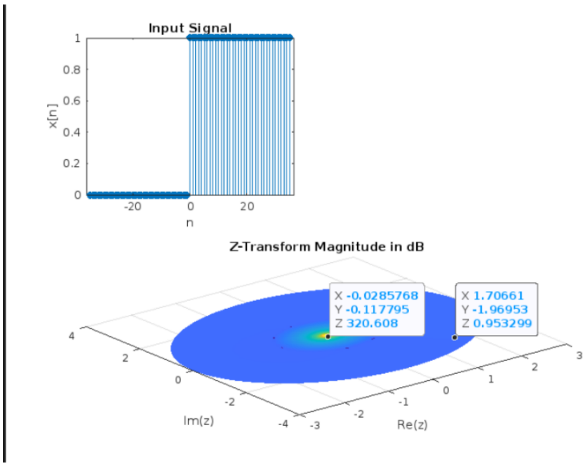
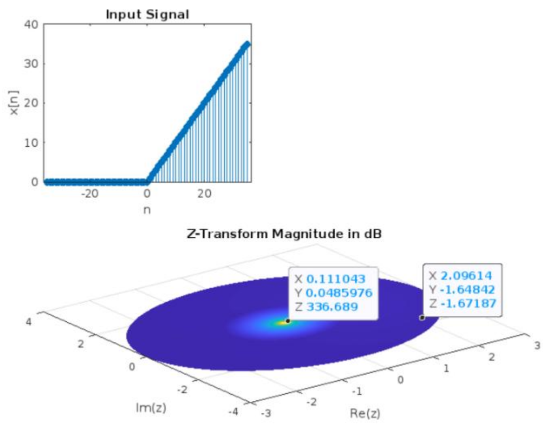
%To find the Z-transform of a given discrete-time signal x[n]

```
clear all;  close all;  clc;

n = -35:1:35;  % Time range for the signal

length_of_sqn = length(n);

x_n = zeros(1, length_of_sqn);  % Initialize i.p signal


% Choose and uncomment the desired signal:

% Unit step signal

for ii = 1 : length_of_sqn

   if (n(ii) < 0)

      x_n(ii) = 0;

   else

      x_n(ii) = 1;

   end end

% Ramp signal

% for ii = 1 : length_of_sqn

%     if (n(ii) < 0)

%        x_n(ii) = 0;

%     else

%        x_n(ii) = n(ii);

%     end end


% Exponential signal

% for ii = 1 : length_of_sqn

%     if (n(ii) >= 0)

%        x_n(ii) = 0.5.^n(ii);

%     else

%        x_n(ii) = 0;

%     end  end

% Double-sided exponential signal

% for ii = 1 : length_of_sqn

%     if (n(ii) >= 0)

%        x_n(ii) = 0.5.^n(ii);

%     else

%        x_n(ii) = 0.8.^n(ii);

%     end end

% Plotting the input signal

subplot(2,2,1);

stem(n, x_n, 'filled');

title('Input Signal');

xlabel('n');

ylabel('x[n]');


omega = linspace(-pi, pi, 100);

z = [];

% Generate Z values

for r = linspace(0, 3, 100)

   z = [z; r .* exp(1j .* omega)];

end

Xtemp = [];

 X = [];

  XFinal = [];

% Compute Z-transform

for c = 1:length(z)

   for b = 1:length(z)

      temp = 0;

      for a = 1:length(n)

         temp = temp + (x_n(a) .* (z(c,b) .^ (-n(a))));

      end

      Xtemp = [Xtemp, temp];

   end

   X = [X; Xtemp];

   Xtemp = [];

end

XFinal = X;
```

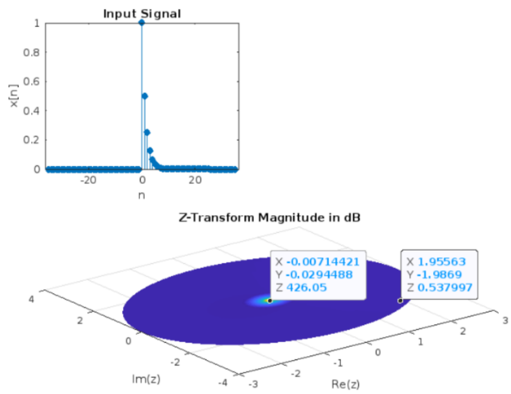
```
% Plot Z-transform result
subplot(2,2,3:4);
surf(real(z), imag(z), 10*log10(abs(XFinal)), 'linestyle', 'none');
title('Z-Transform Magnitude in dB');
xlabel('Re(z)');
ylabel('Im(z)');
zlabel('Magnitude (dB)');
```



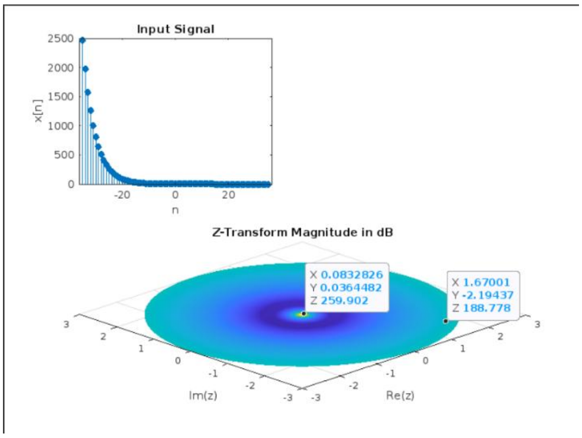
Unit step signal



Unit ramp signal



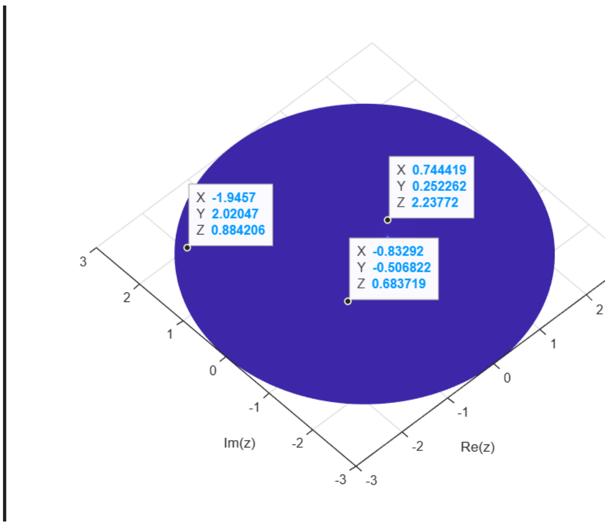
One sided exponential signal

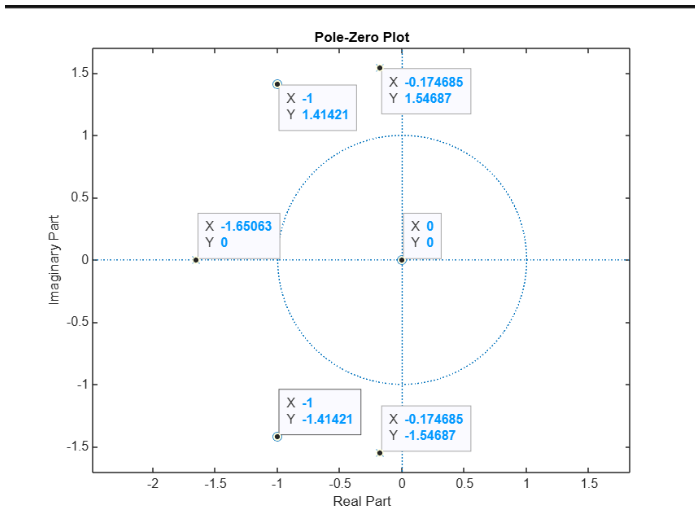


Double sided exponential signal

% 10 a: Visualization of Z plane

```
clear all;close all; clc;

% Define the frequency range

w = linspace(-pi, pi, 1001);  % Frequency range

% Initialize arrays for Z and H values

Z = [];  % Initialize Z values

H = [];  % Initialize H values


% Generate Z values for radius from 0 to 3

for r = linspace(0, 3, 1001)

   Z = [Z; r .* exp(1j * w)];  % Create Z values for
each radius

end

% Calculate the transfer function H(z)

H = (2 .* Z) ./ (2 .* Z - 1);  % H(z) formula


% Plotting the 3D mesh of H(z) in the Z plane

mesh(real(Z), imag(Z), abs(H));  % Plot mesh

grid on;

% Add labels to the plot

xlabel('Re(z)');  % Real part of Z

ylabel('Im(z)');  % Imaginary part of Z

zlabel('H(z)');   % Magnitude of H(z)
```



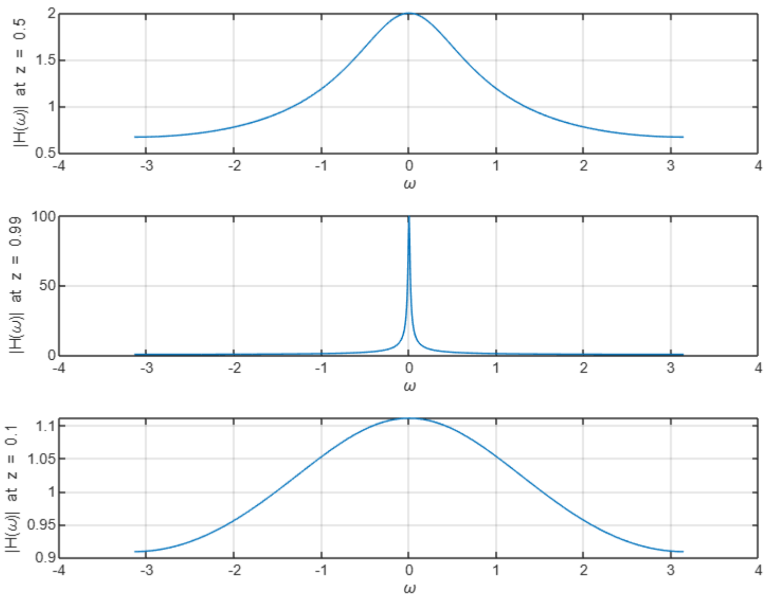


%10 b: Z-plane plot (Pole-Zero Plot)

```
clear all; close all; clc;

% Pole-Zero plot using zplane function

zplane([1 2 3], [1 2 3 4]);
```

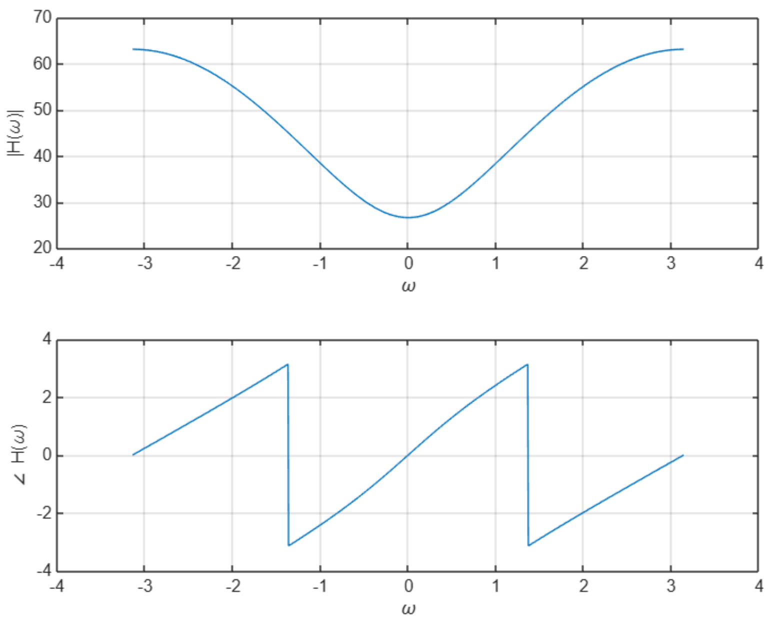
% 10 d Effect of distance of poles from the unit circle on |mag| and p.h spectrum

```
clear all; close all; clc;
 w = linspace(-pi, pi, 1001);  % Frequency range
 r = 1;  % Assuming r = 1 (you can change this as needed)
 z = r .* exp(1j * w);  % Compute Z values

% % First Transfer Function H = z / (z - 0.5)
% H1 = z ./ (z - 0.5);
% % Second Transfer Function H = z / (z - 0.99)
% H2 = z ./ (z - 0.99);
% % Third Transfer Function H = z / (z - 0.1)
% H3 = z ./ (z - 0.1);
% Plot the magnitude of the first transfer function
subplot(3,1,1);
 plot(w, abs(H1));  % Magnitude of H1
xlabel('\omega');
ylabel('|H(\omega)| at z = 0.5');
 grid on;

% Plot the magnitude of the second transfer function
 subplot(3,1,2);
 plot(w, abs(H2));  % Magnitude of H2
 xlabel('\omega');
 ylabel('|H(\omega)| at z = 0.99');
 grid on;

 % Plot the magnitude of the third transfer function
 subplot(3,1,3);
 plot(w, abs(H3));  % Magnitude of H3
 xlabel('\omega');
 ylabel('|H(\omega)| at z = 0.1');
grid on;
```



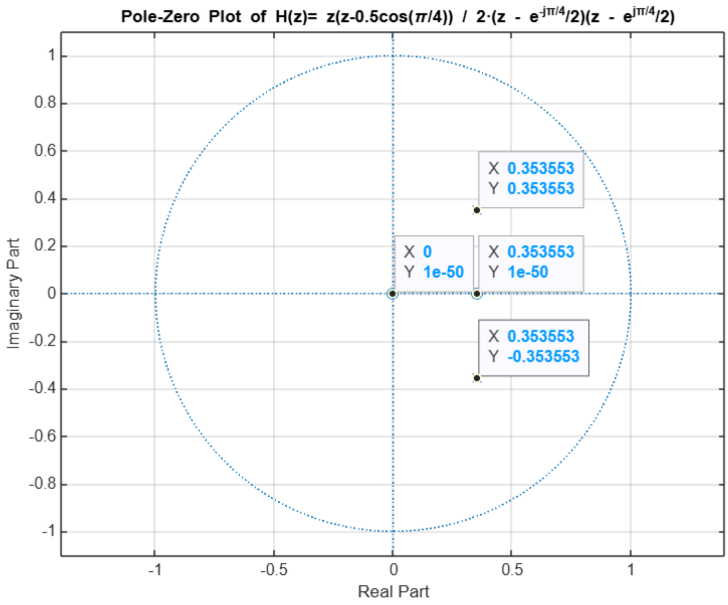
%10 c: Magnitude and Phase of a System

```
clear all; close all; clc;

 w = linspace(-pi, pi, 1001);  % Frequency range

 r = 1;  % Radius for the Z-plane

 z = r .* exp(1j * w);  % Z values


% Define the transfer function H(z)

 H = ((6 .* z) .* (7 .* z - 3)) ./ ((2.8 .* z - 1) ./ (3 .* z -
1));

%% Plot the magnitude and phase of H(z)

figure;


 % Plot magnitude of H(z)

subplot(2,1,1);

plot(w, abs(H));  % Absolute value of H(z)

xlabel('\omega');

ylabel('|H(\omega)|');

grid on;


% Plot phase of H(z)

subplot(2,1,2);

plot(w, angle(H));

xlabel('\omega');

ylabel('\angle H(\omega)');

grid on;
```



%ASSIGMENT: Z-plane plot (Pole-Zero Plot)

```
%H(z) = z(z-0.5cos(\pi/4)) / 2·(z - e^{-jπ/4}/2)(z - e^{jπ/4}/2)

 clear all; close all; clc;

% Define zeros

 z1 = 0;

 z2 = sqrt(2)/4;

 zeros = [z1, z2];

  % Define poles (conjugate complex)

 p1 = (1/2)*exp(1j*pi/4);

 p2 = (1/2)*exp(-1j*pi/4);

 poles = [p1, p2];

 % Get numerator and denominator coefficients

 num = poly(zeros);      % z*(z - sqrt(2)/4)

 den = 2 * poly(poles);   % 2*(z - e^jπ/4/2)*(z - e^-jπ/4/2)

% Plot pole-zero plot

zplane(num, den);

 title(' Pole-Zero Plot of H(z)= z(z-0.5cos(\pi/4)) / 2·(z - e^{-jπ/4}/2)(z - e^{jπ/4}/2)');

 grid on;
```



%ASSIGNMENT: Z-plane plot (Pole-Zero Plot)

```
%H(z) = z·sin(\pi/4) / 2·(z - e^{-jπ/4}/2)(z - e^{jπ/4}/2)

clear all; close all; clc;

 % Numerator zero

 z0 = 0;  % zero at origin

 gain = sin(pi/4);  % gain from numerator

 % Poles (complex conjugates)

 p1 = (1/2) * exp(1j * pi/4);

p2 = (1/2) * exp(-1j * pi/4);


 % Get coefficients

 num = gain * poly([z0]);       % z * sin(pi/4)

 den = 2 * poly([p1, p2]);      % 2 * (z - e^jpi/4/2)(z - e^-jpi/4/2)

 % Plot pole-zero plot

 zplane(num, den);

 title('Pole-Zero Plot of H(z) = z·sin(\pi/4) / 2·(z - e^{-jπ/4}/2)(z - e^{jπ/4}/2)');

 grid on;
```