CENG 384 - Signals and Systems for Computer Engineers
Spring 2023
Homework 1

Geçit, Emre
e2521581@ceng.metu.edu.tr

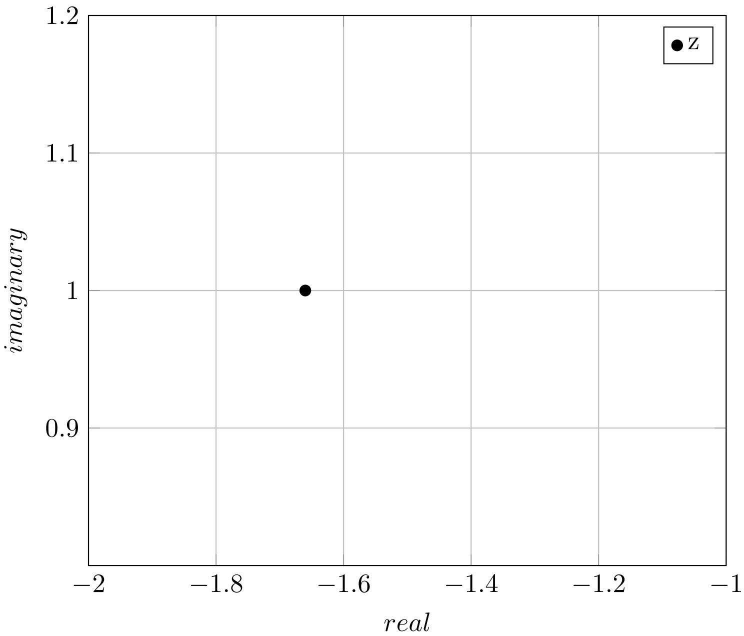
Yancı, Baran
e2449015@ceng.metu.edu.tr

April 1, 2023

---

1. (a)

$$
\begin{aligned}
&z = x + yj \implies \bar{z} = x - yj \\
&2z + 5 = j - \bar{z} \\
&2(x + yj) + 5 = j - (x - yj) \\
&2x + 5 + 2yj = (1 + y)j - x \\
&y = 1, x = \frac{-5}{3} \\
&z = \frac{-5}{3} + j \\
&|z|^2 = \frac{25}{9} + 1 = \frac{34}{9}
\end{aligned}
$$



(b)

$$
\begin{aligned}
&z = re^{j\theta} \implies z^5 = r^5 e^{j5\theta} \\
&32j = 32e^{j\pi/2} \\
&32e^{j\pi/2} = r^5 e^{j5\theta} \implies r = 2, \theta = \pi/10 \\
&z = 2e^{j\pi/10}
\end{aligned}
$$

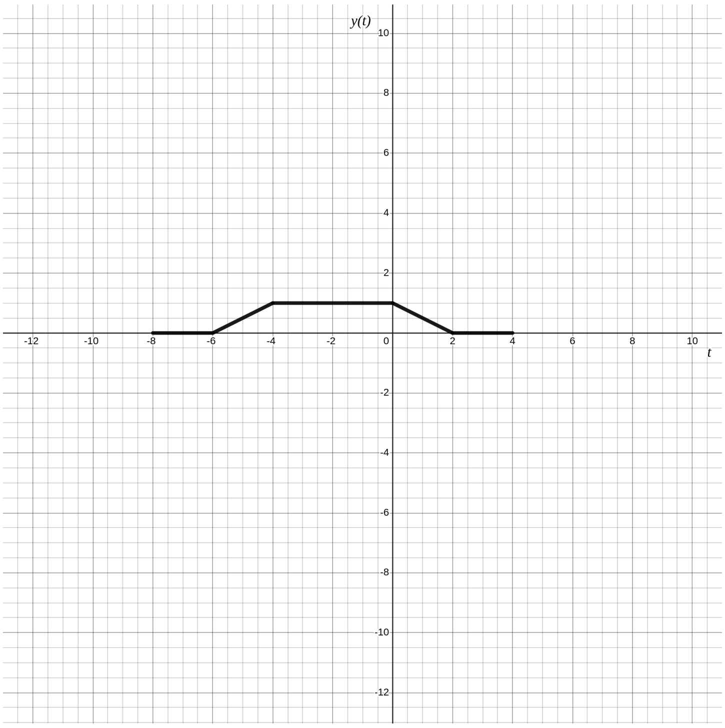
(c)

$$
\begin{aligned}
z &= \frac{(1+j)(\frac{1}{2}+\frac{\sqrt{3}}{2})j}{j-1} \\
&= \frac{(j+1)(1+j)(\frac{1}{2}+\frac{\sqrt{3}}{2})j}{(j+1)(j-1)} \\
&= \frac{(j+1)^2(\frac{1}{2}+\frac{\sqrt{3}}{2})}{-2} \\
&= \frac{(j^2+2j+1)(\frac{1}{2}+\frac{\sqrt{3}}{2})}{-2} \\
&= \frac{(-1+2j+1)(\frac{1}{2}+\frac{\sqrt{3}}{2})}{-2} \\
&= \frac{2j(\frac{1}{2}+\frac{\sqrt{3}}{2})}{-2} \\
&= -j(\frac{1}{2}+\frac{\sqrt{3}}{2}) \\
z &= r\cos\theta + r\sin\theta j \\
j(-\frac{1}{2}-\frac{\sqrt{3}}{2}) &= r\cos\theta + r\sin\theta j \\
r\cos\theta &= 0 \\
r\sin\theta &= -\frac{1}{2}-\frac{\sqrt{3}}{2} \\
\cos\theta &= 0 \\
\sin\theta &= -1 \\
r &= \frac{1}{2}+\frac{\sqrt{3}}{2} \\
\theta &= -\pi/2
\end{aligned}
$$

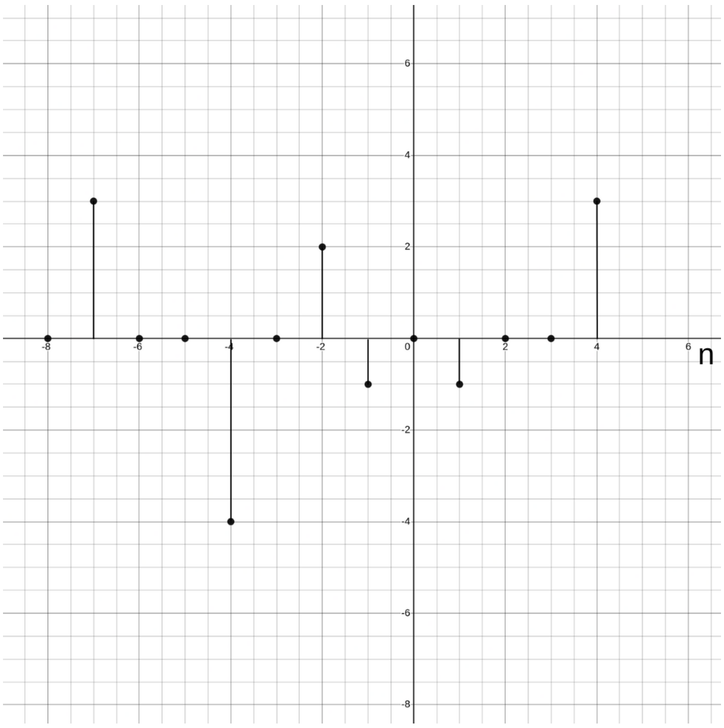
(d)

$$
\begin{aligned}
z &= je^{-j\pi/2} \\
&= e^{j\pi/2}e^{-j\pi/2} \\
&= e^0 = 1
\end{aligned}
$$

2. The graph of the function is given below.



3. (a) The graph of the function $x[-n] + x[2n-1]$ is given below.



(b)

$$\begin{aligned}
x[n] &= -\delta[n-1] + 2\delta[n-2] + -4\delta[n-4] + 3\delta[n-7] \\
x[-n] &= -\delta[-n-1] + 2\delta[-n-2] + -4\delta[-n-4] + 3\delta[-n-7] \\
&= -\delta[n+1] + 2\delta[n+2] + -4\delta[n+4] + 3\delta[n+7] \\
x[2n-1] &= -\delta[2n-2] + 2\delta[2n-3] + -4\delta[2n-5] + 3\delta[2n-8] \\
&= -\delta[n-1] + 3\delta[n-4] \\
x[-n] + x[2n-1] &= -\delta[n+1] + 2\delta[n+2] + -4\delta[n+4] + 3\delta[n+7] + -\delta[n-1] + 3\delta[n-4]
\end{aligned}$$

4. (a)

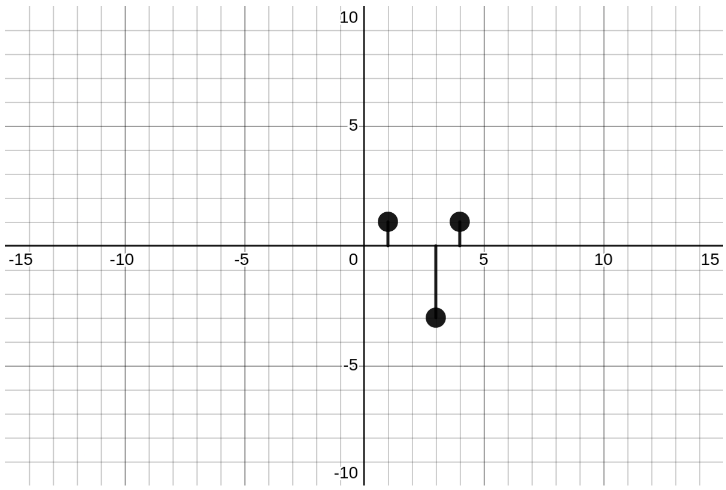
$$\begin{aligned}
period(5\sin(3t-\frac{\pi}{4})) &= period(\sin(3t)) && \text{Scaling the amplitude does not affect the period, neither does shifting.}\\
&= period(\sin(t))/3 && \text{Time scaling inversely affects the period.}\\
&= \frac{2\pi}{3} && \text{Period of sin is } 2\pi.
\end{aligned}$$

(b) $cos[\frac{13\pi}{10}n]$ is periodic with period $\frac{2\pi*10}{13\pi} = \frac{20}{13}$
$sin[\frac{7\pi}{10}n]$ is periodic with period $\frac{2\pi*10}{7\pi} = \frac{20}{7}$
Least common multiple of $\frac{20}{13}$ and $\frac{20}{7}$ is 20

(c) In order $\frac{1}{2}\cos[7n-5]$ to be periodic, its continuous-time counterpart must have a rational period. However, the signal $\frac{1}{2}\cos(7n-5)$ has a fundamental period of $2\pi/7$.
There is no integer t_0 such that,

$$\frac{1}{2}\cos[7n-5] = \frac{1}{2}\cos[7(n+t_0)-5]$$

5. (a) $x(t) = u[t-1] - 3u[t-3] + u[t-4]$
(b) $\frac{dx(t)}{dt} = \delta(t-1) - 3\delta(t-3) + \delta(t-4)$ The graph of $\frac{dx(t)}{dt}$ is given below.



6. (a)

$$y(x) = tx(2x+3)$$

Memory: The system has memory, as the output depends on the future values of the input. For example, for $t=3$, the output value depends on the future value of the input, $y(3) = 3x(9)$.
Stability: The system is **not** stable, as there is no finite number b' such that $|y(t)| \leq b'$ for all t.
Causality: The system is **not** causal, as the output depends on the future values of the input.
Linearity: The system is linear, because the superposition principle holds.

$$\begin{aligned}
x_1 &\implies y_1(t) = tx_1(2t+3)\\
x_2 &\implies y_2(t) = tx_2(2t+3)\\
&\quad x_3 = a_1x_1 + a_2x_2 && \textit{assumption}\\
&\quad y_3 = t(a_1x_1 + a_2x_2) = a_1y_1 + a_2y_2
\end{aligned}$$

Invertibility: The system is invertible for $t \neq 0$.

$$\begin{aligned} y(t) &= tx(2t+3) \\ \frac{1}{t}y(t) &= x(2t+3) \\ \frac{1}{t}y\left(\frac{t-3}{2}\right) &= x(t) \end{aligned}$$

Time invariance: The system is **not** time invariant.

$$\begin{aligned} y(t) &= tx(2t+3) \\ y(t-t_0) = (t-t_0)x(2(t-t_0)+3) &\neq tx(2(t-t_0)+3) \end{aligned}$$

(b)

$$y[n] = \sum_{k=1}^{\infty} x[n-k]$$

Memory: The system has memory, as the output depends on the past values of the input. For example, for $n = 6$, the output value depends on the past values of the input, $y[6] = x[5] + x[4] + x[3] + \cdots$.
Stability: The system is **not** stable, as there is no finite number b' such that $|y(t)| \leq b'$ for all t. For example, if we take $x(n)$ as the unit step function, the signal $\sum_{k=1}^{\infty} u[n-k]$ is unbounded for $n \implies \infty$.
Causality: The system is causal, as the output depends on the past values of the input.
Linearity: The system is linear, because the superposition principle holds.

$$\begin{aligned} x_1 \implies y_1(n) &= \sum_{k=1}^{\infty} x_1[n-k] \\ x_2 \implies y_2(n) &= \sum_{k=1}^{\infty} x_2[n-k] \\ a_1y_1 + a_2y_2 &= \sum_{k=1}^{\infty}(a_1x_1[n-k] + a_2x_2[n-k]) \\ \sum_{k=1}^{\infty} x_3[n-k] &= a_1\sum_{k=1}^{\infty} x_1[n-k] + a_2\sum_{k=1}^{\infty} x_2[n-k] \end{aligned}$$

Invertibility: The system is not invertible, because the output depends on all the past values of the input.
Time invariance: The system is time invariant, because a delay in the input signal $x[n]$ by x, causes a corresponding delay for the output signal $y[n]$ by x as well.

$$\begin{aligned} y[n] &= \sum_{k=1}^{\infty} x[n-k] \\ y[n-n_0] &= \sum_{k=1}^{\infty} x[(n-n_0)-k] \end{aligned}$$

7. (a)

```python
def decompose(signal_name):
    """Read the CSV file with the signal name, decompose the signal into even and odd components,
    and save the results as PNG files."""

    with open(signal_name + ".csv", "r", encoding="ascii") as file:
```

```
    data = [float(item) for item in file.read().split(",")]
start = int(data[0])
signal = data[1:]
end = start + len(signal) - 1

pyplot.title("Original Signal")
pyplot.plot(range(start, end + 1), signal)
pyplot.savefig(IMAGES_PATH + signal_name + "_original.png")
pyplot.clf()

if abs(start) > end:
    signal = signal + [0] * (abs(start) - end)
    end = -start
else:
    signal = [0] * (end - abs(start)) + signal
    start = -end

even = [(x + y) / 2 for x, y in zip(signal, signal[::-1])]
odd = [(x - y) / 2 for x, y in zip(signal, signal[::-1])]

pyplot.title("Even Component")
pyplot.plot(range(start, end + 1), even)
pyplot.savefig(IMAGES_PATH + signal_name + "_even.png")
pyplot.clf()

pyplot.title("Odd Component")
pyplot.plot(range(start, end + 1), odd)
pyplot.savefig(IMAGES_PATH + signal_name + "_odd.png")
pyplot.clf()
```

(a) Original Signal (b) Even Component (c) Odd Component

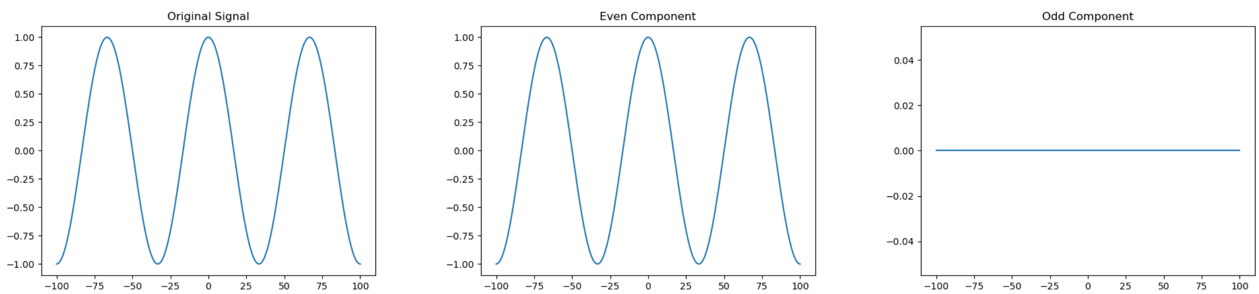


Figure 1: Sinusoidal Signal Decomposition

(a) Original Signal (b) Even Component (c) Odd Component

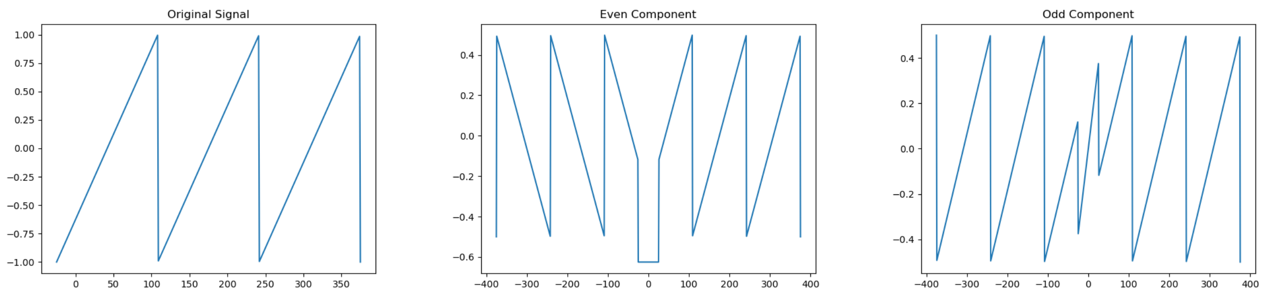


Figure 2: Shifted Sawtooth Signal Decomposition

(a) Original Signal (b) Even Component (c) Odd Component

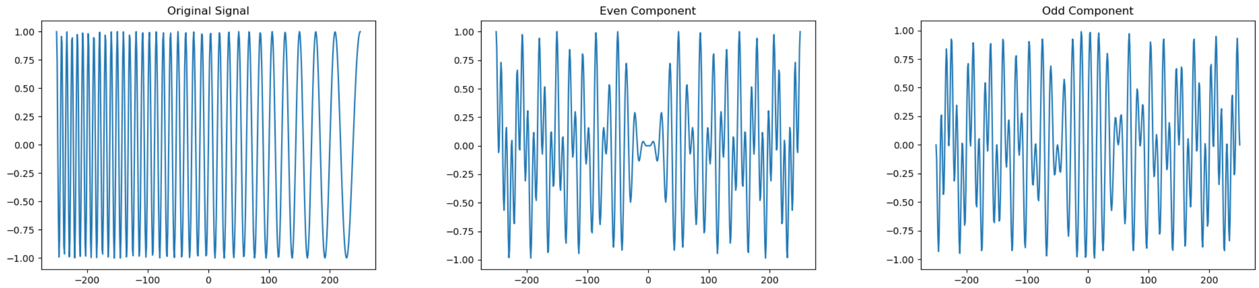


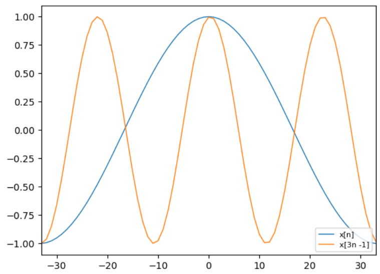
Figure 3: Chirp Signal Decomposition

(b)

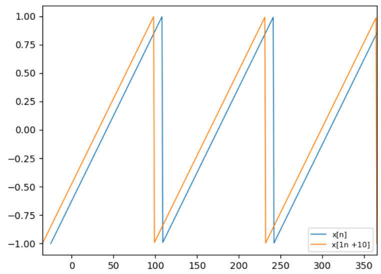
```
class Signal:
    def __init__(self, signal, start, a, b):
        self.signal = signal
        self.start = start
        self.a = a
        self.b = b

    def __getitem__(self, index):
        return self.signal[self.a * index + self.b - self.start]


def shift_n_scale(signal_name):
    """
    Read the CSV file with the signal name, shift and scale the signal,
        and save the results as PNG files.

    This functions reads a signal x[n], and produces x[a*n + b] for a and b
    """

    with open(signal_name + ".csv", "r", encoding="ascii") as file:
        data = [float(item) for item in file.read().split(",")]
    start = int(data[0])
    a = int(data[1])
    b = int(data[2])
    signal = Signal(data[3:], start, a, b)
    end = start + len(signal.signal) - 1

    new_start = (start - b) // a
    new_end = (end - b) // a
    pyplot.xlim(new_start, new_end)

    pyplot.plot(range(start, end + 1), signal.signal,linewidth=1)

    if new_start > new_end:
        domain = range(new_start, new_end, -1)
    else:
        domain = range(new_start, new_end + 1)
    pyplot.plot(
        domain,
        [signal[i] for i in domain],
        linewidth=1,
    )
    pyplot.legend(
        ["x[n]", "x[" + str(a) + "n " + ("+" if b >= 0 else "") + str(b) + "]"],
        loc="lower right",
```

```
    fontsize=8,
)
pyplot.savefig(IMAGES_PATH + signal_name)
pyplot.clf()
```

(a) Sinusoidal Signal



(b) Shifted Sawtooth Signal



(c) Chirp Signal

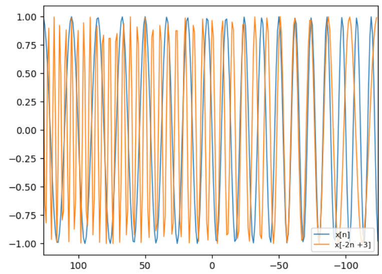


Figure 4: Shift and Scale