Batch: A3	**Roll No.: 1811054**
Experiment No. 04	
Grade: AA / AB / BB / BC / CC / CD /DD	
Signature of the Staff In-charge with date	

Title: Compute DFT & IDFT of discrete time signals using Matlab.

Objective: To learn & understand the Fourier transform operations on discrete time signals.

Expected Outcome of Experiment:

CO	Outcome
CO3	Analyze signals in frequency domain through various image transforms

Books/ Journals/ Websites referred:

1. http://www.mathworks.com/support/
2. www.math.mtu.edu/~msgocken/intro/intro.html
3. www.mccormick.northwestern.edu/docs/efirst/matlab.pdf
4. A.Nagoor Kani “Digital Signal Processing”, 2nd Edition, TMH Education.

Pre Lab/ Prior Concepts:

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:

$$F(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} f(n)\, e^{-j2\pi kn/N}$$

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):

$$f(n) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} F(k)\, e^{+j2\pi nk/N}$$

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 dependent 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*:

$$F(k+N) = F(k) \qquad f(n+N) = f(n)$$

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:

$$FFT_N(k, f) = \sum_{n=0}^{N-1} f(n)\, e^{-j2\pi kn/N} = \sqrt{N} F(k)$$

$$IFFT_N(n, F) = \sum_{k=0}^{N-1} F(k) e^{+j2\pi nk/N} = \sqrt{N} f(n)$$

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:

$$\left[F(0)\right] = \left[1\right]\left[f(0)\right]$$

2 point DFT:

$$\begin{bmatrix} F(0) \\ F(1) \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} +1 & +1 \\ +1 & -1 \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \end{bmatrix}$$

4 point DFT:

$$\begin{bmatrix} F(0) \\ F(1) \\ F(2) \\ F(3) \end{bmatrix} = \frac{1}{\sqrt{4}} \begin{bmatrix} +1 & +1 & +1 & +1 \\ +1 & -j & -1 & +j \\ +1 & -1 & +1 & -1 \\ +1 & +j & -1 & -j \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \\ f(3) \end{bmatrix}$$

3 point DFT:

$$\begin{bmatrix} F(0) \\ F(1) \\ F(2) \end{bmatrix} = \frac{1}{\sqrt{3}} \begin{bmatrix} +1 & +1 & +1 \\ +1 & X & X^* \\ +1 & X^* & X \end{bmatrix} \begin{bmatrix} f(0) \\ f(1) \\ f(2) \end{bmatrix}$$

$$where \qquad X = e^{-j2\pi/3} = COS(2\pi/3) - jSIN(2\pi/3) = -\left(\frac{1 + j\sqrt{3}}{2}\right)$$

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..

$$f(n) = \delta(n - n_0) \qquad n_0 = 0..N-1$$

$$= 1 \qquad if\ (n\,MOD\,N) = n_0$$

$$= 0 \qquad if\ (n\,MOD\,N) \neq n_0$$

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

$$F(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} \delta(n - n_0)\, e^{-j2\pi kn/N} = \frac{e^{-j2\pi k n_0/N}}{\sqrt{N}}$$

This gives us the DFT of a unit impulse at $n=n_0$. Less obvious is this DFT:

If..

$$f(n) = e^{+j2\pi k_0 n/N} \qquad k_0 = 0..N-1$$

$$F(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} e^{+j2\pi k_0 n/N} e^{-j2\pi kn/N} = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} e^{-j2\pi(k-k_0)n/N} = \sqrt{N}\delta(k - k_0)$$

Implementation steps with screenshots for DFT:

1D array:

```
clc;
%dft of 1d
n = input('Enter length of signal: ');
x = input('Enter the signal: ');

twid=zeros(n,n);
for l=0:1:n-1
    for m=0:1:n-1
      twid(l+1,m+1)= exp((-1i*2*pi*l*m/n));
    end
end
fprintf('Twiddle');
disp(twid)
fprintf('DFT');
Y=twid*x';
disp(Y);

fprintf('IDFT');
y=(twid'*Y)/n;
disp(y);
```

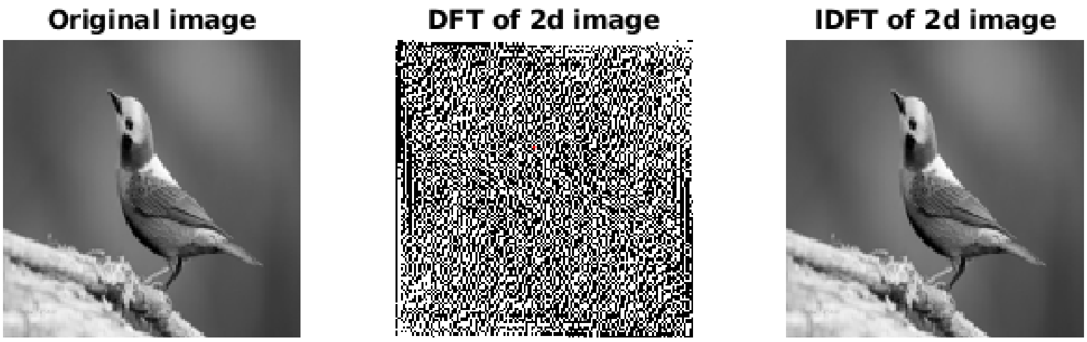
OUTPUT:

```
Command Window
Enter length of signal:
4
Enter the signal:
[0,1,2,1]
Twiddle   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

DFT   4.0000 + 0.0000i
  -2.0000 - 0.0000i
   0.0000 + 0.0000i
  -2.0000 - 0.0000i

IDFT   0.0000 - 0.0000i
   1.0000 - 0.0000i
   2.0000 + 0.0000i
   1.0000 + 0.0000i
```

2D Image:

```
a = imread('bird.jpg');
img = rgb2gray(a);
img = imresize(img,[128,128]);
s = size(img);
subplot(1,3,1)
imshow(img)
title('Original image')
tw = zeros(128,128);
for j=0:127
 for h=0:127
 tw(j+1,h+1) = exp((-1i * 2 * pi * j * h)/128);
 end
end
dftimg = tw * double(img) * tw';
subplot(1,3,2)
imshow(real(dftimg))
title('DFT image')
%idft
idftimg = (1/(128*128)) * (tw' * double(dftimg) * tw);
subplot(1,3,3)
imshow(idftimg,[])
title('IDFT image')
```

OUTPUT:



Conclusion:- We have successfully implemented the DFT and IDFT for one-dimensional form as well as two-dimensional form.

Post Lab Descriptive Questions

1. **Compare and discuss the computational efficiency of DFT and FFT**

Ans: DFT is a linear transform which takes as input a complex signal x of length N and gives as output a complex signal X of length N, $X=Wx$. W is a complex NxN matrix with entiries W_k,n=exp(-2pi*k*n/N), where $0<k,n<N$.

FFT is a collection of algorithms for fast computation of the DFT. Typically the number of operations required by the FFT is on the order of N*log$N$. The most famous

FFT algorithms are for the case that N is a power of 2, but there are FFT for prime orders and for different other factorizations.

2. **Give the properties of DFT and IDFT.**

Ans: It states that the DFT of a combination of signals is equal to the sum of DFT of individual signals. Let us take two signals x1(n) and x2(n), whose DFT s are X1(ω) and X2(ω) respectively. So, if

x1(n)→ X1(ω)andx2(n)→ X2(ω)
Then ax1(n)+bx2(n)→ aX1(ω)+bX2(ω)
where a and b are constants.

Symmetry
The symmetry properties of DFT can be derived in a similar way as we derived DTFT symmetry properties. We know that DFT of sequence x(n) is denoted by X(K). Now, if x(n) and X(K) are complex valued sequence, then it can be represented as under
x(n)=xR(n)+jx1(n) ,0≤n≤N−1
And X(K)=XR(K)+jX1(K) ,0≤K≤N−1

Duality Property
Let us consider a signal x(n), whose DFT is given as X(K). Let the finite duration sequence be X(N). Then according to duality theorem,
If, x(n)⟷ X(K)x(n)⟷ X(K)
Then, X(N)⟷ Nx[((−k))N]
So, by using this theorem if we know DFT, we can easily find the finite duration sequence.

Complex Conjugate Properties
Suppose, there is a signal x(n), whose DFT is also known to us as X(K). Now, if the complex conjugate of the signal is given as x*(n), then we can easily find the DFT without doing much calculation by using the theorem shown below.
If, x(n)⟷ X(K)
Then, x∗(n)⟷ X∗((K))N=X∗(N−K)

Circular Frequency Shift
The multiplication of the sequence x(n) with the complex exponential sequence ej2Πkn/Nej2Πkn/N is equivalent to the circular shift of the DFT by L units in frequency. This is the dual to the circular time shifting property.
If, x(n)⟷ X(K)

Then, x(n)ej2ΠKn/N⟷ X((K−L))

Multiplication of Two Sequence

If there are two signal x1(n) and x2(n) and their respective DFTs are X1(k) and X2(K), then multiplication of signals in time sequence corresponds to circular convolution of their DFTs.

If, x1(n)⟷ X1(K)&x2(n)⟷ X2(K)

Then, x1(n)×x2(n)⟷ X1(K)©X2(K)

Parseval's Theorem

For complex valued sequences x(n) and y(n), in general

If, x(n)⟷X(K)&y(n)⟷Y(K)

Then, ∑N−1n=0 x(n)y∗(n)=(1/N)∑N−1k=0X(K)Y∗(K)

3. **Discuss the impact on computation time & efficiency when the number of samples N increases.**

Ans: Basic FFT resolution is fs/N, where fs is the sampling frequency.

The ability to differentiate two very closely spaced signals depends strongly on relative amplitudes and the windowing function used.

This means that as the number of samples increase the FFT resolution decreases and the efficiency of decreases and computation time increases.

4. **How to compute maximum length N for a circular convolution using DFT and IDFT?**

Ans: We can find the circular convolution by

x1(n)*x2(n)= IDFT(x1(k).x2(k))
where

x1(k)=DFT(x1(n))=WN*x1(n)

x2(k)=DFT(x1(n))=WN*x1(n) and . operator is element wise multiplication.