Experiment No.5

**Aim:** To study magnitude spectrum of the DT signal.

**Theory:**

In [mathematics](https://en.wikipedia.org/wiki/Mathematics), the discrete Fourier transform (DFT) converts a finite sequence of equally-spaced [samples](https://en.wikipedia.org/wiki/Sampling_(signal_processing)) of a [function](https://en.wikipedia.org/wiki/Function_(mathematics)) into an equivalent-length sequence of equally-spaced samples of the [discrete-time Fourier transform](https://en.wikipedia.org/wiki/Discrete-time_Fourier_transform) (DTFT), which is a [complex-valued](https://en.wikipedia.org/wiki/Complex_number) function of frequency. The interval at which the DTFT is sampled is the reciprocal of the duration of the input sequence. An inverse DFT is a [Fourier series](https://en.wikipedia.org/wiki/Fourier_series), using the DTFT samples as coefficients of [complex](https://en.wikipedia.org/wiki/Complex_number) [sinusoids](https://en.wikipedia.org/wiki/Sine_wave) at the corresponding DTFT frequencies. It has the same sample-values as the original input sequence. The DFT is therefore said to be a [frequency domain](https://en.wikipedia.org/wiki/Frequency_domain) representation of the original input sequence. If the original sequence spans all the non-zero values of a function, its DTFT is continuous (and periodic), and the DFT provides discrete samples of one cycle. If the original sequence is one cycle of a periodic function, the DFT provides all the non-zero values of one DTFT cycle.

The DFT is the most important [discrete transform](https://en.wikipedia.org/wiki/Discrete_transform), used to perform [Fourier analysis](https://en.wikipedia.org/wiki/Fourier_analysis) in many practical applications. In [digital signal processing](https://en.wikipedia.org/wiki/Digital_signal_processing), the function is any quantity or [signal](https://en.wikipedia.org/wiki/Signal_(information_theory)) that varies over time, such as the pressure of a [sound wave](https://en.wikipedia.org/wiki/Sound_wave), a [radio](https://en.wikipedia.org/wiki/Radio) signal, or daily [temperature](https://en.wikipedia.org/wiki/Temperature) readings, sampled over a finite time interval (often defined by a window function). In [image processing](https://en.wikipedia.org/wiki/Image_processing), the samples can be the values of [pixels](https://en.wikipedia.org/wiki/Pixel) along a row or column of a [raster image](https://en.wikipedia.org/wiki/Raster_image). The DFT is also used to efficiently solve [partial differential equations](https://en.wikipedia.org/wiki/Partial_differential_equations), and to perform other operations such as [convolutions](https://en.wikipedia.org/wiki/Convolution) or multiplying large integers.Since it deals with a finite amount of data, it can be implemented in [computers](https://en.wikipedia.org/wiki/Computer) by [numerical algorithms](https://en.wikipedia.org/wiki/Numerical_algorithm) or even dedicated [hardware](https://en.wikipedia.org/wiki/Digital_circuit). These implementations usually employ efficient [fast Fourier transform](https://en.wikipedia.org/wiki/Fast_Fourier_transform) (FFT) algorithms so much so that the terms "FFT" and "DFT" are often used interchangeably. Prior to its current usage, the "FFT" [initialism](https://en.wikipedia.org/wiki/Initialism) may have also been used for the ambiguous term *"Finite Fourier transform"*.

Using Euler's formula, the DFT formulae can be converted to the trigonometric forms sometimes used in engineering and computer science:

Given a sequence of *N* samples *f*(*n*), indexed by *n*= 0..*N*-1, the Discrete Fourier Transform (DFT) is defined as *F*(*k*), where *k*=0..*N*-1:

equation

*F*(*k*) are often called the 'Fourier Coefficients' or 'Harmonics'.

The sequence *f*(*n*) can be calculated from *F*(*k*) using the Inverse Discrete Fourier Transform (IDFT):

equation

In general, both *f*(*n*) and *F*(*k*) are complex.

Annex A shows that the IDFT defined above really is an *inverse* DFT.

Conventionally, the sequences *f*(*n*) and *F*(*k*) is referred to as 'time domain' data and 'frequency domain' data respectively. Of course there is no reason why the samples in *f*(*n*) need be samples of a time dependant signal. For example, they could be spatial image samples (though in such cases a 2 dimensional set would be more common).

Although we have stated that both *n* and *k* range over 0..*N*-1, the definitions above have a periodicity of *N*:

equation

So both *f*(*n*) and *F*(*k*) are defined for all (integral) *n* and *k* respectively, but we only need to calculate values in the range 0..*N*-1. Any other points can be obtained using the above periodicity property.

For the sake of simplicity, when considering various Fast Fourier Transform (FFT) algorithms, we shall ignore the scaling factors and simply define the FFT and Inverse FFT (IFFT) like this:

equation

equation

In fact, we shall only consider the FFT algorithms in detail. The inverse FFT (IFFT) is easily obtained from the FFT.

Here are some simple DFT's expressed as matrix multiplications.

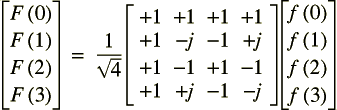
1 point DFT:

equation

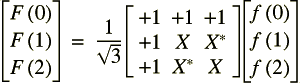
2 point DFT:

equation

4 point DFT:



3 point DFT:



equation

Note that each of the matrix multipliers can be inverted by conjugating the elements. This what we would expect, given that the only difference between the DFT and IDFT is the sign of the complex exponential argument.

Here's another couple of useful transforms:

If..

equation

equation

equation

This is the 'Delta Function'. The usual implied periodicity has been made explicit by using *MOD N*. The DFT is therefore:

equation

**Objective:**

1. Develop a function to perform DFT of N point signal
2. Calculate DFT of a DT signal and Plot spectrum of the signal.
3. Conclude the effect of zero padding on magnitude spectrum.
4. Calculate the number of real multiplications and real additions required to find DFT.

**Input Specifications:**

1. Length of Signal N
2. Signal values

**Problem Definition:**

1. Take any four-point sequence x[n].
2. Find DFT X[k].
3. Compute number of real multiplications and real additions required to find X[k].
4. Plot Magnitude Spectrum of the signal.
5. Append the input sequence by four zeros. Find DFT and plot magnitude spectrum. Repeat the same by appending the sequence by eight zeros. Observe and compare the magnitude spectrum. Give your conclusion.

**Program:**

#include <stdio.h>

#include <math.h>

void star(float[], int);

int max\_find(float [], int );

int main()

{

int n;

float w4[4][4] = {{1, 1, 1, 1}, {1, 0, -1, 0}, {1, -1, 1, -1}, {1, 0, -1, 0}};

float xn[4], j[4] = {0}, ans[4] = {0};

float k[4];

printf("Enter numbers of element in x(n): ");

scanf("%d", &n);

printf("Please enter elements of x(n) one by one: ");

for (int i = 0; i < n; i++)

scanf("%f", &xn[i]);

for (int i = n; i < 4; i++)

xn[i] = 0;

for (int i = 0; i < 4; i++)

for (int k = 0; k < 4 ; k++)

ans[i] = ans[i] + w4[i][k] \* xn[k];

j[1] = -xn[1] + xn[3];

j[3] = xn[1] - xn[3];

for ( int i = 0; i < 4; i++)

{

k[i] = sqrt(ans[i] \* ans[i] + j[i] \* j[i]);

}

for ( int i = 0; i < 4; i++)

{

printf("%.3f ", k[i]);

}

printf("\n");

for (int i = 0; i < n; i++)

{

k[i] = 2 \* k[i]; // representing two || for one decimal

if ( (k[i] - ((int) k[i])) > 0.5 )

{

k[i] = ( (int) k[i] ) + 1;

}

else

k[i] = (int)k[i];

}

star(k, 4);

}

void star(float h3[], int n)

{ char array[50][50] ;

for (int i = 0; i < 50; i++)

for (int j = 0; j < 50; j++)

array[i][j] = ' ';

int max = max\_find(h3, n);

for (int i = 0; i < n; i++)

{

for (int j = 0; j < h3[i]; j++)

{

array[j][i] = '|' ;

}

}

for (int i = max; i >= 0; i--)

{

for (int j = 0; j < n; j++)

{

printf(" ");

printf("%c", array[i][j]);

printf(" ");

}

printf("\n");

}

for (int i = 0; i < n; i++) {

printf(" ");

printf("%d", i);

printf(" ");

}

printf("\n");

}

int max\_find(float h3[], int n)

{

int max = h3[0];

for (int i = 1; i < n; i++) {

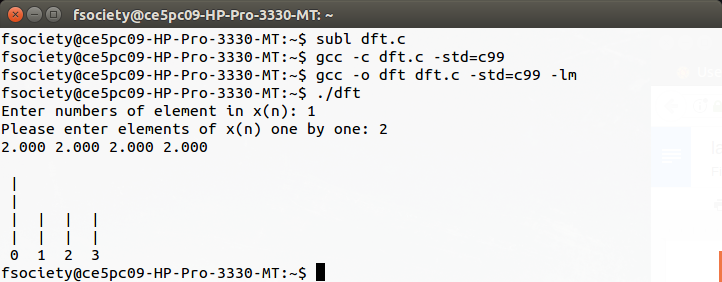
max = (max < h3[i]) ? h3[i] : max;

}

return max;

}

**Output:**

****

**Conclusion:**

Hence we implemented magnitude spectrum of the DT signal.