Discrete-Time Fourier Transform

➔A discrete-time signal can be represented in the frequency domain using discrete-time Fourier transform. Therefore, the Fourier transform of a discrete time sequence is called the *discrete-time Fourier transform (DTFT)*.

Mathematically, if $x(n)$ is a discrete-time sequence, then its discrete-time Fourier transform is defined as –

$$F[x(n)] = X(\omega) = \sum_{n=-\infty}^{\infty} x(n)\, e^{-j\omega n}$$

The discrete-time Fourier transform X(ω) of a discrete-time sequence $x(n)$ represents the frequency content of the sequence $x(n)$. Therefore, by taking the Fourier transform of the discrete-time sequence, the sequence is decomposed into its frequency components. For this reason, the DTFT X(ω) is also called the **signal spectrum.**

Fast Fourier Transform

➔In earlier DFT methods, we have seen that the computational part is too long. We want to reduce that. This can be done through FFT or fast Fourier transform. So, we can say FFT is nothing but computation of discrete Fourier transforms in an algorithmic format, where the computational part will be reduced.

The main advantage of having FFT is that through it, we can design the FIR filters. Mathematically, the FFT can be written as follows;

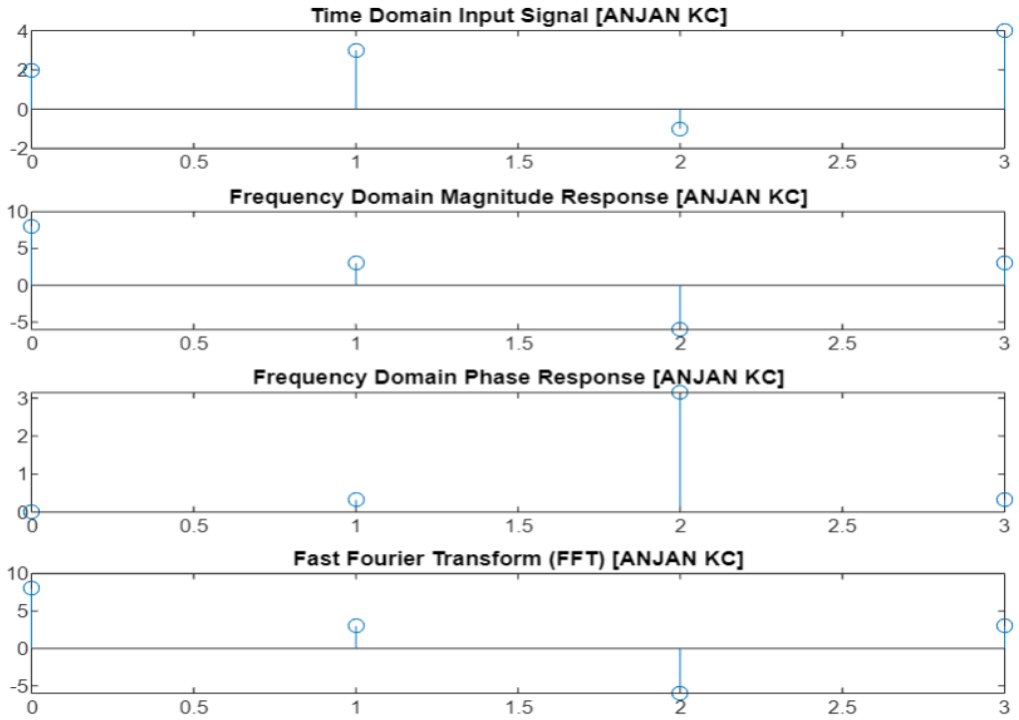
$$x[K] = \sum_{n=0}^{N-1} x[n] W_N^{nk}$$

Name: Anjan KC
Roll No:19070293 (6) Lab: 6 Discrete Fourier Transform and Z-Transform

Code:

```
x=[2 3 -1 4];
N=length(x);
X=zeros(N,1);
for k=0:N-1
    for n=0:N-1
        X(k+1)=X(k+1)+x(n+1)*exp(-1i*2*pi*n*k/N);
    end
end
t=0:N-1;
subplot(4,1,1);
stem(t,x);
title("Time Domain Input Signal [ANJAN KC]");
subplot(4,1,2) ;    %x label
stem(t,X);         %magnitude response
title("Frequency Domain Magnitude Response [ANJAN KC]");
subplot(4,1,3);
stem(t,abs(angle(X)));        %phase response
title("Frequency Domain Phase Response [ANJAN KC]");
FFT=fft(x,N);
subplot(4,1,4);
stem(t,FFT);
title("Fast Fourier Transform (FFT) [ANJAN KC]")
```

Output:

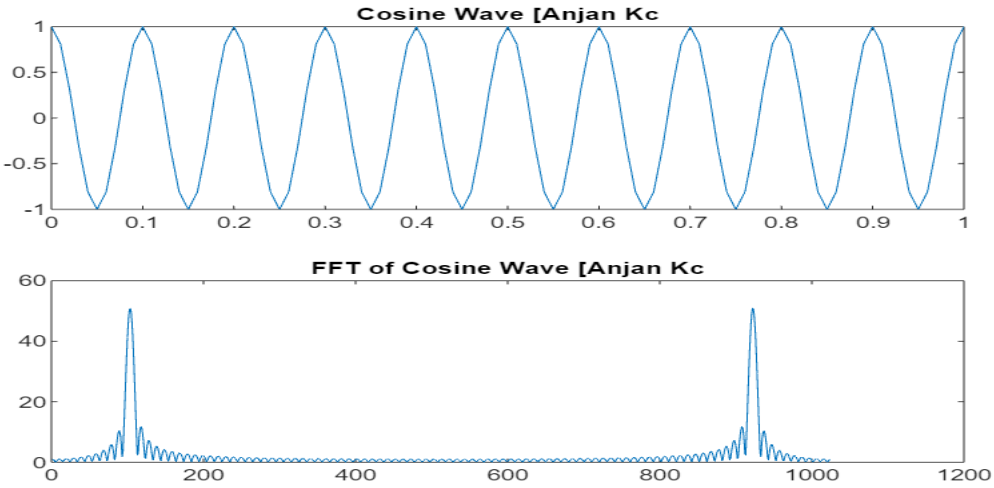


Name: Anjan KC
Roll No:19070293 (6) Lab: 6 Discrete Fourier Transform and Z-Transform

Code For FFT of Cosine Wave:

```
f=10;
a=1;
t=0:0.01:1;
x=a*cos(2*pi*f*t);
subplot(2,1,1);
plot(t,x);
title("Cosine Wave [Anjan Kc")
X=fft(x,1024);
Xabs=abs(X);
subplot(2,1,2);
plot(Xabs);
title("FFT of Cosine Wave [Anjan Kc")
```

Output:



Z-Transforms (ZT)

- Analysis of continuous time LTI systems can be done using z-transforms. It is a powerful mathematical tool to convert differential equations into algebraic equations.
- The bilateral (two sided) z-transform of a discrete time signal x(n) is given as

$$Z.T[x(n)] = X(Z) = \Sigma_{n=-\infty}^{\infty} x(n) z^{-n}$$

- The unilateral (one sided) z-transform of a discrete time signal x(n) is given as

$$Z.T[x(n)] = X(Z) = \Sigma_{n=0}^{\infty} x(n) z^{-n}$$

- Z-transform may exist for some signals for which Discrete Time Fourier Transform (DTFT) does not exist.

The Inverse Z-Transform

The **inverse Z-transform** is defined as the process of finding the time domain signal $x(n)$ from its Z-transform $X(z)$. The inverse Z-transform is denoted as –

$$x(n) = Z^{-1}[X(z)]$$

Since the Z-transform is defined as,

$$X(z) = \sum_{n=-\infty}^{\infty} x(n)\, z^{-n} \quad \cdots (1)$$

```
%Z-Transform
y=[1 2 3 4 5];
syms z;
l=length(y);
Y=0;
for i=0:l-1
   Y=Y+y(i+1)*z^(-i);
end
pretty(Y);
k=iztrans(Y);
pretty(k);
```

Output:

```
>> Z_Transform_and_Inverse_Z_Transform
2   3   4   5
- + -- + -- + -- + 1
z   2   3   4
   z   z   z

2 kroneckerDelta(n - 1, 0) + 3 kroneckerDelta(n - 2, 0) + 4 kroneckerDelta(n - 3, 0) + 5 kroneckerDelta(n - 4, 0) + kroneckerDelta(n, 0)

>>
```