Assignment 7b (Companion Book, Chapter 3)

3.4 Eigenfunctions of Discrete-Time LTI Systems

This exercise examines the eigenfunction property of discrete-time LTI systems. Complex exponentials are eigenfunctions of LTI systems, i.e., when the input sequence is a complex

exponential, the output is the same complex exponential only scaled in amplitude by a complex constant. This constant can be computed from the impulse response h[n]. When

the input to a discrete-time LTI system is $x[n] = z^n$, the output is $y[n] = H(z)z^n$ an, where

$$H(z) = \sum_{n=-\infty}^{\infty} h[n]z^{-n}$$

Consider each of the following input signals:

$$x_1[n] = e^{j(\pi/4)n},$$
$$x_2[n] = sin(\pi n/8 + \pi/16),$$
$$x_3[n] = (9/10)^n,$$
$$x_4[n] = n + 1.$$

You will compute the outputs $y_1[n]$ through $y_4[n]$ that result when each of these signals is the input to the causal LTI system described by the linear,

constant-coefficient difference equation

$$y[n] - 0.25y[n-1] = x[n] + 0.9x[n-1]. \qquad (3.3)$$

Basic Problems

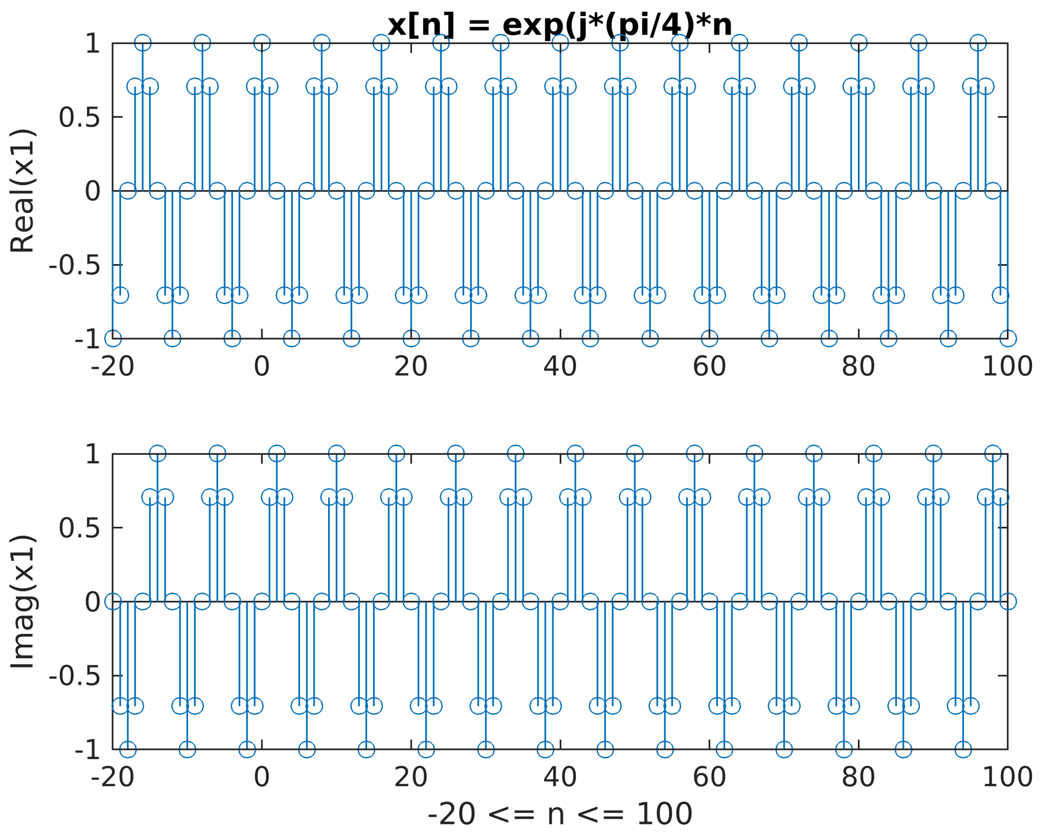
(a). Create a vector **n** containing the time indices for the interval $-20 \leq n \leq 100$ using the colon (:) operator. Using this vector, define x_1, x_2, x_3 and x_4 to be vectors

containing the values of the four signals $x_1[n]$ through $x_4[n]$ over the interval described by **n**. Produce a clearly labeled plot of each signal over this interval. Since the vector

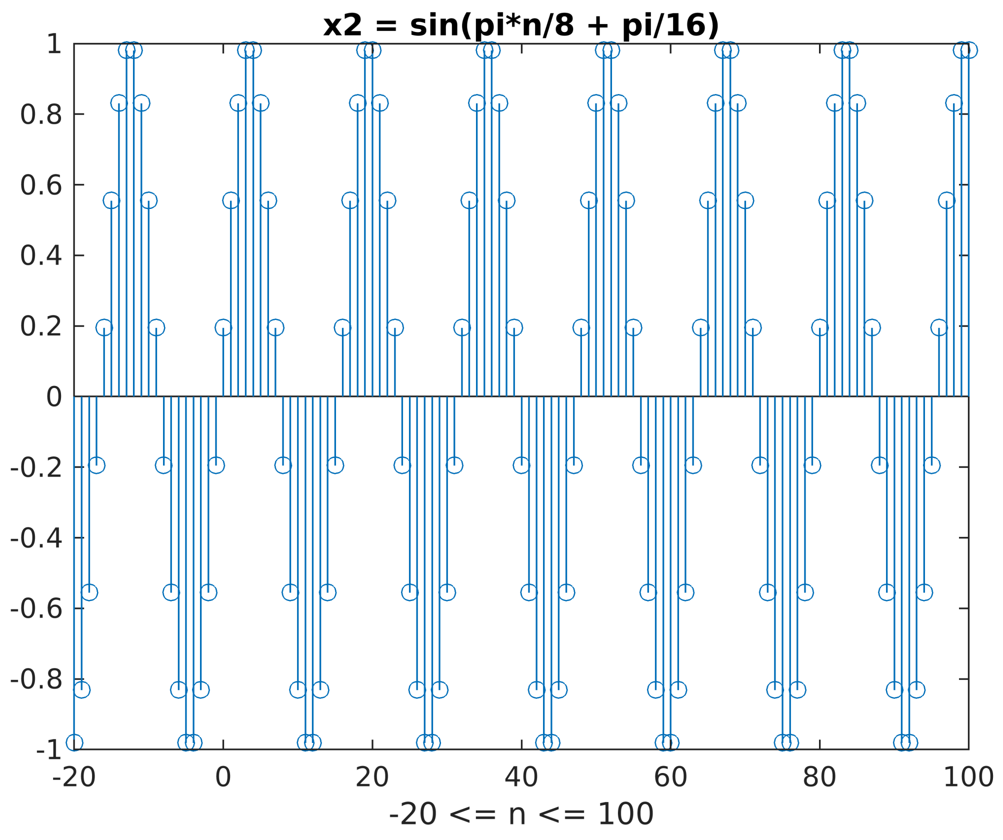
x_1 is complex, you will need to produce two separate plots for the real and imaginary parts. You can combine these in a single figure using either subplot or hold.

```
%REVISION
```

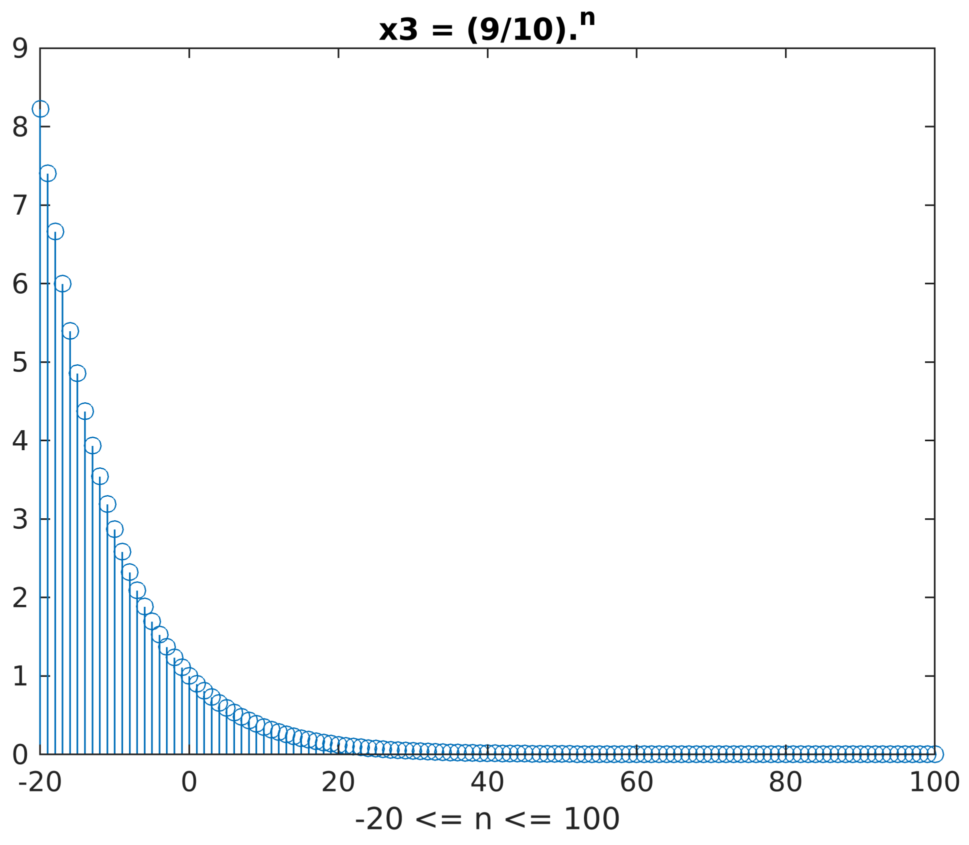
```
%3.4a
n = [-20:100];
x1 = exp(1j*(pi/4)*n);
x2 = sin(pi*n/8 + pi/16);
x3 = (9/10).^n;
x4 = n + 1;

figure;
subplot(2, 1, 1);
stem(n, real(x1));
title('x[n] = exp(j*(pi/4)*n');
ylabel('Real(x1)');

subplot(2, 1, 2);
stem(n, imag(x1)); xlabel('-20 <= n <= 100');
ylabel('Imag(x1)');
```



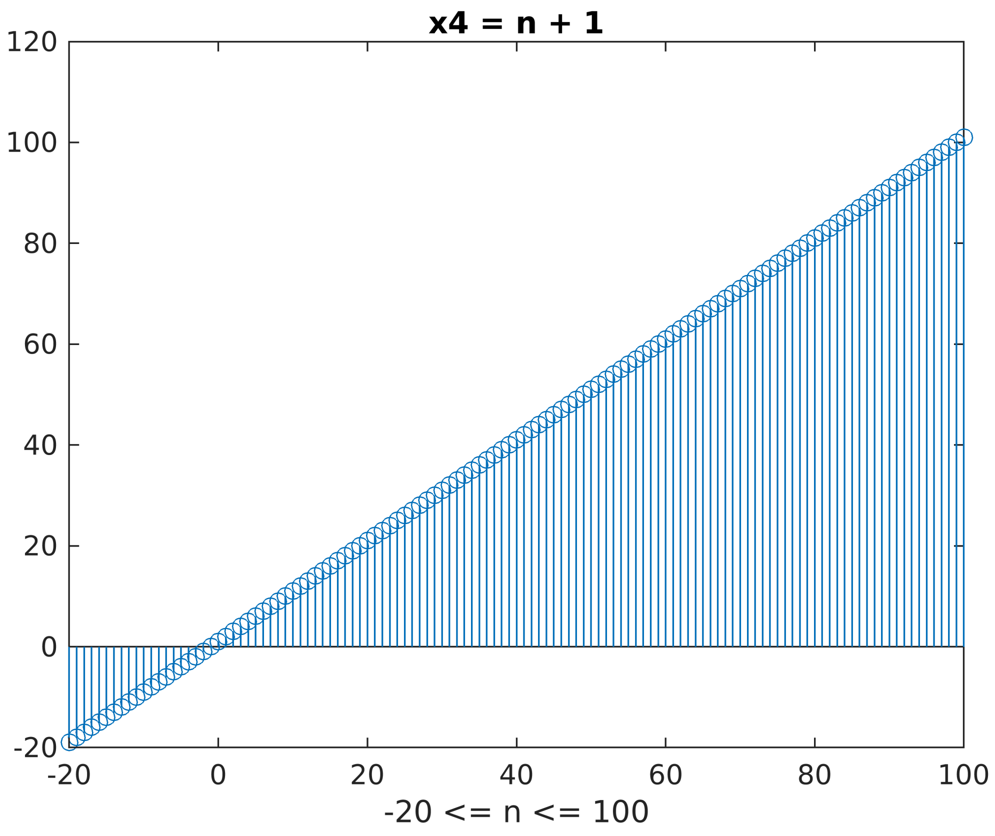
```
figure;
stem(n, x2);
title('x2 = sin(pi*n/8 + pi/16)');
xlabel('-20 <= n <= 100');
```



```
figure;
stem(n, x3);
title('x3 = (9/10).^n');
xlabel('-20 <= n <= 100');
```



```
figure;
stem(n, x4);
title('x4 = n + 1');
xlabel('-20 <= n <= 100');
```



(b). The **filter** command described in Tutorial 2.2 computes the output of the causal, LTI system described by a difference equation for a given input sequence. Define **a**

and **b** to specify the system shown in Eq. (3.3). Use these vectors and **filter** to compute the vectors $y_1,\ y_2,\ y_3$ and y_4, containing the output of the system specified

by Eq. (3.3) when the input is x_1 through x_4, respectively. For each of the outputs you obtain, produce appropriately labeled plots of the portion of the outputs over the

interval $-20 \le n \le 100$. For y_1, you will again need to plot the real and imaginary parts separately. Comparing your plots of inputs and outputs, indicate which of the input

signals are eigenfunctions of this LTI system.

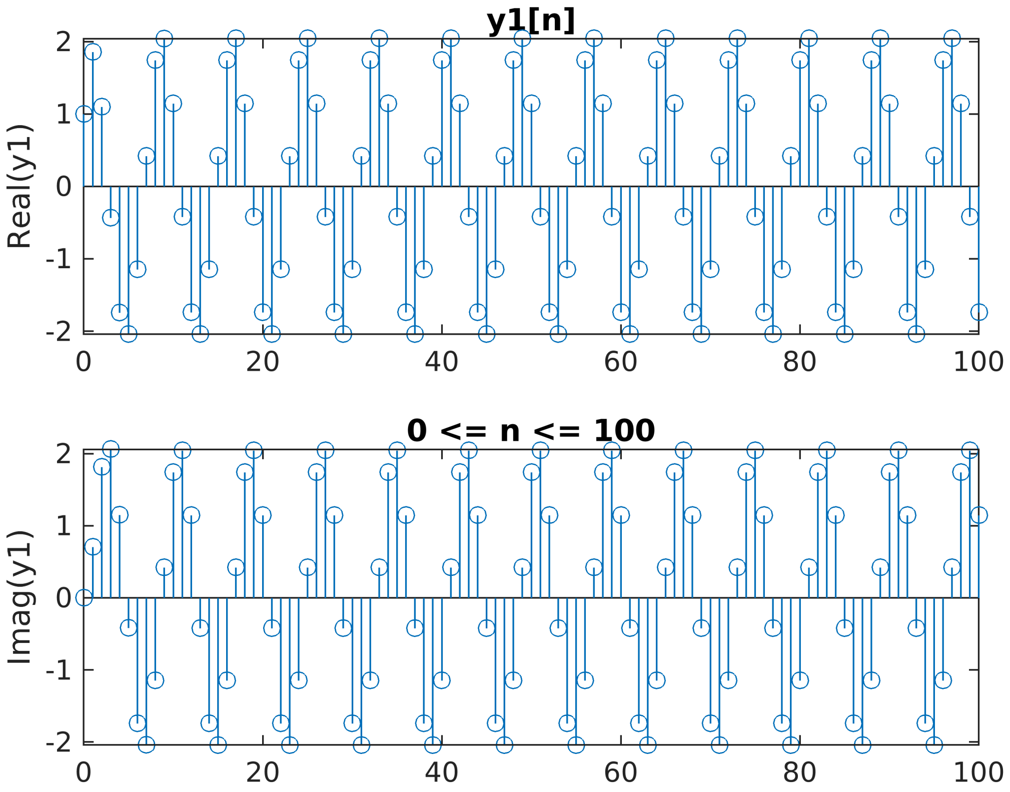
Note: In both this part and the next part we will ignore the output samples over the interval $-20 \le n \le -1$. These samples include transients due to the natural response

of the system because MATLAB is unable to work with infinitely long input signals. MATLAB assumes the input and output were zero before the values given in x. The

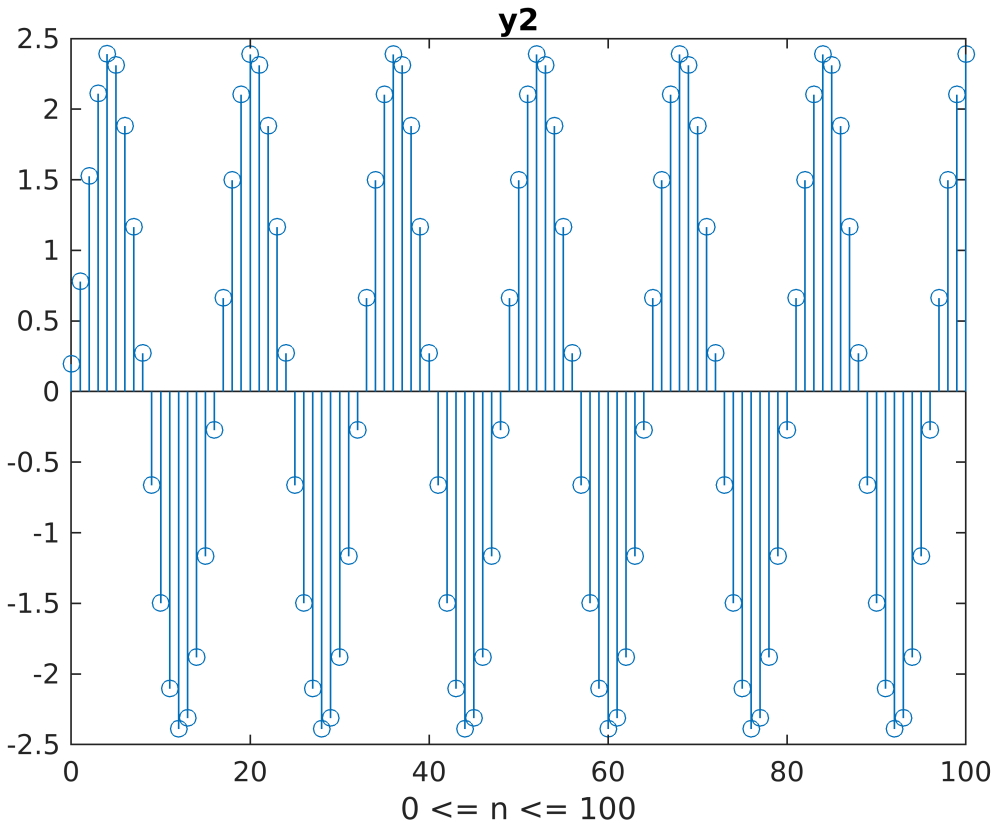
eigenfunction property of the system relates only to the steady-state solution. The signals and systems in this exercise have been chosen to insure the transients have died out completely within 20 samples, so by restricting the interval examined to be $0 \leq n \leq 100$, you are working with a portion of the output where the effect of the

natural response is insignificant.

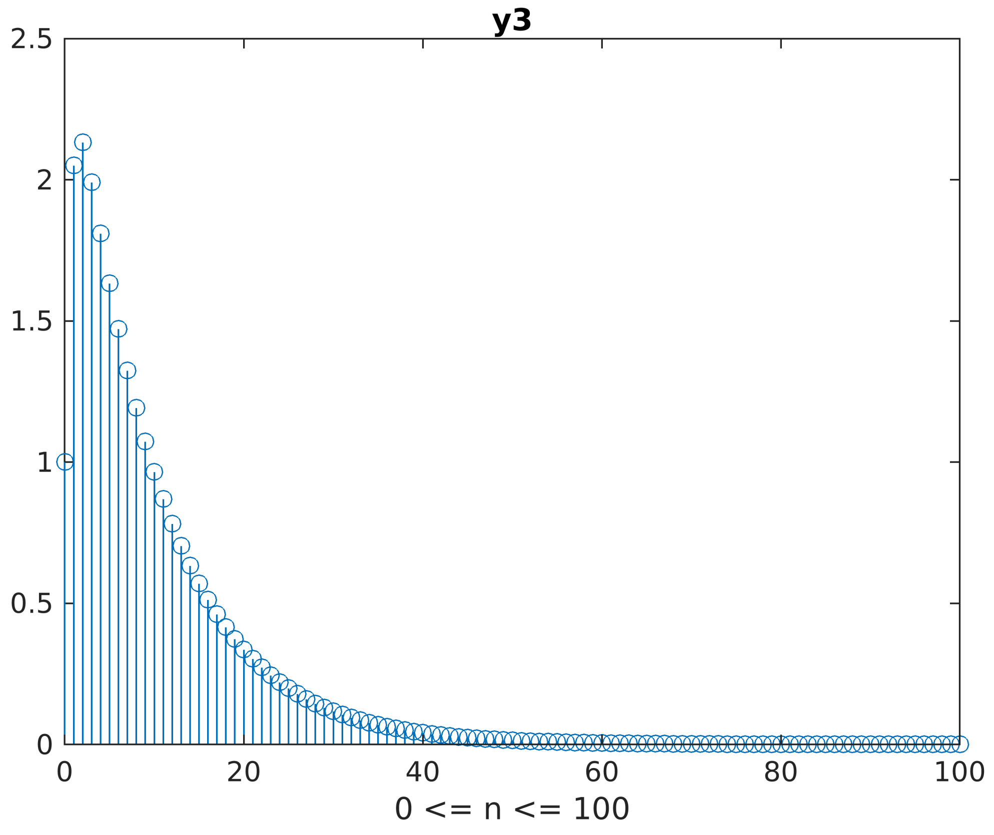
```
%3.4b
% REVISION
n = [0:100];

x1 = exp(1j*(pi/4)*n);
x2 = sin(pi*n/8 + pi/16);
x3 = (9/10).^n;
x4 = n + 1;

a = [1 -0.25];
b = [1 0.9];

y1 = filter(b, a, x1);
y2 = filter(b, a, x2);
y3 = filter(b, a, x3);
y4 = filter(b, a, x4);

figure;
subplot(2, 1, 1);
stem(n, real(y1)); title('y1[n]');
ylabel('Real(y1)');

subplot(2, 1, 2);
stem(n, imag(y1)); title('0 <= n <= 100');
ylabel('Imag(y1)');
```



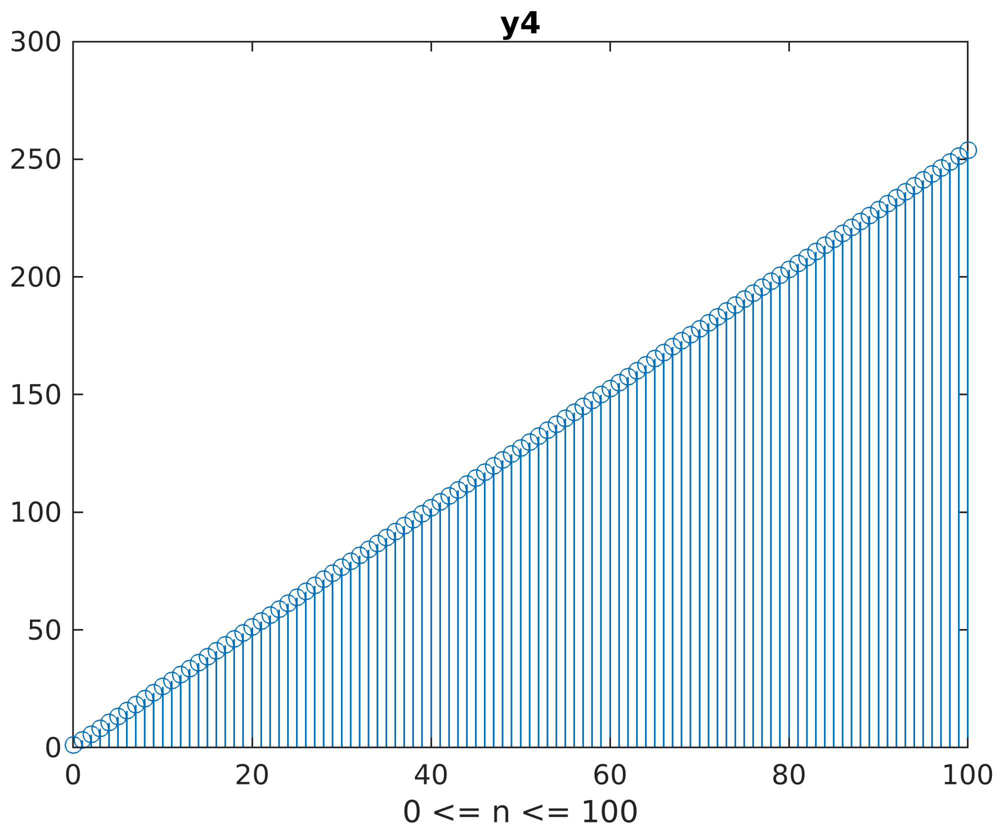
```
figure;
stem(n, y2);
title('y2');
xlabel('0 <= n <= 100');
```



```
figure;
stem(n, y3);
title('y3');
xlabel('0 <= n <= 100');
```



```
figure;
stem(n, y4);
title('y4');
xlabel('0 <= n <= 100');
```



(c). For this part, you will verify which of the inputs were eigenfunctions and compute the corresponding eigenvalues for those eigenfunctions. If the vectors **x** and **y** describe the

input and output sequences of a system, and the input sequence is an eigenfunction of the system, **y** should be equal to **x** scaled by a constant. This can be verified

by computing **H=y. /x,** which computes the ratio of the output to input sequence ateach time index. If the resulting vector **H** is a constant, the input signal was an

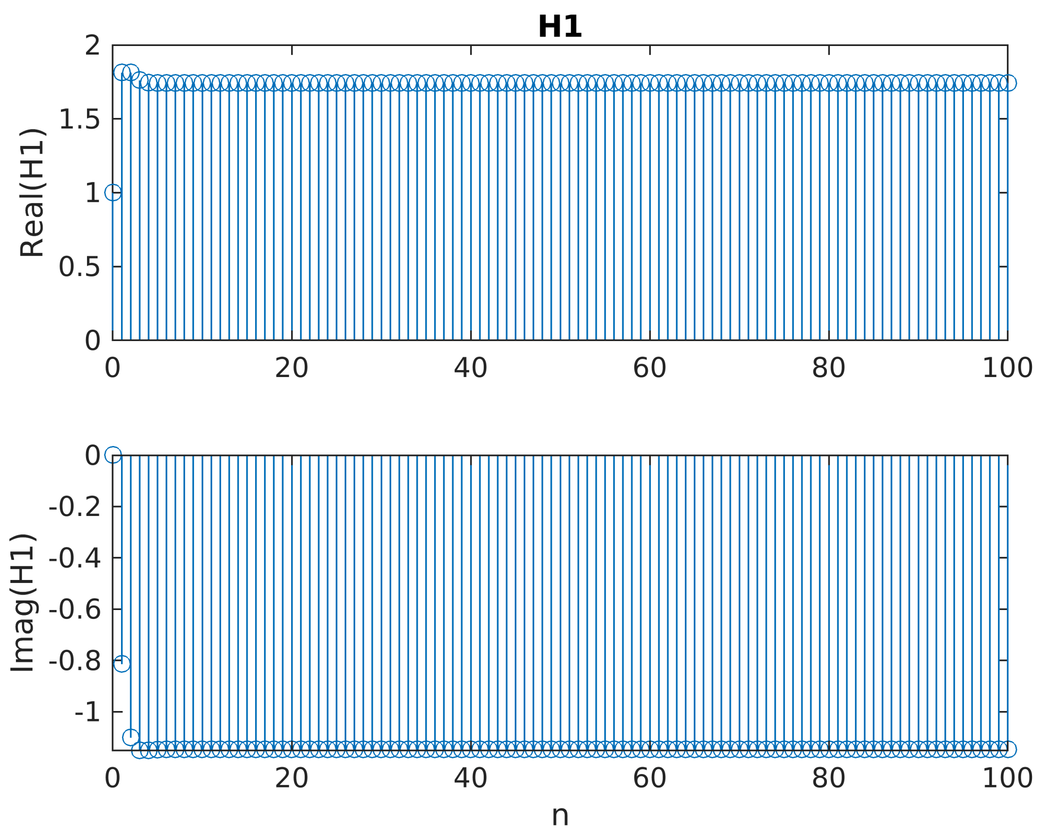
eigenfunction of the system. Compute H_1 through H_4 for each of the input/output signal pairs you obtained above, and produce appropriately labeled plots of **H** over

the interval $0 \le n \le 100$ again. Again, H_1 will require separate plots for its real and imaginary parts. For the inputs which are eigenfunctions of the system, find the

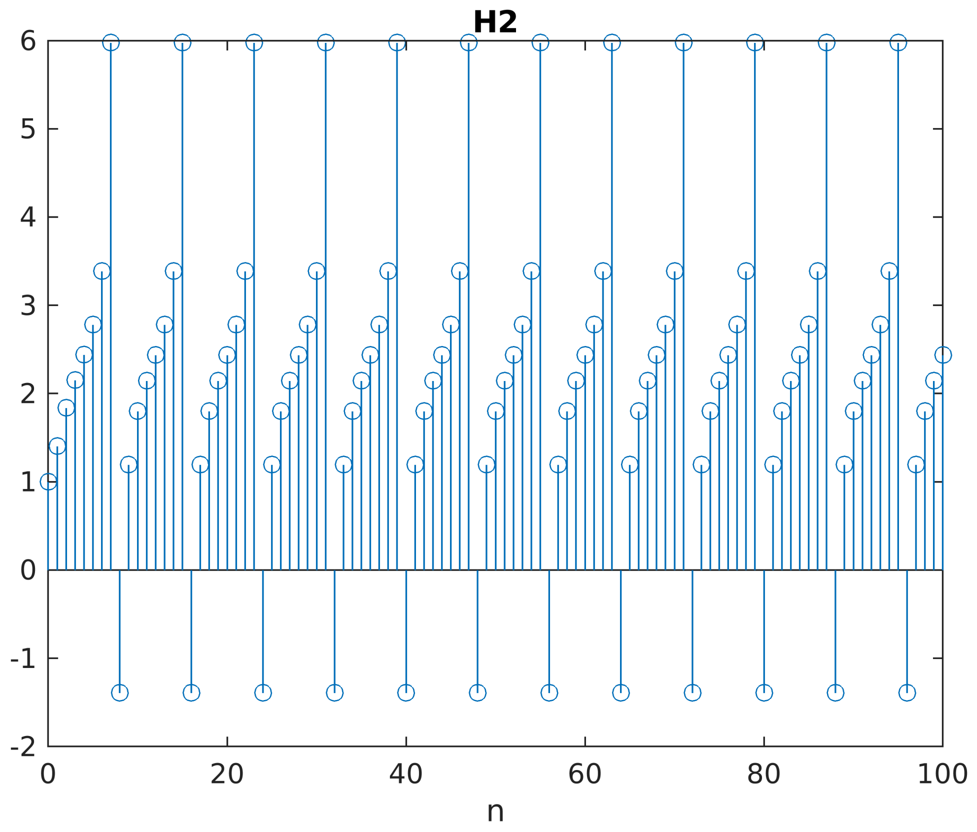
eigenvalue $H(z)$ from your plots or from the vector **H**.

```
%3.4c
% REVISION

H1 = y1./x1;
H2 = y2./x2;
H3 = y3./x3;
```

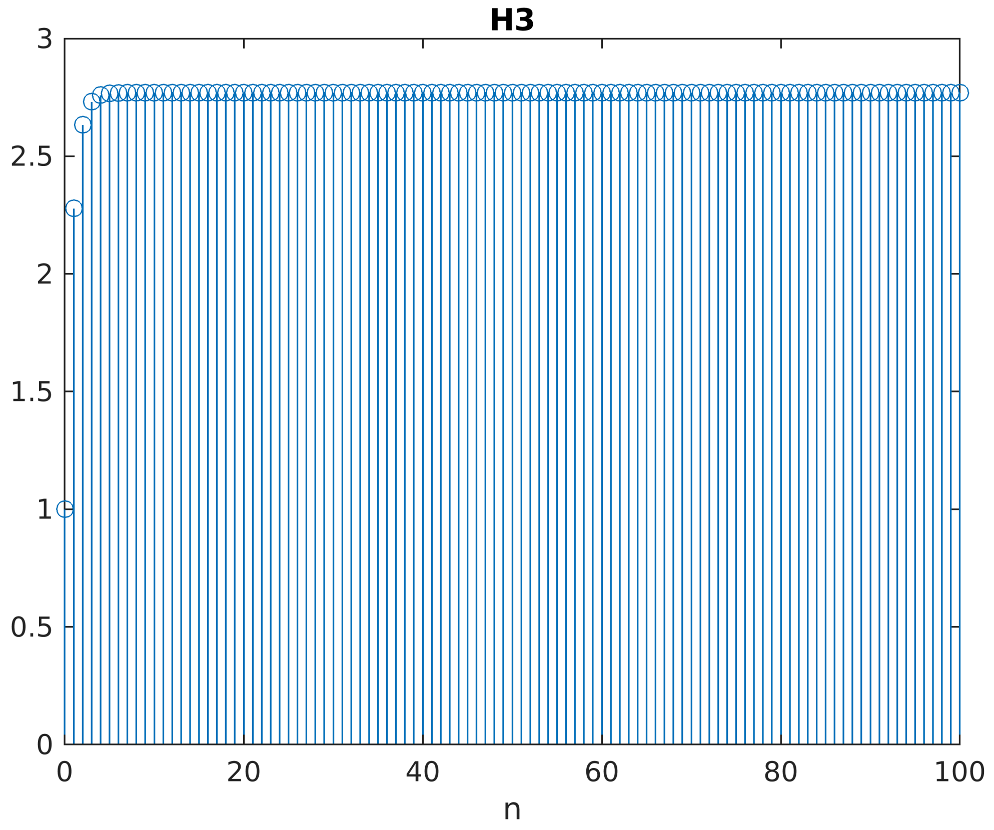
```
H4 = y4./x4;

figure;
subplot(2, 1, 1);
stem(n, real(H1));
title('H1');
ylabel('Real(H1)');

subplot(2, 1, 2);
stem(n, imag(H1));
xlabel('n');
ylabel('Imag(H1)');
```



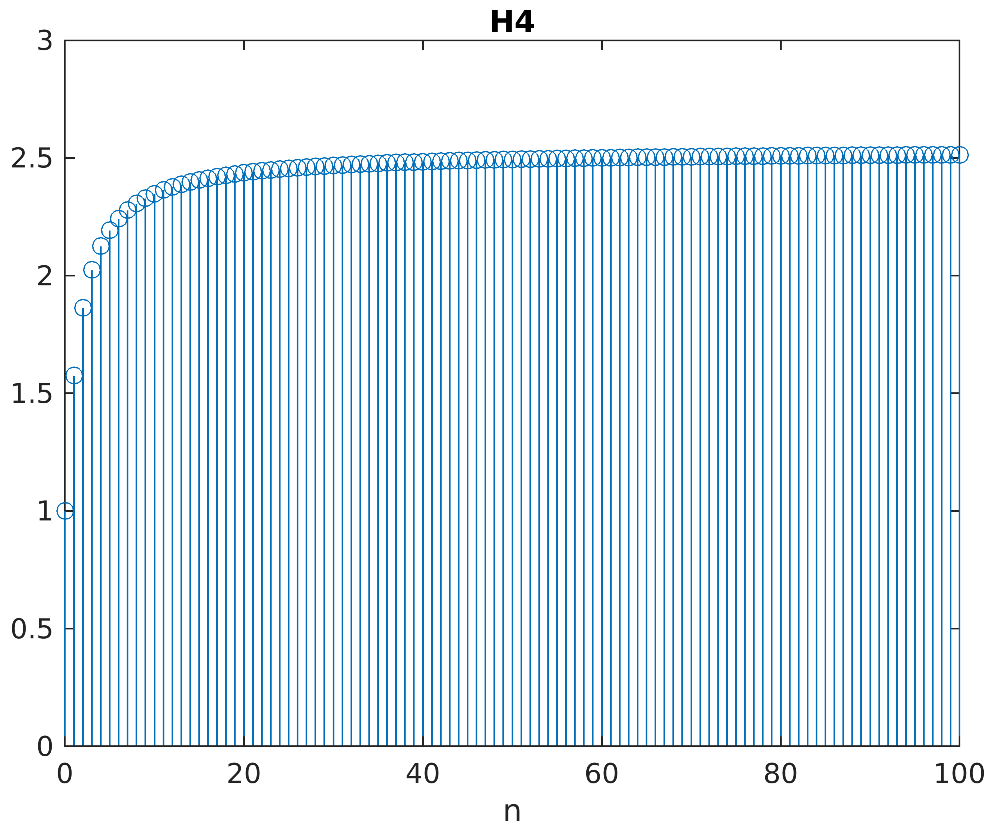
```
figure;
stem(n, H2);
title('H2');
xlabel('n');
```



```
figure;
stem(n, H3);
title('H3');
xlabel('n');
```



```
figure;
stem(n, H4);
title('H4');
xlabel('n');
```



Results and Analysis:

From the plot for H(z), we can find out that only H1 is a constant. So x1 is an eigen function of system, and thr eigenvalue of H(z) = H(e^{j\pi /4})

3.5 Synthesizing Signals with the Discrete-Time Fourier Series

The discrete-time Fourier series (DTFS) is a frequency-domain representation for periodic discrete-time sequences. The synthesis and analysis equations for the DTFS are given by

Eqs. (3.1) and (3.2). This exercise contains three sets of problems to give you practice working with these equations. The Basic Problems allow you to synthesize a very simple

periodic discrete-time signal from its DTFS coefficients. In the Intermediate Problems, you will both analyze a set of periodic discrete-time signals to obtain their DTFS coefficients,

and construct one of these signals by adding in a few coefficients at a time. For the Advanced Problem, you will write a function to find the DTFS coefficients of an arbitrary periodic

discrete-time signal from one period of samples.

Basic Problems

In these problems, you will synthesize a periodic discrete-time signal with period $N = 5$ and the following DTFS coefficients

$a_0 = 1,\ a_2 = a^*_{-2} = e^{j\pi/4},\ a_4 = a^*_{-4} = 2e^{j\pi/3}$

(a). Based on the DTFS coefficients, do you expect x[n] to be complex-valued, purely real, or purely imaginary? Why?

3.5a Analysis

x[n] = will be a complex-valued number since the answer will be about

$x[n] = 2 + 2sin(\frac{\pi}{6}n + \frac{\pi}{6}) + 2sin(\frac{\pi}{6}n + \frac{\pi}{6}) + sin(\frac{\pi}{6}n + \frac{\pi}{3}) + sin(\frac{\pi}{6}n + \frac{\pi}{3})$. Since it contains π, the value will be a complex-value.

==

REVISION

Given N=5 and $\omega = 2\pi/N = 2\pi/5$, using the synthese equation (3/1), $x[n] = \frac{1}{N}\sum_{k=0}^{N-1} a_k e^{jk(2\pi/N)n}$ and main textbook Eq(1.27), we have:

$x[n] = a_0 + a_2 e^{j(4\pi/5)n} + a_{-2}e^{-j(4\pi/5)n} + a_4 e^{j(8\pi/5)n} + a_{-4}e^{-j(8\pi/5)n}$

$x[n] = 1 + e^{j(4\pi/5)n} + e^{-j\pi/4}e^{-j(4\pi/5)n} + 2e^{j\pi/3}e^{j(8\pi/5)n} + 2e^{-j\pi/3}e^{-j(8\pi/5)n}$

$x[n] = 1 + e^{j(4n\pi/5+\pi/4)} + e^{-j(4n\pi/5+\pi/4)} + 2e^{j(8n\pi/5+\pi/3)} + 2e^{-j(8n\pi/5+\pi/3}$

$x[n] = 1 + 2cos(4n\pi/5 + \pi/4) + 4cos(8n\pi/5 + \pi/3)$

This expects to be purley real.

==

(b). Using the DTFS coefficients given above, determine the values of **a0** through **a4** and specify a vector **a** containing these values.

==

Revision

In a discrete-time Fourier series, the coefficients (ak) are coresponses of the harmonics at certain frequences. When we represent them in a vector, it needs to follow the sequency (or order) in accordancy to the frequence domain.

$a_k = (a_0, a_1, a_2, a_3, a_4)$

We have the vector as

$a = [1, 0, e^{j\pi/4}, 0, 2e^{j(\pi/3)}]$

==

(c). Using the vector **a** of DTFS coefficients and the synthesis equation, define a new vector x containing one period of the signal $x[n]$ for $0 \leq n \leq 4$. You can either write

out the summation explicitly or you may find it helpful to use a **for** loop. Generate an appropriately labeled plot of $x[n]$ for $0 \leq n \leq 4$ using **stem**. Was your prediction

in Part (a) correct? Note that if you predicted a purely imaginary or real signal, it may still have a very small $(< 10^{-10})$ nonzero real or imaginary part due to roundoff

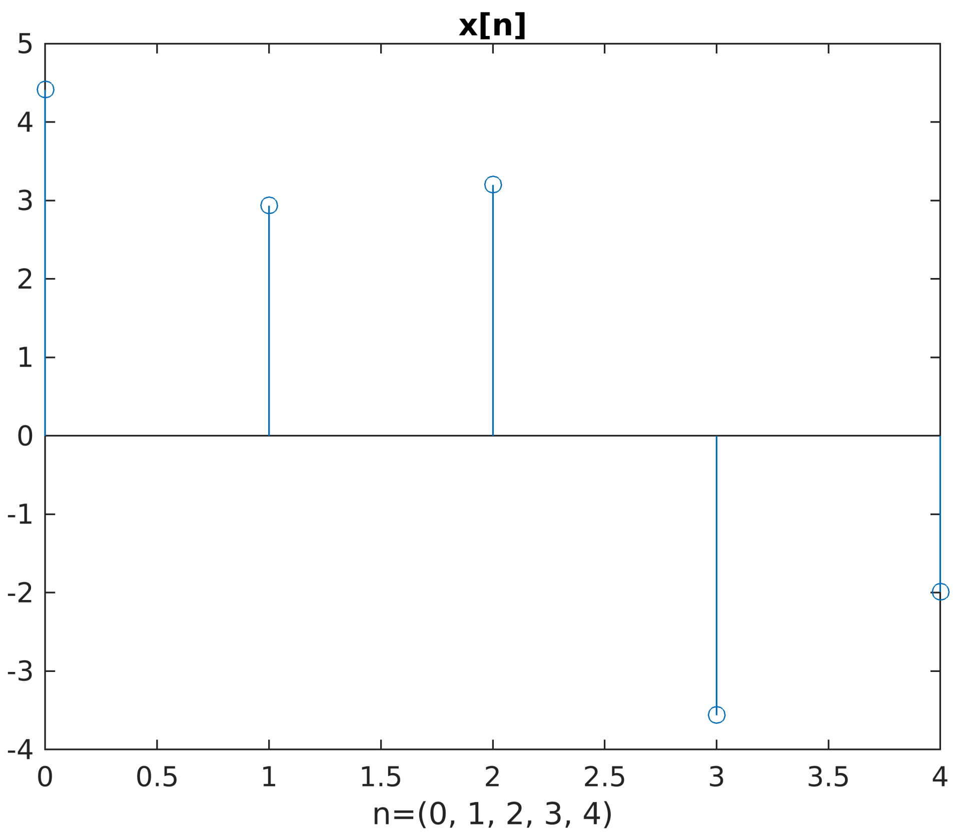
errors. If this is the case, set this part to be zero using **real** or **imag** as appropriate before making your plot.

==

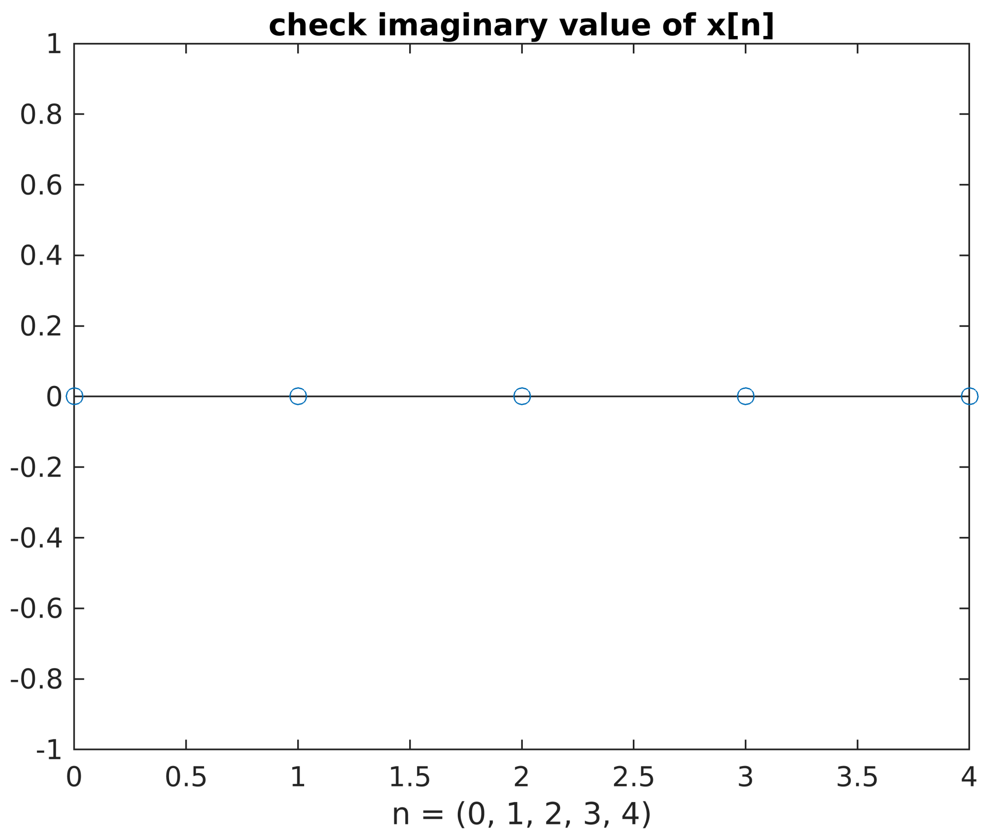
Using the synthesis equation (3/1) and the Fourier series coefficients above, we can express x[n] as

$$x[n] = \sum_{k=0}^{4} a_k e^{-jk(2\pi/N)n}$$

```
%3.5c

clf;

n = [0:4];
x = n*0;
x = 1 + 2*cos(4*pi.*n/5+pi/4) + 4*cos(8*pi.*n/5+pi/3);

figure;
stem(n, x);
title('x[n]');
xlabel('n=(0, 1, 2, 3, 4)');
```



```
stem(n, imag(x));
title('check imaginary value of x[n]');
xlabel('n = (0, 1, 2, 3, 4)');
```



==

3.6 Properties of the Continuous-Time Fourier Series

This exercise examines properties of the continuous-time Fourier series (CTFS) representation for periodic continuous-time signals.

Consider the signal

$$x_1(t) = cos(\omega_0 t) + sin(2\omega_0 t), \qquad (3.7)$$

where $\omega_0 = 2\pi$. To evaluate this signal in MATLAB, use the time vector

```
>> t=linspace(-1,1,1000);
```

which creates a vector of 1000 time samples over the region $-1 \le t \le 1$.

Intermediate Problems

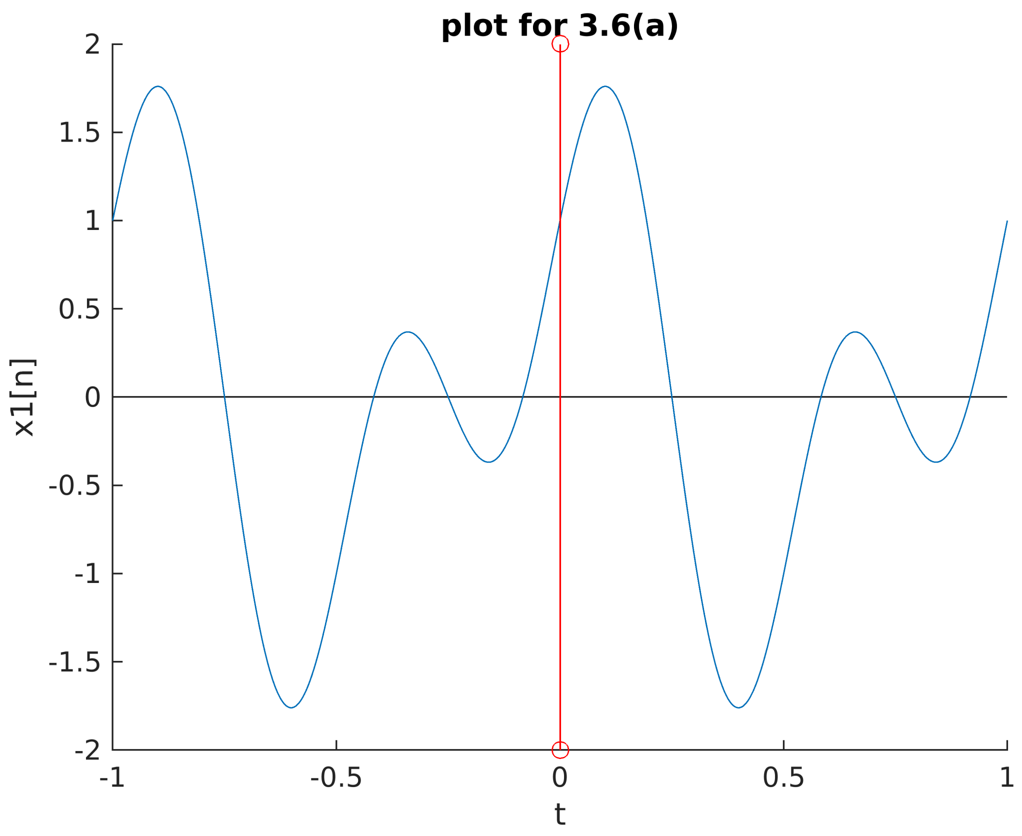
(a). What is the smallest period, **T**, for which x $x_1(t) = x_1(t+T)$? Analytically find the coefficients of the CTFS for $x_1(t)$ using this value of **T**.

3.6a Analysis

```
% Revision:

clf;

t = linspace(-1, 1, 1000);
x1 = cos(2*pi.*t) + sin(4*pi.*t);

figure;
hold;
```

```
Current plot held
```

```

plot(t, x1);
xlabel('t');
ylabel('x1[n]');
title('plot for 3.6(a)');
stem(0, 2, 'r');
stem(0, -2, 'r');
hold off;
```



Given $\omega = 2\pi/T$ and $\omega_0 = 2\pi$, we have T=1. The plot above also showws that T=1. Using the analysis equation (main textbook 3.39), we have:

$$a_k = \frac{1}{T}\int_0^1 cos(2\pi t) + sin(4\pi t)e^{j2\pi kt}dt$$

where $\omega_0 = 2\pi$ and T=1.

(b). Consider the signal $y(t) = x_1(t) + x_1(-t)$. Using the time-reversal and conjugation properties of the CTFS, determine the coefficients for the CTFS of $y(t)$.

3.6b Analysis

Revision:

By the timne-reversal property, if $x_1(t) < - > a_k$ then $x_1(-t) < - > b_k = a_{-k}$.

By conjugation property, we know that Real(a_k) = Real(a_{-k}) and Imag(a_k) = -Imag(a_{-k}).

We also know y(t) has the same period, T=1/ SO it only has two non-zero coeffients, a_{-1} and a_1 where $a_0 = 0$.

By the linearity property, if $y(t) = x_1(t) + x_1(-t)$, we have $y(t) < - > c_k = a_k + a_{-k}$.

Therefore we can derive that Real(c_k) = Real(a_k) + Real(a_{-k}) = 2Real(a_k) and Imag(a_k) + Imag(a_{-k}) = 0.

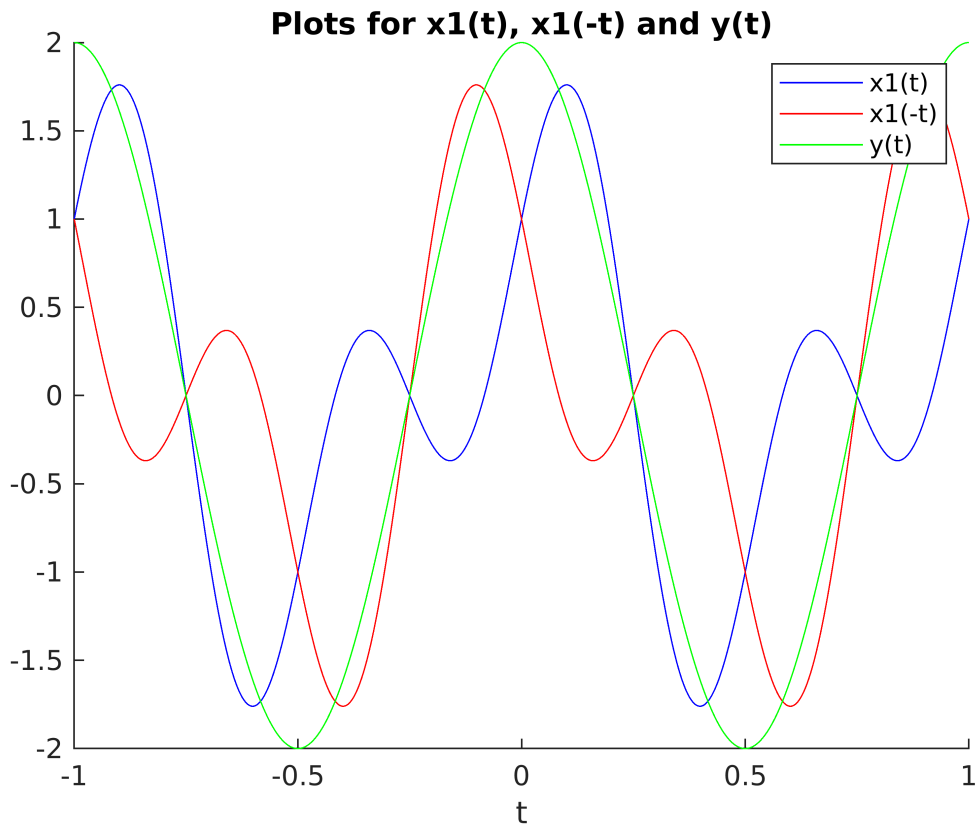
(c). Plot the signal $y(t)$ over $-1 \leq t \leq 1$. What type of symmetry do you expect? Can you explain what you see in terms of the symmetry properties of the CTFS?

```
%3.6c
%REVISION

clf;

t = linspace(-1, 1, 1000);
x1 = cos(2*pi.*t) + sin(4*pi.*t);
x2 = cos(2*pi.*(-t)) + sin(4*pi.*(-t));
y = x1 + x2;

figure;
hold;
```

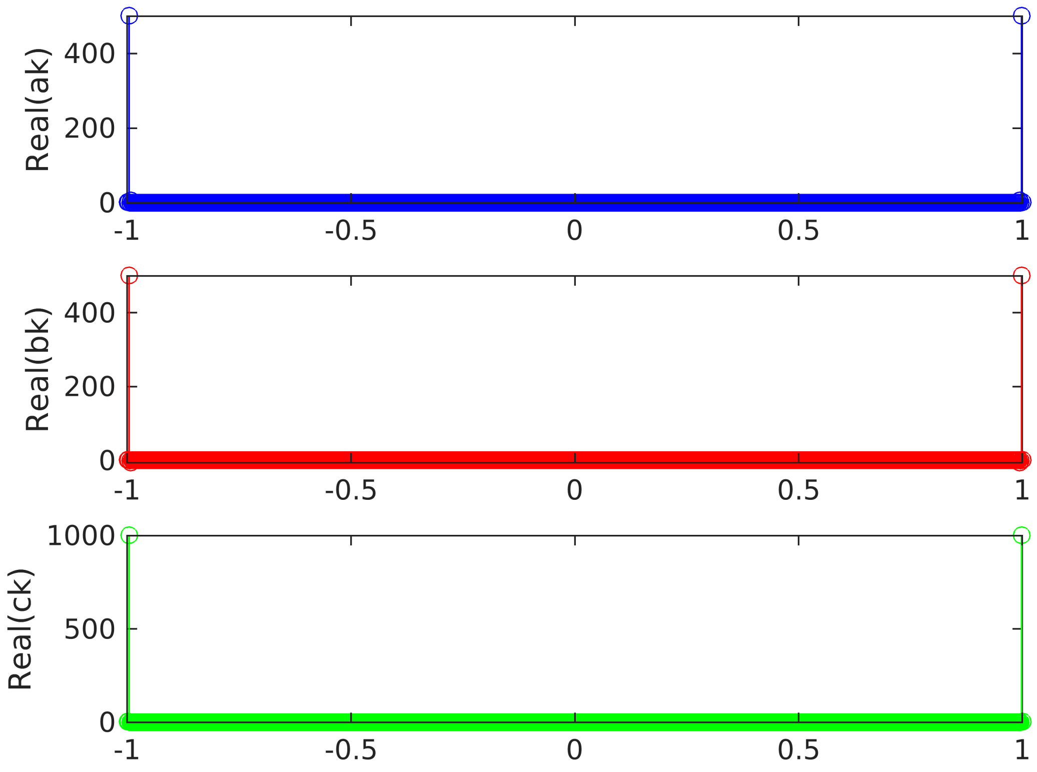
```
Current plot held
```

```

plot(t, x1, 'b');
plot(t, x2, 'r');
plot(t, y, 'g');
legend('x1(t)', 'x1(-t)', 'y(t)');
xlabel('t');
```

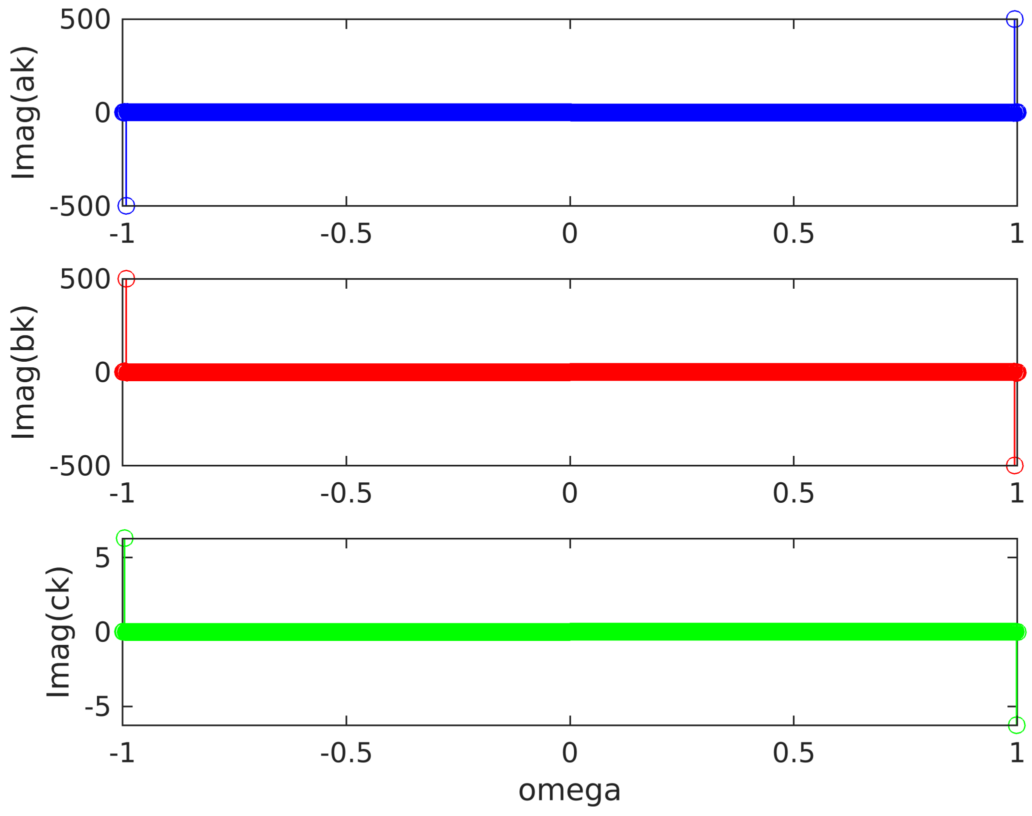
```
title('Plots for x1(t), x1(-t) and y(t)');
```



```
N = 1;
a = fft(x1)/N;
b = fft(x2)/N;
c = fft(y)/N;

figure;

subplot(3, 1, 1);
stem(t, real(a), 'b');
ylabel('Real(ak)');

subplot(3, 1, 2);
stem(t, real(b), 'r');
ylabel('Real(bk)');

subplot(3, 1, 3);
stem(t, real(c), 'g');
ylabel('Real(ck)');
```



```
figure;

subplot(3, 1, 1);
stem(t, imag(a), 'b');
ylabel('Imag(ak)');

subplot(3, 1, 2);
stem(t, imag(b), 'r');
ylabel('Imag(bk)');

subplot(3, 1, 3);
stem(t, imag(c), 'g');
ylabel('Imag(ck)');
xlabel('omega');
```



Analysis:

The plot of signal y(t) shows that it is an even symmetry. The plot of ck shows the Fourier series coefficients are real and even with the imaginary part being zero.

(d). Consider the signal $z(t) = x_1(t) - x_1^*(-t)$. Using the time-reversal and conjugation properties of the CTFS, determine the coefficients for the CTFS of $z(t)$.

3.6d Analysis

Revision

As stated in the conjugation and time reversal Fourier series properties (Eq 3.65, main textbook), if $x_1(t) <-> a_k$ then $x_1(t)* <-> a_{-k}^*$

and by (Eq 3.66, main textbook)

$x_1^*(-t) <-> b_k = a_k^* = a_{-k}$

Therefore z(t) is real and even, $a_k = a_{-k}$ therefore $c_k = 0$.

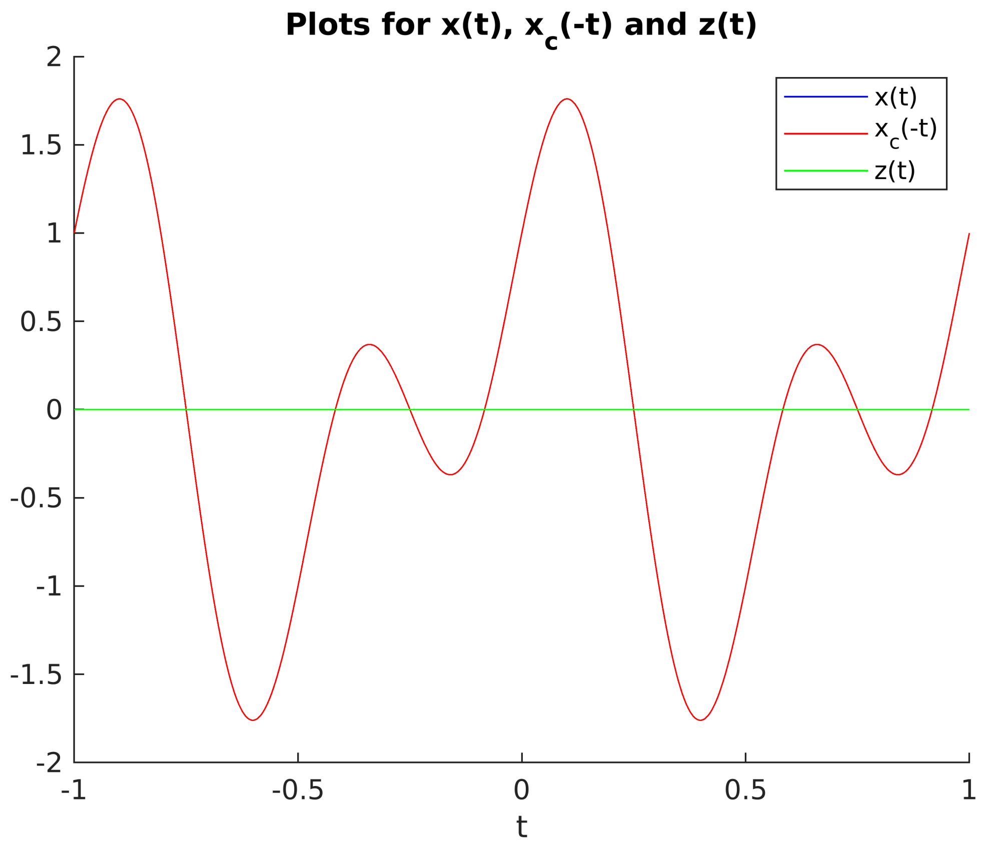
The plots below confirm the results we got from mathematical analysis.

(e). Plot the signal $z(t)$ over $-1 \le t \le 1$. What type of symmetry do you expect to see? Can you explain what you see in terms of the symmetry properties of the CTFS?

```
%3.6e
% Revision
clf;

t = linspace(-1, 1, 1000);
x = cos(2*pi.*t) + sin(4*pi.*t);
x_c = cos(2*pi.*(-t)) + sin(-4*pi.*(-t));
z = x - x_c;

figure;
hold;
```

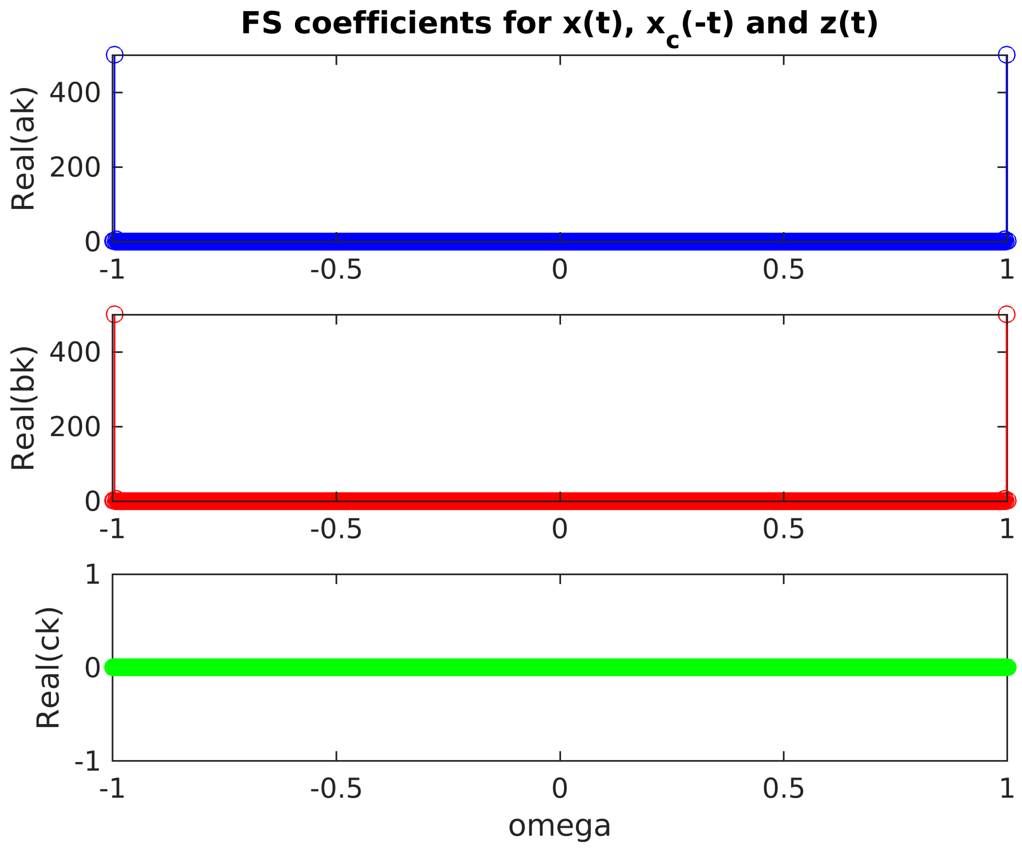
```
Current plot held
```

```

plot(t, x, 'b');
plot(t, x_c, 'r');
plot(t, z, 'g');
legend('x(t)', 'x_c(-t)', 'z(t)');
xlabel('t');
title('Plots for x(t), x_c(-t) and z(t)');
```



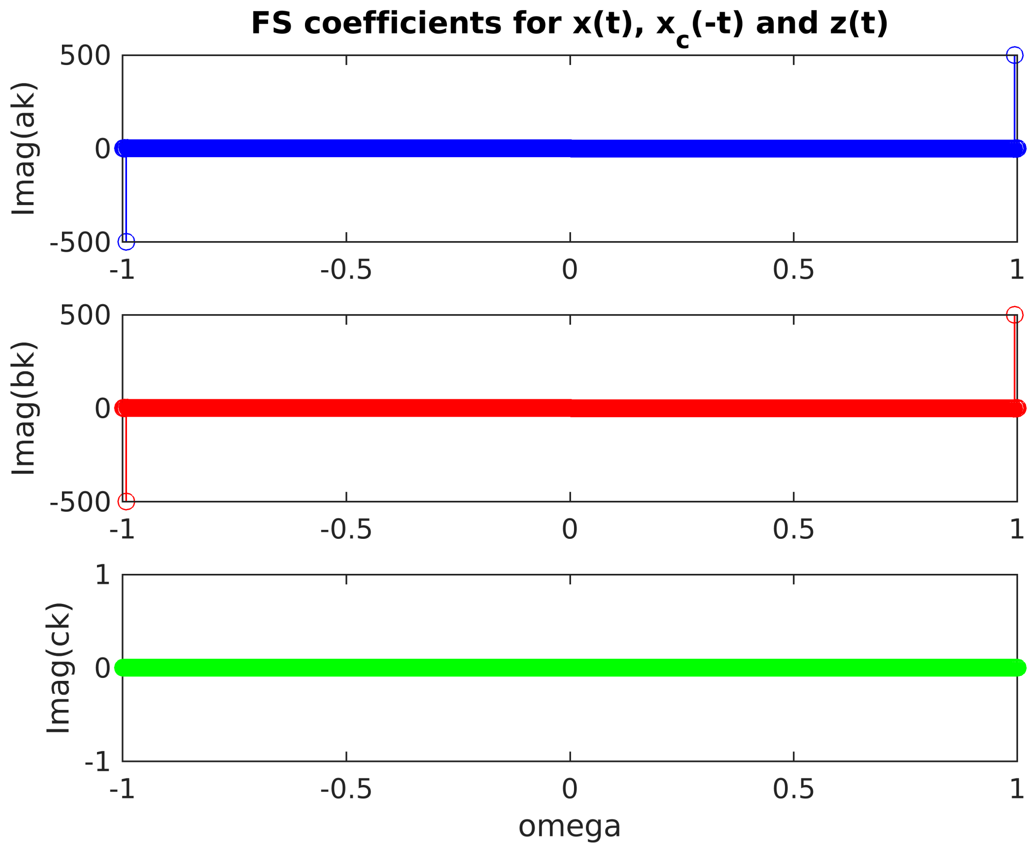
```

N = 1;
```

```
a = fft(x)/N;
b = fft(x_c)/N;
c = fft(z)/N;

figure;

subplot(3, 1, 1);
stem(t, real(a), 'b');
ylabel('Real(ak)');
title('FS coefficients for x(t), x_c(-t) and z(t)');

subplot(3, 1, 2);
stem(t, real(b), 'r');
ylabel('Real(bk)');

subplot(3, 1, 3);
stem(t, real(c), 'g');
ylabel('Real(ck)');
xlabel('omega');
```



```
figure;
subplot(3, 1, 1);
stem(t, imag(a), 'b');
ylabel('Imag(ak)');
title('FS coefficients for x(t), x_c(-t) and z(t)');

subplot(3, 1, 2);
```

```
stem(t, imag(b), 'r');
ylabel('Imag(bk)');

subplot(3, 1, 3);
stem(t, imag(c), 'g');
ylabel('Imag(ck)');
xlabel('omega');
```



(f). Repeat Parts (a)-(e) using

$$x_2(t) = cos(\omega_0 t) + isin(2\omega_0 t) \tag{3.8}$$

Note that $x_2(t)$ is complex. When plotting $x_2(t)$, $y(t)$, and $z(t)$, be sure to plot the real and imaginary parts separately and take note of the symmetry that you see in

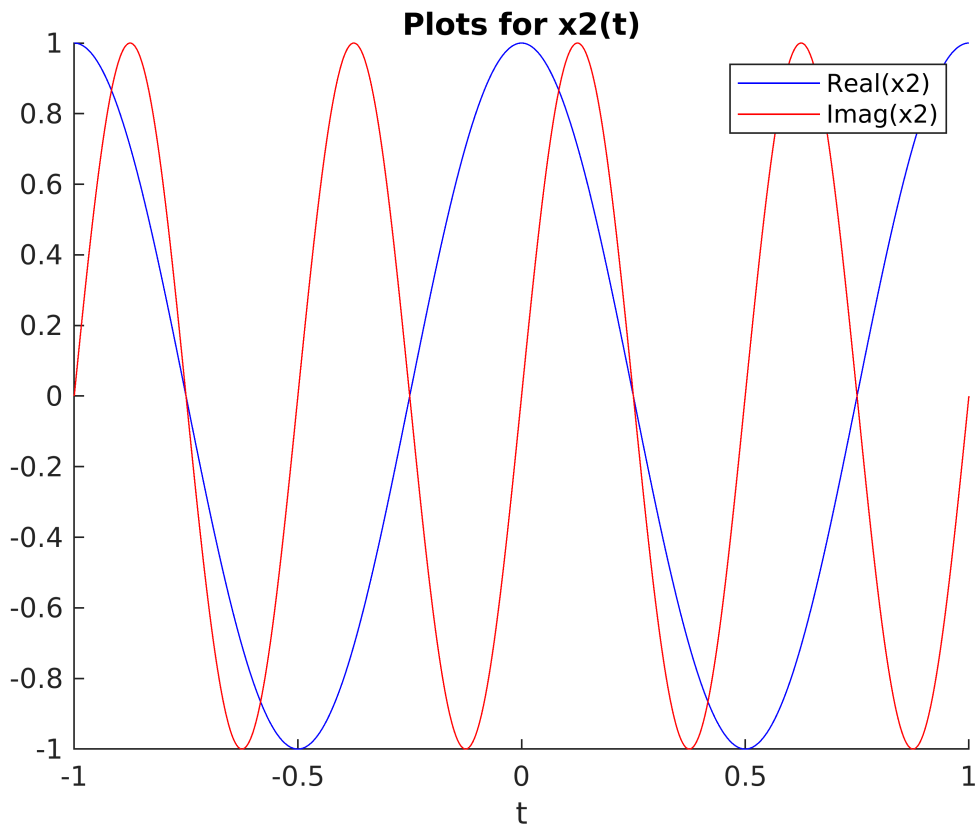
each.

```
%3.6f
%Revision:

clf;

t = linspace(-1, 1, 1000);
```

```
x2 = cos(2*pi.*t) + 1j*sin(4*pi.*t);
x2_t = cos(2*pi.*(-t) + 1j*sin(4*pi.*(-t)));
x2_c = cos(2*pi.*(-t) - 1j*sin(4*pi.*(-t)));

figure;
hold;
```

```
Current plot held
```

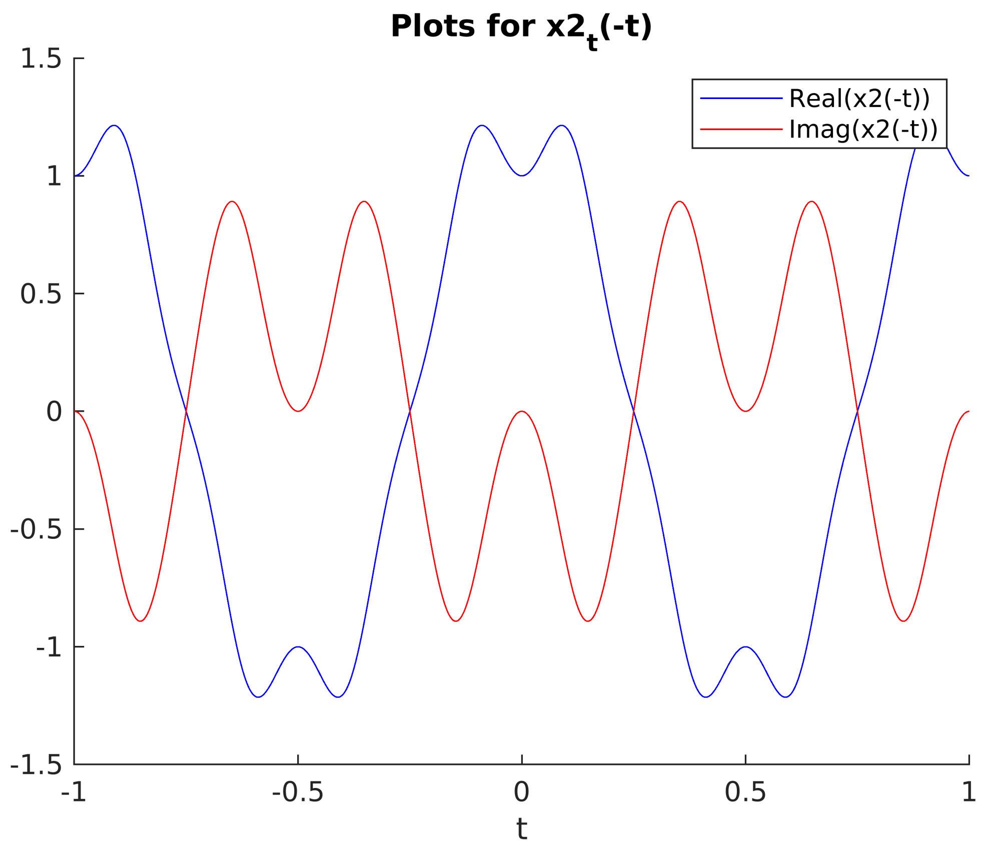
```
plot(t, real(x2), 'b');
plot(t, imag(x2), 'r');
legend('Real(x2)', 'Imag(x2)');
xlabel('t');
title('Plots for x2(t)');
```



```
figure;
hold;
```

```
Current plot held
```

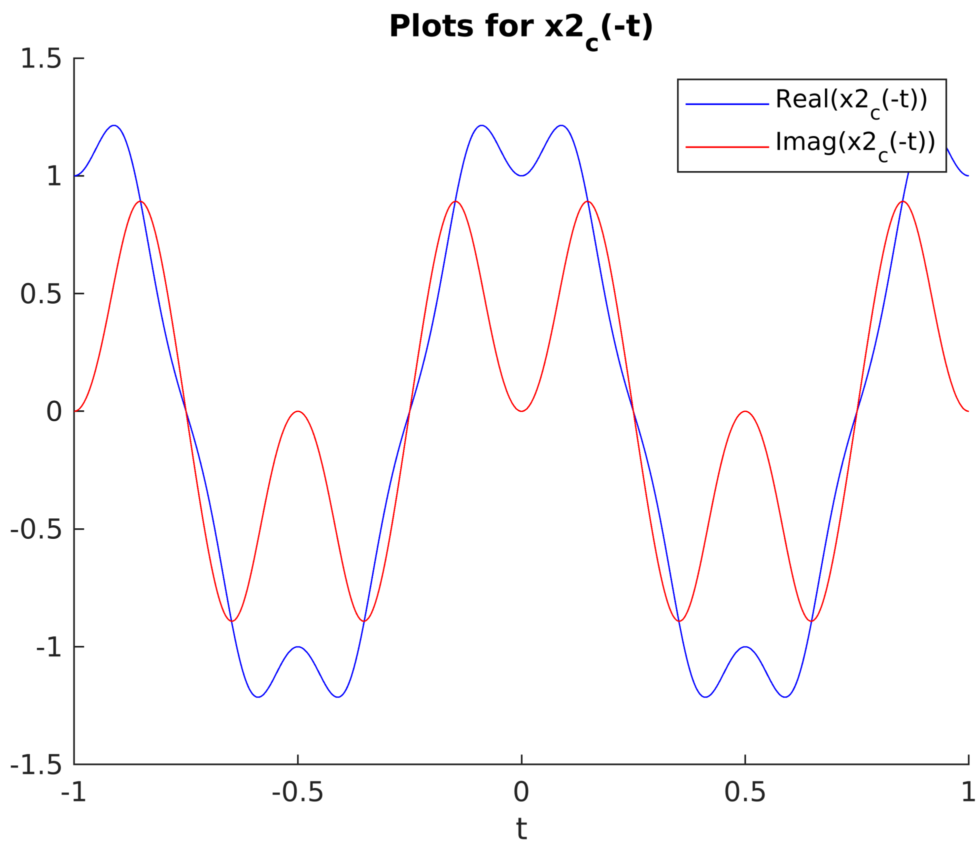
```
plot(t, real(x2_t), 'b');
plot(t, imag(x2_t), 'r');
legend('Real(x2(-t))', 'Imag(x2(-t))');
xlabel('t');
title('Plots for x2_t(-t)');
```



```
figure;
hold;
```

```
Current plot held
```

```
plot(t, real(x2_c), 'b');
plot(t, imag(x2_c), 'r');
legend('Real(x2_c(-t))', 'Imag(x2_c(-t))');
xlabel('t');
title('Plots for x2_c(-t)');
```

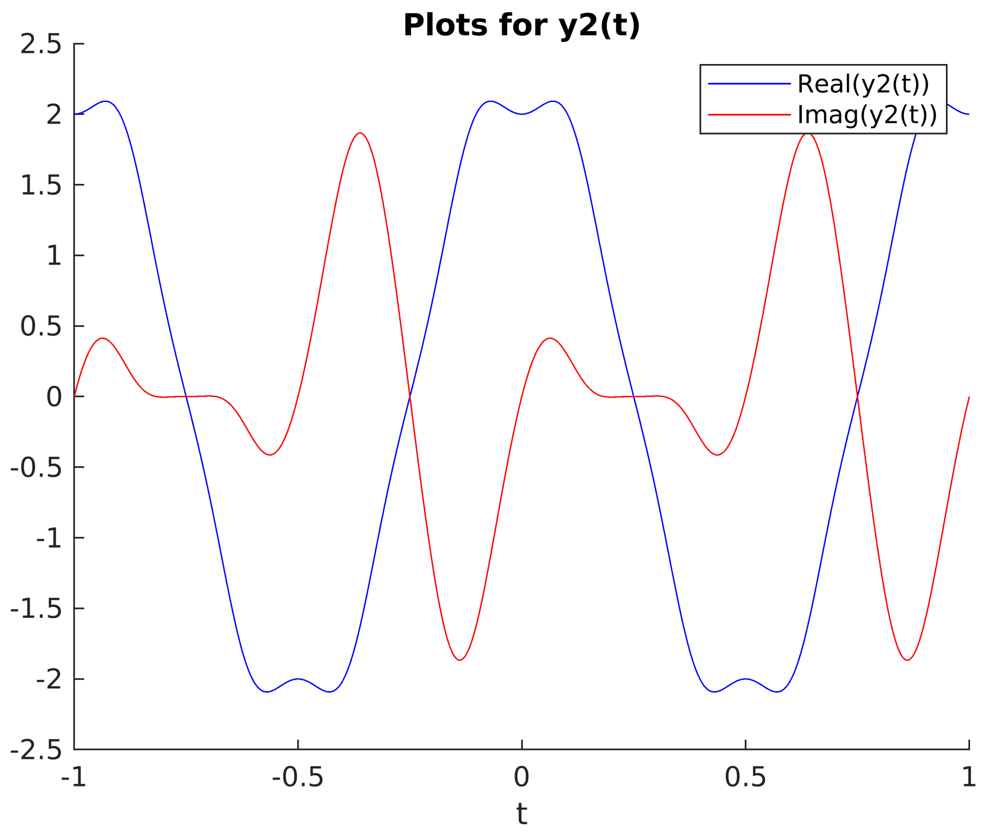


The $x_2^*(-t)$ plot shows the Real part is an even symmetry and the Imag partt is an odd symmetry. The plot shows $x_2^*(-t) = x_2(t)$

```
y2 = x2 + x2_t;

figure;
hold;
```

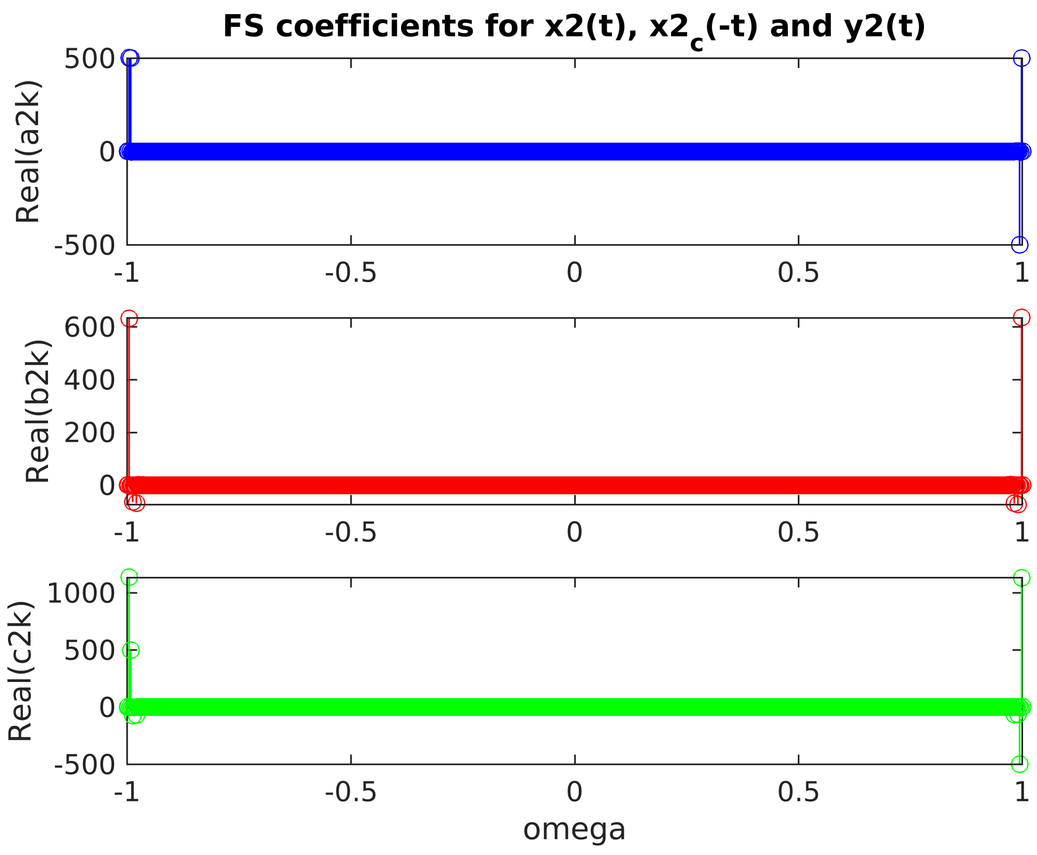
```
Current plot held
```

```

plot(t, real(y2), 'b');
plot(t, imag(y2), 'r');
legend('Real(y2(t))', 'Imag(y2(t))');
xlabel('t');
title('Plots for y2(t)');
```

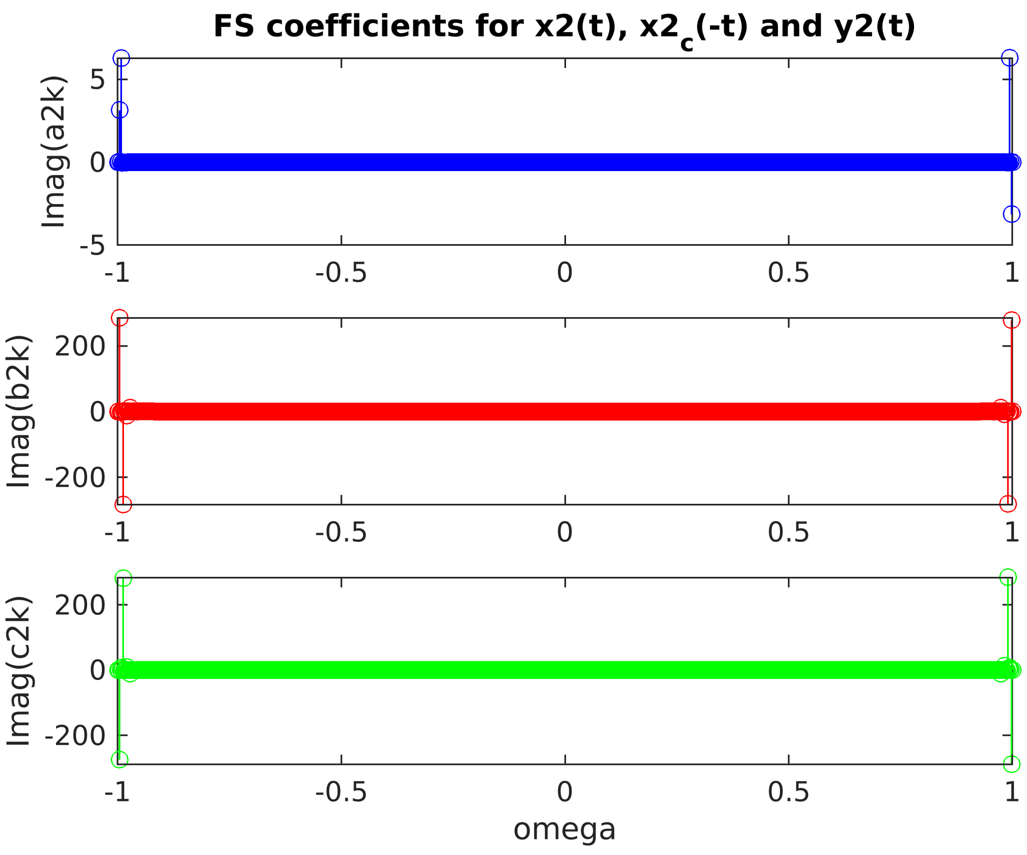


The y2(t) plot shows the Real part is an even symmetry and the Imag part is an odd symmetry.

```
N = 1;
a2 = fft(x2)/N;
b2 = fft(x2_c)/N;
c2 = fft(y2)/N;

figure;

subplot(3, 1, 1);
stem(t, real(a2), 'b');
ylabel('Real(a2k)');
title('FS coefficients for x2(t), x2_c(-t) and y2(t)');

subplot(3, 1, 2);
stem(t, real(b2), 'r');
ylabel('Real(b2k)');

subplot(3, 1, 3);
stem(t, real(c2), 'g');
ylabel('Real(c2k)');
xlabel('omega');
```



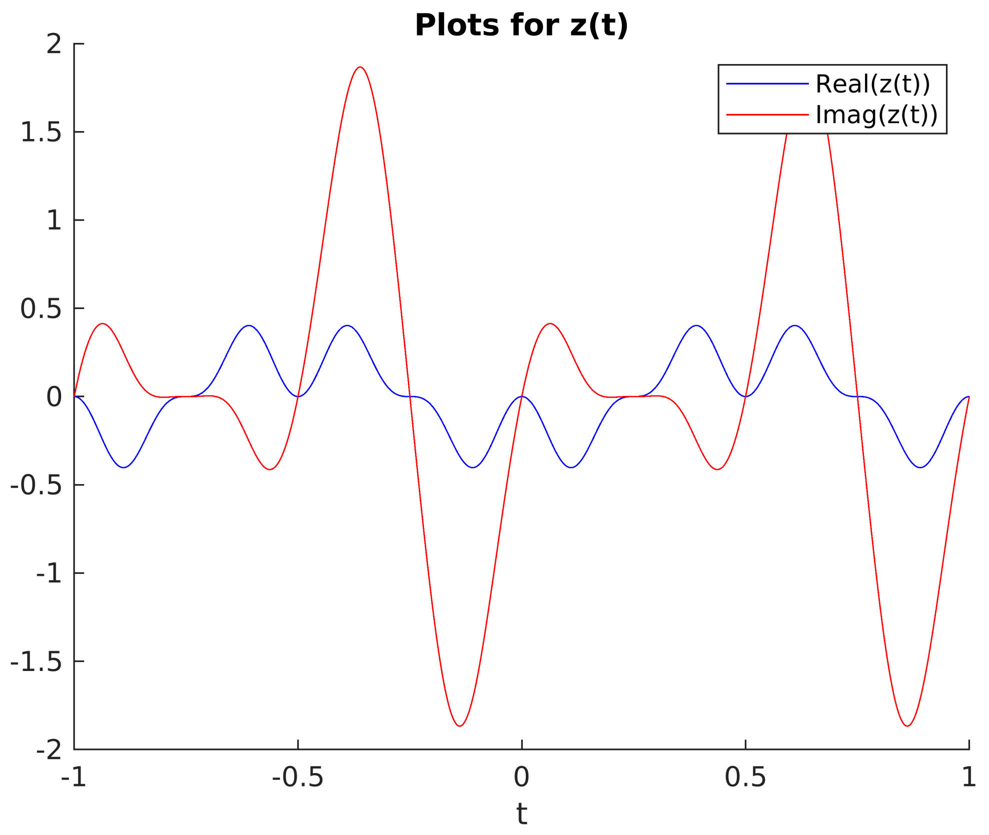
```
figure;

subplot(3, 1, 1);
stem(t, imag(a2), 'b');
ylabel('Imag(a2k)');
title('FS coefficients for x2(t), x2_c(-t) and y2(t)');

subplot(3, 1, 2);
stem(t, imag(b2), 'r');
ylabel('Imag(b2k)');

subplot(3, 1, 3);
stem(t, imag(c2), 'g');
ylabel('Imag(c2k)');
xlabel('omega');
```



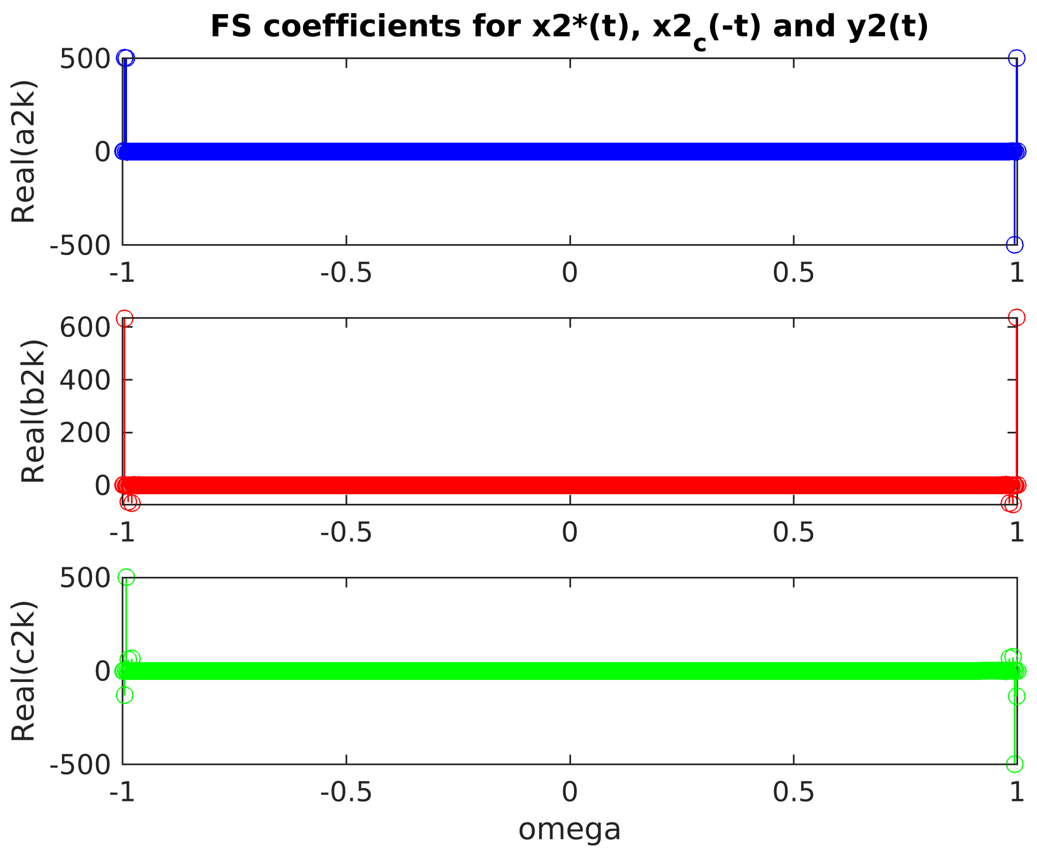
```
z2 = x2 - x2_c;
figure;
hold;
```

```
Current plot held
```

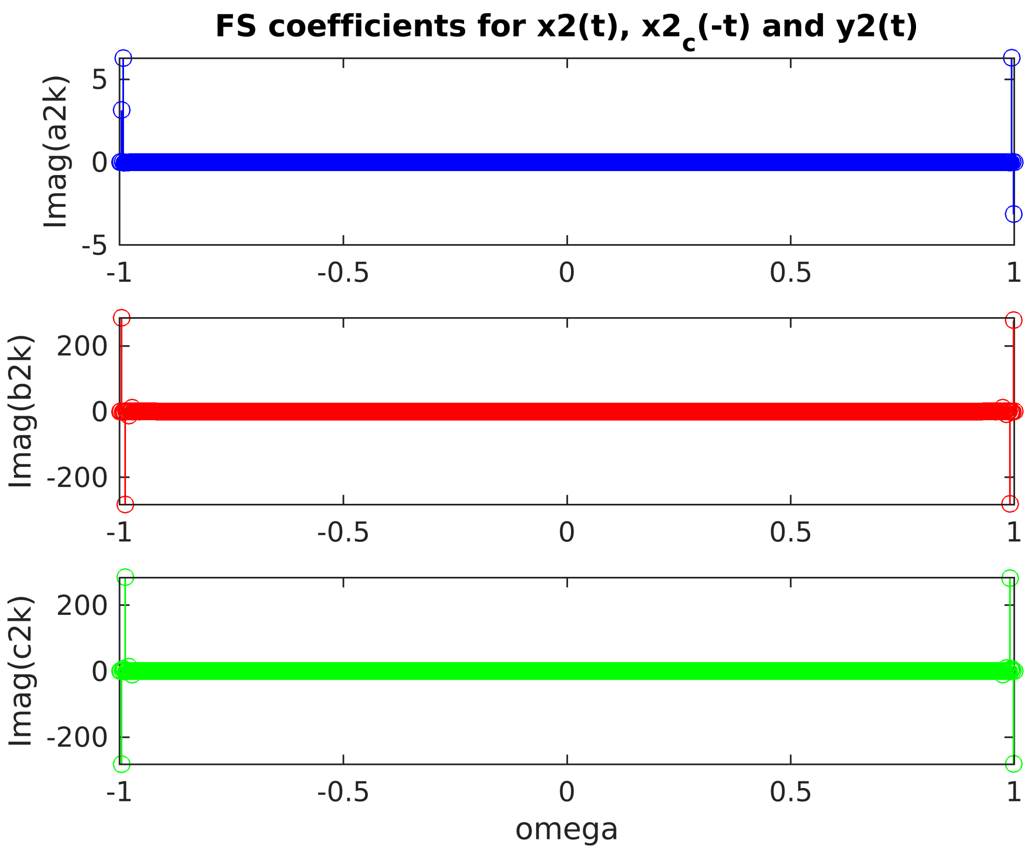
```
plot(t, real(z2), 'b');
plot(t, imag(z2), 'r');
xlabel('t');
title('Plots for z(t)');
legend('Real(z(t))', 'Imag(z(t))');
```



```
a2 = fft(x2);
b2 = fft(x2_c);
c2 = fft(z2);

figure;

subplot(3, 1, 1);
stem(t, real(a2), 'b');
ylabel('Real(a2k)');
title('FS coefficients for x2*(t), x2_c(-t) and y2(t)');

subplot(3, 1, 2);
stem(t, real(b2), 'r');
ylabel('Real(b2k)');

subplot(3, 1, 3);
stem(t, real(c2), 'g');
ylabel('Real(c2k)');
xlabel('omega');
```



```
figure;

subplot(3, 1, 1);
stem(t, imag(a2), 'b');
ylabel('Imag(a2k)');
title('FS coefficients for x2(t), x2_c(-t) and y2(t)');

subplot(3, 1, 2);
stem(t, imag(b2), 'r');
ylabel('Imag(b2k)');

subplot(3, 1, 3);
stem(t, imag(c2), 'g');
ylabel('Imag(c2k)');
xlabel('omega');
```


FS coefficients for x2(t), x2$_c$(-t) and y2(t)
Imag(a2k)
5
0
-5
-1
-0.5
0
0.5
1
Imag(b2k)
200
0
-200
Imag(c2k)
200
0
-200
omega