# 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
```

2. 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.
```

3. Use the more efficient function (shortest time for test values) to calculate and plot the following values of omega: (T=400 μs)

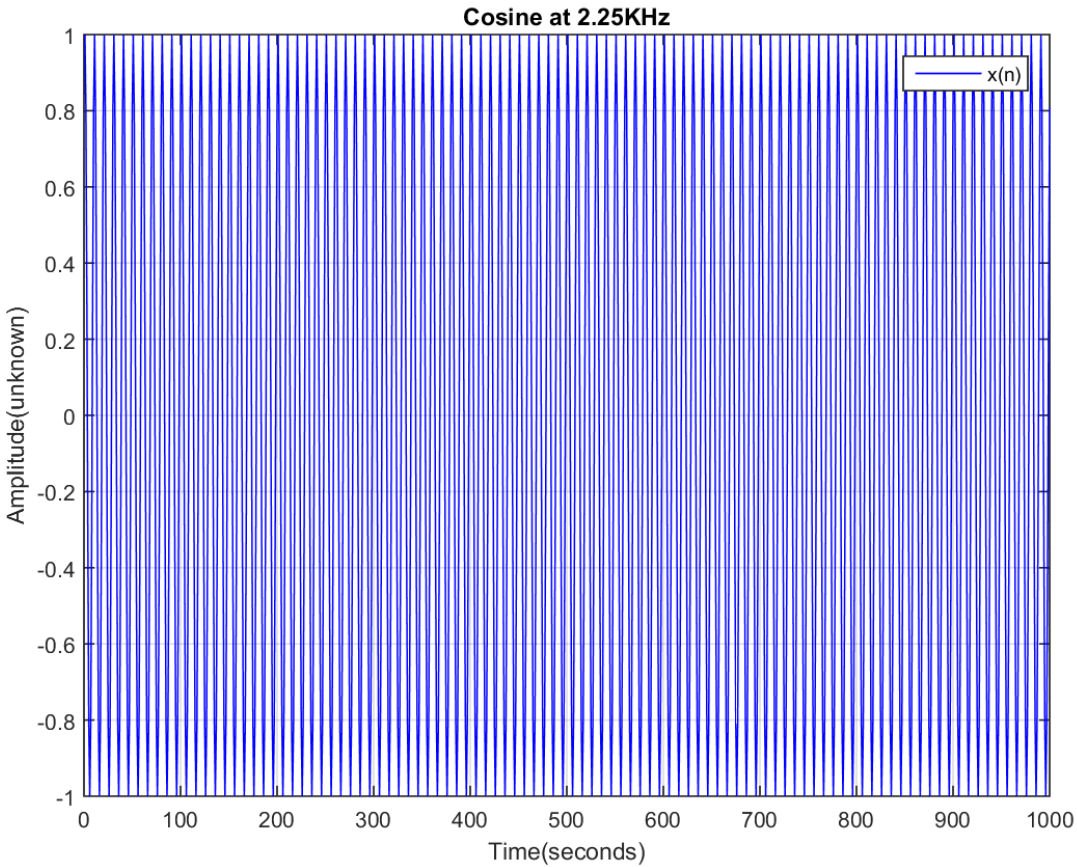
a) Ω = π*4.5e3 rad/s
b) Ω = π*4.75e3 rad/s
c) Ω = π*5.00e3 rad/s
d) Ω = π*5.25e3 rad/s
e) Ω = π*5.5e3 rad/s

*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.*

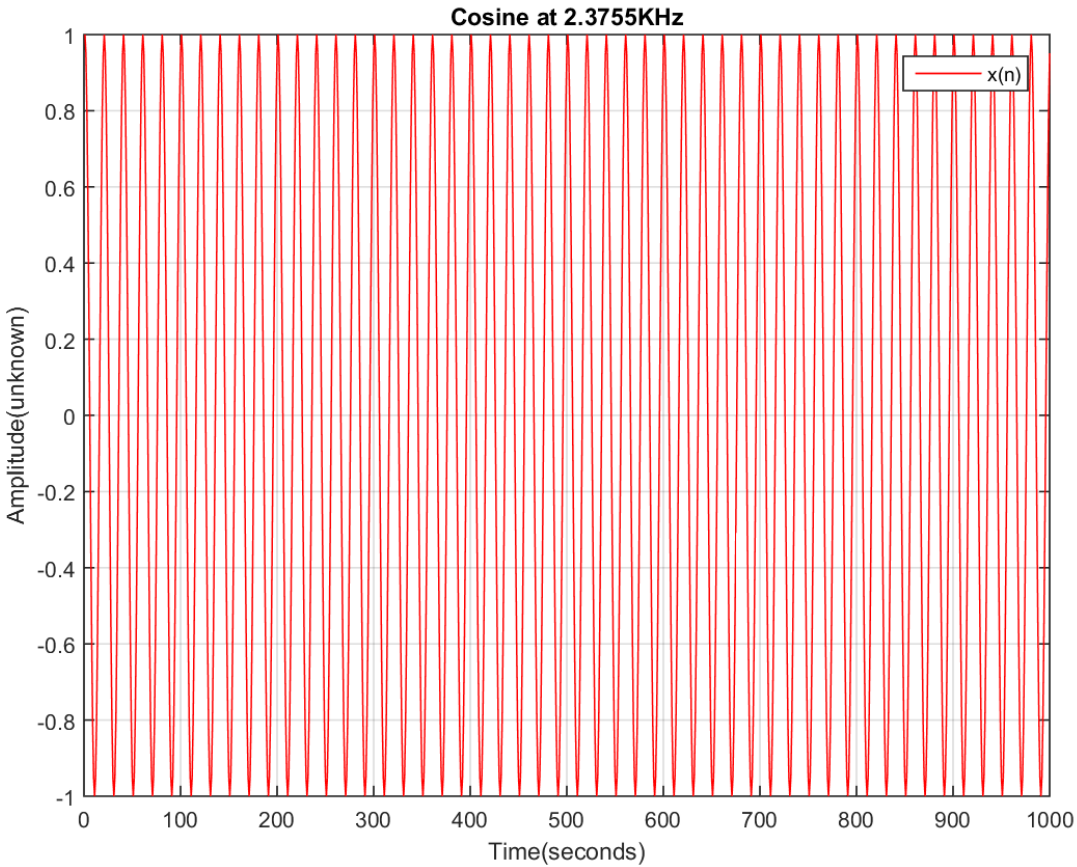
```
% Clear Workspace clear
clc


% 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
```

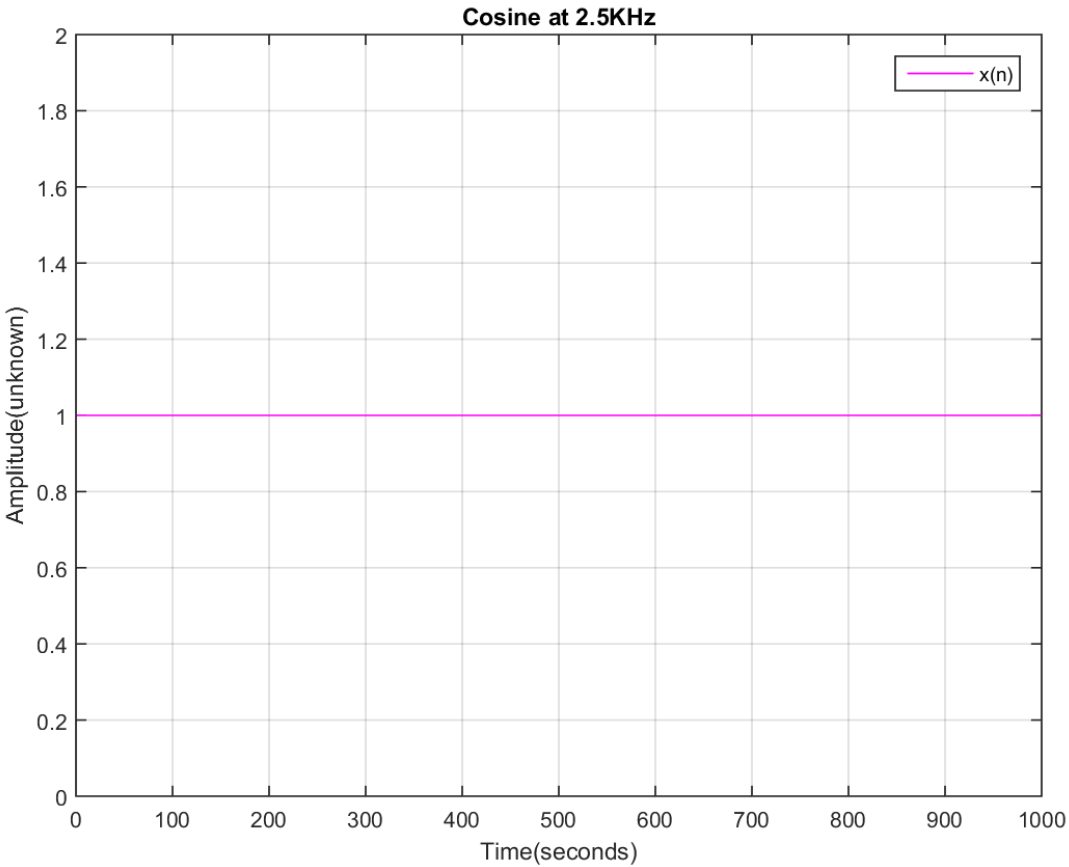
```
% Plotting all the graphs
figure; % Automatically creates a new figure
plot(Array_A,'b');
hold on
title('Cosine at 2.25KHz');
xlabel('Time(seconds)');
ylabel('Amplitude(unknown)');
legend('x(n)');
grid;
```


Cosine at 2.25KHz
x(n)
Amplitude(unknown)
Time(seconds)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
100
200
300
400
500
600
700
800
900
1000

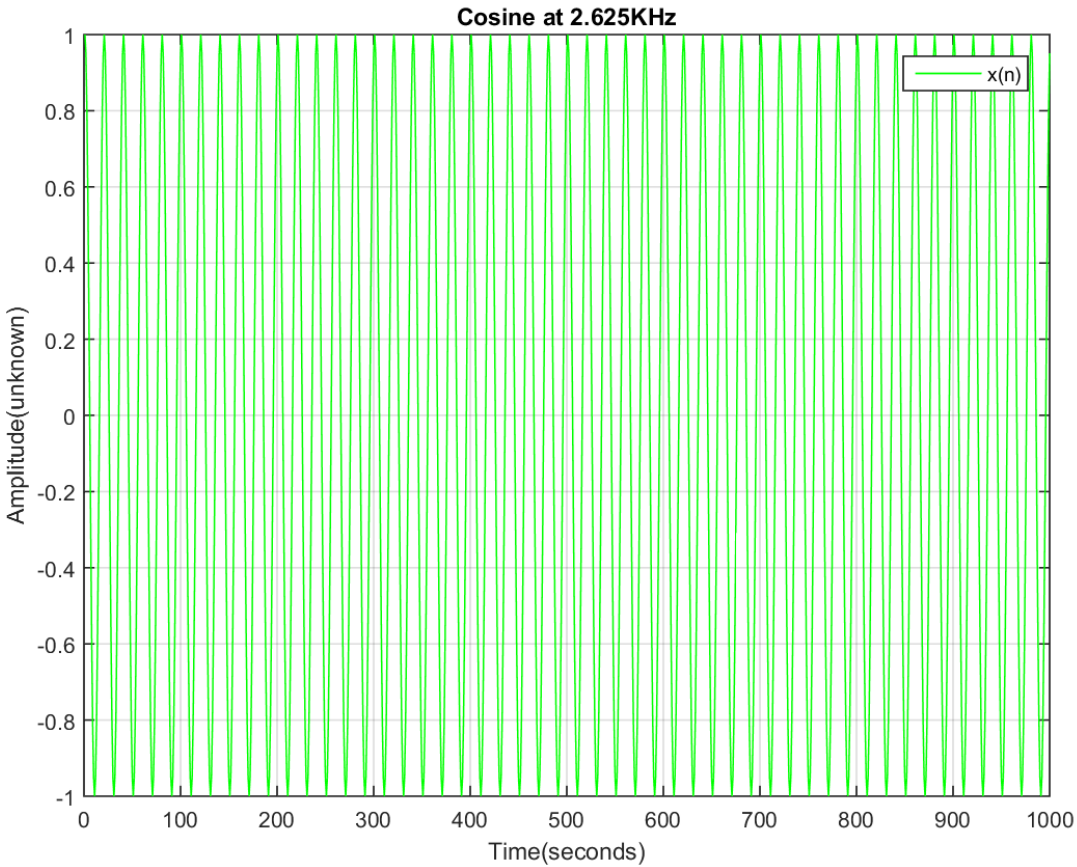
```
figure; % Automatically creates a new figure
plot(Array_B,'r');
hold on
title('Cosine at 2.3755KHz');
xlabel('Time(seconds)');
ylabel('Amplitude(unknown)');
legend('x(n)');
grid;
```


Cosine at 2.3755KHz
x(n)
Amplitude(unknown)
Time(seconds)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
100
200
300
400
500
600
700
800
900
1000

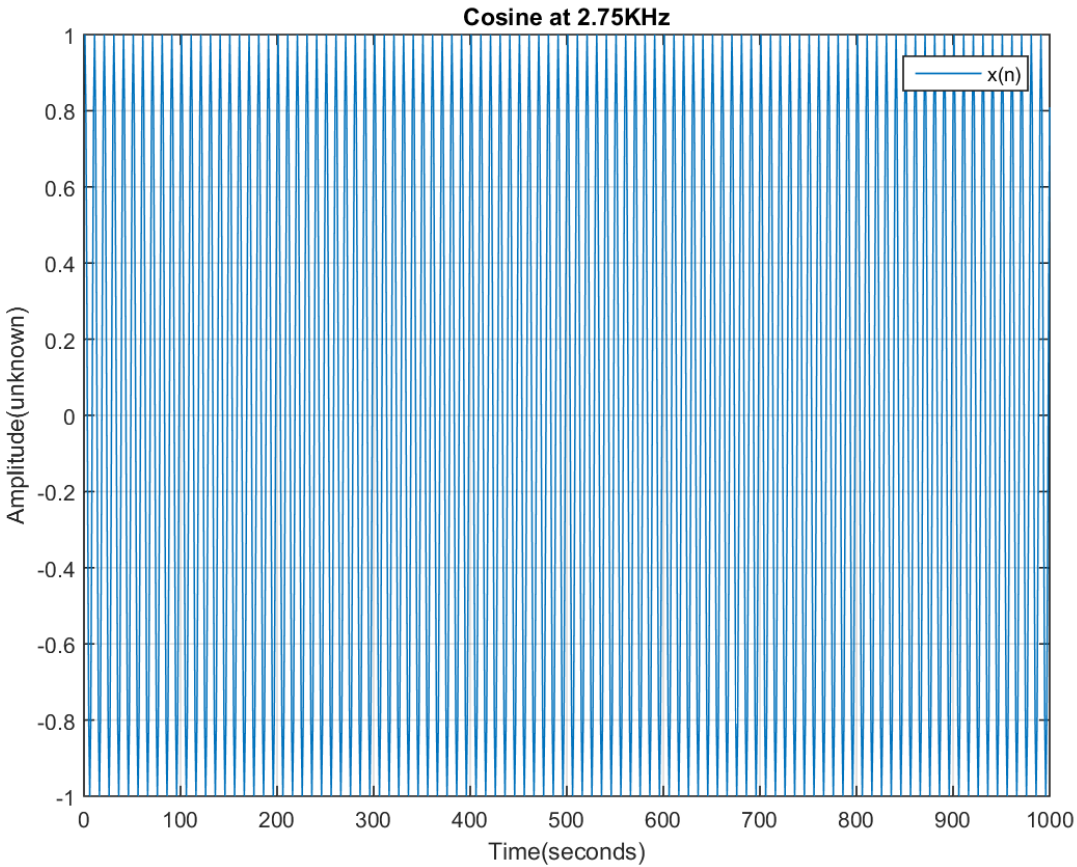
```
figure; % Automatically creates a new figure
plot(Array_C,'m');
hold on
title('Cosine at 2.5KHz');
xlabel('Time(seconds)');
ylabel('Amplitude(unknown)');
legend('x(n)');
grid;
```


Cosine at 2.5KHz
x(n)
Amplitude(unknown)
Time(seconds)
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
0
100
200
300
400
500
600
700
800
900
1000

```
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;
```

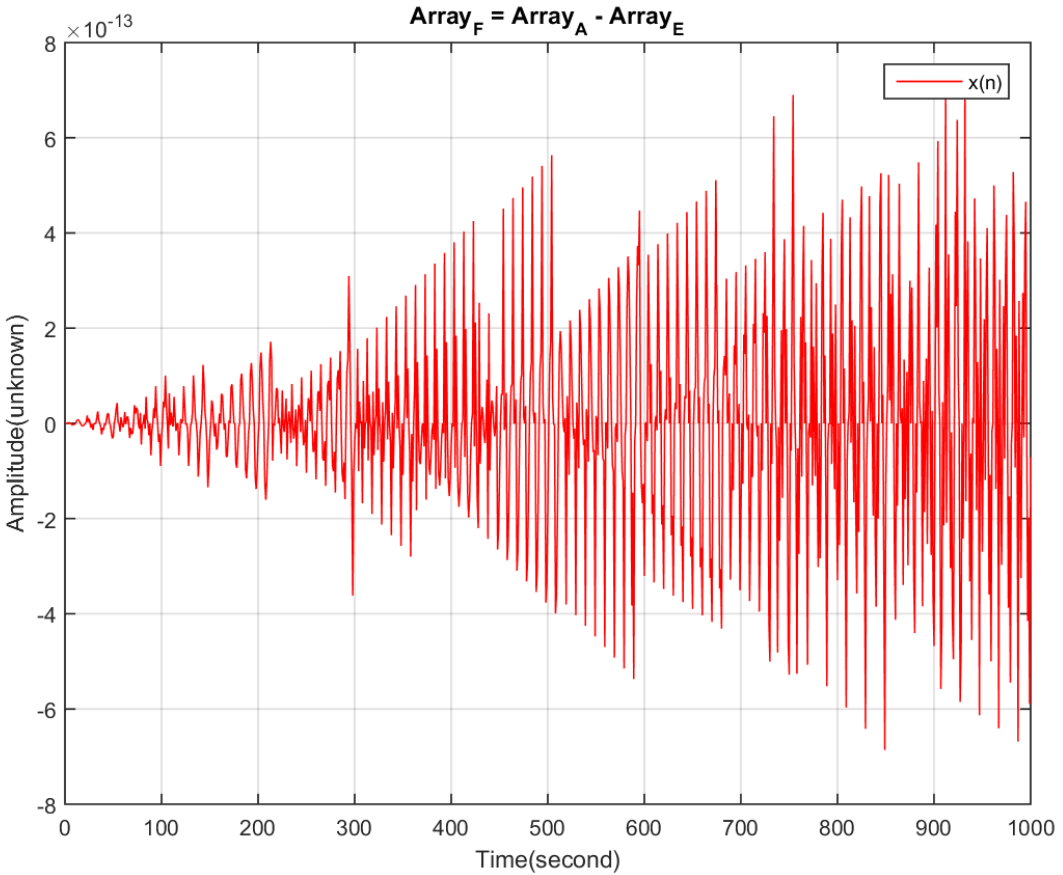

Cosine at 2.625KHz
x(n)
Amplitude(unknown)
Time(seconds)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
100
200
300
400
500
600
700
800
900
1000

```
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;
```


Cosine at 2.75KHz
x(n)
Amplitude(unknown)
Time(seconds)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
100
200
300
400
500
600
700
800
900
1000

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;
```


Array$_F$ = Array$_A$ - Array$_E$
×10^-13
8
6
4
2
0
-2
-4
-6
-8
Amplitude(unknown)
0
100
200
300
400
500
600
700
800
900
1000
Time(second)
x(n)