



Subject: Programming With Python (01CT1309)

Aim: Analysis of LTI System Responses to Standard Inputs Using Python

Experiment No: 19

Date:

Enrollment No:92400133131

GITHUB LINK:- <https://github.com/farhan-web404/farhankaladiya.git>

Aim: Analysis of LTI System Responses to Standard Inputs Using Python

IDE:

Analyzing Discrete-Time Systems Using Z-Transform

The Z-transform is used for analyzing discrete-time signals and systems. The Z-transform of a discrete-time signal $x[n]$ is given by:

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

where z is a complex variable, $X(z)$ represents the Z-transform of the signal.

Z-Transform Function

For an LTI system, the Z-transform function $H(z)$ is defined as:

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + \dots + b_m z^{-m}}{a_0 + a_1 z^{-1} + \dots + a_n z^{-n}}$$

where $B(z)$ is the numerator polynomial, $A(z)$ is the denominator polynomial.

Stability

A discrete-time system is stable if all poles of its Z-transfer function lie inside the unit circle in the Z-plane. To check stability:

Calculate the poles of $H(z)$

Check if the magnitude of each pole is less than 1.

 Marwadi University <small>Marwadi Chandarana Group</small>	 NAAC A+	Marwadi University Faculty of Engineering & Technology Department of Information and Communication Technology
Subject: Programming With Python (01CT1309)	Aim: Analysis of LTI System Responses to Standard Inputs Using Python	
Experiment No: 19	Date:	Enrollment No:92400133131

Causality

A system is causal if its impulse response $h[n]$ is zero $n < 0$. This generally means that the numerator polynomial should not have terms that depend on future values.

Time Invariance

A system is time-invariant if a time shift in the input results in an equivalent time shift in the output. For LTI systems, if the system is defined properly, it is generally assumed to be time-invariant. **Example**

$$H(z) = \frac{(z^2 + 0.5)}{(z^2 - 1.5z + 0.5)}$$

Bode Plot Analysis

Stability:

- Check the gain and phase margins.
- Ensure that both margins are positive for stability.

Causality:

- Examine the magnitude and phase at low frequencies.
- Confirm that the system behaves as a causal system (magnitude starts lower, phase starts near 0 and decreases).

Time Invariance:

- If the system is LTI, it is inherently time-invariant.
- Analyse the impulse response (if available) to verify consistent responses to delayed inputs.

Python Implementation `import numpy as np`

`import matplotlib.pyplot as plt from`

`scipy.signal import TransferFunction, lti`



Subject: Programming With Python (01CT1309)

Aim: Analysis of LTI System Responses to Standard Inputs Using Python

Experiment No: 19

Date:

Enrollment No:92400133131

def analyze z transfer function(num, den):

 # Create a Transfer Function object

 system = TransferFunction(num, den)

 # Get the poles and zeros zeros = system.zeros

 poles = system.poles print("Zeros:", zeros)

 print("Poles:", poles) # Stability Analysis stable =

 all(np.abs(pole) < 1 for pole in poles)

 print("Stability:", "Stable" if stable else "Unstable")

 # Causality Analysis causal = all(num[i] == 0 for i in range(len(num))

 - 1) if num[i + 1] == 0) print("Causality:", "Causal" if causal else

 "Non-Causal")

 # Time Invariance Analysis time_invariant = True # For Z-transforms, generally time-

invariant if system defined properly print("Time Invariance:", "Time Invariant" if

time_invariant else "Time Variant")

 # Bode plot (magnitude and phase)

w, mag, phase = bode(system)

 # Plot Bode plot

 plt.figure(figsize=(12, 8))



Marwadi University
Faculty of Engineering & Technology
Department of Information and Communication Technology

Subject: Programming With Python (01CT1309)

Aim: Analysis of LTI System Responses to Standard Inputs Using Python

Experiment No: 19

Date:

Enrollment No:92400133131

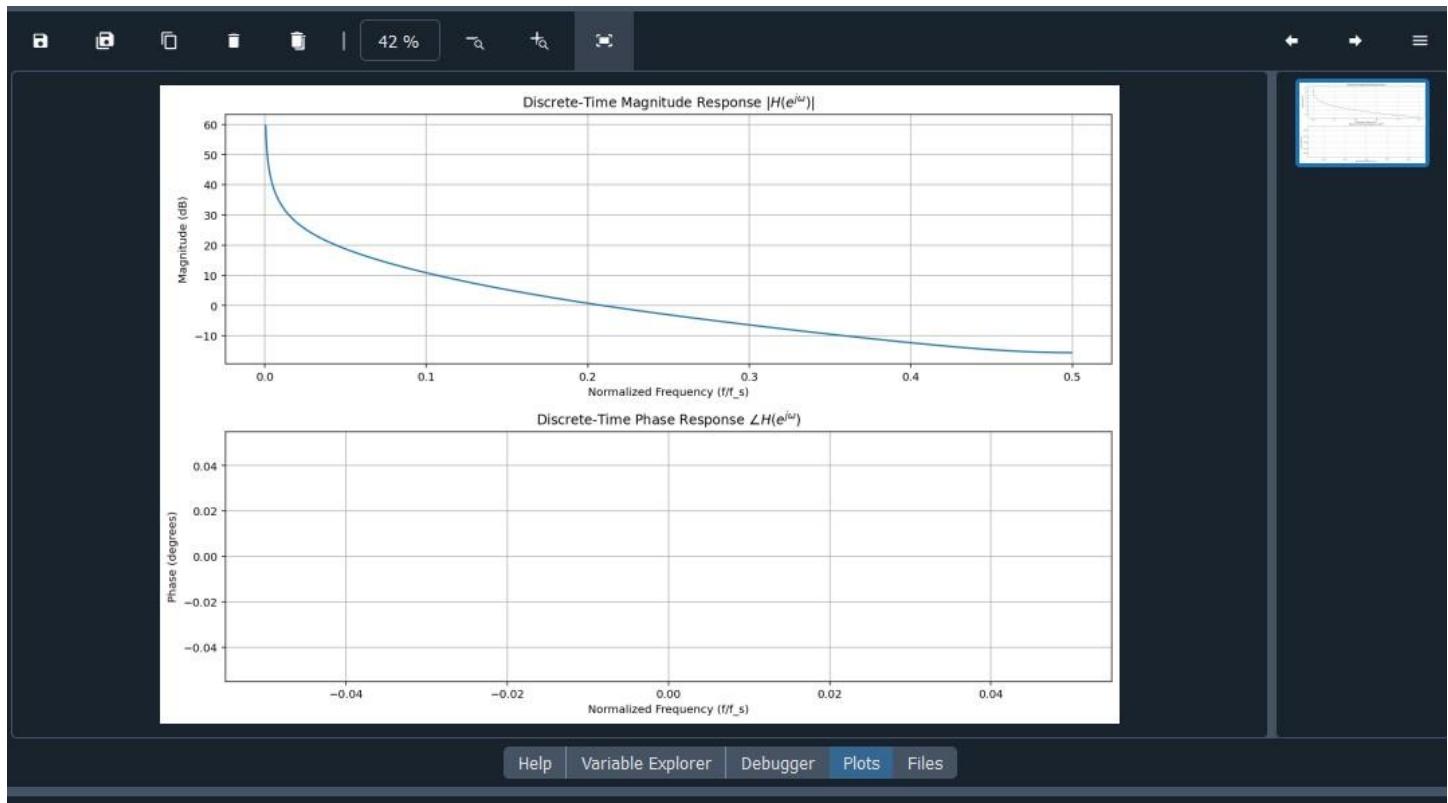
```
plt.subplot(2, 1, 1) plt.semilogx(w,
mag) # Bode magnitude plot
plt.title('Bode Magnitude Plot')
plt.xlabel('Frequency [rad/s]')
plt.ylabel('Magnitude [dB]') plt.grid()
plt.subplot(2, 1, 2) plt.semilogx(w,
phase) # Bode phase plot
plt.title('Bode Phase Plot')
plt.xlabel('Frequency [rad/s]')
plt.ylabel('Phase [degrees]') plt.grid()
plt.tight_layout() plt.show()

# Example: Analyzing a specific system H(z) = (z^2 + 0.5)/(z^2 - 1.5z + 0.5)

num = [1, 0.5] # Numerator coefficients den = [1, -1.5, 0.5] #

Denominator coefficients analyze z transfer function(num, den)
```

Subject: Programming With Python (01CT1309)
Aim: Analysis of LTI System Responses to Standard Inputs Using Python

Experiment No: 19
Date:
Enrollment No:92400133131
OUTPUT:

Transfer Function :

$$H(z) = \frac{0.5}{1 - 0.8z^{-1}}$$

Causality: This system is causal because the denominator has a non-negative exponent (i.e., all powers of z^{-1} are non-negative).

Stability: The system is stable if the poles (the roots of the denominator) lie inside the unit circle. Here, the pole is $z = 0.8$, which is inside the unit circle, so the system is stable.

Time Invariance: The system is time-invariant because the coefficients do not depend on time.

Transfer function:

Subject: Programming With Python (01CT1309)	Aim: Analysis of LTI System Responses to Standard Inputs Using Python	
Experiment No: 19	Date:	Enrollment No: 92400133131

$$H(z) = \frac{1 - z^{-1}}{0.5z^{-1}}$$

Causality: This system is causal.

Stability: The pole at $z = 0.5$ is inside the unit circle, making the system stable. Time

Invariance: The system is time-invariant

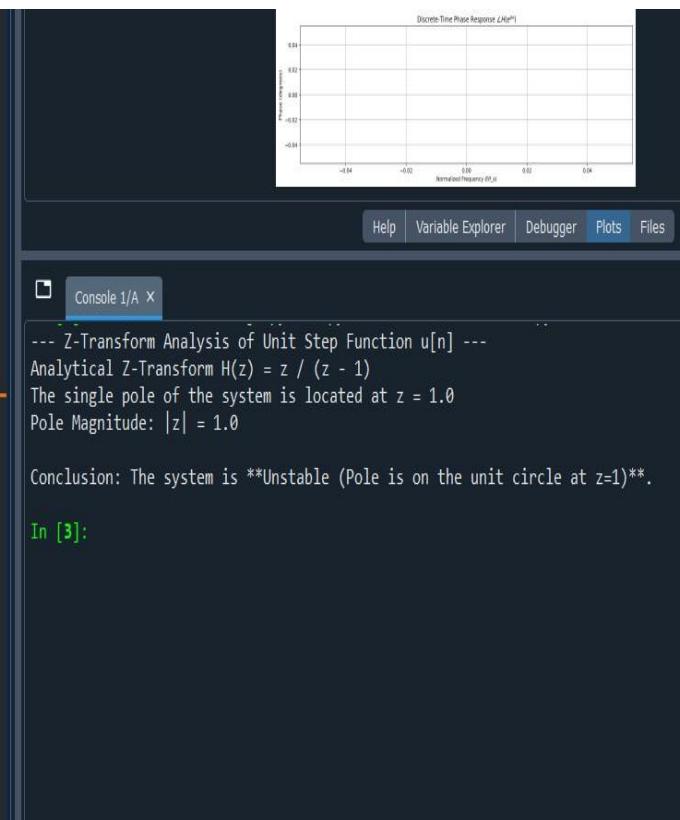
Post Lab Exercise:

- Write a Python function to compute the Z-transform of an unit step function. verify whether the system is stable or unstable. Code and output:-

```

6  """
7
8 import numpy as np
9
10 def analyze_unit_step_z_transform():
11     """
12         Computes the Z-transform of the unit step function u[n] and verifies
13         its stability based on the pole location.
14
15         The analytical Z-transform of the unit step is H(z) = z / (z - 1).
16     """
17     pole = 1.0
18
19     print("--- Z-Transform Analysis of Unit Step Function u[n] ---")
20     print(f"Analytical Z-Transform H(z) = z / (z - 1)")
21     print(f"The single pole of the system is located at z = {pole}")
22     magnitude = np.abs(pole)
23
24     print(f"Pole Magnitude: |z| = {magnitude}")
25
26     if magnitude < 1:
27         stability_result = "Stable"
28     else:
29         stability_result = "Unstable (Pole is on the unit circle at z=1)"
30
31     print(f"\nConclusion: The system is **{stability_result}**.")
32
33 # Run the analysis
34 analyze_unit_step_z_transform()

```



Subject: Programming With Python (01CT1309)		
Experiment No: 19	Date:	Enrollment No:92400133131

- Implement this for the system $H(z) = \frac{0.5(z-0.7)(z-0.9)}{(z-0.6)(z-0.4)}$ and verify whether the system is stable or

$$(z-0.6)(z-0.4)$$

```

7 import numpy as np
8
9
10 def analyze_z_system_stability(num_coeffs, den_coeffs):
11     """
12         Analyzes the stability of a discrete-time system defined by its
13         Z-transfer function H(z) = N(z) / D(z).
14
15     Args:
16         num_coeffs (list): Coefficients of the numerator N(z).
17         den_coeffs (list): Coefficients of the denominator D(z).
18             These should be in descending powers of z,
19             after clearing all negative powers of z.
20
21         Example: For H(z) = 1 / (1 - 0.5z^-1),
22             the characteristic equation is z - 0.5 = 0.
23             Use den_coeffs = [1, -0.5].
24
25     """
26     print("\n--- Z-System Stability Analysis ---")
27
28     poles = np.roots(den_coeffs)
29
30     print(f"Denominator Coefficients D(z): {den_coeffs}")
31     print(f"System Poles (roots of D(z)=0): {poles}")
32     pole_magnitudes = np.abs(poles)
33     print(f"Pole Magnitudes: {pole_magnitudes}")
34     is_unstable = np.any(pole_magnitudes >= 1)
35
36     if is_unstable:
37         stability_result = "Unstable (At least one pole is outside or on the unit circle, |z| >= 1)."
38     else:
39         stability_result = "Stable (All poles are strictly inside the unit circle, |z| < 1)."
40
41     print(f"\nConclusion: The system is **{stability_result}**.")
42
43     num = [1]          # Numerator
44     den = [1, -0.5]    # Denominator D(z)
45
46     print("Analyzing Example System: H(z) = 1 / (1 - 0.5z^-1)")
47     analyze_z_system_stability(num, den)

```

unstable.

Code and output:

