Aim ↬ To understand and implement the Z-Transform and its inverse for analyzing and synthesizing discrete-time signals using MATLAB.

Software Required ↬ MATLAB

Theory ↬

The Z-Transform is a fundamental tool in digital signal processing, analogous to the Continuous-Time Fourier Transform (CTFT) for continuous signals. It transforms a discrete-time signal from the time domain to the complex frequency domain, providing valuable insight into the signal's frequency content and system characteristics. This transformation is crucial in digital signal processing, control systems, and communications.

The Z-Transform of a discrete-time signal x[n] is defined as:

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

where:

- X(z) is the Z-Transform of x[n], representing the signal in the complex frequency domain.
- z is a complex variable, often expressed as $z = re^{j\omega}$, where r is the radius and ω is the angle in the complex plane.

The Z-transform maps a sequence x[n] to a function of the complex variable z, revealing the signal's poles and zeros, which are critical for understanding system stability and frequency response.

Properties of Z-Transform ↴

1] Linearity ↪

$$Z\{ax_1[n] + bx_2[n]\} = aX_1(z) + bX_2(z)$$

2] Time Shifting ↪

$$Z\{x[n - n_0]\} = z^{-n_0}X(z)$$

3] Time Reversal ↪

$$Z\{-x[n]\} = X\left(\frac{1}{z}\right)$$

4] Time Scaling ↪

$$Z\left\{x\left[\frac{n}{k}\right]\right\} = X(z^k)$$

5] Scaling in the Z-Domain ↪

$$Z\{r^n x[n]\} = X\left(\frac{z}{r}\right)$$

6] Differentiation in the Z-Domain ↪

$$Z\{nx[n]\} = -z\frac{dX(z)}{dx}$$

7] Convolution in Time Domain ↪

$$Z\{x[n] * y[n]\} = X(z) \cdot Y(z)$$

8] Multiplication in Time Domain ↪

$$Z\{x[n].y[n]\} = \frac{1}{2\pi j}[X(z) * Y(z)]$$

9] Conjugation ↪

$$Z\{x^*[n]\} = X^*(z^*)$$

10] First Difference ↪

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

11] Initial Value Theorem ↪

If X(z) is known, $x[0] = \lim_{z\to\infty} X(z)$

12] Final Value Theorem ↪

If X(z) has all poles inside the unit circle,

$$\lim_{n\to\infty} x[n] = \lim_{z\to 1}(z-1)X(z)$$

Z-Transform of Basic Signals ↴

x[n]	X(z)
δ[n]	1
u[n]	$\frac{z}{z-1}$

u[-n-1]	$-\frac{z}{z-1}$
δ[n-m]	z^{-m}
$r^n u[n]$	$\frac{z}{z-r}$
$nr^n u[n]$	$\frac{rz}{\lvert z-r\rvert^2}$
$r^n \cos(\omega n)\, u[n]$	$\frac{z^2 - rz\cos(\omega)}{z^2 - 2rz\cos(\omega) + r^2}$
$r^n \sin(\omega n)\, u[n]$	$\frac{rz\sin(\omega)}{z^2 - 2rz\cos(\omega) + r^2}$

Inverse Z-Transform ↴

The Inverse Z-Transform reconstructs the original discrete-time signal from its Z-domain representation:

$$x[n] = \frac{1}{2\pi j}\oint_C X(z)z^{n-1}dz$$

where C is a closed contour in the complex plane that encircles all the poles of X(z). This contour integral sums the contributions of all the poles to reconstruct the time-domain signal.

The Z-Transform and its inverse allow for the analysis and synthesis of discrete-time signals, providing a powerful framework for signal processing and system analysis in various engineering fields.

Code ↬

```
% Z-Transform
syms z n a;
x = a^(0.1*n)*sin(a*n);
fprintf('Original Signal-1:');
display(x);

% Using built-in functions
X_z = ztrans(x);
fprintf('Z-Transform of signal:');
```

```
display(X_z);

x_remake = iztrans(X_z);
fprintf('Reconstructed Signal-1:');
display(x_remake);

l = linspace(0,10);
x_val = double(subs(x, {a, n}, {4.7,l}));
x_remake_val = double(subs(x_remake, {a, n}, {4.7,l}));

figure;
subplot(2,2,1);
stem(l, x_val, 'r');
xlabel('Time');
ylabel('Amplitude');
title('Original Signal-1');

subplot(2,2,2);
stem(l, x_remake_val, 'b');
xlabel('Time');
ylabel('Amplitude');
title('Reconstructed Signal-1');

% Using Formula
k = [2, -3, 4, -1, 3, 0, 1];
fprintf('Original Signal-2:');
display(k);
N = length(k);

for n = 1:N
    F_z(n) = k(n)*(z^-n);
end

K_z = 0;
for n = 1:N
    K_z = K_z + F_z(n);
end
fprintf('Z-Transform of signal:');
display(K_z);
```

```
for n = 1:N
    k_remake(n) = F_z(n)*(z^n);
end
fprintf('Reconstructed Signal-2:');
display(k_remake);

subplot(2,2,3);
stem(1:N, k, 'r');
xlabel('Time');
ylabel('Amplitude');
title('Original Signal-2');

subplot(2,2,4);
stem(1:N, k_remake, 'b');
xlabel('Time');
ylabel('Amplitude');
title('Reconstructed Signal-2');
```

Output ↬

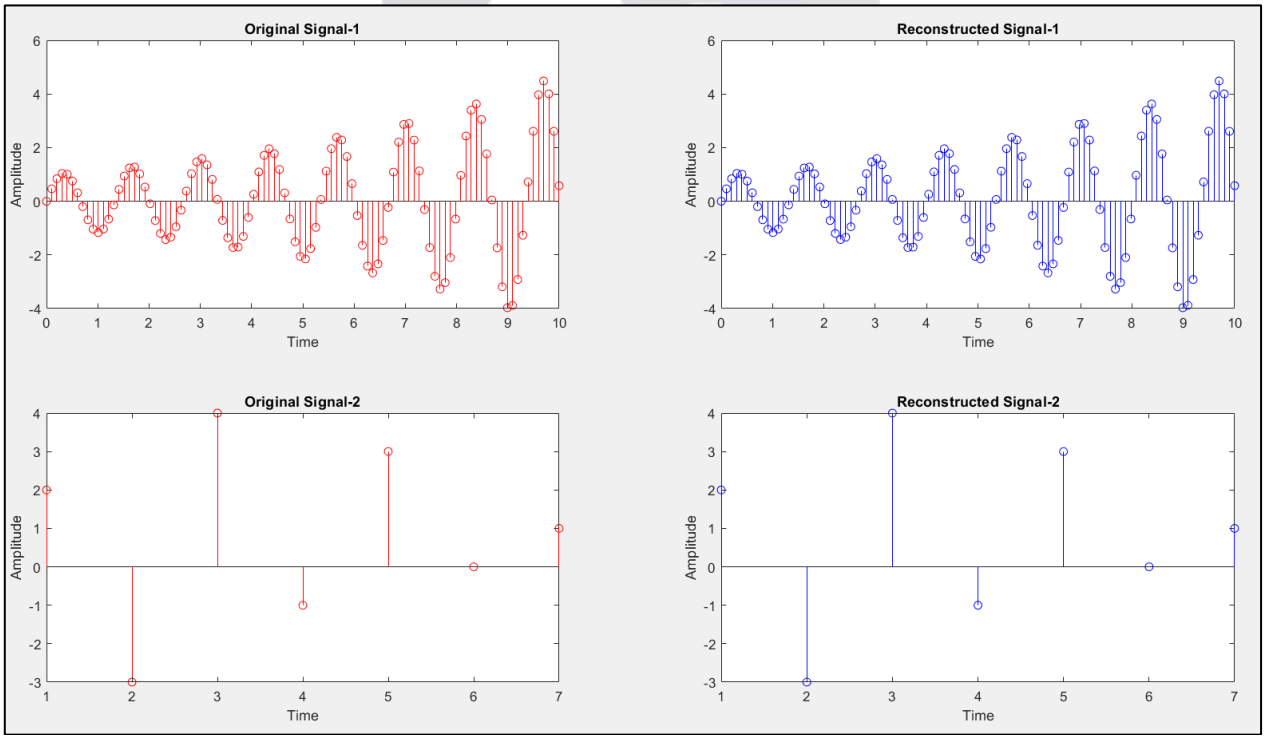

Fig. i) Plots for Z-Transform and Inverse Z-Transform

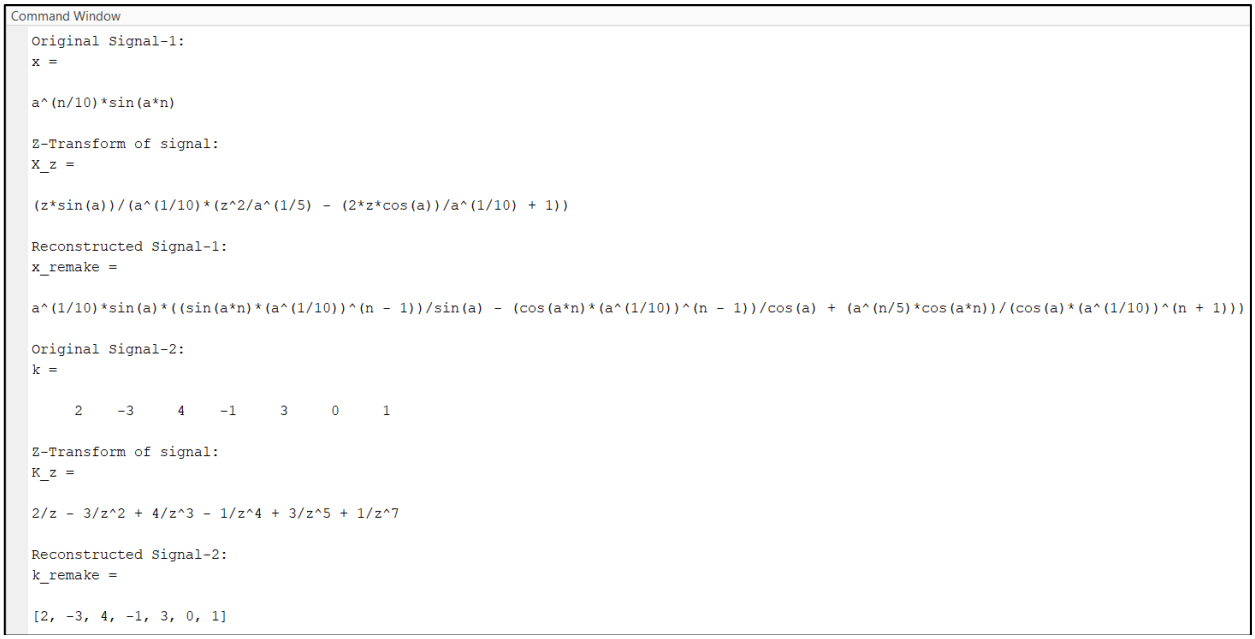

Fig. ii) Expressions obtained as result

Result ↬

The Z-Transform and its inverse were successfully applied to analyze discrete-time signals. The MATLAB code accurately computed the Z-domain representations and reconstructed the original signals.

Conclusion ↬

Z-Transform enables the transformation of complex time-domain operations into simpler algebraic forms in the z-domain. This transformation is crucial for understanding the stability, frequency response, and overall behavior of discrete-time systems.

Precautions ↬

- Define the region of convergence carefully.
- Apply properties like scaling and shifting with precision.
- Cross-check results using the inverse Z-Transform.