

***NAMAL UNIVERSITY MIANWALI***

***DEPARTMENT OF ELECTRICAL ENGINEERING***

***EE 345 (L) – Digital Signal Processing (Lab)***

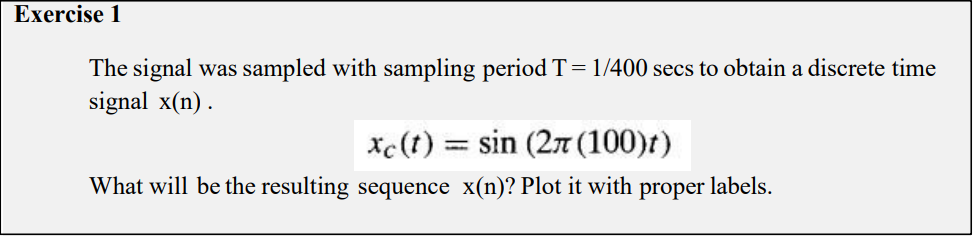
***LAB # 07***

***REPORT***

***Title :***

***Digital Processing of Continuous Signal (Sampling) in MATLAB***

|  |  |
| --- | --- |
| ***Name*** | ***Fahim Ur Rehman Shah*** |
| ***Roll No*** | ***NIM-BSEE-2021-24*** |
| ***Intructor*** | ***Zulaikha Kiran*** |
| ***Lab Engineer*** | ***Engr. Faizan Ahmad*** |
| ***Date Performed*** | ***17-April-2024*** |
| ***Marks*** |  |



% Sampling period

T = 1/400;

% Sampling frequency

fs = 1/T;

% Time array

t = 0:T:1-T;

% Discrete-time signal

x = sin(2\*pi\*100\*t);

subplot(2,1,1)

% Plot the discrete-time signal

plot(t, x, 'DisplayName', 'x(t)');

xlabel('Time (sec)');

ylabel('Amplitude');

title('Continous-time Signal x(t)');

legend;

% Plot the discrete-time signal

subplot(2,1,2)

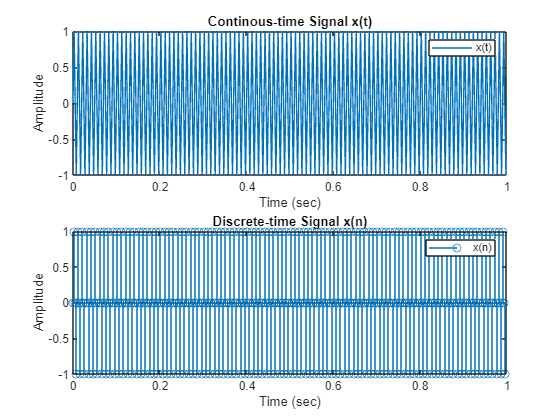
stem(t, x, 'DisplayName', 'x(n)');

xlabel('Time (sec)');

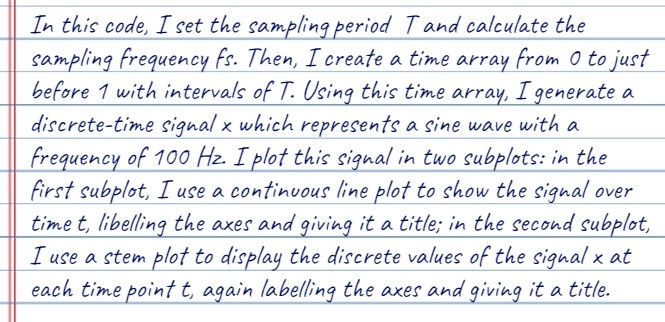
ylabel('Amplitude');

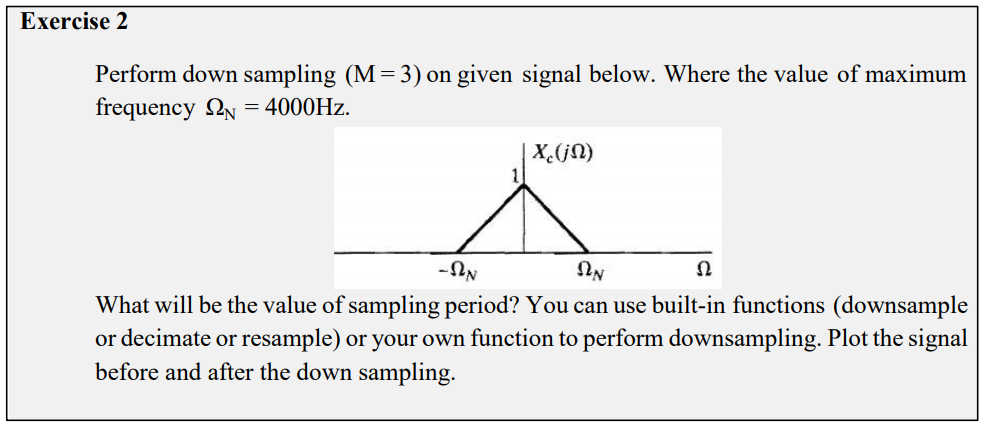
title('Discrete-time Signal x(n)');

legend;



***Explanation:***





% Given parameters

Omega\_N = 4000; % Maximum frequency in Hz

M = 3; % Downsampling factor

% Calculate sampling period

Ts = 1 / Omega\_N;

% Define the range of Omega

Omega = linspace(-Omega\_N, Omega\_N, 1000);

% Define the original signal X(Ω) as a triangular pulse

X\_Omega = zeros(size(Omega));

X\_Omega(Omega >= -Omega\_N & Omega <= 0) = (Omega(Omega >= -Omega\_N & Omega <= 0) + Omega\_N) / Omega\_N;

X\_Omega(Omega > 0 & Omega <= Omega\_N) = (Omega\_N - Omega(Omega > 0 & Omega <= Omega\_N)) / Omega\_N;

% Perform downsampling

X\_downsampled = downsample(X\_Omega, M);

Omega\_downsampled = downsample(Omega, M)/3;

% Plot the original and downsampled signals

figure;

subplot(2, 1, 1);

plot(Omega, X\_Omega, 'b', 'LineWidth', 2);

title('Original Signal');

xlabel('Omega');

ylabel('X(Omega)');

grid on;

subplot(2, 1, 2);

stem(Omega\_downsampled, X\_downsampled, 'r', 'LineWidth', 2);

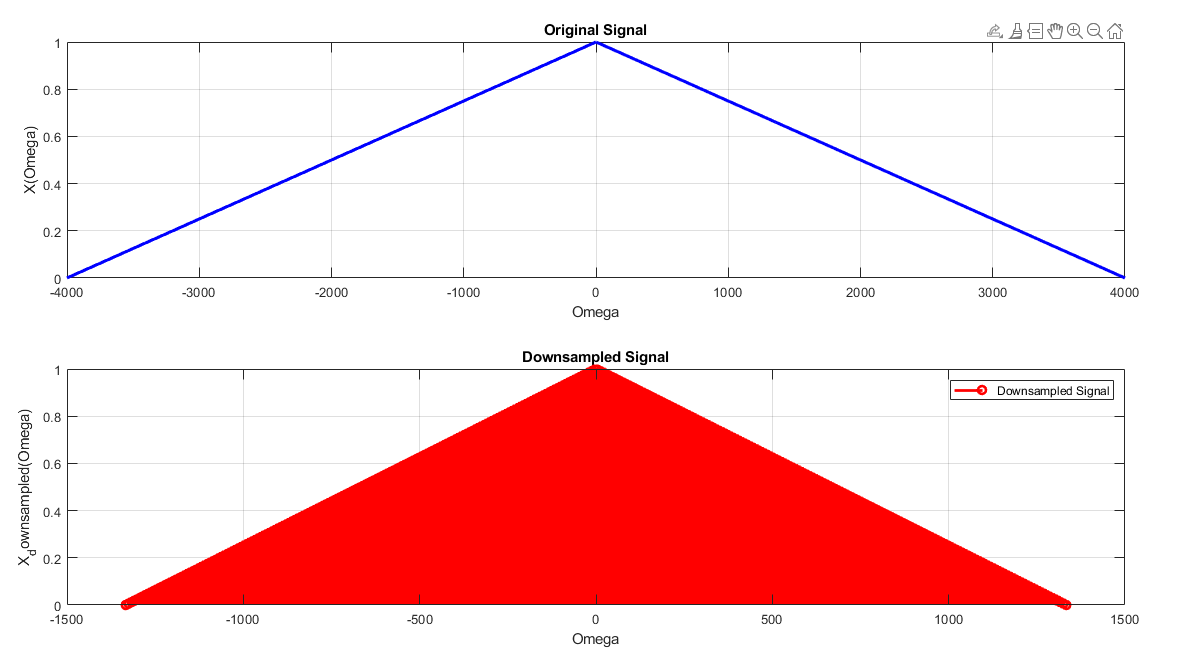
title('Downsampled Signal');

xlabel('Omega');

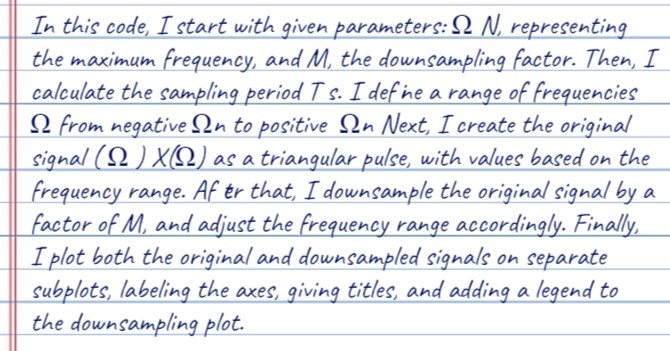
ylabel('X\_downsampled(Omega)');

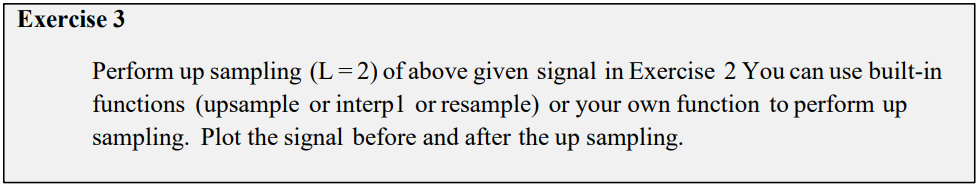
grid on;

legend('Downsampled Signal');



***Explanation:***





% Given parameters

Omega\_N = 4000; % Maximum frequency in Hz

M = 3; % Downsampling factor

L = 2; % Upsampling factor

% Calculate sampling period

Ts = 1 / Omega\_N;

% Define the original signal X(Ω) as a triangular pulse (same as Exercise 2)

Omega = linspace(-Omega\_N, Omega\_N, 1000);

X\_Omega = zeros(size(Omega));

X\_Omega(Omega >= -Omega\_N & Omega <= 0) = (Omega(Omega >= -Omega\_N & Omega <= 0) + Omega\_N) / Omega\_N;

X\_Omega(Omega > 0 & Omega <= Omega\_N) = (Omega\_N - Omega(Omega > 0 & Omega <= Omega\_N)) / Omega\_N;

% Perform downsampling (same as Exercise 2)

X\_downsampled = X\_Omega(1:M:end);

Omega\_downsampled = Omega(1:M:end);

% Perform upsampling by inserting zeros

upsampled\_length = length(X\_downsampled) \* L;

X\_upsampled = zeros(1, upsampled\_length);

X\_upsampled(1:L:end) = X\_downsampled\*3;

% Upsample the Omega array accordingly

Omega\_upsampled = linspace(-Omega\_N, Omega\_N, upsampled\_length)\*3;

% Plot the original and upsampled signals

figure;

subplot(2, 1, 1);

plot(Omega, X\_Omega, 'b', 'LineWidth', 2);

title('Original Signal');

xlabel('Omega');

ylabel('X(Omega)');

grid on;

legend('Original Signal');

subplot(2, 1, 2);

plot(Omega\_upsampled, X\_upsampled, 'r', 'LineWidth', 2);

xlim([-14000 14000])

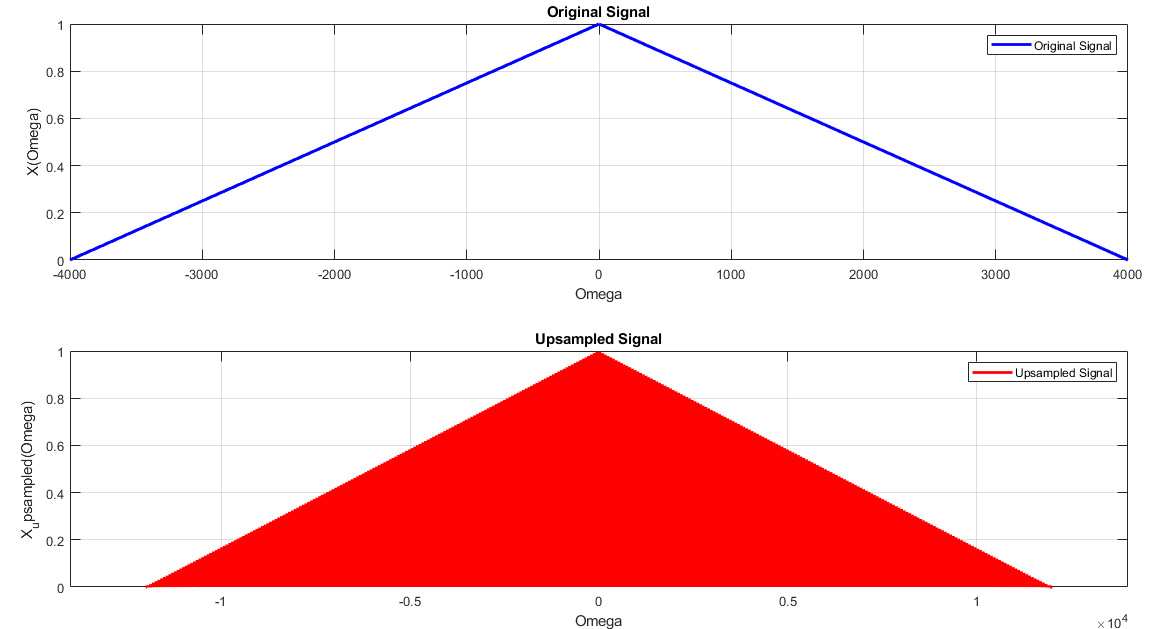
title('Upsampled Signal');

xlabel('Omega');

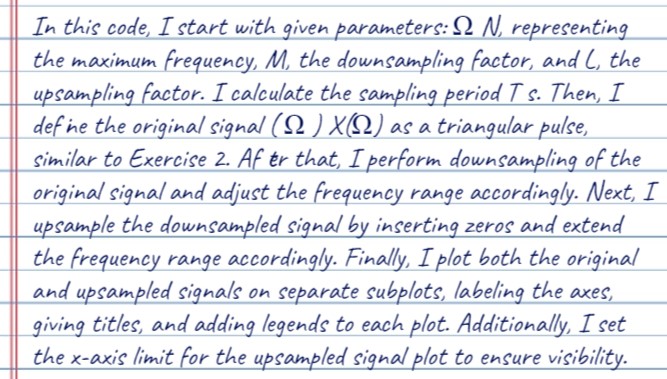
ylabel('X\_upsampled(Omega)');

grid on;

legend('Upsampled Signal');



***Explanation:***



***Conclusion:***

