Presentation - Matgeo

Aryansingh Sonaye
AI25BTECH11032
EE1030 - Matrix Theory

September 30, 2025

Problem Statement

Problem 12.6

Compute the 4-point Discrete Fourier Transform (DFT) of the sequence

$$x[n] = \{1, 0, 2, 3\}, \quad n = 0, 1, 2, 3. \tag{1.1}$$

Description of Variables used

Symbol	Description	Value
N	Length of sequence	4
$x[n]$	Input sequence	$\{1, 0, 2, 3\}$
W_N	Twiddle factor	$e^{-j2\pi/N}$

Table

Theoretical Solution

The N-point DFT is defined as

$$X[k] = \sum_{n=0}^{N-1} x[n]\, W_N^{kn}, \quad k = 0, 1, \ldots, N-1, \tag{2.1}$$

where

$$W_N = e^{-j\frac{2\pi}{N}}. \tag{2.2}$$

For $N = 4$,

$$W_4 = e^{-j\frac{2\pi}{4}} = -j, \tag{2.3}$$

so that

$$W_4^0 = 1, \quad W_4^1 = -j, \quad W_4^2 = -1, \quad W_4^3 = j. \tag{2.4}$$

Theoretical Solution

The DFT matrix is

$$F_4 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -j & -1 & j \\ 1 & -1 & 1 & -1 \\ 1 & j & -1 & -j \end{pmatrix}. \tag{2.5}$$

The input vector is

$$\mathbf{x} = \begin{pmatrix} 1 \\ 0 \\ 2 \\ 3 \end{pmatrix}. \tag{2.6}$$

Thus,

$$\mathbf{X} = F_4 \, \mathbf{x}. \tag{2.7}$$

Theoretical Solution

Row-by-row computation:

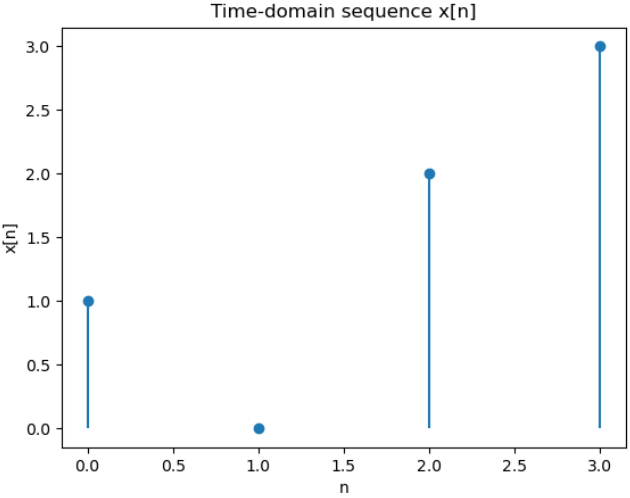
$$X[0] = 1 + 0 + 2 + 3 = 6, \tag{2.8}$$
$$X[1] = 1 + 0(-j) - 2 + 3j = -1 + 3j, \tag{2.9}$$
$$X[2] = 1 + 0(-1) + 2 - 3 = 0, \tag{2.10}$$
$$X[3] = 1 + 0(j) - 2 - 3j = -1 - 3j. \tag{2.11}$$

Therefore, the DFT vector is

$$\mathbf{X} = \begin{pmatrix} 6 \\ -1 + 3j \\ 0 \\ -1 - 3j \end{pmatrix}. \tag{2.12}$$

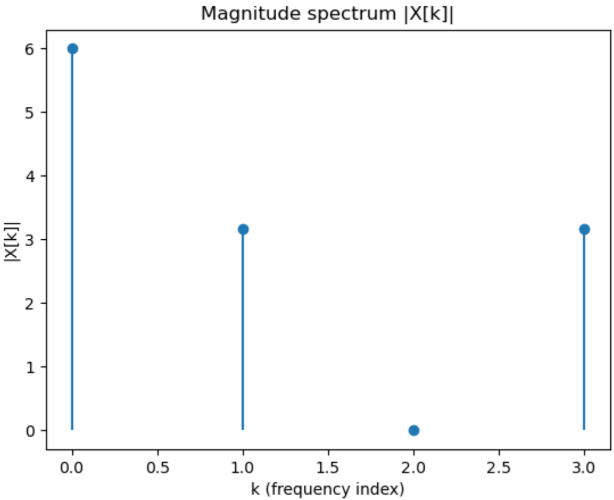
$$\mathbf{X} = \begin{pmatrix} 6 \\ -1 + 3j \\ 0 \\ -1 - 3j \end{pmatrix} \tag{3.1}$$

Plot


Time-domain sequence x[n]
x[n]
n
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0

Figure

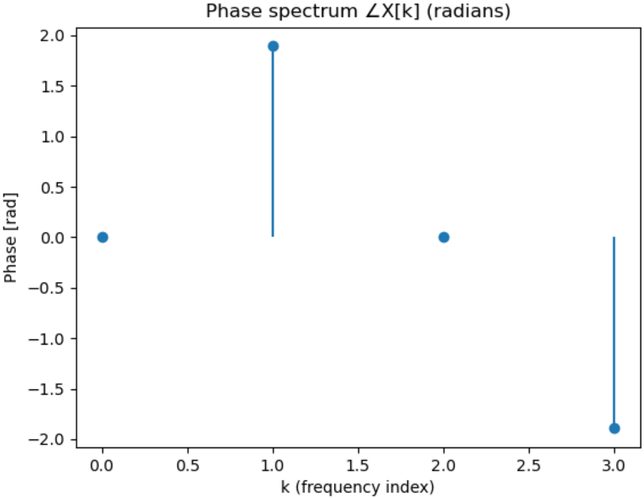
Plot



Magnitude spectrum |X[k]|
|X[k]|
6
5
4
3
2
1
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
k (frequency index)

Figure

Plot



Phase spectrum ∠X[k] (radians)
Phase [rad]
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
k (frequency index)

Figure

Code - C

```c
#include <math.h>

#ifndef M_PI
#define M_PI 3.14159265358979323846
#endif
void dft(const double* x, int N, double* Xr, double* Xi) {
    for (int k = 0; k < N; k++) {
        double sum_r = 0.0, sum_i = 0.0;
        for (int n = 0; n < N; n++) {
            double angle = -2.0 * M_PI * k * n / N;
            sum_r += x[n] * cos(angle);
            sum_i += x[n] * sin(angle);
        }
        Xr[k] = sum_r;
        Xi[k] = sum_i;
    }
}
```

Code - Python(with shared C code)

The code to obtain the required plot is

```
import ctypes
import numpy as np
import matplotlib.pyplot as plt
from numpy.ctypeslib import ndpointer

# Load the C library
lib = ctypes.CDLL("./libdft.so")

# Define function signature
lib.dft.argtypes = [
    ndpointer(ctypes.c_double, flags="C_CONTIGUOUS"), # input x
    ctypes.c_int, # N
    ndpointer(ctypes.c_double, flags="C_CONTIGUOUS"), # output Xr
    ndpointer(ctypes.c_double, flags="C_CONTIGUOUS"), # output Xi
]
lib.dft.restype = None
```

Code - Python(with shared C code)

```
# Input signal
x = np.array([1.0, 0.0, 2.0, 3.0], dtype=np.float64)
N = len(x)

# Allocate outputs
Xr = np.zeros(N, dtype=np.float64)
Xi = np.zeros(N, dtype=np.float64)
# Call C function
lib.dft(x, N, Xr, Xi)

# Convert to magnitude and phase
X = Xr + 1j*Xi
mag = np.abs(X)
phase = np.angle(X)

# --- Plot ---
k = np.arange(N)
```

Code - Python(with shared C code)

```
plt.figure()
plt.stem(k, mag, basefmt=" ")
plt.title("Magnitude Spectrum |X[k]|")
plt.xlabel("k")
plt.ylabel("|X[k]|")
plt.savefig("fig1.png")
plt.show()

plt.figure()
plt.stem(k, phase, basefmt=" ")
plt.title("Phase Spectrum of X[k]")
plt.xlabel("k")
plt.ylabel("Phase [rad]")
plt.savefig("fig2.png")
plt.show()
```

Code - Python only

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

# Input sequence
x = np.array([1, 0, 2, 3])
N = len(x)
n = np.arange(N)

# DFT
X = np.fft.fft(x)
k = np.arange(N)

# --- Plot time-domain sequence ---
plt.figure()
plt.stem(n, x, basefmt=" ")
plt.title("Time-domain sequence x[n]")
plt.xlabel("n")
```

Code - Python only

```
plt.ylabel("x[n]")
plt.savefig("fig3.png")
plt.show()

# --- Plot magnitude spectrum ---
plt.figure()
plt.stem(k, np.abs(X), basefmt=" ")
plt.title("Magnitude spectrum |X[k]|")
plt.xlabel("k (frequency index)")
plt.ylabel("|X[k]|")
plt.savefig("fig4.png")
plt.show()
```

Code - Python only

```
# --- Plot phase spectrum ---
plt.figure()
plt.stem(k, np.angle(X), basefmt=" ")
plt.title("Phase spectrum of X[k] (radians)")
plt.xlabel("k (frequency index)")
plt.ylabel("Phase [rad]")
plt.savefig("fig5.png")
plt.show()
```