Homework 1: Matlab and Aliasing

1. Write two MATLAB function that generate an array of numbers that represent the samples of the simple signal, X(n) = cos( (n-1) \* Omega \* T ).

(a). The first function should use a loop to generate the samples

% Initialization of the function

function x = SampleSignal( Samples, omega, TimInterval)

% -----------------------------------------------------------------

% TITLE: Sample Signal

%

% Purpose: This function generates an array of numbers that represent the samples of

% given signal

%

% Operation: x = SampleSignal( sample, omega, Time Interval)

%

% Inputs: ( Samples ) = number of samples taken

% ( omega ) = frequency in rad/sec

% ( TimInterval ) = sampling time interval in seconds

%

% Outputs: Array of numbers that represent the samples of

% given signal

%

% Other variables: k

%

% Date created: 06/18/2016 Author: Tamoghna Chattopadhyay

% Date modified: rev1 - 06/22/2016

% -----------------------------------------------------------------

% Function body

for k = 1:Samples

x(k) = (cos((k-1)\*omega\*TimInterval));

end

return

# Output

>> omega=0.5;

>> TimInterval=2;

>> Samples=10;

>> SampleSignal( Samples, omega, TimInterval)

ans =

1.0000 0.5403 -0.4161 -0.9900 -0.6536 0.2837 0.9602 0.7539 -0.1455 -0.9111

(b). While the second will generate the array using the internal generation capability of MatLab.

% Initialization of the function

function y = TestSignal( Samples, omega, TimInterval)

% -----------------------------------------------------------------

% TITLE: Sample Signal

%

% Purpose: This function generates an array of numbers that represent the samples of

% given signal

%

% Operation: y = TestSignal( sample, omega, Time Interval)

%

% Inputs: ( Samples ) = number of samples taken

% ( omega ) = frequency in rad/sec

% ( TimInterval ) = sampling time interval in seconds

%

% Outputs: Array of numbers that represent the samples of

% given signal

%

% Other variables: none

%

% Date created: 06/18/2016 Author: Tamoghna Chattopadhyay

% Date modified: rev1 - 06/22/2016

% -----------------------------------------------------------------

% Function body

y = cos( omega \* TimInterval \* [0 : Samples-1] );

return

# Output

>> omega=0.5;

>> TimInterval=2;

>> Samples=10;

>> TestSignal( Samples, omega, TimInterval)

ans =

1.0000 0.5403 -0.4161 -0.9900 -0.6536 0.2837 0.9602 0.7539 -0.1455 -0.9111

1. Measure the time required by the two functions for the case of generating 1000 samples, Omega = 2\*pi\*10 and T = 200 m sec.

% Clear Workspace close all

clear

% Input of values for number of samples, Omega and time interval Samples = 1000;

omega = 62.8;

TimInterval = 0.0002;

% Calculate the time taken to create an array by SampleSignal and also display it

disp(' ')

disp('Time taken by SampleSignal')

tic

Array\_1 = SampleSignal(Samples,omega,TimInterval);

toc

% Calculate the time taken to create an array by TestSignal and also display it

disp(' ')

disp('Time taken by TestSignal')

tic

Array\_2 = TestSignal(Samples, omega, TimInterval);

toc

**Output**

>> TimeElapsed

Time taken by SampleSignal Elapsed time is 0.000370 seconds.

Time taken by TestSignal Elapsed time is 0.000057 seconds.

1. Use the more efficient function (shortest time for test values) to calculate and plot the following values of omega: (T=400 µs)
2. Ω = π\*4.5e3 rad/s
3. Ω = π\*4.75e3 rad/s
4. Ω = π\*5.00e3 rad/s
5. Ω = π\*5.25e3 rad/s
6. Ω = π\*5.5e3 rad/s

*Clearly, TestSignal seems to be the function which uses less time to generate the same array. Thus, we make use of that function to generate arrays A through E.*

% Clear Workspace clear

clc

% Initial Data

Samples = 1000; % n = Number of samples

TimInterval= 0.0004; % T = 400 micro seconds

%defining values of omega

omega = (4.5e3:250:5.5e3).\*pi;

% Creates an array for different values of omega

% Calculate the time taken to create an array by TestSignal and also display it

disp('Array\_A:')

tic % Start calculating time

Array\_A = TestSignal(Samples,omega(1),TimIntervals); % Define Array\_A

toc % Stop calculating time

disp('Array\_B:')

tic % Start calculating time

Array\_B = TestSignal(Samples,omega(2),TimInterval); % Define Array\_B

toc % Stop calculating time

disp('Array\_C:')

tic % Start calculating time

Array\_C = TestSignal(Samples,omega(3),TimInterval); % Define Array\_C

toc % Stop calculating time

disp('Array\_D:')

tic % Start calculating time

Array\_D = TestSignal(Samples,omega(4),TimInterval); % Define Array\_D

toc % Stop calculating time

disp('Array\_E:')

tic % Start calculating time

Array\_E = TestSignal(Samples,omega(5),TimInterval); % Define Array\_E

toc % Stop calculating time

% Plotting all the graphs

figure; % Automatically creates a new figure

plot(Array\_A,'b');

hold on

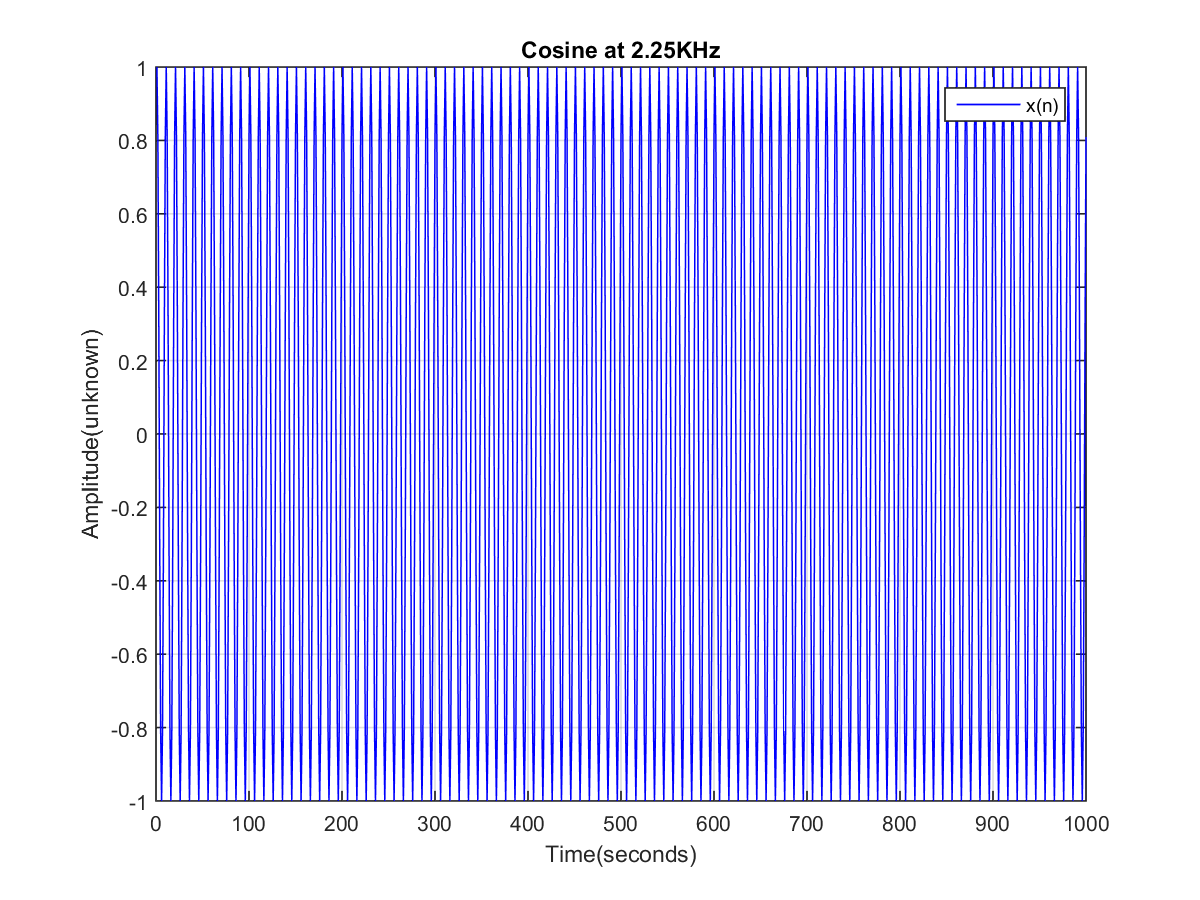
title('Cosine at 2.25KHz');

xlabel('Time(seconds)');

ylabel('Amplitude(unknown)');

legend('x(n)');

grid;



figure; % Automatically creates a new figure

plot(Array\_B,'r');

hold on

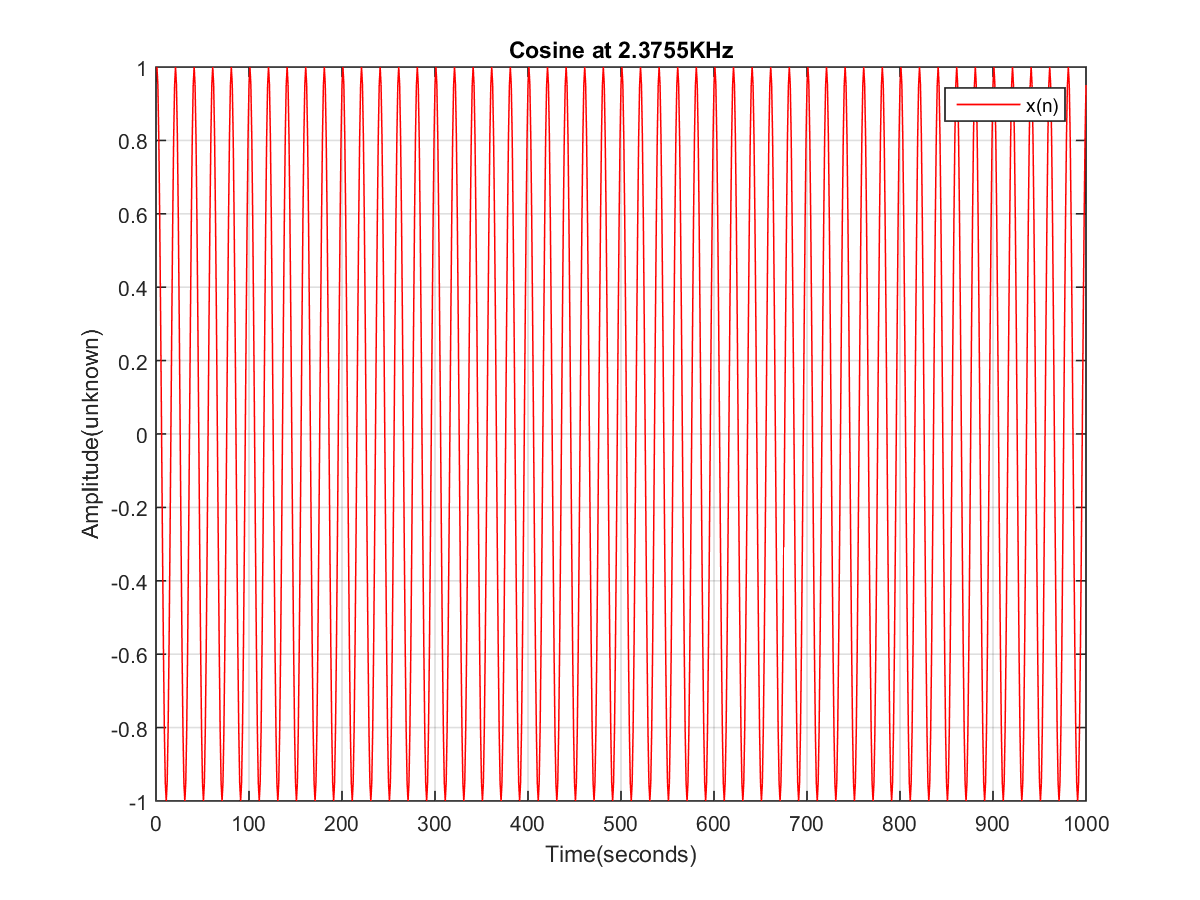
title('Cosine at 2.3755KHz');

xlabel('Time(seconds)');

ylabel('Amplitude(unknown)');

legend('x(n)');

grid;



figure; % Automatically creates a new figure

plot(Array\_C,'m');

hold on

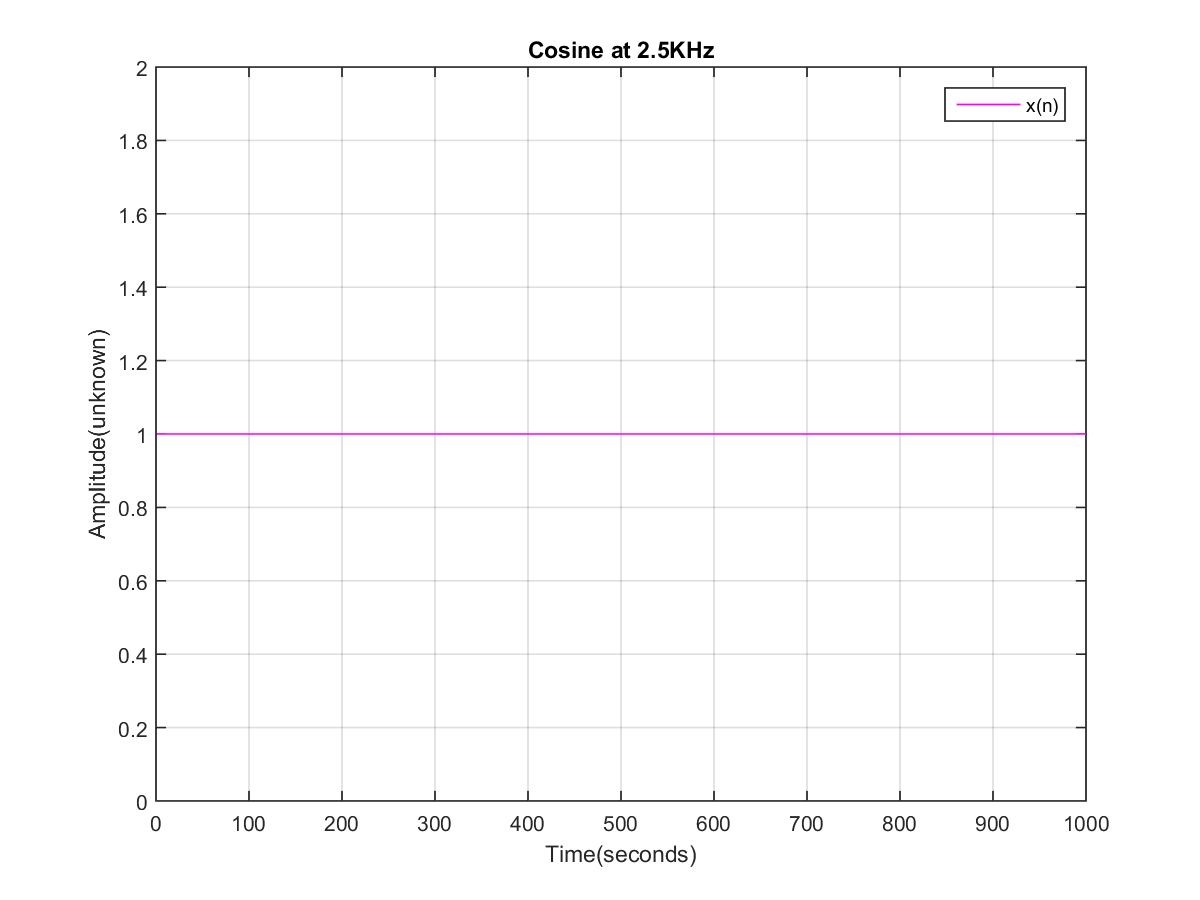
title('Cosine at 2.5KHz');

xlabel('Time(seconds)');

ylabel('Amplitude(unknown)');

legend('x(n)');

grid;



figure; % Automatically creates a new figure

plot(Array\_D,'g');

hold on

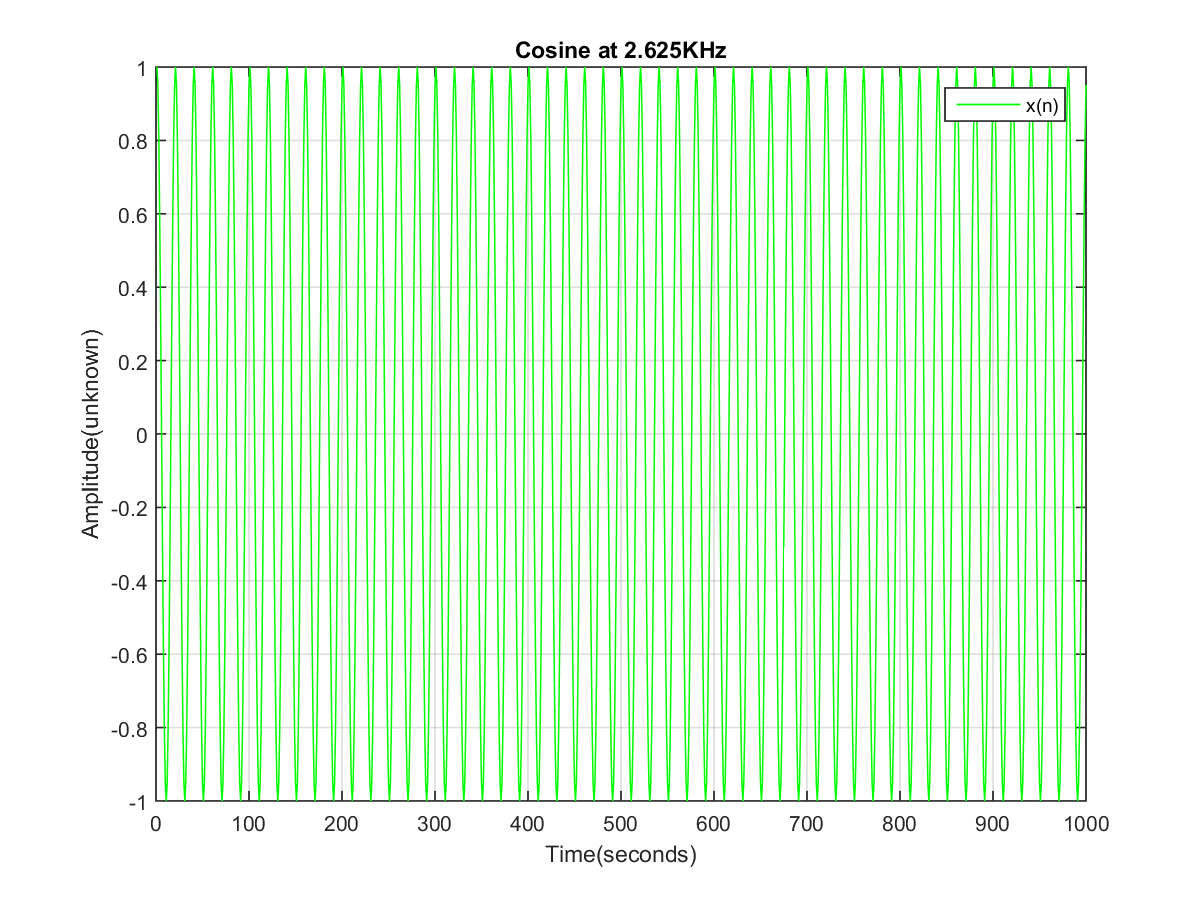
title('Cosine at 2.625KHz');

xlabel('Time(seconds)');

ylabel('Amplitude(unknown)');

legend('x(n)');

grid;



figure; % Automatically creates a new figure

plot(Array\_E);

hold on

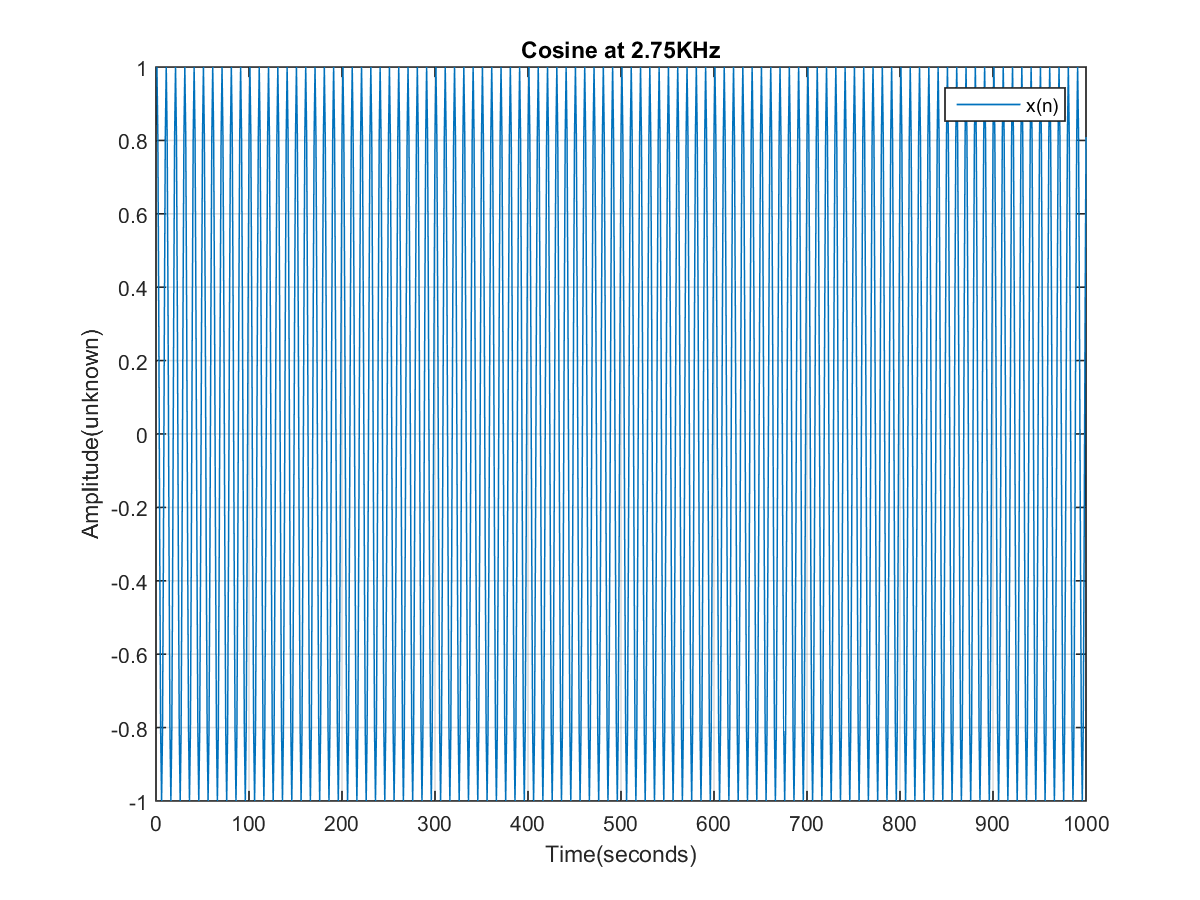
title('Cosine at 2.75KHz');

xlabel('Time(seconds)');

ylabel('Amplitude(unknown)');

legend('x(n)');

grid;



4.Calculate the difference between arrays A and E and plot the difference between them.

% Calculating Difference

Array\_F = Array\_A - Array\_E;

% Plotting the Difference Array

figure; % Automatically creates a new figure

plot(Array\_F,'r');

hold on

title('Array\_F = Array\_A - Array\_E');

xlabel('Time(second)');

ylabel('Amplitude(unknown)');

legend('x(n)');

grid;

