**Experiment No. 4**

**Aim:** To apply Discrete Fourier Transform on DT signal

**Objective:**

1. Write a code to perform DFT of N point signal

2. Calculate DFT of a DT signal

3.Calculate FFT of same signal

**Input Specifications:**

1. Length of Signal N

2. Signal values

**Theory:**

**Discrete Fourier Transform**

**Discrete Fourier transform** (**DFT**) converts a finite list of equally spaced [samples](http://en.wikipedia.org/wiki/Sampling_(signal_processing)) of a [function](http://en.wikipedia.org/wiki/Function_(mathematics)) into the list of [coefficients](http://en.wikipedia.org/wiki/Coefficient) of a finite combination of [complex](http://en.wikipedia.org/wiki/Complex_number) [sinusoids](http://en.wikipedia.org/wiki/Sine_wave), ordered by their [frequencies](http://en.wikipedia.org/wiki/Frequency), that has those same sample values. It can be said to convert the sampled function from its original domain (often [time](http://en.wikipedia.org/wiki/Time_domain) or position along a line) to the [frequency domain](http://en.wikipedia.org/wiki/Frequency_domain).

The input samples are [complex numbers](http://en.wikipedia.org/wiki/Complex_number) (in practice, usually [real numbers](http://en.wikipedia.org/wiki/Real_number)), and the output coefficients are complex as well. The frequencies of the output sinusoids are integer multiples of a fundamental frequency, whose corresponding period is the length of the sampling interval. The combination of sinusoids obtained through the DFT is therefore [periodic](http://en.wikipedia.org/wiki/Periodic_function) with that same period. The DFT differs from the [discrete-time Fourier transform](http://en.wikipedia.org/wiki/Discrete-time_Fourier_transform) (DTFT) in that its input and output sequences are both finite; it is therefore said to be the Fourier analysis of finite-domain (or periodic) discrete-time functions.

The [sequence](http://en.wikipedia.org/wiki/Sequence) of **N** [complex numbers](http://en.wikipedia.org/wiki/Complex_number) x_0, x_1, \ldots, x_{N-1} is transformed into an **N**-periodic sequence of complex numbers as follows

X_k\ \stackrel{\text{def}}{=}\ \sum_{n=0}^{N-1} x_n \cdot e^{-i 2 \pi k n / N},  \quad k\in\mathbb{Z}\,

**Problem Definition:**

1. Take any four-point & eight-point sequence x[n].

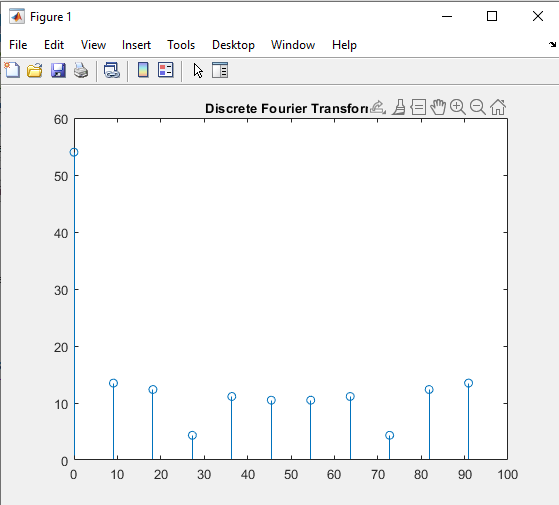
* Find DFT X[k].

**Code**

| **clc; signal\_input=input("Enter Signal: "); y=fft(signal\_input); x = (0:length(y)-1)\*100/length(y); m = abs(y); y(m<1e-6) = 0; subplot(1,1,1),stem(x, m); title("Discrete Fourier Transform", 'FontSize', 10);** |
| --- |

**Output**

| **Enter Signal: [1 3 4 6 2 6 4 9 12 2 5]** |
| --- |

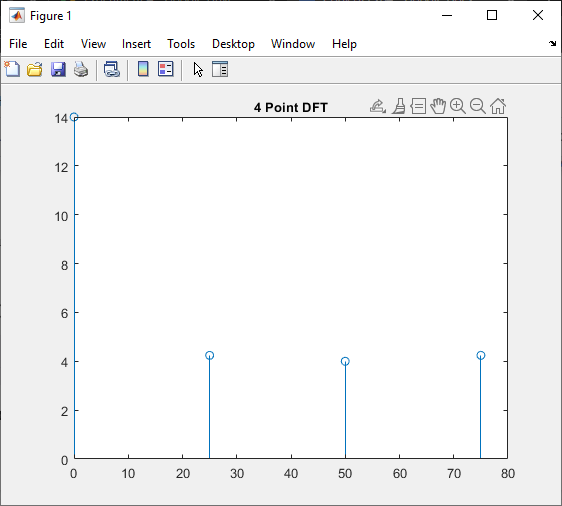
****

**Code ( 4 Point DFT )**

| **clc; signal\_input=input("Enter Signal: "); y=fft(signal\_input, 4); x = (0:length(y)-1)\*100/length(y); m = abs(y); y(m<1e-6) = 0; subplot(1,1,1),stem(x, m); title("4 Point DFT", 'FontSize', 10);** |
| --- |

**Output**

| **Enter Signal: [1 3 4 6 2 6 4 9 12 2 5]** |
| --- |

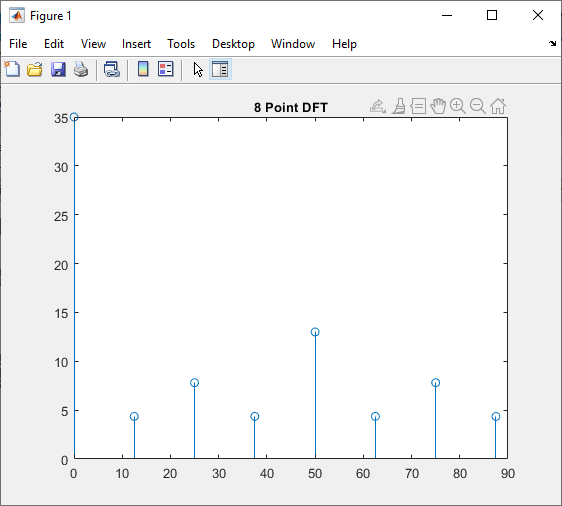
****

**Code ( 8 Point DFT )**

| **clc; signal\_input=input("Enter Signal: "); y=fft(signal\_input, 8); x = (0:length(y)-1)\*100/length(y); m = abs(y); y(m<1e-6) = 0; subplot(1,1,1),stem(x, m); title("8 Point DFT", 'FontSize', 10);** |
| --- |

**Output:**

| **Enter Signal: [1 3 4 6 2 6 4 9 12 2 5]** |
| --- |

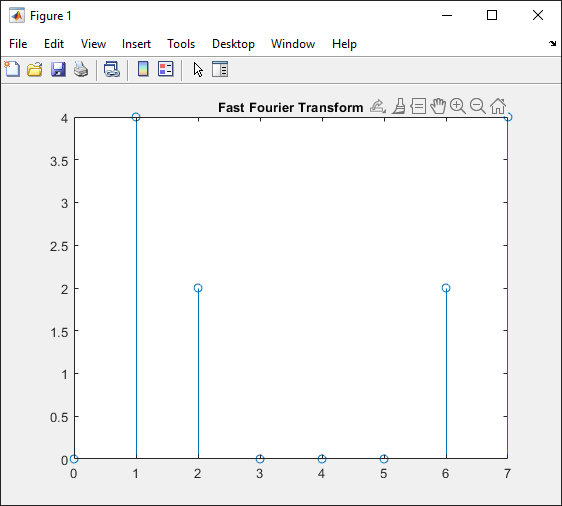
****

**Code (FFT)**

| **clc; input\_signal = input('Input N : '); signal = zeros(1, input\_signal); for k = 0:input\_signal-1 signal(1,k+1) = sin((pi/4)\*k) + 0.5\*sin((pi/2)\*k); end FFT = fft(signal); FFT\_Value = abs(FFT); disp(FFT\_Value); range = 0:input\_signal-1; subplot(1,1,1),stem(range, FFT\_Value); title("Fast Fourier Transform", 'FontSize', 10);** |
| --- |

**Output:**

| Input N : 8  0.0000 4.0000 2.0000 0.0000 0.0000 0.0000 2.0000 4.0000 |
| --- |

****

**Conclusion**

In this experiment, we learnt about Discrete Fourier Transform and Fast Fourier Transform. We then implemented these using MATLAB.

**SAP ID: 60004190057**

**Name: Junaid Altaf Girkar**