# EXPERIMENT 8

# Fast Fourier Transform and Z-Transform

**Objective**

To study the fast and efficient way of computing the discrete Fourier transform, difference between this Fast Fourier transform and Discrete Fourier transform and to learn how to find the Z-Transform using MATLAB.

**Introduction**

The Fast Fourier Transform is an efficient algorithm for computing the Discrete Fourier Transform.

**Difference in DFT and FFT**

FFT (Fast Fourier Transform) is a faster version of the DFT that can be applied when the number of samples in the signal is a power of two. FFT computation takes approximately **$O(N\log_2(N))$** operations, whereas a DFT takes approximately **$O(N^2)$** operations, so the FFT is significantly faster.

To obtain one sample of $X(k)$ we need $N$ complex multiplications and $(N-1)$ complex additions. Clearly, the number of DFT computations for an $N$-point sequence depends quadraticallyon $N$. The quadratic dependence on $N$ can be reduced by realizing that most of the computations can be eliminated using the periodicity property and the symmetry property.

Let $X(k) = \sum_{n=0}^{3} x(n).W_4^{nk}$ , $\quad 0 \le k \le 3$,

This computation can be written in matrix form as:

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} . \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix}$$

which requires 16 (**$N^2$**) complex multiplications.

**MATLAB Implementation**

MATLAB provides a function called FFT to compute the DFT of a vector $x$. If length of $x$ is less than$N$, then $x$ is padded with zeros. This fft function is written in machine language therefore it executes very fast. If $N$ is a power of 2 then a high speed radix-2 FFT algorithm is employed. If $N$ is not a multiple of 2, then $N$ is decomposed into prime factors and a slower mixed radix FFT algorithm is used.

There are many algorithms to compute FFT, a few are listed below:

- Radix-2 FFT algorithm
- Split-radix FFT algorithm
- Prime-factor FFT algorithm
- Bruun's FFT algorithm
- Rader's FFT algorithm
- Bluestein's FFT algorithm

**Radix-2-FFT Algorithm**

Let $N$ be a multiple of 2 then we divide$x(n)$into two$\frac{N}{2}$pointsequences:

$$\boldsymbol{g_1(n) = x(2n); \quad 0 \le n \le \frac{N}{2} - 1}$$

$$\boldsymbol{g_2(n) = x(2n+1); \quad 0 \le n \le \frac{N}{2} - 1}$$

Let $G_1(k)$ and $G_2(k)$ be $N/2$-point DFTs of $g_1(n)$ and $g_2(n)$then we have:

$$\boldsymbol{X(k) = G_1(k) + W_n^k G_2(k), \qquad 0 \le k \le N-1}$$

This algorithm has a complexity of**O(Nlog$_2$(N)).** The input sequence for the FFT is divided into two sequences in the following way:

If $x(n) = [0\ 1\ 2\ 3\ 4\ 5\ 6\ 7]$, then

$x_1(n_1) = [0\ 2\ 4\ 6]$% even samples of $x(n)$

$x_2(n_2) = [1\ 3\ 5\ 7]$% odd samples of $x(n)$

**Exercise 1:**

Write a script to implement the equation $X(k) = G_1(k) + W_n^k G_2(k)$in MATLAB given only the following two vectors of size N/2 each.

$x_1(n_1) = [0\ 2\ 4\ 6]$% even samples of $x(n)$

$x_2(n_2) = [1\ 3\ 5\ 7]$% odd samples of $x(n)$

$W_N^k = e^{-j2\pi(\frac{k}{N})}$

You may use the MATLAB function fft to check your results.

(Hint: *Use the symmetry property of*$W_{N/2}^{km}$)

Write the code in the space provided below:

**<u>Exercise 2</u>:**

2- point FFT can be found as $\begin{bmatrix} X(0) \\ X(1) \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} . \begin{bmatrix} x(0) \\ x(1) \end{bmatrix}$

Write a MATLAB function to find 2-point FFT.

```
function y = fft_2pt(x,N)

%   x   :   input vector of length 2

% if N == 2

%        Calculate 2-point FFT of x

% else

%        print 'error'
```

# Z-Transform

The Z-transform is simply a power series representation of a discrete-time sequence. The Z-transform of a discrete time signal is defined as

$$\boldsymbol{X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n}}$$

Where, $z$ is the complex variable. This equation is sometimes called direct z-transform because it transforms the time-domain signal $x(n)$ into its complex-plane representation $X(z)$. For convenience the z-transform of a signal $x(n)$ is denoted by $X(z) \equiv Z\{x(n)\}$

The procedure of transforming z-transform to the time domain is called ***inverse z-transform***. The inversion formula for obtaining $x(n)$from $X(z)$ is

$$\boldsymbol{x(n) = \frac{1}{2\pi j} \oint X(z) z^{n-1} dz}$$

Z-transform gives us the complex domain information of a sequence. MATLAB provides us with built-in functions to compute Z-transform as well as Inverse Z-transform. The region of convergence can also be plotted using a function called z-plane.

**Objective:**

The objective of this experiment is to learn how to find z-Transform and Inverse z-transform from multiple methods using MATLAB. The pole-zero plot is found using z-plane (num, den).

**Exercise 1:**

If $h(n) = 5\left(\frac{1}{4}\right)^n u(n)$

Write $H(z)$ and also sketch its pole-zero plot.

```
num=[5];den=[1, -1/4];
zplane (num,den)
[r,p,c]=residue (num,den)
```

Attach the screen shots of the plots in the space provided below:

**Inverse z-Transform**

The analysis equation of z-transform is given by:

$$\boldsymbol{X(z) = \sum_{n=-\infty}^{n=+\infty} x(n)\, z^{-n}}$$

The inverse z-transform can be found by

1. Inspection Method
2. Partial Fraction Expansion Method
3. Power Series Expansion
4. Residue function in MATLAB

**Inspection Method**:

The inverse z-transform of $X(z) = \frac{1}{1-\frac{1}{2}z^{-1}}$ is $\rightarrow x(n) = \left(\frac{1}{2}\right)^n u(n)$

**Partial Fraction Expansion Method**:

Express $G(z)$ in a partial fraction expansion form and then determine $g(n)$ by summing the inverse transform of the individual simpler terms in the expansion.

Example:

$$X(z) = \frac{1}{\left(1-\frac{1}{4}z^{-1}\right)\left(1-\frac{1}{2}z^{-1}\right)}, \qquad ROC: |z| > \frac{1}{2}$$

$$X(z) = \frac{-1}{\left(1-\frac{1}{4}z^{-1}\right)} + \frac{2}{\left(1-\frac{1}{2}z^{-1}\right)}, \qquad with|z| > \frac{1}{2}$$

The inverse z-transform is given by: $x(n) = 2\left(\frac{1}{2}\right)^n u(n) - \left(\frac{1}{4}\right)^n u(n)$.

And, this is a right sided sequence.

**Exercise 2:**

Find all possible sequences (which is inverse z-transforms) for this $X(z)$.

$$X(z) = \frac{z}{3z^2 - 4z + 1}$$

Write the sequences in the space provided below:

## **Partial Fraction Expansion Using MATLAB**

The MATLAB function [r, p, c] = residue (num, den) computes the partial fraction expansion of a rational z-transform with numerator and denominator coefficients given by vectors num and den.

1. Vector r contains the residues
2. Vector p contains the poles
3. Vector c contains the constants

Example:

$$X(z) = \frac{z}{3z^2 - 4z + 1}$$

$$X(z) = \frac{(0 + z^{-1})}{(3 - 4z^{-1} + z^{-2})}$$

```
num=[0,1];den=[3,-4,1];
zplane(num,den)
[r,p,c]=residue(num,den)
```

---

```
r =
0.5000
-0.5000

p =
1.0000
0.3333
c =
[]
```

So from above, we obtain

---

$$X(z) = \frac{\frac{1}{2}}{(1 - z^{-1})} - \frac{\frac{1}{2}}{\left(1 - \frac{1}{3}z^{-1}\right)}$$

$$x(n) = \left(\frac{1}{2} - \frac{1}{2}\left(\frac{1}{3}\right)^n\right)u(n)$$

**<u>Exercise 3:</u>**

Find inverse z-transform by using MATLAB and by the method of partial fractions expansion for the following z-transforms. Show working in the space provided:

The denominator polynomial can also be calculated using MATLAB's built-in function poly, which computes the polynomial coefficients when roots are given.

1)

$$X(z) = \frac{(1 - z^{-1} - 4z^{-2} + 4z^{-3})}{(1 - \frac{11}{4}z^{-1} + \frac{13}{8}z^{-2} - \frac{1}{4}z^{-4})}$$

2)

$$X(z) = \frac{z}{(z^3 + 2z^2 + 1.25z + 0.25)}$$

3)

$$X(z) = \frac{(z^3 - 3z^2 + 4z + 1)}{(z^3 - 4z^2 + z - 0.16)}$$