CENG 384 - Signals and Systems for Computer Engineers
Spring 2023
Homework 4

Aydoğdu, Yusuf
e2237014@ceng.metu.edu.tr

Karaca, Bengisu
e2448538@ceng.metu.edu.tr

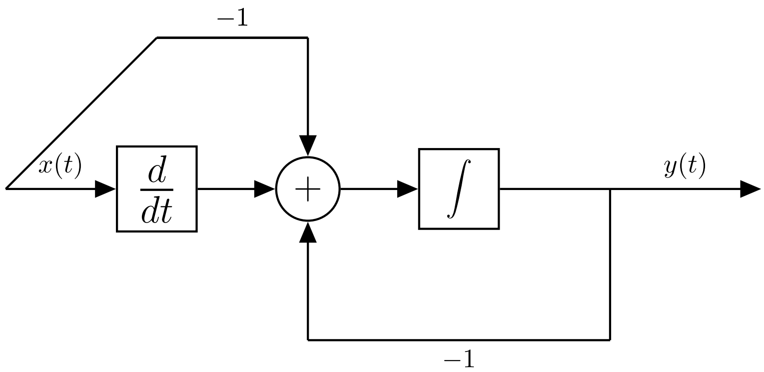
June 8, 2023

---

1. (a) $H(j\omega) = \frac{j\omega - 1}{j\omega + 1}$
Taking inverse Fourier Transform, we get,
$y'(t) + y(t) = x'(t) - x(t)$

(b) $H(j\omega) = \frac{j\omega}{j\omega+1} + \frac{-1}{j\omega+1} = \frac{A}{j\omega+1} - B$
solving for A and B,
$A - Bj\omega - B = j\omega - 1$,
$B = -1$,
$A = -2$,
$H(j\omega) = \frac{-2}{j\omega+1} + 1$

(c) $X(j\omega) = \frac{1}{j\omega+2}$
$Y(j\omega) = X(j\omega)H(j\omega) = \frac{1}{j\omega+2}\frac{-2}{j\omega+1} + \frac{1}{j\omega+2} = \frac{A}{j\omega+2} - \frac{B}{j\omega+1} + \frac{C}{j\omega+2} + D$
$1 + j\omega - 4 - 2j\omega + 1 + j\omega - \omega^2 + 3j\omega + 2 = A + Aj\omega + -2B - Bj\omega + C + Cj\omega - D\omega^2 + 3Dj\omega + 2D$
$A = 1$, $B = 2$, $C = 1$, $D = 1$
$y(t) = (2e^{-2t} - 2e^{-t})u(t) + \delta(t)$



(d)

2. (a) $Y(e^{j\omega})e^{j\omega} - \frac{1}{2}Y(e^{j\omega})e^{j\omega} = X(e^{j\omega})e^{j\omega}$
$H(e^{j\omega}) = \frac{Y(e^{j\omega})}{X(e^{j\omega})}$
$H(e^{j\omega}) = \frac{e^{j\omega}}{e^{j\omega} - \frac{1}{2}} = \frac{1}{1 - \frac{1}{2}e^{-j\omega}}$
$H(e^{j\omega}) = \frac{1}{1 - \frac{1}{2}e^{-j\omega}}$

(b) From table 5.2, we can take the Inverse Fourier Transform of $H(e^{j\omega})$
and reach $h[n] = (\frac{1}{2})^n u[n]$

(c) From table 5.2, we can take the Fourier Transform of $x[n]$
and reach $X(e^{jw}) = (\frac{1}{1 - \frac{3e^{-jw}}{4}})$
$Y(e^{j\omega}) = \frac{A}{1 - \frac{1}{2}e^{-j\omega}} + \frac{B}{1 - \frac{3e^{-jw}}{4}}$ We solve the equations and get A=-2, B=3
$Y(e^{j\omega}) = -\frac{2}{1 - \frac{1}{2}e^{-j\omega}} + \frac{3}{1 - \frac{3e^{-jw}}{4}}$
From table 5.2, we can take the Inverse Fourier Transform of $Y(e^{j\omega})$
and reach $y[n] = (-2(\frac{1}{2})^n + 3(\frac{3}{4})^n)u[n]$

3. (a) $Y(j\omega) = X(j\omega)[H_1(j\omega)H_2(j\omega)]$
$H(j\omega) = (\frac{1}{j\omega+1})\frac{1}{j\omega+2})$
$\frac{Y(j\omega)}{X(j\omega)} = \frac{1}{(j\omega)^2+3j\omega+2}$
$y''(t) + 3y'(t) + 2y(t) = x(t)$

(b) $H(j\omega) = (\frac{1}{j\omega+1})(\frac{1}{j\omega+2}) = \frac{A}{j\omega+1} + \frac{B}{j\omega+2}$
$A + B = 0$,
$2A + B = 1$,
$A = 1$,
$B = -1$
$h(t) = e^{-t}u(t) - e^{-2t}u(t)$

(c) $Y(j\omega) = X(j\omega)H(j\omega) = (j\omega)[\frac{1}{j\omega+1} - \frac{1}{j\omega+2}] = \frac{j\omega}{(j\omega+1)(j\omega+2)}$
$Y(j\omega) = \frac{j\omega}{(j\omega+1)(j\omega+2)} = \frac{A}{j\omega+1} + \frac{B}{j\omega+2}$
$A + B = 1$,
$2A + B = 0$,
$A = -1$,
$B = 2$
$Y(j\omega) = \frac{j\omega}{(j\omega+1)(j\omega+2)} = \frac{-1}{j\omega+1} + \frac{2}{j\omega+2}$
$y(t) = -e^{-t}u(t) + 2e^{-2t}u(t)$

4. (a) $Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega}) = X(e^{j\omega})[H_1(e^{j\omega}) + H_2(^{j\omega})]$
$H(e^{j\omega}) = (\frac{3}{3+e^{-j\omega}}) + (\frac{2}{2+e^{-j\omega}})$
$H(e^{j\omega}) = \frac{6+3e^{-j\omega}+6+2e^{-j\omega}}{6+5e^{-j\omega}+e^{-2j\omega}} = \frac{12+5e^{-j\omega}}{6+5e^{-j\omega}+e^{-2j\omega}}$
$y[n-2] - 5y[n-1] + 6y[n] = 12x[n] - 5x[n-1]$

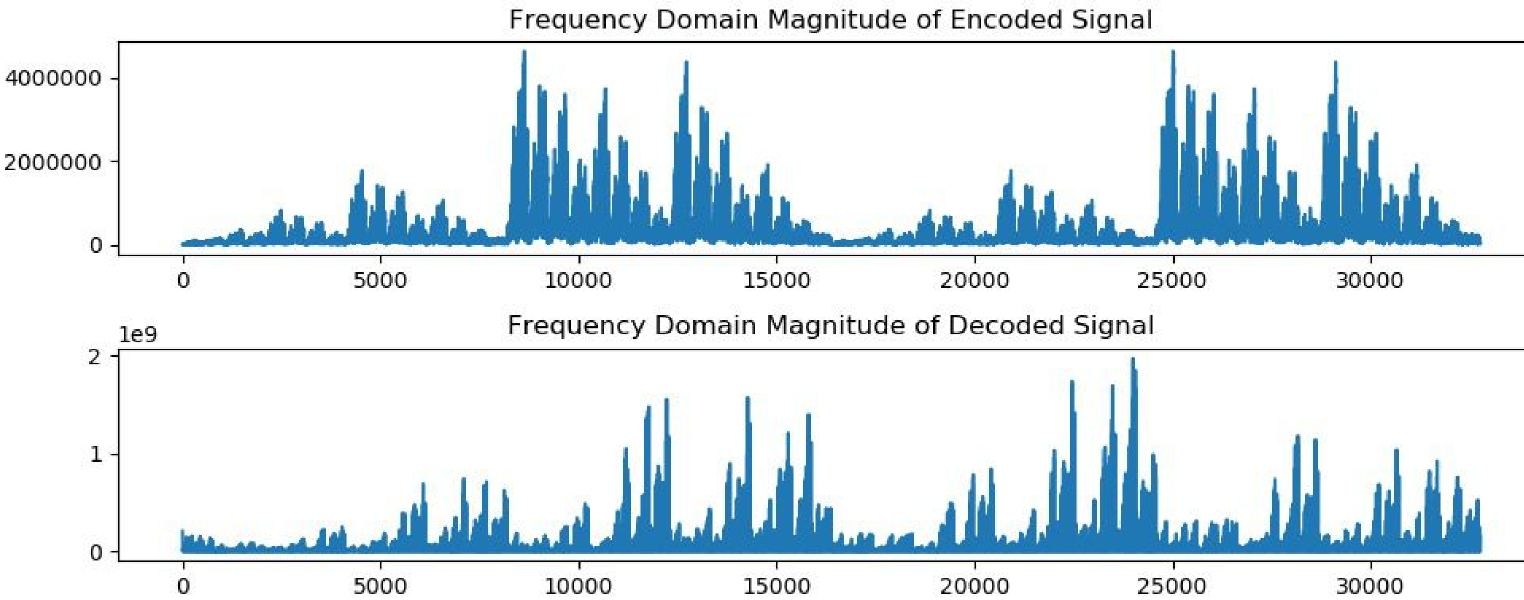
(b) $H(e^{j\omega}) = \frac{6+3e^{-j\omega}+6+2e^{-j\omega}}{6+5e^{-j\omega}+e^{-2j\omega}}$

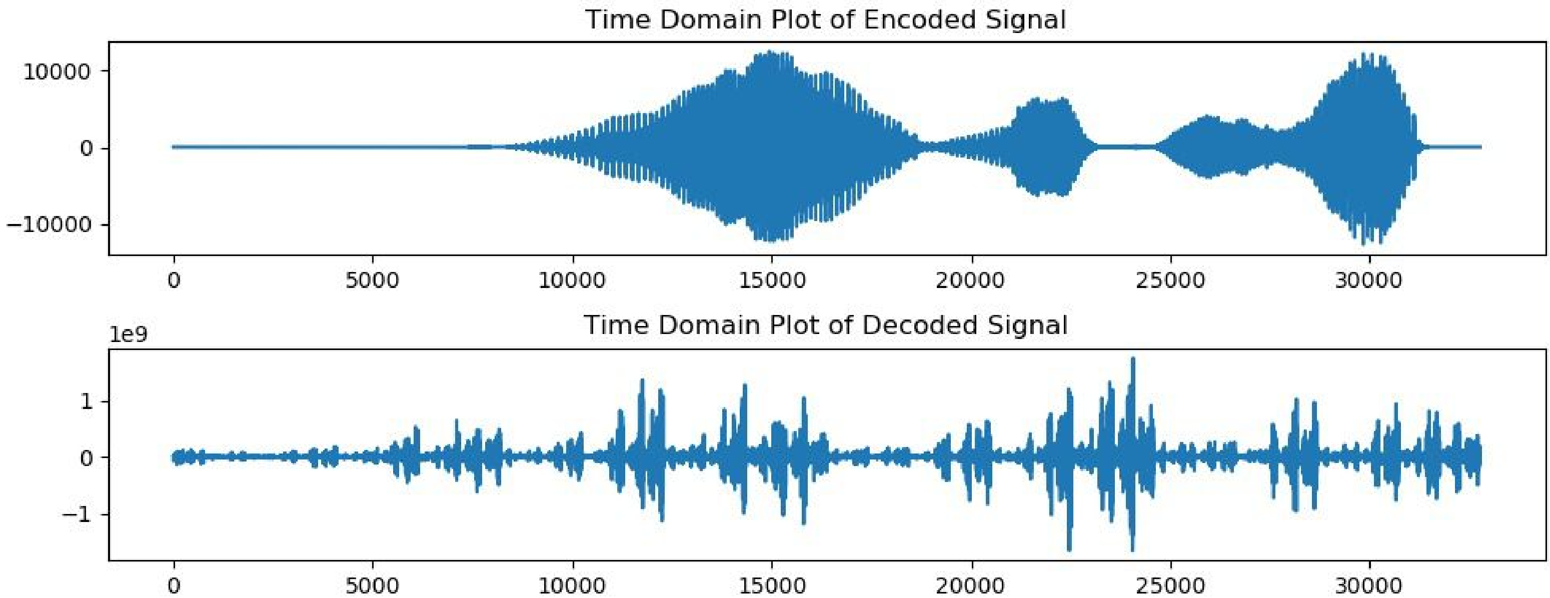
(c) $y[n-2] - 5y[n-1] + 6y[n] = 12x[n] - 5x[n-1]$
Substituting $x[n] = \delta[n]$
$h[n-2] - 5h[n-1] + 6h[n] = 12\delta[n] - 5\delta[n-1]$
for n=0;
$h[-2] - 5h[-1] + 6h[0] = 12\delta[0] - 5\delta[-1] = 12$
for n=1;
$h[-1] - 5h[0] + 6h[1] = 12\delta[1] - 5\delta[0] = -5$
for n=2;
$h[0] - 5h[1] + 6h[2] = 12\delta[2] - 5\delta[1] = 0$
for n≥3;
$h[n-2] - 5h[n-1] + 6h[n] = 12\delta[3] - 5\delta[2] = 0$
Combining all; $h[n] = 5h[n-1] - 6h[n-2] + 12\delta[n] - 5\delta[n-1]$

5.
```
import numpy as np
import matplotlib.pyplot as plt
import scipy.io.wavfile as wav

def fft(signal):
    N = len(signal)
    if N <= 1:
        return signal

    even = fft(signal[::2])
    odd = fft(signal[1::2])

    X = np.zeros(N, dtype=np.complex128)
    for k in range(N//2):
        t = np.exp(-1j * 2 * np.pi * k / N)
        X[k] = even[k] + t * odd[k]

    X[:N//2] += X[N//2:]
    X[N//2:] = X[:N//2] - 2 * np.exp(-1j * np.pi * np.arange(N//2) / N) * X[N//2:]

    return X

def ifft(X):
    N = len(X)
    if N <= 1:
        return X
```

```
    even = ifft(X[0:N:2])
    odd = ifft(X[1:N:2])
    T = [np.exp(2j * np.pi * k / N) * odd[k] for k in range(N // 2)]
    return [even[k] + T[k] for k in range(N // 2)] + \
           [even[k] - T[k] for k in range(N // 2)]

def reverse_concatenate(fourier):
    n = len(fourier)
    positive_freq = fourier[:n//2]
    negative_freq = fourier[n//2:]
    reversed_positive_freq = positive_freq[::-1]
    reversed_negative_freq = negative_freq[::-1]
    return np.concatenate((reversed_positive_freq, reversed_negative_freq))

sample_rate, encoded_data = wav.read('encoded.wav')
fft_data = fft(encoded_data)
fft_data_reversed = reverse_concatenate(fft_data)
decoded_data = ifft(fft_data_reversed)

plt.figure(figsize=(10, 4))
plt.subplot(2, 1, 1)
magnitude_encoded = np.abs(fft_data)
plt.plot(magnitude_encoded)
plt.subplot(2, 1, 2)
magnitude_decoded = np.abs(ifft(fft_data_reversed))
plt.plot(magnitude_decoded)
plt.title('Frequency Domain Magnitude of Decoded Signal')
plt.tight_layout()
plt.show()

plt.figure(figsize=(10, 4))
plt.subplot(2, 1, 1)
plt.plot(encoded_data)
plt.title('Time Domain Plot of Encoded Signal')
plt.subplot(2, 1, 2)
plt.plot(decoded_data)
plt.title('Time Domain Plot of Decoded Signal')
plt.tight_layout()
plt.show()

# Print the decoded message
#decoded_data = decoded_data.real.astype(np.int16)
decoded_npdata = np.array(decoded_data, dtype=np.int16)
decoded_message = decoded_npdata.tobytes().decode('utf-8').strip()
print("Decoded Message:", decoded_message)
```

Decoded Message: I have a dream


Frequency Domain Magnitude of Encoded Signal
4000000
2000000
0
0
5000
10000
15000
20000
25000
30000
Frequency Domain Magnitude of Decoded Signal
1e9
2
1
0
0
5000
10000
15000
20000
25000
30000


Time Domain Plot of Encoded Signal
10000
0
−10000
0
5000
10000
15000
20000
25000
30000
Time Domain Plot of Decoded Signal
1e9
1
0
−1
0
5000
10000
15000
20000
25000
30000