CENG 384 - Signals and Systems for Computer Engineers
20232
Written Assignment 2 Solutions

April 3, 2024

---

1. $x(t) = u(t+3) - u(t-7)$ and $h(t) = u(t-1) - u(t-15)$.

   $x(\tau)$ will be nonzero in the interval of $[-3, 7]$.

   $h(t-\tau)$ will be nonzero in the interval of $[t-15, t-1]$.

$$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau$$

   According to overlaps, we will have three different integrals:

   For $-2 \leq t < 8$, partially overlap:

$$y(t) = \int_{-3}^{t-1} 1d\tau = t+2.$$

   For $8 \leq t < 12$, fully overlap:

$$y(t) = \int_{-3}^{7} 1d\tau = 10.$$

   For $12 \leq t < 22$, partially overlap:

$$y(t) = \int_{t-15}^{7} 1d\tau = 22-t.$$

   Therefore,

$$y(t) = \begin{cases} 0, & t < -2, \\ t+2, & -2 \leq t < 8, \\ 10, & 8 \leq t < 12, \\ 22-t, & 12 \leq t < 22, \\ 0, & 22 < t. \end{cases}$$

2. $x[n] = \delta[n] + 2\delta[n-2] - 3\delta[n-4]$ and $h[n] = 2\delta[n+2] + \delta[n-2]$

   (a) $y_1[n] = x[n] * h[n] = \sum_{k=-\infty}^{k=\infty} x[k]h[n-k]$

   $x[k]h[n-k] = (\delta[k] + 2\delta[k-2] - 3\delta[k-4])(2\delta[n-k+2] + \delta[n-k-2])$

   Since $\delta[k] = 0$ for $k \neq 0$, $\delta[k-2] = 0$ for $k \neq 2$ and $\delta[k-4] = 0$ for $k \neq 4$,

   $\sum_{k=-\infty}^{k=\infty} (\delta[k] + 2\delta[k-2] - 3\delta[k-4])(2\delta[n-k+2] + \delta[n-k-2])$ becomes:

   $y_1[n] = 2\delta[n-0+2] + \delta[n-0-2] + 4\delta[n-2+2] + 2\delta[n-2-2] - 6\delta[n-4+2] - 3\delta[n-4-2]$

   $= 2\delta[n+2] + 4\delta[n] - 5\delta[n-2] + 2\delta[n-4] - 3\delta[n-6]$

   (b) $y_2[n] = x[n+2] * h[n] = \sum_{k=-\infty}^{k=\infty} x[k]h[n+2-k] = y_1[n+2]$

   which is a shift of +2:

   $y_2[n] = y_1[n+2] = 2\delta[n+4] + 4\delta[n+2] - 5\delta[n] + \delta[n-2] - 3\delta[n-4]$

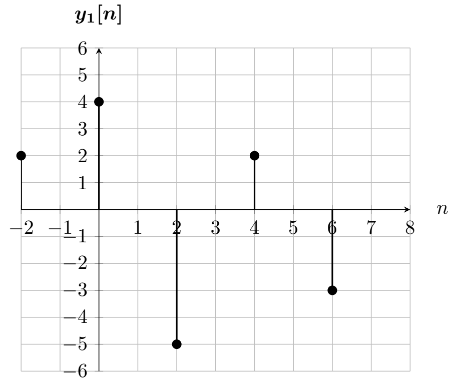


Figure 1: n vs. $y_1[n]$ plot

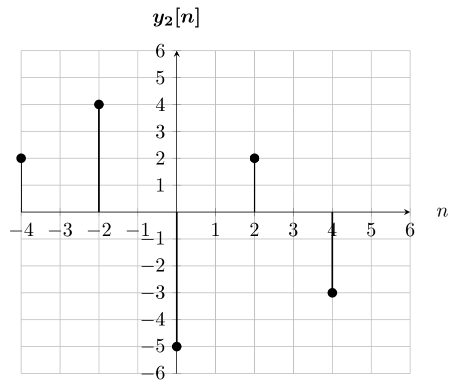


Figure 2: n vs. $y_2[n]$ plot

(c) We know that in LTI systems:

$$x[n] * h[n] = y[n] \Rightarrow x[n - n_0] * h[n - n_1] = y[n - n_0 - n_1]$$

$$y_3[n] = x[n+2] * h[n-2] = y_3[n+2-2] = x[n] * h[n] = y_1[n]$$

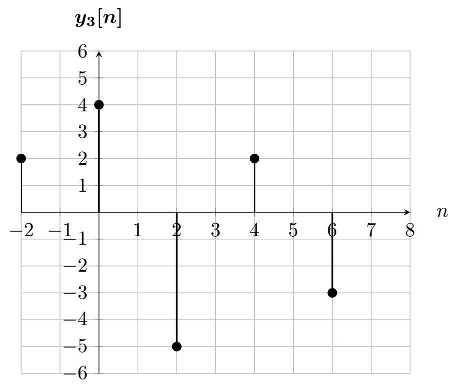


Figure 3: n vs. $y_3[n]$ plot

3. (a) For the input $\delta[n]$, the output will be equal to $h[n]$. Therefore,

$$h[n] = \frac{1}{5}\delta[n-1] + \delta[n].$$

(b)

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k].$$

$$= x[2]h[n-2],$$
$$= \frac{1}{5}\delta[n-3] + \delta[n-2].$$

(c) The system is stable if

$$\sum_{k=-\infty}^{\infty} |h[k]| < \infty.$$

In our case, $\sum_{k=-\infty}^{\infty} |h[k]| = \frac{6}{5}$. Therefore, the system is **stable**.

(d) It **has memory** since $h[n] \neq K\delta[n], \forall K$.

(e) It is **not invertible**. Let's show a counter example. Define an input $x_1[2n] = -2$ and $x_1[2n-1] = 10$. This will generate $-2 + \frac{1}{5}10 = 0$. Let's define another input $x_2[2n] = -4$ and $x_2[2n-1] = 20$. This will generate $-4 + \frac{1}{5}20 = 0$. As we can see different input functions evaluated to same output.

4. (a)

$$\frac{Y(\lambda)}{X(\lambda)} = \frac{2\lambda}{\lambda^2 - 2\lambda + 1}$$
$$y''(t) - 2y'(t) + y(t) = 2x'(t)$$

(b) char eqn. : $r^2 - 2r + 1 = 0 \quad \Rightarrow \quad r_{1,2} = 1 \quad \Rightarrow \quad y_h(t) = C_1 \cdot e^{-t} + C_2 \cdot te^{-t}$

(c) $y_p(t) = At + B$
$y_p'(t) = A$
$y_p''(t) = 0$

$$\begin{aligned} 0 - 2A + At + B &= 4 \\ A(t-2) + B &= 4 \end{aligned}$$

$A = 0$
$B = 4$

$$\begin{aligned} y(t) &= y_h(t) + y_p(t) = C_1 e^{-t} + C_2 te^{-t} + 4 \\ y(0) &= C_1 + 4 = 0 \quad \Rightarrow \quad C_1 = -4 \\ y'(t) &= -C_1 e^{-t} - C_2 e^{-t}(t-1) \\ y'(0) &= -C_1 + C_2 = 0 \quad \Rightarrow \quad C_2 = C_1 = -4 \\ y(t) &= -4e^{-t} - 4te^{-t} + 4 \end{aligned}$$

5. (a) $y[n] = \frac{1}{5}y[n-1] + 2x[n-2]$, putting $x[n] = \delta[n]$ gives $h[n]$:

$h[n] = \frac{1}{5}h[n-1] + 2\delta[n-2]$

$n = 0 \Rightarrow h[0] = \frac{1}{5}h[-1] + 2\delta[-2] = 0$

$n = 1 \Rightarrow h[1] = \frac{1}{5}h[0] + 2\delta[-1] = 0$

$n = 2 \Rightarrow h[2] = \frac{1}{5}h[1] + 2\delta[0] = 2$

We know that when $n > 2$, $\delta[n] = 0$

$n > 2 \Rightarrow h[n] = \frac{1}{5}h[n-1] + 2\delta[n] = \frac{1}{5}h[n-1] = (\frac{1}{5})^2 h[n-2] = ... = (\frac{1}{5})^{n-2} h[2] = 2(\frac{1}{5})^{n-2} = 2(\frac{1}{5})^{n-2} u[n-2]$

(b) $y[n] = \frac{1}{5}y[n-1] + 2x[n-2]$ can be written as a form of difference function, where:

$$\sum_{k=0}^{N} a_k y[n-k] = \sum_{k=0}^{M} b_k x[n-k] \tag{1}$$

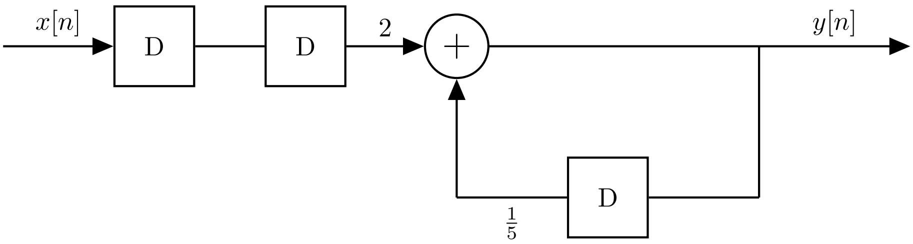
Using Equation 1, coefficients of $y[n] = \frac{1}{5}y[n-1] + 2x[n-2]$ becomes:

$a_0 = 1, a_1 = -\frac{1}{5}, a_{n>1} = 0$ and $b_0 = 0, b_1 = 0, b_2 = 1, b_{n>2} = 0$.

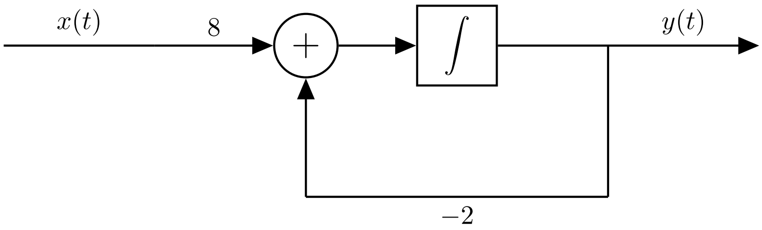
Remember that transfer function of a Discrete Time LTI System is:

$$K(e^{\lambda}) = \frac{\sum_{k=0}^{M} b_k x[n-k]}{\sum_{k=0}^{N} a_k y[n-k]} = \frac{2e^{-2\lambda}}{1 - \frac{1}{5}e^{-\lambda}} \tag{2}$$

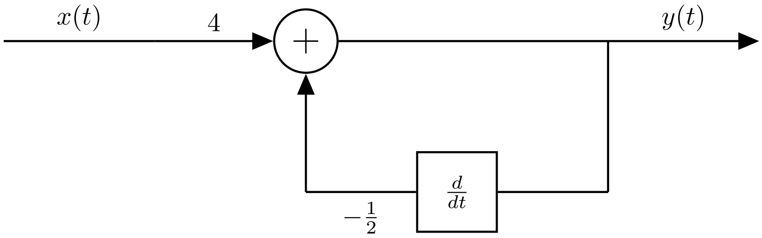
(c) The block diagram representation of this system using the adders and unit delay operators:



6. (a) The block diagram representation of this system using integrators and adders:



(b) The block diagram representation of this system using differentiators and adders:



7.

```python
# Define a recursive function to calculate y[n]
def y(n, y_values):
    # The base case: if n is less than 1, y[n] is 0 because the system is causal
    if n < 1:
        return 0
    # If y[n] has already been calculated, use that value
    if n in y_values:
        return y_values[n]
    # The recursive case: calculate y[n] based on the difference equation
    y_values[n] = (1/4) * y(n-1, y_values) + x(n)
    return y_values[n]

# Define x[n] as the delta function shifted by 1
def x(n):
    return 1 if n == 1 else 0

# Initialize the dictionary to store calculated values of y[n]
y_values = {}

# Calculate y[n] for a range of n
n_values = range(1, 6) # Assuming we want to calculate from n=1 to n=10
y_results = [y(n, y_values) for n in n_values]

import matplotlib.pyplot as plt
plt.stem(n_values, y_results, 'g', markerfmt='go', basefmt=" ", use_line_collection=True)
plt.xlabel('n')
plt.ylabel('y[n]')
plt.title('Recursive Calculation of y[n]')
plt.grid(True)
plt.show()
```