Institute of Information Technology (IIT)

Jahangirnagar University

Lab Report: 04

Course Code: ICT-4104

Submitted by:

Name: Md Shakil Ahmed

Roll No: 2013

Lab Date: 10-08-2023
Submission Date: 14-08-2023

EXP.NO: 04

Name Of the Experiment: To find the FFT of a given sequence

Objective:

1.To analyses the frequency content of a signal.
2. To efficiently compute the discrete Fourier transform (DFT).

Software: MATLAB

Theory: DFT of a sequence

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

Where N= Length of sequence.
K= Frequency Coefficient.
n = Samples in time domain.

FFT: -Fast Fourier transform.
There are two methods.
1. Decimation in time (DIT) FFT.
2. Decimation in Frequency (DIF) FFT.

The Fast Fourier Transform (FFT) enables rapid analysis of signals by converting them from the time domain to the frequency domain. This aids in identifying underlying frequencies, crucial for applications like audio processing and communication systems. FFT's speed makes real-time analysis efficient and supports tasks requiring frequency pattern understanding.

The no of multiplications in DFT = N^2.
The no of Additions in DFT = N (N-1).
For FFT.
The no of multiplication = N/2 log 2N.
The no of additions = N log2 N.

Algorithm:

Step 1 : Give input sequence x[n].

Step 2 : Find the length of the input sequence using length command.

Step 3 : Find FFT and IFFT using matlab commands fft and ifft.

Step 4 : Plot magnitude and phase response

Step 5 : Display the results.

Flow Chart:

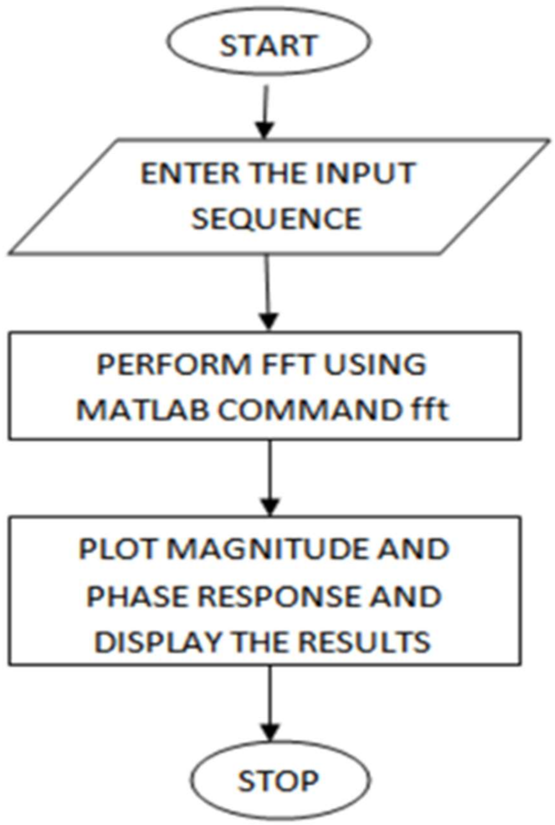


Fig-01: Flowchart

PROGRAM:

```
clc;

clear all;

close all;

x=input('Enter the sequence : ')

N=length(x)

xK=fft(x,N)

xn=ifft(xK)
```

```
n=0:N-1;
subplot (2,2,1);
stem(n,x);
xlabel('n---->');

ylabel('amplitude');
title('input sequence');
 subplot (2,2,2);
stem(n,abs(xK));
xlabel('n---->');
ylabel('magnitude');
title('magnitude response');
 subplot (2,2,3);
stem(n,angle(xK));
xlabel('n---->');
ylabel('phase');
title('Phase responce');
 subplot (2,2,4);
stem(n,xn);
xlabel('n---->');
ylabel('amplitude');


title('IFFT');
```

Output:

Enter the sequence: [1 2 3 4 5]

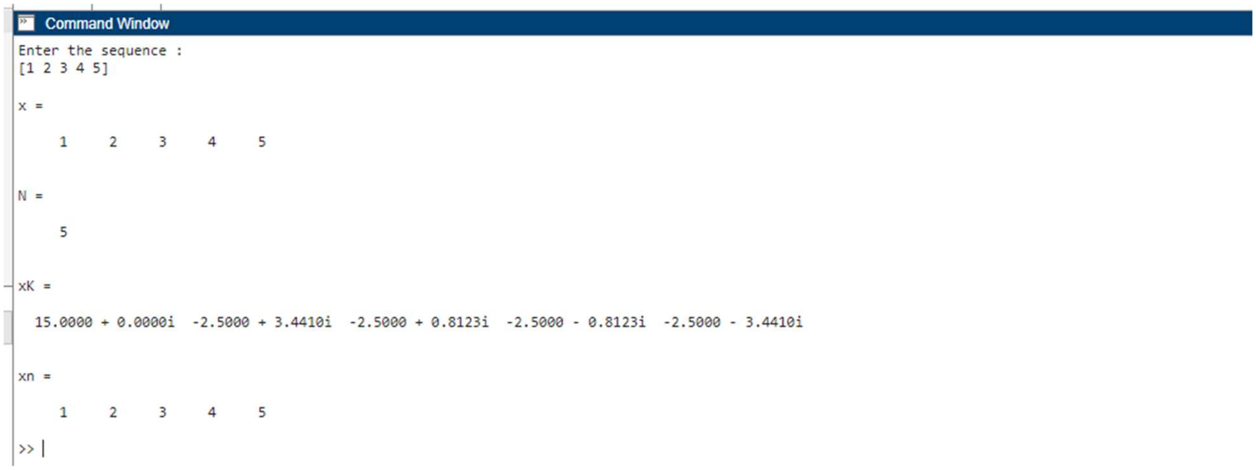


Fig 02: command window

Output Waveforms:

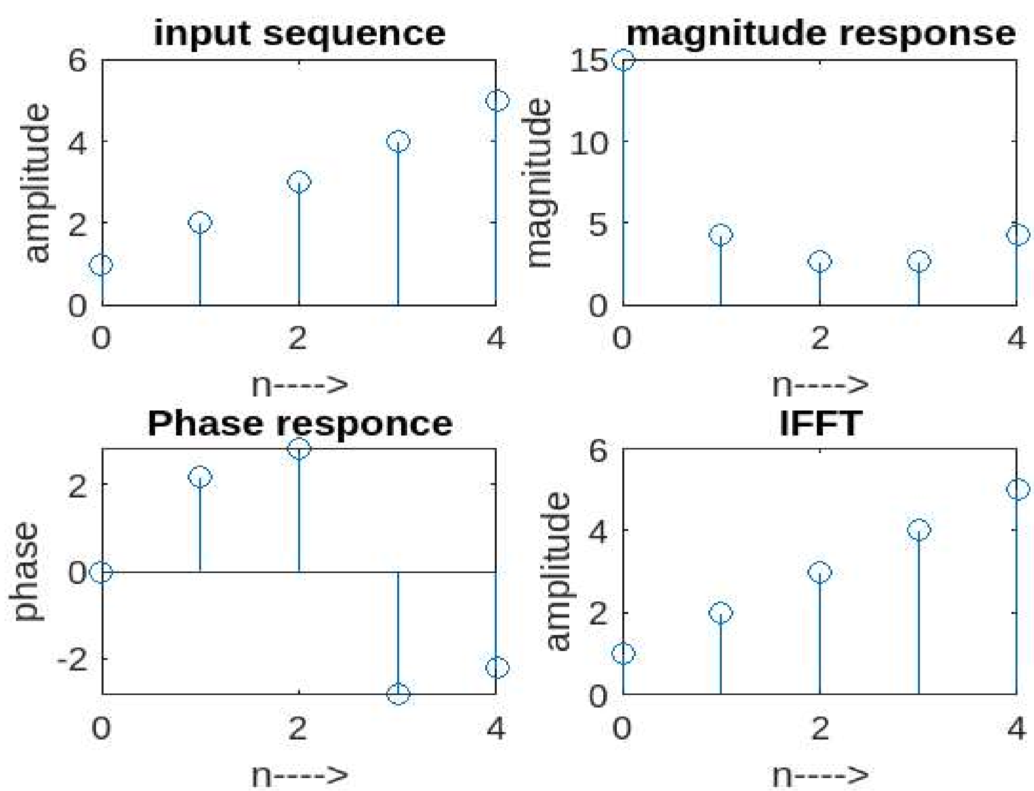


Fig 03: wavelength

Hand Calculation:

Hand calculation:-

Given,

$x(n) = [1\ 2\ 3\ 4\ 5]$

$$x(k) = \sum_{0}^{4} x(n)\, e^{-j\frac{2\pi}{N}kn} \qquad 0 \le k \le 4$$

$$= x(0) + x(1)\, e^{-j\frac{2\pi}{N}k} + x(2)\, e^{-j\frac{2\pi}{N}k\cdot 2} + x(3)\, e^{-j\cdots} + x(4)\, e^{-j\frac{2\pi}{N}k\cdot 4}$$

$K=0 \quad X(0) = 1+2+3+4+5 = 15$

$$k=1 \quad X(1) = 1 + 2\cdot e^{-j\frac{2\pi}{N}} + 3\cdot e^{-j\frac{2\pi}{N}\cdot 2} + 4\cdot e^{-j\frac{2\pi}{N}\cdot 3}$$

$$= 1 + 2e^{-j\frac{2\pi}{5}} + 3e^{-j4\pi/5} + 4e^{-j6\pi/5} + 5e^{-j\frac{8\pi}{5}}$$

$$= 1 + 2(0.309 - 0.951j) + 3(0.809 - j0.5877) + 4(-0.809 + j0.58877) + 5(0.309 + j0.95)$$

$$= -2.5 + j3.6207$$

$$k=2 \quad X(2) = 1 + 2e^{-j\frac{2\pi}{N}\cdot 2} + 3e^{-j\frac{2\pi}{N}2\cdot 2} + 4e^{-j\frac{2\pi}{N}2\times 3} + 5e^{-j\frac{2\pi}{N}2\times 4}$$

$$= 1 + 2e^{-j\frac{4\pi}{5}} + 3e^{-j\frac{8\pi}{5}} + 4e^{-j\frac{12\pi}{5}} + 5e^{-j\frac{16\pi}{5}}$$

$$= 1 + 2(-0.809 - j0.5877) + 3(0.309 + j0.951) + 4(0.309 - j0.951) + 5(-0.809 + j0.58778)$$

Fig 04: hand calculation-01

$$= -2.5 + 0.8123j$$

$$K=3 \quad X(3) = 1 + 2e^{-j\frac{2\pi}{5}\cdot 3} + 3e^{-j\frac{2\pi}{5}\cdot 3\times 2} + 4e^{-j\frac{2\pi}{5}3\times 3} + 5e^{-j\frac{2\pi}{5}3\times 4}$$

$$= 1 + 2(-0.809 + j0.58777) + 3(0.309 - 0.951j) + 4(0.309 + j\,0.951) + 5(-0.809 - j\,0.58778)$$

$$= -2.5 - j0.81234$$

$$K=4 \quad X(4) = 1 + 2e^{-j\frac{2\pi}{5}\cdot 4} + 3e^{-j\frac{2\pi}{5}\cdot 4\times 2} + 4e^{-j\frac{2\pi}{5}4\times 3} + 5e^{-j\frac{2\pi}{5}4\times 4}$$

$$= 1 + 2(0.309 + j0.951) + 3(-0.809 + j0.588) + 4(-0.809 - j0.58778) + 5(0.309 - j0.951)$$

$$= -2.5 - j3.44112$$

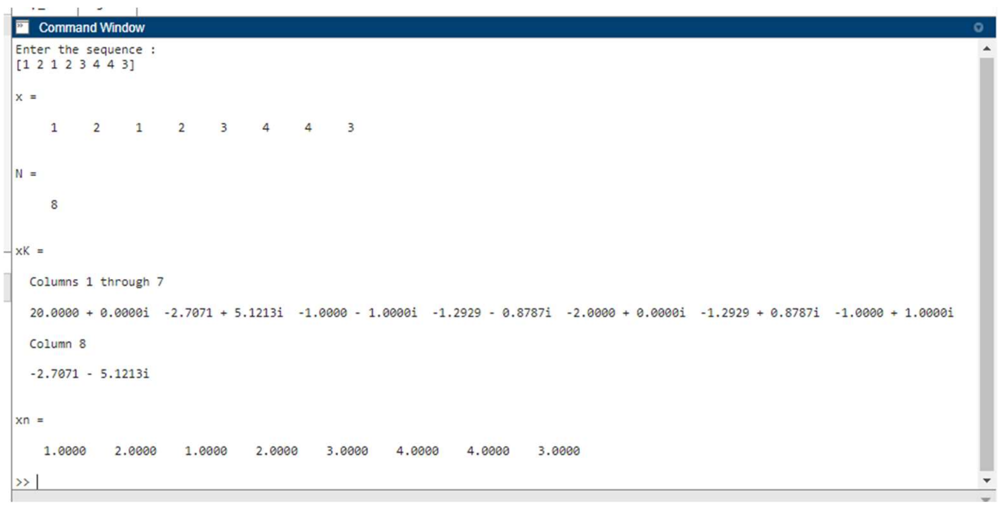
$$X(k) = \{15 \quad -2.5 + j3.6207 \quad -2.5 + j0.8123j \quad -2.5 - j.81234 \quad -2.5 - j\,3.44112\}$$

Fig 05: hand calculation-02

Exercise:

1. Find 8-point DFT of sequence x(n)=[1 2 1 2 3 4 4 3] using FFT algorithm.

Output:



```
Command Window
Enter the sequence :
[1 2 1 2 3 4 4 3]

x =

     1     2     1     2     3     4     4     3


N =

     8


xK =

  Columns 1 through 7

  20.0000 + 0.0000i  -2.7071 + 5.1213i  -1.0000 - 1.0000i  -1.2929 - 0.8787i  -2.0000 + 0.0000i  -1.2929 + 0.8787i  -1.0000 + 1.0000i

  Column 8

  -2.7071 - 5.1213i


xn =

    1.0000    2.0000    1.0000    2.0000    3.0000    4.0000    4.0000    3.0000

>>
```

Fig 06: Command window

Output Waveforms:

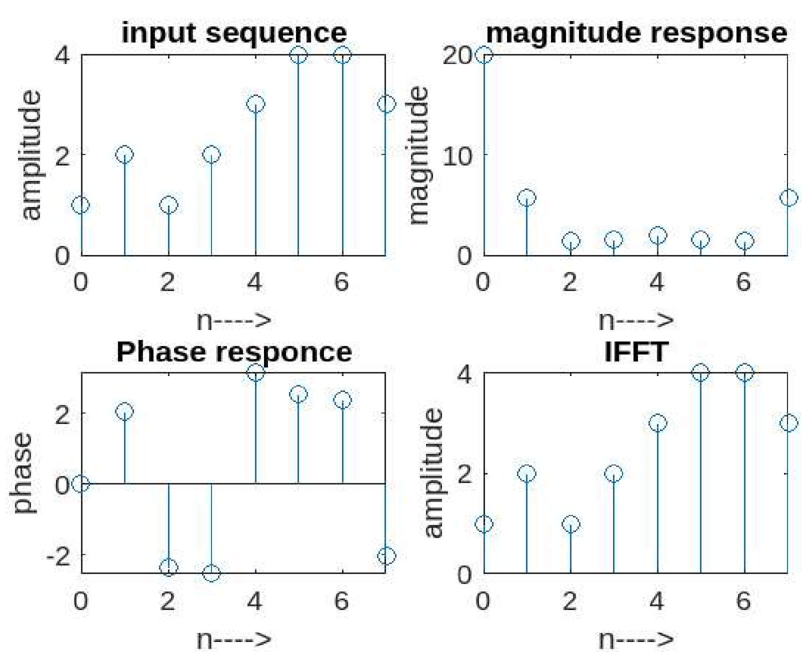


Fig 07: Waveforms

DISCUSSION:

For input sequence [1 2 3 4 5]:

FFT Result (xK):

- The initial value (15.0000 + 0.0000i) signifies the average value (DC component) of the input sequence.
- Complex values (-2.5000 + 3.4410i, -2.5000 + 0.8123i, -2.5000 - 0.8123i, -2.5000 - 3.4410i) represent distinct frequency components.
- Magnitudes indicate contribution strength, while phase angles denote positions within the sequence.

Reconstructed Sequence (xn):

- xn, derived from IFFT on xK, matches the input [1 2 3 4 5], validating FFT and IFFT accuracy.

For input sequence [1 2 3 4 5]:

FFT Result (xK):

- The initial value (20.0000 + 0.0000i) represents the DC component, signifying the average value of the input sequence.
- Complex values like (-2.7071 + 5.1213i) indicate different frequency components. Magnitudes reflect strengths, phase angles indicate positions.
- Negative frequency components have symmetric counterparts with conjugate values.

Reconstructed Sequence (xn):

- The reconstructed sequence xn corresponds to the IFFT result, accurately restoring the original sequence.

The code shows how to use FFT and IFFT to convert an input sequence from the time domain to the frequency domain and back again. Magnitude/phase visualization provides insights into frequency. IFFT precisely reconstructs the original, demonstrating the accuracy of the transformation. This approach aids signal processing by studying frequency content and manipulation. The code underscores FFT/IFFT's strength in signal analysis, showcasing their power.

CONCLUSION:

An essential tool for studying frequency-domain signals is the Fast Fourier Transform (FFT). It effectively separates a given sequence into its individual frequency components, revealing important details about its spectral properties. The FFT improves our comprehension of signals in a variety of applications, from audio processing to picture analysis, by highlighting the presence of particular frequencies and their amplitudes. It is a cornerstone of contemporary signal processing due to its computational effectiveness and capacity for handling big datasets.

VIVA QUESTIONS:

1. Define transform. What is the need for transform?

A transform is a mathematical procedure that changes the domain of an input signal. To analyse the signal in a different domain and learn more about it, a transform is required. When a signal is transformed from the time domain to the frequency domain, as is the case with the Fourier transform, we may observe the many frequencies that are present in the signal.

2. Differentiate Fourier transform and discrete Fourier transform.

Key Differences:

1. Fourier Transform (FT) is for continuous-time signals and provides a continuous frequency spectrum.
2. Discrete Fourier Transform (DFT) is for discrete-time signals (samples) and provides a discrete frequency spectrum.
3. FT uses integrals for continuous functions, while DFT uses summations for discrete data.
4. FT is for analog signals, and DFT is for digital signals.
5. DFT is efficiently calculated using algorithms like FFT for digital processing.

3. Differentiate DFT and DTFT.

The Discrete Fourier Transform (DFT) and Discrete-Time Fourier Transform (DTFT) are distinct tools in signal processing:
DFT: Analyzes discrete sequences, producing a finite set of frequency components in a periodic spectrum. Useful for digital signals and computations like audio analysis and image processing. FFT is an efficient DFT implementation.
DTFT: Analyzes continuous signals after discrete sampling, yielding a continuous frequency spectrum. It extends infinitely in time, revealing detailed frequency characteristics. Essential for understanding continuous signals in applications like analog-to-digital conversion and signal analysis.

4. What are the advantages of FFT over DFT?

The FFT is a faster algorithm for computing the DFT. It is based on the divide-and-conquer algorithm, which breaks the DFT into smaller DFTs that can be computed recursively. This makes the FFT much faster than the naive DFT algorithm, which computes the DFT directly.

5. Differentiate DITFFT and DIFFFT algorithms.

The key difference between DIT-FFT and DIF-FFT lies in the domain where butterfly operations are performed: DIT-FFT uses butterfly operations in the time domain and starts with smaller subsequences, while DIF-FFT uses butterfly operations in the frequency domain and starts with smaller frequency components. Both algorithms aim to optimize the calculations and reduce redundancy, achieving computational efficiency.

6. What is meant by radix?

The base of the number system used to represent the DFT coefficients is referred to as the radix in the context of the FFT. The 2, 4, and 8 radixes are the most used ones for the FFT. The radix of the FFT affects the speed and memory efficiency of the algorithm.

For example, a radix-2 FFT divides the DFT into two halves, which can be computed recursively. This makes the radix-2 FFT much faster than a radix-1 FFT, which computes the DFT directly.

7. What is meant by twiddle factor and give its properties?

A twiddle factor is a complex exponential used in the Discrete Fourier Transform and FFT algorithms. Its properties, including symmetry, multiplicative behavior, unity roots, and phase shift, simplify calculations by reducing redundancy. Twiddle factors are precomputed and play a crucial role in optimizing the efficiency of signal transformation between time and frequency domains.

8. How FFT is useful to represent a signal?

The FFT is useful to represent a signal because it can be used to decompose the signal into its constituent frequencies. This can be helpful for analyzing the signal and for performing signal processing tasks such as filtering and modulation.

For example, if we have a signal that contains a sine wave at 100 Hz, the FFT will show us a peak at 100 Hz in the frequency domain. This tells us that the signal contains a sine wave at 100 Hz.

9. Compare FFT and DFT with respect to number of calculation required?

The Discrete Fourier Transform (DFT) and the Fast Fourier Transform (FFT) are both algorithms for computing the Fourier transform of a sequence. The DFT requires O(N^2) operations to compute, while the FFT requires O(N log N) operations. This means that the FFT is much faster than the DFT, especially for large data sets.

10. How the original signal is reconstructed from the FFT of a signal?

The original signal can be reconstructed from the FFT of a signal by taking the inverse FFT of the FFT coefficients. This process is called the inverse Fourier transform. The inverse FFT is essentially the same as the FFT, but it works in reverse. It takes the DFT coefficients and reconstructs the original signal.

References:

[1]"Fast Fourier transform -Algorithms for Competitive Programming," *Cp-algorithms.com*,2022.Available:https://cpalgorithms.com/algebra/fft.html?fbclid=IwAR3nRgPlxN8IhhAcxx6aZ5EkV2nBL4vc7lpPr_byB2deUzn99g0eI4zJqBA#improved-implementation-in-place-computation. [Accessed: Aug. 13, 2023].

[2]tutorialspoint,"dspfastfouriertransform".[Online].Available:https://www.tutorialspoint.com/digital_signal_processing/dsp_fast_fourier_transform.h tm[Accessed: august. 13, 2023,09:10am]