**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 (For Signal Length = 4)**

| **clc clear all; x1=input('Enter the signal of length equal to 4:'); N= length(x1); for n=1:N for k=1:N W(n,k)=exp(-2\*1i\*pi\*(n-1)\*(k-1)/N); end end W** |
| --- |

**Output**

| Enter the signal of length equal to 4:[1,2,3,4]  W =   1.0000 + 0.0000i 1.0000 + 0.0000i 1.0000 + 0.0000i 1.0000 + 0.0000i  1.0000 + 0.0000i 0.0000 - 1.0000i -1.0000 - 0.0000i -0.0000 + 1.0000i  1.0000 + 0.0000i -1.0000 - 0.0000i 1.0000 + 0.0000i -1.0000 - 0.0000i  1.0000 + 0.0000i -0.0000 + 1.0000i -1.0000 - 0.0000i 0.0000 - 1.0000i  >> Y = W \* x1'; >> Y  Y =   10.0000 + 0.0000i  -2.0000 + 2.0000i  -2.0000 - 0.0000i  -2.0000 - 2.0000i |
| --- |

**Code (For Signal Length = 8)**

| **clc clear all; x1=input('Enter the signal of length equal to 8:'); N= length(x1); for n=1:N for k=1:N W(n,k)=exp(-2\*1i\*pi\*(n-1)\*(k-1)/N); end end W** |
| --- |

**Output**

| Enter the signal of length equal to 8:[1,2,3,4,5,6,7,8]  W =   1.0000 + 0.0000i 1.0000 + 0.0000i 1.0000 + 0.0000i 1.0000 + 0.0000i 1.0000 + 0.0000i 1.0000 + 0.0000i 1.0000 + 0.0000i 1.0000 + 0.0000i  1.0000 + 0.0000i 0.7071 - 0.7071i 0.0000 - 1.0000i -0.7071 - 0.7071i -1.0000 - 0.0000i -0.7071 + 0.7071i -0.0000 + 1.0000i 0.7071 + 0.7071i  1.0000 + 0.0000i 0.0000 - 1.0000i -1.0000 - 0.0000i -0.0000 + 1.0000i 1.0000 + 0.0000i 0.0000 - 1.0000i -1.0000 - 0.0000i -0.0000 + 1.0000i  1.0000 + 0.0000i -0.7071 - 0.7071i -0.0000 + 1.0000i 0.7071 - 0.7071i -1.0000 - 0.0000i 0.7071 + 0.7071i 0.0000 - 1.0000i -0.7071 + 0.7071i  1.0000 + 0.0000i -1.0000 - 0.0000i 1.0000 + 0.0000i -1.0000 - 0.0000i 1.0000 + 0.0000i -1.0000 - 0.0000i 1.0000 + 0.0000i -1.0000 - 0.0000i  1.0000 + 0.0000i -0.7071 + 0.7071i 0.0000 - 1.0000i 0.7071 + 0.7071i -1.0000 - 0.0000i 0.7071 - 0.7071i -0.0000 + 1.0000i -0.7071 - 0.7071i  1.0000 + 0.0000i -0.0000 + 1.0000i -1.0000 - 0.0000i 0.0000 - 1.0000i 1.0000 + 0.0000i -0.0000 + 1.0000i -1.0000 - 0.0000i -0.0000 - 1.0000i  1.0000 + 0.0000i 0.7071 + 0.7071i -0.0000 + 1.0000i -0.7071 + 0.7071i -1.0000 - 0.0000i -0.7071 - 0.7071i -0.0000 - 1.0000i 0.7071 - 0.7071i  >> Y = W \* x1'; >> Y  Y =   36.0000 + 0.0000i  -4.0000 + 9.6569i  -4.0000 + 4.0000i  -4.0000 + 1.6569i  -4.0000 - 0.0000i  -4.0000 - 1.6569i  -4.0000 - 4.0000i  -4.0000 - 9.6569i |
| --- |

**Code (For FFT Function)**

| **clc clear all; x1=input('Enter the signal of length equal to 8:'); fft(x1)** |
| --- |

**Output:**

| Enter the signal of length equal to 8:[1 2 3 4 5 6 7 8]  ans =   36.0000 + 0.0000i -4.0000 + 9.6569i -4.0000 + 4.0000i -4.0000 + 1.6569i -4.0000 + 0.0000i -4.0000 - 1.6569i -4.0000 - 4.0000i -4.0000 - 9.6569i |
| --- |

**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**