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

Geçit, Emre
e2521581@ceng.metu.edu.tr

Yancı, Baran
e2449015@ceng.metu.edu.tr

June 7, 2023

1. (a)

$$H(j\omega) = \frac{j\omega - 1}{j\omega + 1}$$
$$\frac{Y(j\omega)}{X(j\omega)} = \frac{j\omega - 1}{j\omega + 1}$$
$$Y(j\omega)(j\omega + 1) = X(j\omega)(j\omega - 1)$$
$$y'(t) + y(t) = x'(t) - x(t)$$

(b)

$$H(j\omega) = \frac{j\omega - 1}{j\omega + 1}$$
$$h(t) = \mathcal{F}^{-1}\{H(j\omega)\}$$
$$= \mathcal{F}^{-1}\{\frac{j\omega - 1}{j\omega + 1}\}$$
$$= \mathcal{F}^{-1}\{\frac{j\omega + 1 - 2}{j\omega + 1}\}$$
$$= \mathcal{F}^{-1}\{\frac{j\omega + 1}{j\omega + 1}\} - \mathcal{F}^{-1}\{\frac{2}{j\omega + 1}\}$$
$$= \mathcal{F}^{-1}\{1\} - 2\mathcal{F}^{-1}\{\frac{1}{j\omega + 1}\}$$
$$= \delta(t) - 2e^{-t}u(t)$$

(c)

$$y'(t) + y(t) = x'(t) - x(t)$$
$$y'(t) + y(t) = -2e^{-2t}u(t) - e^{-2t}u(t)$$
$$y'(t) + y(t) = -3e^{-2t}u(t)$$
$$y_p(t) = Ae^{-2t}$$
$$y_p'(t) = -2Ae^{-2t}$$
$$-2Ae^{-2t} + Ae^{-2t} = -3e^{-2t}u(t)$$
$$A = 3$$
$$y_p(t) = 3e^{-2t}$$
$$y_h(t) = c_1 e^{-t}u(t)$$
$$y(t) = y_p(t) + y_h(t)$$
$$= 3e^{-2t} + c_1 e^{-t}u(t)$$
$$y(0) = 0$$
$$0 = 3e^{-2(0)} + c_1 e^{-0}u(0)$$
$$0 = 3 + c_1$$
$$c_1 = -3$$
$$y(t) = 3e^{-2t} - 3e^{-t}u(t)$$

(d)

2. (a)

$$y[n+1] - \frac{1}{2}y[n] = x[n+1]$$
$$e^{j\omega}Y(e^{j\omega}) - \frac{1}{2}Y(e^{j\omega}) = e^{j\omega}X(e^{j\omega})$$
$$H(e^{j\omega}) = \frac{Y(e^{j\omega})}{X(e^{j\omega})}$$
$$H(e^{j\omega}) = \frac{e^{j\omega}}{e^{j\omega} - \frac{1}{2}}$$

(b)

$$H(e^{j\omega}) = \frac{e^{j\omega}}{e^{j\omega} - \frac{1}{2}}$$
$$h[n] = \mathcal{F}^{-1}\{H(e^{j\omega})\}$$
$$= \mathcal{F}^{-1}\{\frac{e^{j\omega}}{e^{j\omega} - \frac{1}{2}}\}$$
$$= \mathcal{F}^{-1}\{\frac{e^{j\omega} - \frac{1}{2} + \frac{1}{2}}{e^{j\omega} - \frac{1}{2}}\}$$
$$= \mathcal{F}^{-1}\{\frac{e^{j\omega} - \frac{1}{2}}{e^{j\omega} - \frac{1}{2}} + \frac{\frac{1}{2}}{e^{j\omega} - \frac{1}{2}}\}$$
$$= \mathcal{F}^{-1}\{1 + \frac{\frac{1}{2}}{e^{j\omega} - \frac{1}{2}}\}$$
$$= \mathcal{F}^{-1}\{1\} + \mathcal{F}^{-1}\{\frac{\frac{1}{2}}{e^{j\omega} - \frac{1}{2}}\}$$
$$= \delta[n] + \frac{1}{2}\mathcal{F}^{-1}\{\frac{1}{e^{j\omega} - \frac{1}{2}}\}$$
$$= \delta[n] + \frac{1}{2}e^{\frac{1}{2}n}u[n]$$

2

(c)

$$y[n+1] - \frac{1}{2}y[n] = x[n+1]$$

$$y[n+1] - \frac{1}{2}y[n] = \left(\frac{3}{4}\right)^{n+1} u[n+1]$$

$$y_p[n] = A\left(\frac{3}{4}\right)^n$$

$$y_p[n+1] = A\left(\frac{3}{4}\right)^{n+1}$$

$$A\left(\frac{3}{4}\right)^{n+1} - \frac{1}{2}A\left(\frac{3}{4}\right)^n = \left(\frac{3}{4}\right)^{n+1} u[n+1]$$

$$\frac{3}{4}A\left(\frac{3}{4}\right)^n - \frac{1}{2}A\left(\frac{3}{4}\right)^n = \frac{3}{4}\left(\frac{3}{4}\right)^n u[n+1]$$

$$\frac{3}{4}A - \frac{1}{2}A = \frac{3}{4}u[n+1]$$

$$A = 3$$

$$y_p[n] = 3\left(\frac{3}{4}\right)^n$$

$$y_h[n] = c_1 \left(\frac{1}{2}\right)^n u[n]$$

$$y[n] = y_p[n] + y_h[n]$$

$$= 3\left(\frac{3}{4}\right)^n + c_1 \left(\frac{1}{2}\right)^n u[n]$$

$$y[0] = 0$$

$$0 = 3\left(\frac{3}{4}\right)^0 + c_1 \left(\frac{1}{2}\right)^0 u[0]$$

$$0 = 3 + c_1$$

$$c_1 = -3$$

$$y[n] = 3\left(\frac{3}{4}\right)^n - 3\left(\frac{1}{2}\right)^n u[n]$$

3. (a)

$$H(j\omega) = H_1(j\omega)H_2(j\omega)$$

$$= \frac{1}{j\omega+1}\frac{1}{j\omega+2}$$

$$= \frac{Y(j\omega)}{X(j\omega)}$$

$$Y(j\omega)(j\omega+1)(j\omega+2) = X(j\omega)$$

$$Y(j\omega)(j^2\omega^2 + 3j\omega + 2) = X(j\omega)$$

$$y''(t) + 3y'(t) + 2y(t) = x(t)$$

(b)

$$H(j\omega) = H_1(j\omega)H_2(j\omega)$$

$$H_1(j\omega) = \frac{1}{j\omega + 1}$$

$$H_2(j\omega) = \frac{1}{j\omega + 2}$$

$$H(j\omega) = \frac{1}{j\omega + 1}\frac{1}{j\omega + 2}$$

$$h_1(t) = \mathcal{F}^{-1}\{H(j\omega)\}$$

$$= \mathcal{F}^{-1}\{\frac{1}{j\omega + 1}\frac{1}{j\omega + 2}\}$$

$$= \mathcal{F}^{-1}\{\frac{1}{j\omega + 1}\} * \mathcal{F}^{-1}\{\frac{1}{j\omega + 2}\}$$

$$= e^{-t}u(t) * e^{-2t}u(t)$$

$$= \int_{-\infty}^{\infty} e^{-(t-\tau)}u(t - \tau)e^{-2\tau}u(\tau)d\tau$$

$$= \int_{0}^{t} e^{-(t-\tau)}e^{-2\tau}d\tau$$

$$= \int_{0}^{t} e^{-t+\tau}e^{-2\tau}d\tau$$

$$= \int_{0}^{t} e^{-t}e^{\tau}e^{-2\tau}d\tau$$

$$= e^{-t}\int_{0}^{t} e^{-\tau}d\tau$$

$$= e^{-t}\left[-e^{-\tau}\right]_{0}^{t}$$

$$= e^{-t}\left[-e^{-t} + 1\right]$$

$$= e^{-t} - e^{-2t}$$

(c)

$$X(j\omega) = j\omega$$

$$Y(j\omega) = H(j\omega)X(j\omega)$$

$$= \frac{1}{j\omega + 1}\frac{1}{j\omega + 2}j\omega$$

$$y(t) = \mathcal{F}^{-1}\{Y(j\omega)\}$$

$$= \mathcal{F}^{-1}\{\frac{j\omega}{(j\omega + 1)(j\omega + 2)}\}$$

$$= \mathcal{F}^{-1}\{\frac{-1}{j\omega + 1} + \frac{2}{j\omega + 2}\}$$

$$= \mathcal{F}^{-1}\{\frac{-1}{j\omega + 1}\} + \mathcal{F}^{-1}\{\frac{2}{j\omega + 2}\}$$

$$= -e^{-t}u(t) + 2e^{-2t}u(t)$$

4. (a)

$$H(e^{j\omega}) = \frac{Y(e^{j\omega})}{X(e^{j\omega})} = \frac{5e^{-j\omega} + 12}{e^{-2j\omega} + 5e^{-j\omega} + 6} \qquad \text{Found in part 4b}$$

$$Y(e^{j\omega})(e^{-2j\omega} + 5e^{-j\omega} + 6) = X(e^{j\omega})(5e^{-j\omega} + 12)$$

$$y[n - 2] + 5y[n - 1] + 6y[n] = 5x[n - 1] + 12x[n]$$

(b)

$$H(e^{j\omega}) = H_1(e^{j\omega}) + H_2(e^{j\omega})$$
$$= \frac{3}{3 + e^{-j\omega}} + \frac{2}{2 + e^{-j\omega}}$$
$$= \frac{5e^{-j\omega} + 12}{e^{-2j\omega} + 5e^{-j\omega} + 6}$$

(c)

$$h[n] = \mathcal{F}^{-1}\{H_1(j\omega) + H_2(j\omega)\}$$
$$= \mathcal{F}^{-1}\{\frac{3}{3 + e^{-j\omega}} + \frac{2}{2 + e^{-j\omega}}\}$$
$$= \mathcal{F}^{-1}\{\frac{3}{3 + e^{-j\omega}}\} + \mathcal{F}^{-1}\{\frac{2}{2 + e^{-j\omega}}\}$$
$$= \frac{-1}{3}u[n] + \frac{1}{2}u[n]$$

5. After decoding **encoded.wav** according to the recipe given, a wav file **decoded.wav** is obtained. When played by a media player, the decoded file says "I have a dream".

The code for the decoding process is given below.

```python
import numpy
from scipy.io import wavfile
from tester import test # This is to verify the implementation, uses numpy.fft

def fft(x: numpy.ndarray) -> numpy.ndarray:
    N = len(x)
    if N <= 1:
        return x

    # Even and odd with mathematical indexing
    odd = fft(x[::2])
    even = fft(x[1::2]) * numpy.exp(-2j * numpy.pi * numpy.arange(N // 2) / N)
    return numpy.concatenate([odd + even,
                              odd - even])

def ifft(X: numpy.ndarray) -> numpy.ndarray:
    return fft(X)[::-1] / len(X)


rate, encoded = wavfile.read("encoded.wav")
N = len(encoded)
dft_coefficients = fft(encoded)
flipped = numpy.flip(dft_coefficients)
decoded = ifft(numpy.concatenate([flipped[N//2 + 1:], flipped[:N//2 + 1]]))
test(encoded, decoded)
wavfile.write("decoded.wav", rate, decoded.real.astype(numpy.int16))
```