Experiment 3. Discrete fourier transform

Name	Prathamesh Mane
UID no. & Branch	**2022200078 (B1)**
Experiment No.	3

AIM:	The aim of this experiment is to study magnitude spectrum of the DT signal.
OBJECTIVE:	1. Develop a function to perform DFT of N point signal 2. Calculate DFT of a DT signal and Plot Spectrum of Signal. 3. Calculate the effect of zero padding on magnitude spectrum
INPUT SPECIFICATION:	1. Length of first Signal N 2. DT Signal values
PROBLEM DEFINITION:	(1) Take any four-point sequence x[n]. Find DFT X[k]. Plot Magnitude Spectrum. (2) Append the input signal by four zeros. Find DFT and plot Magnitude Spectrum Give your conclusion. (3) Expand the input signal by inserting alternate zero. Find DFT and plot Magnitude Spectrum Give your conclusion. (4) Expand the input signal by inserting alternate two zeros. Find DFT and plot Magnitude Spectrum Give your conclusion.
THEORY:	Definition: The Discrete Fourier Transform (DFT) is a mathematical algorithm used to transform a discrete sequence of values from the time domain into the frequency domain. It is a key tool in digital signal processing for analyzing the frequency content of discrete signals. The DFT of a sequence (x[n]) with (N) points is defined as:

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j\frac{2\pi}{N}kn}$$

where:
- (x[n]) is the input time-domain sequence.
- (X[k]) is the output frequency-domain sequence (the DFT).
- (N) is the number of points in the sequence.
- (k) is the index of the frequency component.
- (j) is the imaginary unit.

Physical Significance:

1. Frequency Analysis: The DFT breaks down a discrete-time signal into its constituent frequencies. Each frequency component (X[k]) represents the amplitude and phase of the sinusoidal component of the signal at that frequency.

2. Signal Decomposition: It allows us to analyze how the signal's energy is distributed across different frequencies, which is crucial for understanding signal characteristics and behaviors.

3. Sampling Theorem: It provides insight into the sampling theorem, which states that a continuous signal can be fully reconstructed from its samples if it is band-limited and sampled at a rate greater than twice the highest frequency present.

Magnitude Spectrum:

The magnitude spectrum is derived from the DFT by calculating the magnitude of each complex DFT coefficient:

$$|X[k]| = \sqrt{\mathrm{Re}(X[k])^2 + \mathrm{Im}(X[k])^2}$$

- Re(X[k]) is the real part of the DFT coefficient.
- Im(X[k]) is the imaginary part.

The magnitude spectrum provides a visual representation of how the signal's power or energy is distributed across different frequency components. Peaks in the magnitude spectrum indicate the presence of significant frequency components in the original signal.

	Applications: 1. Signal Processing: The DFT is widely used in digital signal processing to filter, analyze, and modify signals. It helps in tasks like noise reduction, signal enhancement, and feature extraction. 2. Image Processing: In image processing, the DFT is used for tasks like image filtering, compression, and enhancement. It allows operations to be performed in the frequency domain, where certain manipulations can be more efficient.

Theoretical solution using formula

Case 1 : **finding DFT of four point signal**

A=[6, 12 , 7 ,14]

X[0]:
[X[0] = 6 * exp(0) + 12 * exp(0) + 7 * exp(0) + 14 * exp(0)
[X[0] = 6 + 12 + 7 + 14 = 39

X[1]:
X[1] = 6 * exp(-j * 0) + 12 * exp(-j * (pi / 2)) + 7 * exp(-j * pi) + 14 * exp(-j * (3pi / 2))
X[1] = 6 + 12 * (-j) + 7 * (-1) + 14 * (j)
X[1] = 6 - 7 + j * (14 - 12) = -1 + 2j

X[2]:
X[2] = 6 * exp(0) + 12 * exp(-j * pi) + 7 * exp(-j * (2pi)) + 14 * exp(-j * (3pi))
X[2] = 6 - 12 + 7 - 14 = -13

X[3]:
X[3] = 6 * exp(0) + 12 * exp(-j * (3pi / 2)) + 7 * exp(-j * (3pi)) + 14 * exp(-j * (9pi / 2))
X[3] = 6 + 12 * (j) + 7 * (-1) + 14 * (-j)
X[3] = -1 - 2j
Result:
X[k] = [39, -1 + 2j, -13, -1 - 2j]
Magnitude : [39 , 2.24 , 13 , 2.24]

Case 2 : **add four zeros at the end of signal and find the DFT**

A=[6, 12 , 7 ,14 , 0, 0, 0, 0]
Using the result of case one and using the property of DFT
X[k] = [**39, 4.59-25.38j ,-1 + 2j ,7.41-11.38j ,-13, 7.41-11.38j ,-1 - 2j , 4.59-25.38j**]
Magnitude = [39 , 25.80 ,2.24 ,13.59 , 13 ,13.59 , 2.24 ,25.80]

Case 3 : **add one zero between every element and find the DFT**

A=[6,0, 12 ,0, 7 ,0 ,14, 0]
Using the result of case one and using the property of DFT
X[k] = [**39, -1 + 2j, -13, -1 - 2j , 39, -1 + 2j, -13, -1 - 2j**]
Magnitude = [39 , 2.24 , 13 , 2.24, 39 , 2.24 , 13 , 2.24]

Case 4 : **add two zeros between every element and find the DFT**

A=[6,0,0, 12 ,0,0, 7 ,0 ,0,14, 0,0]
Using the result of case one and using the property of DFT
X[k] = [39, -1 + 2j, -13, -1 - 2j, 39, -1 + 2j, -13, -1 - 2j, 39, -1 + 2j, -13, -1 - 2j]
Magnitude = [39 , 2.24 , 13 , 2.24, 39 , 2.24 , 13 , 2.24, 39 , 2.24 , 13 , 2.24]

Developing algorithm using programming language	
PROBLEM STATEMENT:	Write a program in any programming language to perform the discrete fourier transform of the given discrete signal

PROGRAM:	#include <stdio.h>
	#include <math.h>
	
	void compute_dft(int n, float real_in[], float imag_in[], float real_out[], float imag_out[]);
	void compute_idft(int n, float real_in[], float imag_in[], float real_out[], float imag_out[]);
	void compute_magnitude(int n, float real[], float imag[], float magnitude[]);
	void display_magnitude(int n, float magnitude[]);
	
	int main() {
	  int n, i;
	  float real_in[16], imag_in[16];
	  float real_out[16], imag_out[16];
	  float magnitude[16];
	
	  // Initialize input arrays to 0
	  for (i = 0; i < 16; i++) {
	  real_in[i] = 0;
	  imag_in[i] = 0;
	  }
	
	  printf("Enter the length of the signal (N): ");

```
    scanf("%d", &n);

    printf("Enter the real part of the signal x[n]: ");
    for (i = 0; i < n; i++) {
        scanf("%f", &real_in[i]);
    }

    // Compute DFT
    compute_dft(n, real_in, imag_in, real_out, imag_out);

    printf("\nX[k] computed by DFT:\n");
    for (i = 0; i < n; i++) {
        printf("%6.2f + j%6.2f\n", real_out[i], imag_out[i]);
    }

    // Compute Magnitude
    compute_magnitude(n, real_out, imag_out, magnitude);

    // Display Magnitude
    display_magnitude(n, magnitude);

    // Compute IDFT
    compute_idft(n, real_out, imag_out, real_in, imag_in);

    printf("\nx[n] computed by IDFT:\n");
    for (i = 0; i < n; i++) {
        printf("%6.2f + j%6.2f\n", real_in[i], imag_in[i]);
    }

    return 0;
}

void compute_dft(int n, float real_in[], float imag_in[], float real_out[],
float imag_out[]) {
    int k, t;
    float angle;

    for (k = 0; k < n; k++) {
        real_out[k] = 0;
        imag_out[k] = 0;

        for (t = 0; t < n; t++) {
            angle = 2 * M_PI * k * t / n;
```

```
            real_out[k] += real_in[t] * cos(angle) + imag_in[t] * sin(angle);
            imag_out[k] += imag_in[t] * cos(angle) - real_in[t] * sin(angle);
        }
    }
}

void compute_idft(int n, float real_in[], float imag_in[], float real_out[],
float imag_out[]) {
    int t, k;
    float angle;

    for (t = 0; t < n; t++) {
        real_out[t] = 0;
        imag_out[t] = 0;

        for (k = 0; k < n; k++) {
            angle = 2 * M_PI * k * t / n;
            real_out[t] += real_in[k] * cos(angle) - imag_in[k] * sin(angle);
            imag_out[t] += real_in[k] * sin(angle) + imag_in[k] * cos(angle);
        }

        real_out[t] /= n;
        imag_out[t] /= n;
    }
}

void compute_magnitude(int n, float real[], float imag[], float magnitude[])
{
    int i;
    for (i = 0; i < n; i++) {
        magnitude[i] = sqrt(real[i] * real[i] + imag[i] * imag[i]);
    }
}

void display_magnitude(int n, float magnitude[]) {
    int i;
    printf("\nMagnitude of X[k]:\n");
    for (i = 0; i < n; i++) {
        printf("%6.2f\n", magnitude[i]);
    }
}
```

RESULT:

CASE 1 : DFT OF 4 POINT SEQUENCE

```
Enter the length of the signal (N): 4
Enter the real part of the signal x[n]: 6 12 7 14

X[k] computed by DFT:
 39.00 + j  0.00
 -1.00 + j  2.00
-13.00 + j  0.00
 -1.00 + j -2.00

Magnitude of X[k]:
 39.00
  2.24
 13.00
  2.24

x[n] computed by IDFT:
  6.00 + j  0.00
 12.00 + j  0.00
  7.00 + j -0.00
 14.00 + j  0.00
```

CASE 2 : ZERO PADDED SIGNAL (ZEROS AT LAST)

```
Enter the length of the signal (N): 8
Enter the real part of the signal x[n]: 6 12 7 14 0 0 0 0

X[k] computed by DFT:
 39.00 + j  0.00
  4.59 + j-25.38
 -1.00 + j  2.00
  7.41 + j-11.38
-13.00 + j  0.00
  7.41 + j 11.38
 -1.00 + j -2.00
  4.59 + j 25.38

Magnitude of X[k]:
 39.00
 25.80
  2.24
 13.59
 13.00
 13.59
  2.24
 25.80
```

```
x[n] computed by IDFT:
  6.00 + j -0.00
 12.00 + j -0.00
  7.00 + j  0.00
 14.00 + j  0.00
 -0.00 + j  0.00
  0.00 + j -0.00
  0.00 + j  0.00
  0.00 + j  0.00
```

CASE 3 : ZERO PADDED SIGNAL (ALTERNATE ZERO)

```
Enter the length of the signal (N): 8
Enter the real part of the signal x[n]: 6 0 12 0 7 0 14 0

X[k] computed by DFT:
 39.00 + j  0.00
 -1.00 + j  2.00
-13.00 + j  0.00
 -1.00 + j -2.00
 39.00 + j -0.00
 -1.00 + j  2.00
-13.00 + j  0.00
 -1.00 + j -2.00

Magnitude of X[k]:
 39.00
  2.24
 13.00
  2.24
 39.00
  2.24
 13.00
  2.24
```

```
x[n] computed by IDFT:
  6.00 + j -0.00
  0.00 + j -0.00
 12.00 + j -0.00
 -0.00 + j -0.00
  7.00 + j  0.00
 -0.00 + j  0.00
 14.00 + j  0.00
 -0.00 + j  0.00
```

CASE 4 : ZERO PADDED SIGNAL (TWO ZEROS BETWEEN EVERY NUMBER)

```
Enter the length of the signal (N): 12
Enter the real part of the signal x[n]: 6 0 0 12 0 0 7 0 0 14 0 0

X[k] computed by DFT:
 39.00 + j  0.00
 -1.00 + j  2.00
-13.00 + j  0.00
 -1.00 + j -2.00
 39.00 + j -0.00
 -1.00 + j  2.00
-13.00 + j  0.00
 -1.00 + j -2.00
 39.00 + j -0.00
 -1.00 + j  2.00
-13.00 + j  0.00
 -1.00 + j -2.00
```

```
Magnitude of X[k]:
 39.00
  2.24
 13.00
  2.24
 39.00
  2.24
 13.00
  2.24
 39.00
  2.24
 13.00
  2.24

x[n] computed by IDFT:
  6.00 + j -0.00
 -0.00 + j  0.00
  0.00 + j -0.00
 12.00 + j -0.00
 -0.00 + j  0.00
  0.00 + j  0.00
  7.00 + j  0.00
  0.00 + j  0.00
 -0.00 + j -0.00
 14.00 + j  0.00
  0.00 + j  0.00
  0.00 + j  0.00
```

CONCLUSION:	• In first case we calculated the DFT using the formula • In second case we padded zero at the last of the signal and then the result obtained was compared with the result of first case and it was found that the terms at even position remains the same. • In third case we padded zero between every element of the signal and then the result obtained was compared with the result of first case and it was found that the original result is repeated once • When we padded two zeros in between it was found that the result is repeated twice. This means if the input signal is padded by 'm' number of zeros in between then the resultant sequence is also repeated 'm' times. • When the signal is expanded in time domain, spectrum is compressed in frequency domain. That means, Expansion in the time domain by a factor corresponds to the compression of the signal in the Frequency domain by the same factor • DFT converts sequence from Time Domain to Frequency Domain • DFT Converts N samples from time domain to N coefficients in frequency domain • Frequency domain coefficients are separated by W(omega) = 2 pi / N