/3)) * (z - 0.9 * np.exp(-1j * np.pi/3))
    H = K * numerator / denominator
    magnitude = np.abs(H)
    phase = np.angle(H)
    return H, magnitude, phase

K = 1
w = np.linspace(-np.pi, np.pi, 100)

H, mag, phase = H_omega(w, K)
fig, ax = plt.subplots(2, 1, figsize=(8, 6))

ax[0].plot(w, mag)
ax[0].set_title('Magnitude Response')
ax[0].set_xlabel('Frequency omega')
ax[0].set_ylabel('Magnitude')

ax[1].plot(w, phase)
ax[1].set_title('Phase Response')
ax[1].set_xlabel('Frequency omega')
ax[1].set_ylabel('Phase')

plt.tight_layout()
plt.show()
```



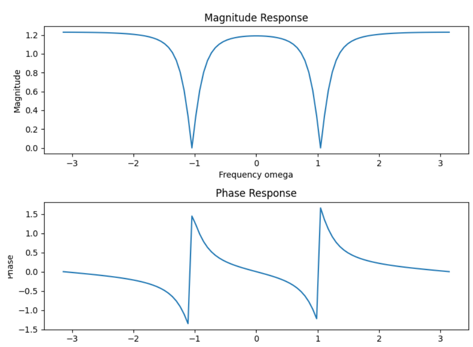
5) Filter Stability

Given that I used a similar filter to one found in the notes,the poles give the filter a stable respose since they lay inside the unit circle. The radius of 0.9 for the poles can be changed to either a smaller radius for a wider notched magnitude response or they can be widened to 0.99 for a sharper notched magnitude response.

0.99 Poles



0.80 Poles



6) Given $x(n) = 6 + 3cos(\frac{\pi}{3}n) + sin(\frac{\pi}{6}n + \frac{\pi}{2}) + 2cos(\frac{\pi}{2}n)$ input what is the output $y(n)$?

We know that the output function can be split up into the respective terms, then multiplied by the magnitude of the frequency component of each term. Such as...
$y1 = 6 \times |H(0)|$
$y2 = 3 \times |H(\frac{\pi}{3})|$
$y3 = 1 \times |H(\frac{\pi}{6})|$
$y4 = 2 \times |H(\frac{\pi}{2})|$
Since our filter is blocking all frequency at $\pi/3$ the second term of x(n) will be zero since the notch filter will completely block it. I reworked the code earlier to get our output y(n). We essentially just plug in our frequency component to the magnitude and then mulptiply it out.

```python
import numpy as np

def H_mag(omega, K=1):
    numerator = np.abs((np.exp(1j * omega) - np.exp(1j * np.pi/3)) * (np.exp(1j * omega) - np.exp(-1j * np.pi/3)))
    denominator = np.abs((np.exp(1j * omega) - 0.9 * np.exp(1j * np.pi/3)) * (np.exp(1j * omega) - 0.9 * np.exp(-1j * np.pi/3)))
    return K * numerator / denominator

H_0 = H_mag(0)
H_pi_3 = H_mag(np.pi/3)
H_pi_6 = H_mag(np.pi/6)
H_pi_2 = H_mag(np.pi/2)

print(f"|H(0)| = {H_0:.4f}")
print(f"|H(pi/3)| = {H_pi_3:.4f}")
print(f"|H(pi/6)| = {H_pi_6:.4f}")
print(f"|H(pi/2)| = {H_pi_2:.4f}")
```

Output:

```
|H(0)| = 1.0989
|H(π/3)| = 0.0000
|H(π/6)| = 1.0858
|H(π/2)| = 1.0871
```

$y1 = 6 \times 1.0989 = 6.5934$
$y2 = 3 \times 0 = 0$
$y3 = 1 \times 1.0858 = 1.0858$
$y4 = 2 \times 1.0871 = 2.1742$
$y(n) = 6.5934 + 1.0858sin(\frac{\pi}{6}n + \frac{\pi}{2}) + 2.1742cos(\frac{\pi}{2}n)$

7) What kind of filter?

This filter is a notch filter which is a type of band stop filter. This is used for blocking signals in a specific range, while letting all the rest of the signal to pass through. You would use this when you want a specific portion of the signal to be blocked and the rest of the signal to be unaltered. In our case we blocked the frequency of $\omega_0 = \pi/3$

8) Relation between y(n) and x(n) from 6?

So basically each component in x(n) can be treated as a separate input signal to get y1 y2 y3 y4, and the final output y(n) can be obtained by summing the responses of the system to each input component. y(n) is essentially just showing the way the signal would be when passing through the filter I designed.

Problem #2 (25 Points)

Given

Use the specifications of the following ideal filter to design a BP FIR Filter, answer the following questions:

$$H_{\text{Ideal}}(F) = \begin{cases} 1 & 10 \leq |F| \leq 15 \\ 0 & \text{otherwise} \end{cases}$$

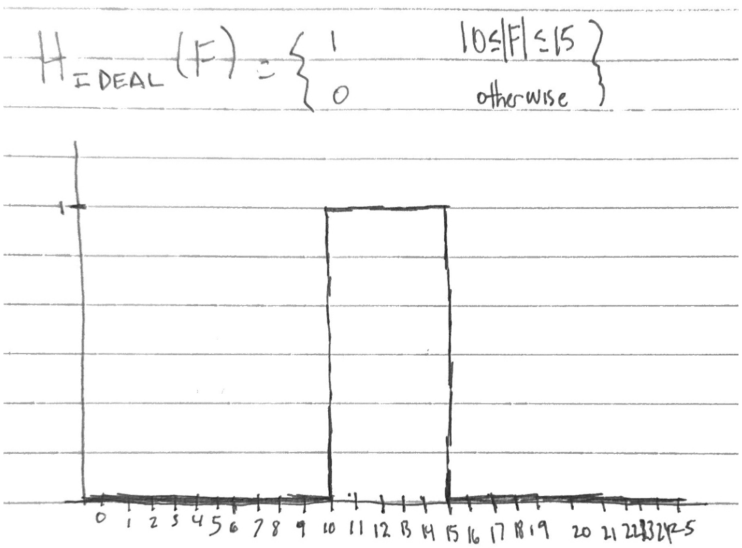
With sampling frequency $F_s = 50kHz$ and all given F values above are in kHz. And using,

$$W_{\text{hamming}}(n) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2\pi n}{M}\right) & 0 \leq n \leq M \\ 0 & \text{otherwise} \end{cases}$$

1. Sketch the frequency response for the desired filter using your hand drawing.
2. Using the above window, determine the impulse response of FIR filter which approximates this frequency response.
3. Compute the first 4 coefficients of the impulse response $h(n)$ coefficients.
4. Use MATLAB to plot both the desired $h(n)$, and the frequency response magnitude and phase of the filter you designed.
5. Comment on the expected performance of the filter you designed, and verify that it holds.
6. Comment on the nature of the filters phase.
7. Discuss how this window would be a better choice over the rectangular window.

Solution

1) Hand Sketch The Frequency Response



2) Determine the Impulse Response

I used MATLAB code to calculate the impulse response:

```
Fs = 50;
M = 50;

syms n;

h_ideal = (sin(2*pi*15*n/Fs) - sin(2*pi*10*n/Fs)) / (pi*n);
h_ideal(n==0) = (15 - 10) / Fs;

w = 0.54 - 0.46*cos(2*pi*n/M);

h = h_ideal * w;

h_eq = simplify(h);

disp('Equation for h(n):');
pretty(h_eq);
```

Here is the output when running the code

```
>> p2p2
Equation for h(n):
/    / pi n \          \
| cos| ---- | 23       |
|    \  25  /       27 | /    / 2 pi n \       / 3 pi n \ \
| -------------- - -- | | sin| ------ | - sin| ------ | |
\       50          50 / \    \    5   /       \    5   / /
---------------------------------------------------------
                           n pi
```

$$h(n) = \frac{\left(\frac{\cos\left(\frac{\pi n}{25}\right)23}{50} - \frac{27}{50}\right)\left(\sin\left(\frac{2\pi n}{5}\right) - \sin\left(\frac{3\pi n}{5}\right)\right)}{n\pi}$$

3) Compute the first 4 coefficients of the impulse response

I edited the previous code for finding the impulse response. Here is the MATLAB code:

```
Fs = 50;
M = 50;

syms n;

h_ideal = (sin(2*pi*15*n/Fs) - sin(2*pi*10*n/Fs)) / (pi*n);
w = 0.54 - 0.46*cos(2*pi*n/M);

h = h_ideal * w;

h_eq = simplify(h);

h_coeffs = zeros(1, 4);

%Handle the case when n=0
h_coeffs(1) = double(limit(h_eq, n, 0));
%Handle the rest
for i = 2:4
    h_coeffs(i) = double(subs(h_eq, n, i-1));
end

disp('First 4 coefficients of h(n):');
disp(h_coeffs);
```

Here is the output when running the code

```
>> p2p3
First 4 coefficients of h(n):
    0.0160         0   -0.0177         0
```

Problem #3 (30 Points)

Given

Consider the following filter $H(s)$

$$H(s) = \frac{1}{s}$$

Using $F_s = 10Hz$, design an IIR filter using these methods.

- Method 1: Impulse Invariant
- Method 2: Bilinear Transformation

In your design, address these questions

1. Give the model for $H(z)$ and $H(\omega)$ for each method.
2. Sketch the magnitude frequency response of $H(s)$, $H_i(\omega)$ of *each of the methods listed above.* You can use any tools.
3. Discuss the performance of both designs in how they approximate the analog filter for certain frequencies, that is discuss filter performance for low range frequencies, and high frequencies in comparison to the analog filter.
4. What type of a filter is this? Explain your answer.
5. Does your answer in part 4 align with the advantages and disadvantages of these two design methods? Keep your answer short and precise statements.
6. Discuss warping effect on $H_z(\omega)$ if any.

Solution

Thank you