Name: SOLUTION (Havlicek)

Section:

Laboratory Exercise 3

DISCRETE-TIME SIGNALS: FREQUENCY-DOMAIN REPRESENTATIONS

3.1 DISCRETE-TIME FOURIER TRANSFORM

Project 3.1 DTFT Computation

A copy of Program P3_1 is given below:

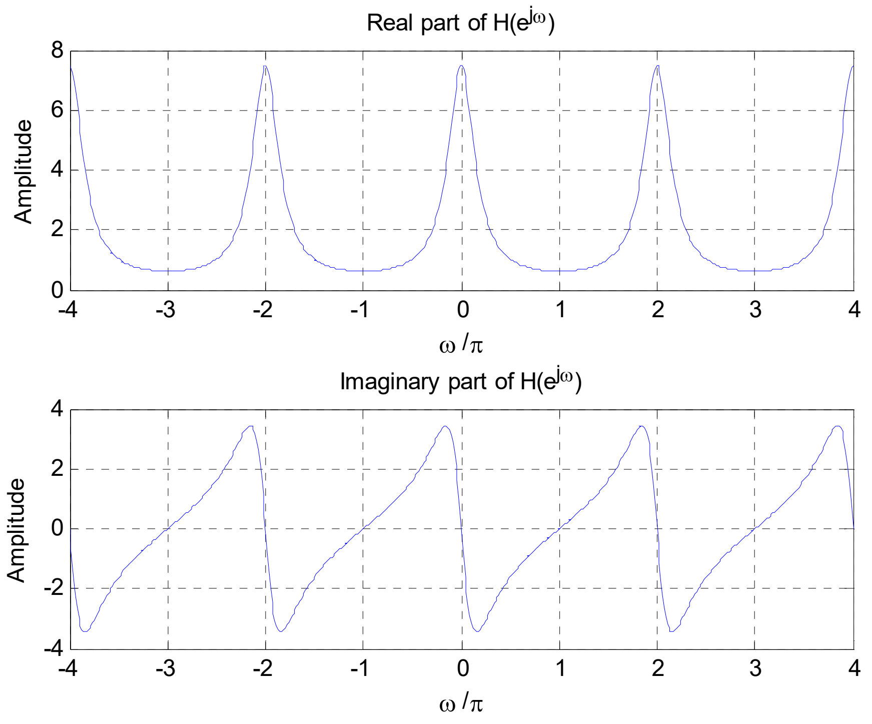
```
% Program P3_1
% Evaluation of the DTFT
clf;
% Compute the frequency samples of the DTFT
w = -4*pi:8*pi/511:4*pi;
num = [2 1];den = [1 -0.6];
h = freqz(num, den, w);
% Plot the DTFT
subplot(2,1,1)
plot(w/pi,real(h));grid
title('Real part of H(e^{j\omega})')
xlabel('\omega /\pi');
ylabel('Amplitude');
subplot(2,1,2)
plot(w/pi,imag(h));grid
title('Imaginary part of H(e^{j\omega})')
xlabel('\omega /\pi');
ylabel('Amplitude');
pause
subplot(2,1,1)
plot(w/pi,abs(h));grid
title('Magnitude Spectrum |H(e^{j\omega})|')
xlabel('\omega /\pi');
ylabel('Amplitude');
subplot(2,1,2)
plot(w/pi,angle(h));grid
title('Phase Spectrum arg[H(e^{j\omega})]')
xlabel('\omega /\pi');
ylabel('Phase in radians');
```

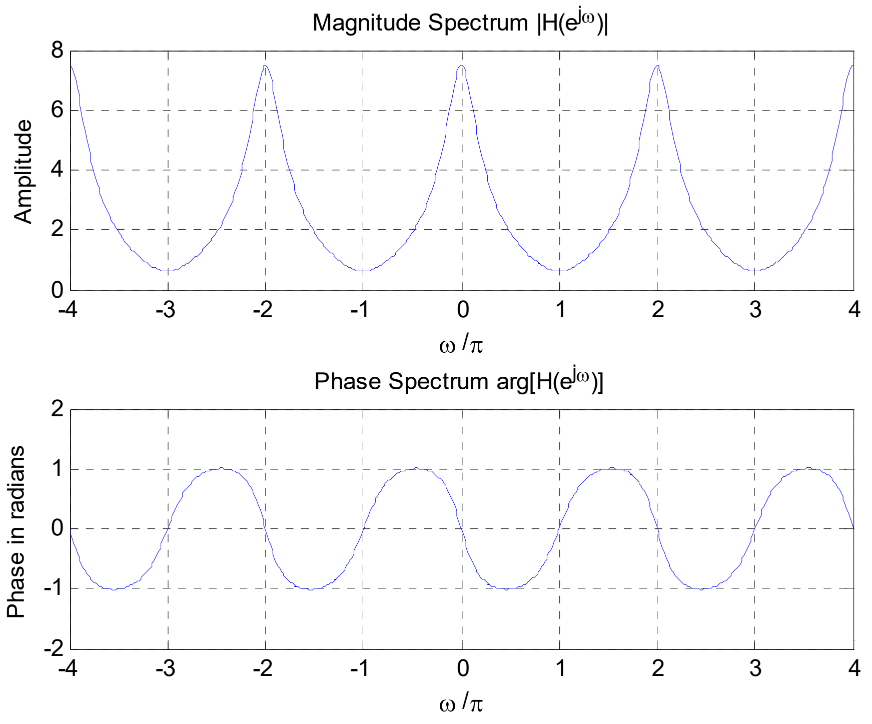
Answers:

Q3.1 The expression of the DTFT being evaluated in Program P3_1 is - $H\left(e^{j\omega}\right)=\dfrac{2+z^{-1}}{1-0.6z^{-1}}$

The function of the `pause` command is - to pause execution of a Matlab program. Without arguments, `pause` waits for the user to type any key. With an argument, `pause` pauses for a number of seconds specified by the argument.

Q3.2 The plots generated by running Program P3_1 are shown below:





The DTFT is a _periodic_ function of ω.

Its period is - 2π

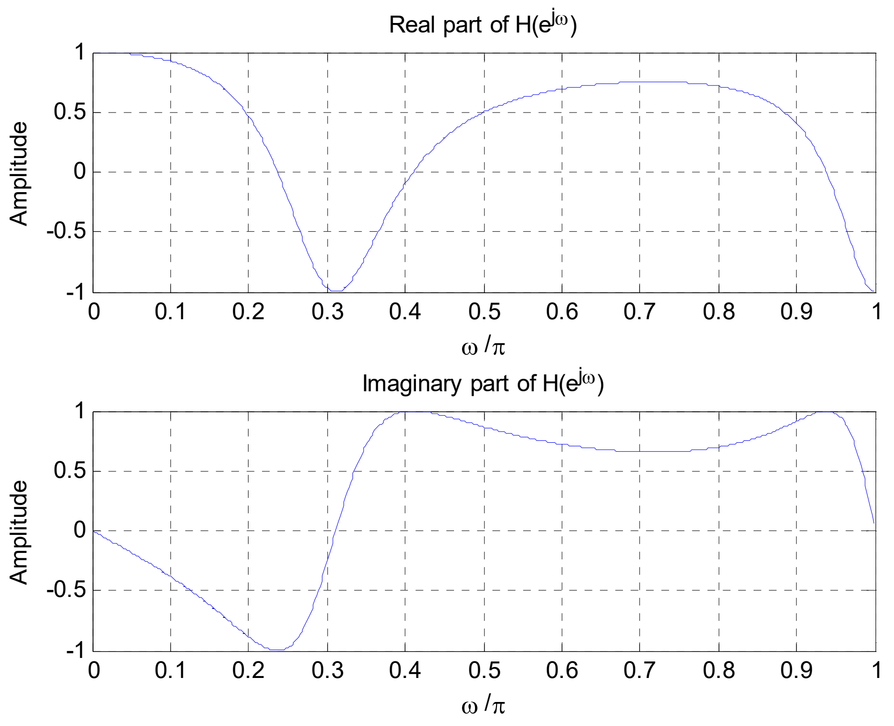
The types of symmetries exhibited by the four plots are as follows:

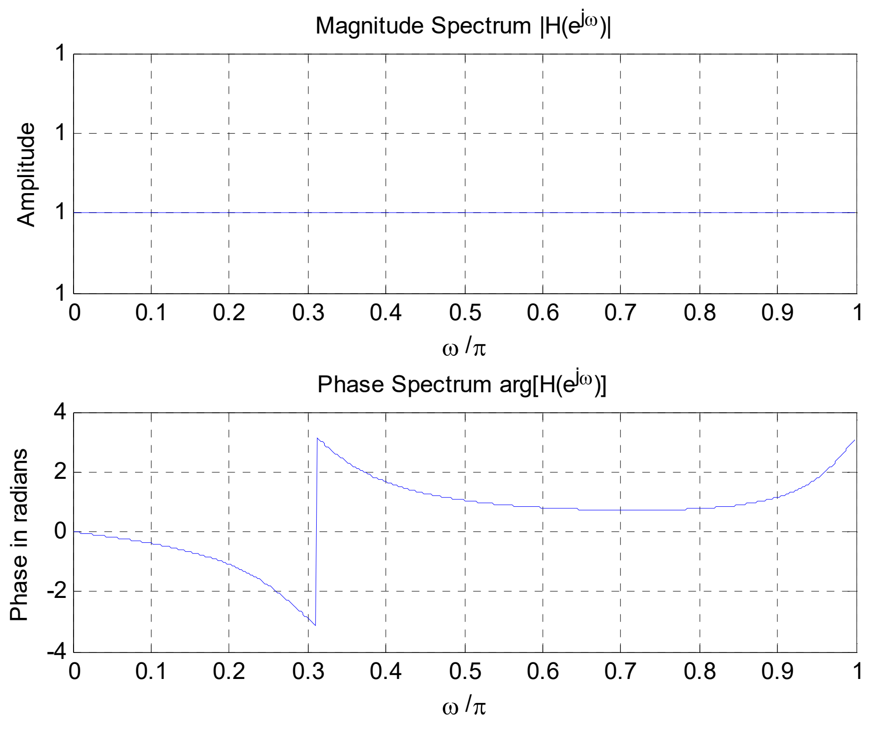
- The real part is 2π periodic and EVEN SYMMETRIC.
- The imaginary part is 2π periodic and ODD SYMMETRIC.
- The magnitude is 2π periodic and EVEN SYMMETRIC.
- The phase is 2π periodic and ODD SYMMETRIC.

Q3.3 The required modifications to Program P3_1 to evaluate the given DTFT of Q3.3 are given below:

```
% Program P3_1B
% Evaluation of the DTFT
clf;
% Compute the frequency samples of the DTFT
%  because 0 \leq w \leq pi is the default for "freqz",
%  the vector "w" is now an output of freqz instead of an input.
N = 512;
num = [0.7 -0.5 0.3 1];
den = [1 0.3 -0.5 0.7];
[h,w] = freqz(num, den, N);
% Plot the DTFT
subplot(2,1,1)
plot(w/pi,real(h));grid
title('Real part of H(e^{j\omega})')
xlabel('\omega /\pi');
ylabel('Amplitude');
subplot(2,1,2)
plot(w/pi,imag(h));grid
title('Imaginary part of H(e^{j\omega})')
xlabel('\omega /\pi');
ylabel('Amplitude');
pause
subplot(2,1,1)
plot(w/pi,abs(h));grid
title('Magnitude Spectrum |H(e^{j\omega})|')
xlabel('\omega /\pi');
ylabel('Amplitude');
subplot(2,1,2)
plot(w/pi,angle(h));grid
title('Phase Spectrum arg[H(e^{j\omega})]')
xlabel('\omega /\pi');
ylabel('Phase in radians');
```

The plots generated by running the modified Program P3_1 are shown below:



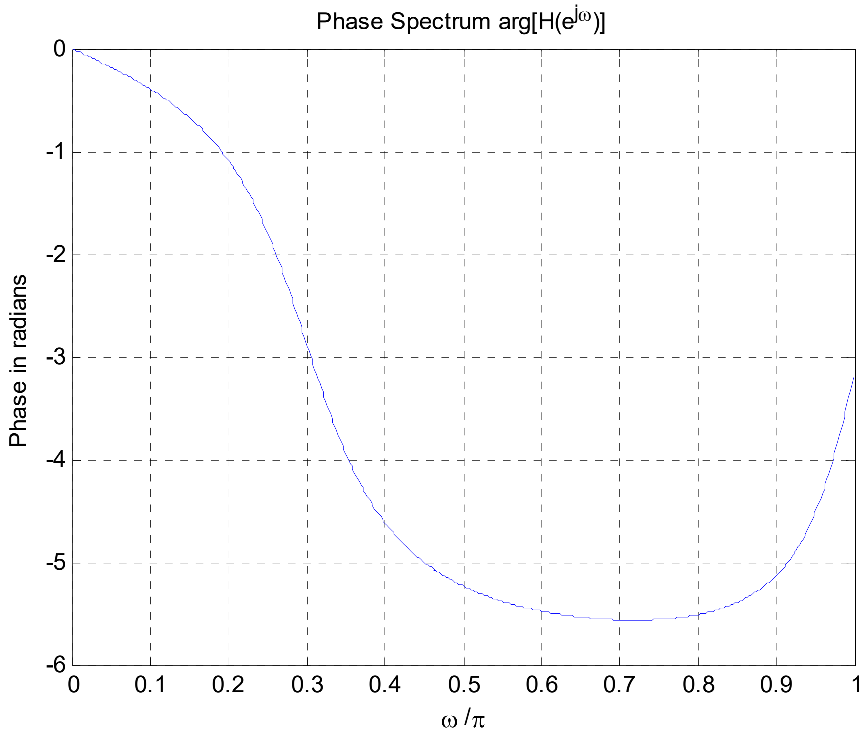


The DTFT is a _periodic__ function of ω.

Its period is - 2π

The jump in the phase spectrum is caused by - a branch cut in the arctan function used by `angle` in computing the phase. "`angle`" returns the principal branch of arctan.

The phase spectrum evaluated with the jump removed by the command `unwrap` is as given below:

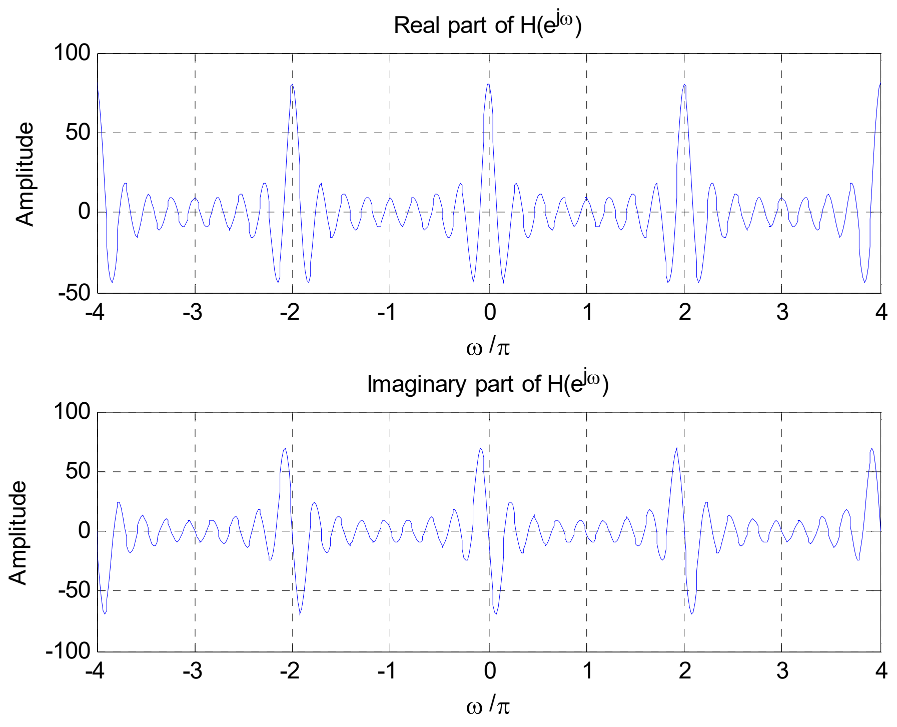


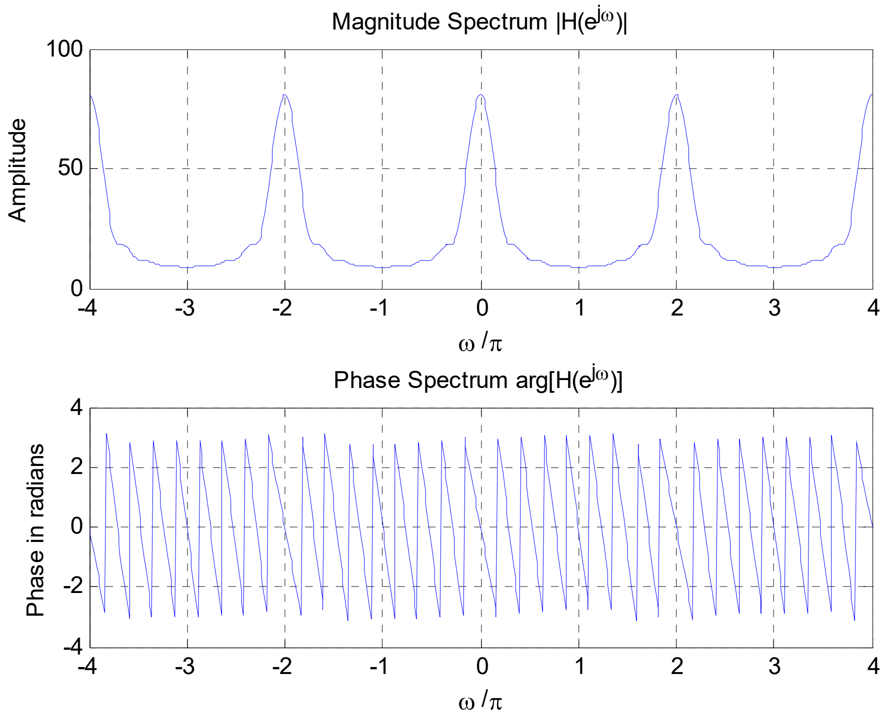
Q3.4 The required modifications to Program P3_1 to evaluate the given DTFT of Q3.4 are given below:

```
% Program P3_1D
% Evaluation of the DTFT
clf;
% Compute the frequency samples of the DTFT
w = -4*pi:8*pi/511:4*pi;
num = [1 3 5 7 9 11 13 15 17];
den = 1;
h = freqz(num, den, w);
% Plot the DTFT
subplot(2,1,1)
plot(w/pi,real(h));grid
title('Real part of H(e^{j\omega})')
xlabel('\omega /\pi');
ylabel('Amplitude');
subplot(2,1,2)
plot(w/pi,imag(h));grid
title('Imaginary part of H(e^{j\omega})')
xlabel('\omega /\pi');
ylabel('Amplitude');
pause
subplot(2,1,1)
```

```
plot(w/pi,abs(h));grid
title('Magnitude Spectrum |H(e^{j\omega})|')
xlabel('\omega /\pi');
ylabel('Amplitude');
subplot(2,1,2)
plot(w/pi,angle(h));grid
title('Phase Spectrum arg[H(e^{j\omega})]')
xlabel('\omega /\pi');
ylabel('Phase in radians');
```

The plots generated by running the modified Program P3_1 are shown below:





The DTFT is a _periodic_ function of ω.

Its period is - 2π

The jump in the phase spectrum is caused by - "`angle`" returns the principal value of the arc tangent.

Q3.5 The required modifications to Program P3_1 to plot the phase in degrees are indicated below:

Only the last paragraph of the code needs to be changed to:

```
% plot phase in degrees
subplot(2,1,2)
plot(w/pi,180*angle(h)/pi);grid
title('Phase Spectrum arg[H(e^{j\omega})]')
xlabel('\omega /\pi');
ylabel('Phase in degrees');
```

Project 3.2 DTFT Properties

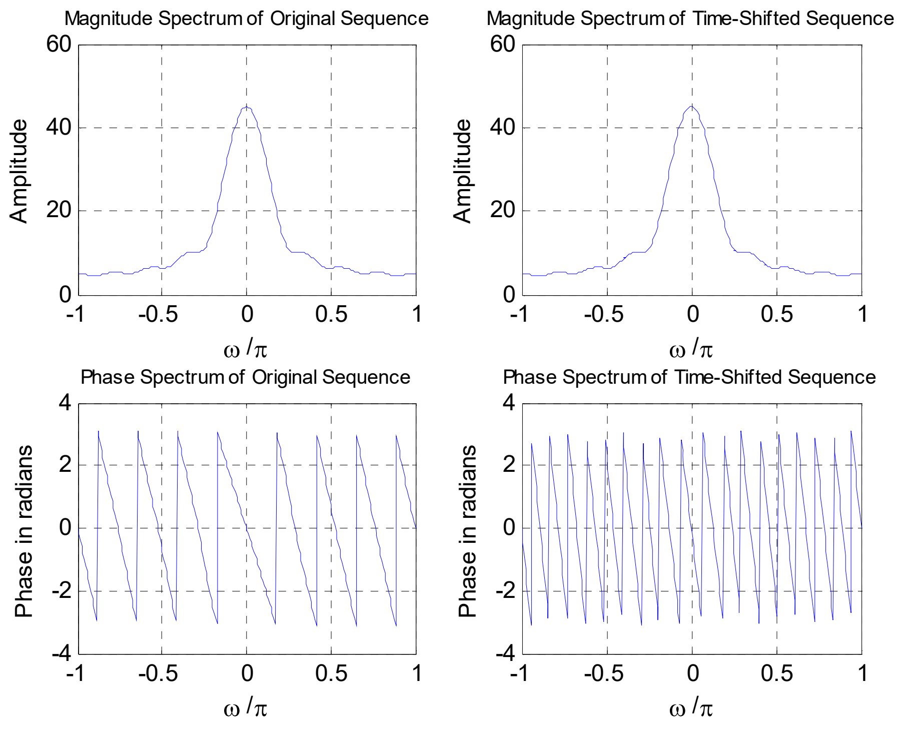
Answers:

Q3.6 The modified Program P3_2 created by adding appropriate comment statements, and adding program statements for labeling the two axes of each plot being generated by the program is given below:

```
% Program P3_2B
% Time-Shifting Properties of DTFT
clf;
w = -pi:2*pi/255:pi; % freqency vector for evaluating DTFT
D = 10; % Amount of time shift in samples
num = [1 2 3 4 5 6 7 8 9];
% h1 is the DTFT of original sequence
% h2 is the DTFT of the time shifted sequence
h1 = freqz(num, 1, w);
h2 = freqz([zeros(1,D) num], 1, w);
subplot(2,2,1)
% plot the DTFT magnitude of the original sequence
plot(w/pi,abs(h1));grid
title('Magnitude Spectrum of Original Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the DTFT magnitude of the shifted sequence
subplot(2,2,2)
plot(w/pi,abs(h2));grid
title('Magnitude Spectrum of Time-Shifted Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the DTFT phase of the original sequence
subplot(2,2,3)
plot(w/pi,angle(h1));grid
title('Phase Spectrum of Original Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Phase in radians');
% plot the DTFT phase of the shifted sequence
subplot(2,2,4)
plot(w/pi,angle(h2));grid
title('Phase Spectrum of Time-Shifted Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Phase in radians');
```

The parameter controlling the amount of time-shift is - D

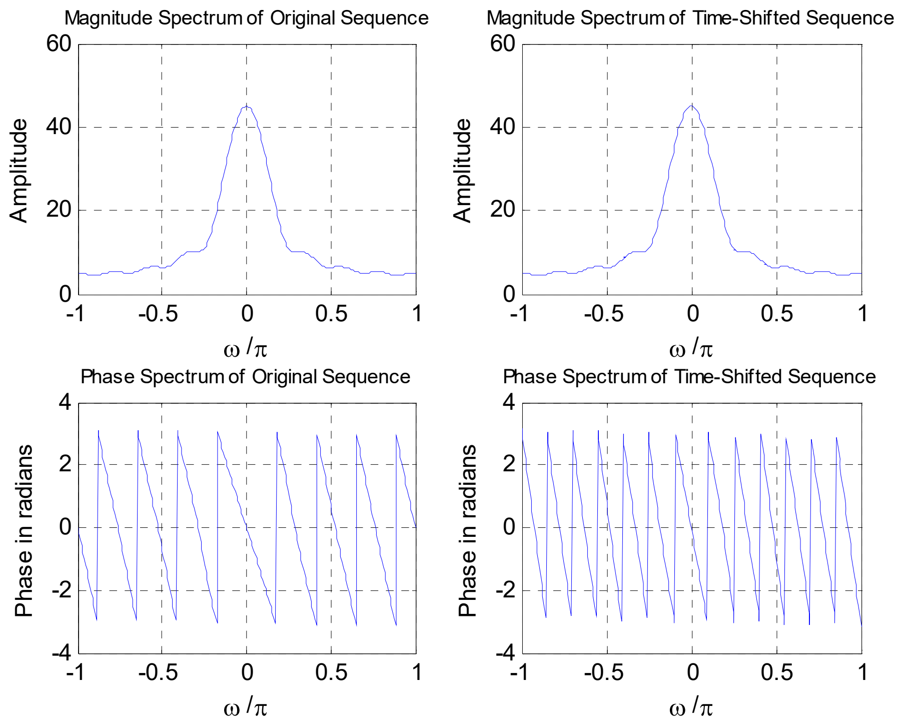
Q3.7 The plots generated by running the modified program are given below:



From these plots we make the following observations: The time shift does not have any effect at all on the magnitude spectrum. However, it has a significant effect on the phase spectrum. The effect is to add phase, which makes the slope of the phase function steeper (more negative in this case).

Q3.8 Program P3_2 was run for the following value of the time-shift – `D=5`.

The plots generated by running the modified program are given below:

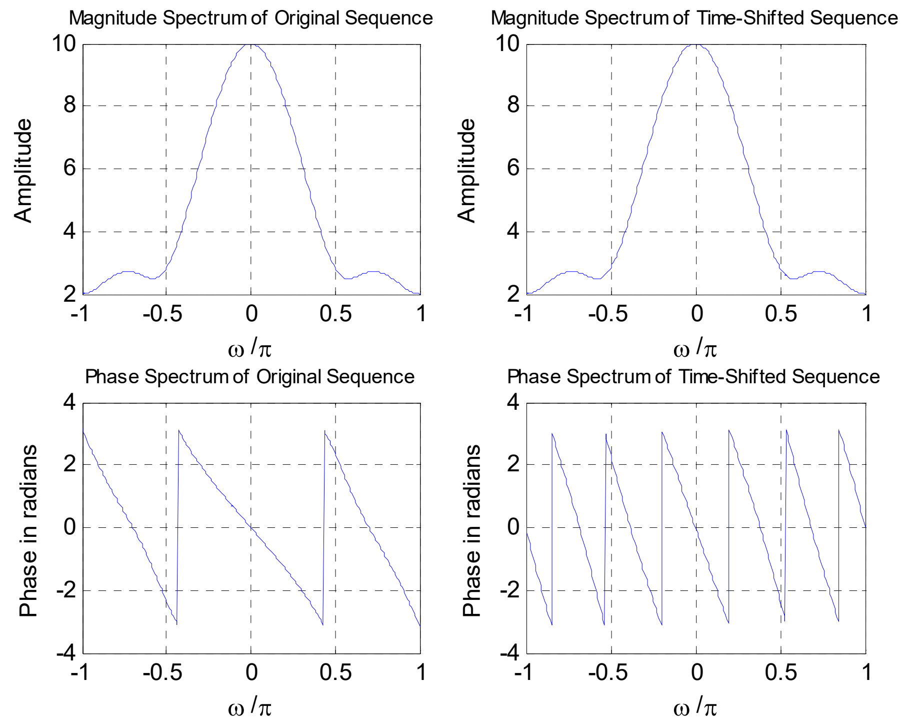


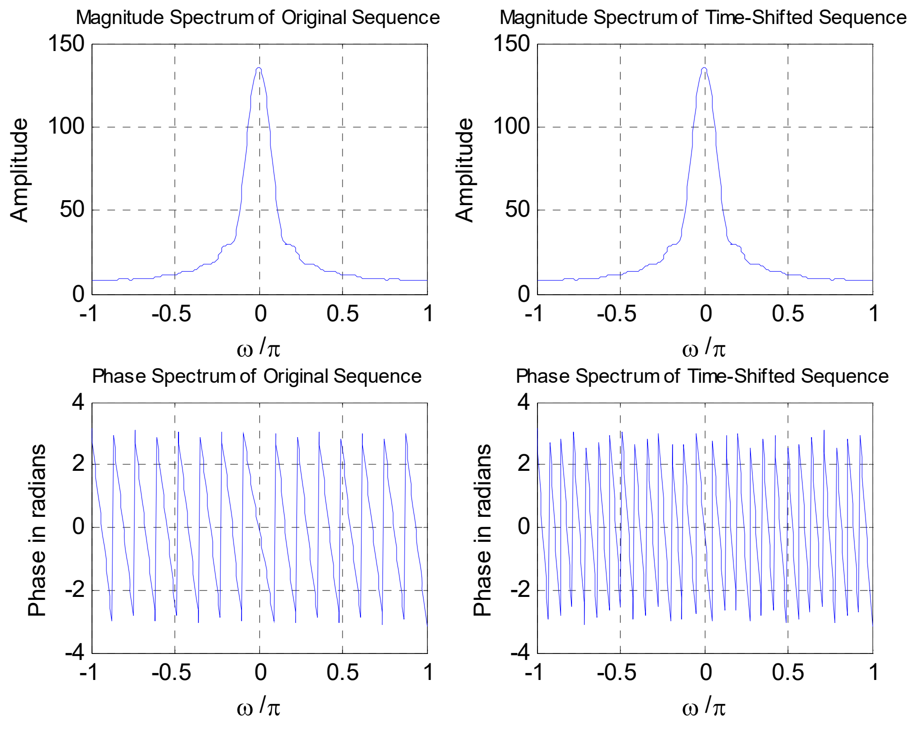
From these plots we make the following observations: As before, of course, the time shift has no effect on the magnitude spectrum. However, there is a very significant change to the phase. As before, the time shift adds phase to the DTFT, making the slope of the phase spectrum steeper. However, in this case with `D=5` instead of `D=10`, the (negative) increase in the slope is less than it was before.

Q3.9 Program P3_2 was run for the following values of the time-shift and for the following values of length for the sequence –

1. Length 4, time shift `D=3`.
2. Length 16, time shift `D=12`.

The plots generated by running the modified program are given below:





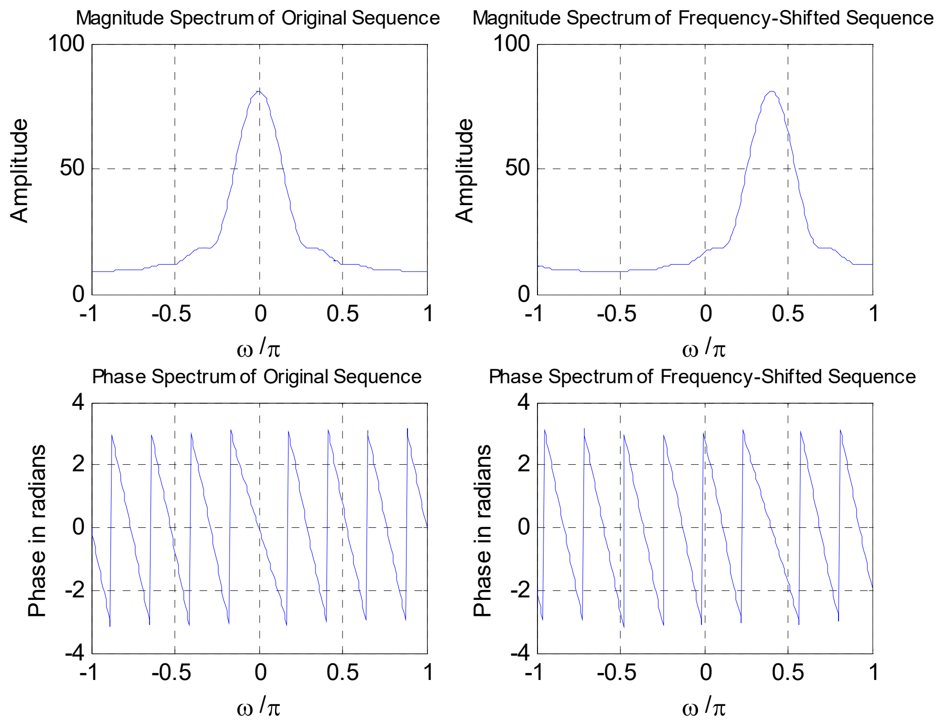
From these plots we make the following observations: Increasing the length makes the magnitude spectrum more narrow (i.e., makes the signal more "low pass"). It also makes the phase steeper (i.e., the slope more negative). This is because, if we think of the sequence as the impulse response of an LTI system, increasing the length adds more delay between the input and output of the system. As before, the time shift does not have any effect on the magnitude spectrum. However, it makes the slope of the phase spectrum more negative. The larger the value of the delay, the more negative slope is added to the phase spectrum.

Q3.10 The modified Program P3_3 created by adding appropriate comment statements, and adding program statements for labeling the two axes of each plot being generated by the program is given below:

```
% Program P3_3B
% Frequency-Shifting Properties of DTFT
clf;
w = -pi:2*pi/255:pi; % freqency vector for evaluating DTFT
wo = 0.4*pi; % Amount of frequency shift in radians
% h1 is the DTFT of the original sequence
% h2 is the DTFT of the frequency shifted sequence
num1 = [1 3 5 7 9 11 13 15 17];
L = length(num1);
h1 = freqz(num1, 1, w);
n = 0:L-1;
num2 = exp(wo*i*n).*num1;
h2 = freqz(num2, 1, w);
% plot the DTFT magnitude of the original sequence
subplot(2,2,1)
plot(w/pi,abs(h1));grid
title('Magnitude Spectrum of Original Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the DTFT magnitude of the freq shifted sequence
subplot(2,2,2)
plot(w/pi,abs(h2));grid
title('Magnitude Spectrum of Frequency-Shifted Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the DTFT phase of the original sequence
subplot(2,2,3)
plot(w/pi,angle(h1));grid
title('Phase Spectrum of Original Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Phase in radians');
% plot the DTFT phase of the shifted sequence
subplot(2,2,4)
plot(w/pi,angle(h2));grid
title('Phase Spectrum of Frequency-Shifted Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Phase in radians');
```

The parameter controlling the amount of frequency-shift is - `wo`.

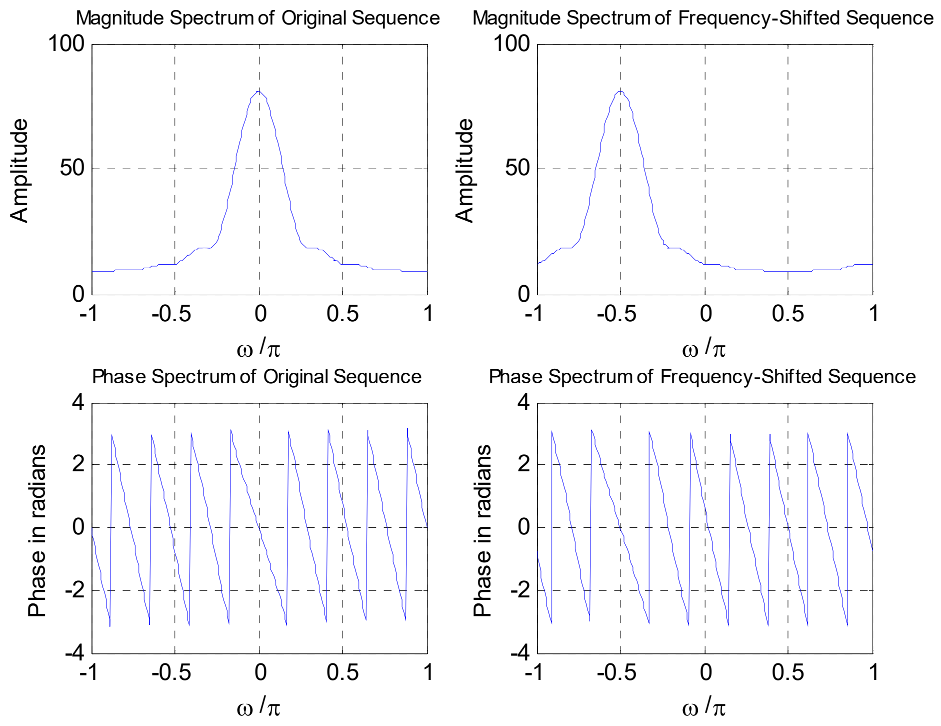
Q3.11 The plots generated by running the modified program are given below:



From these plots we make the following observations: Both the magnitude and phase spectra are shifted right by `wo`, which is given by 0.4π in this case. Note that the frequency shifted signal was obtained by multiplying the original sequence pointwise with a complex-valued exponential sequence. Thus, the frequency shifted sequence is also complex-valued and it's DTFT does not have conjugate symmetry.

Q3.12 Program P3_3 was run for the following value of the frequency-shift – `wo` = -0.5π.

The plots generated by running the modified program are given below:



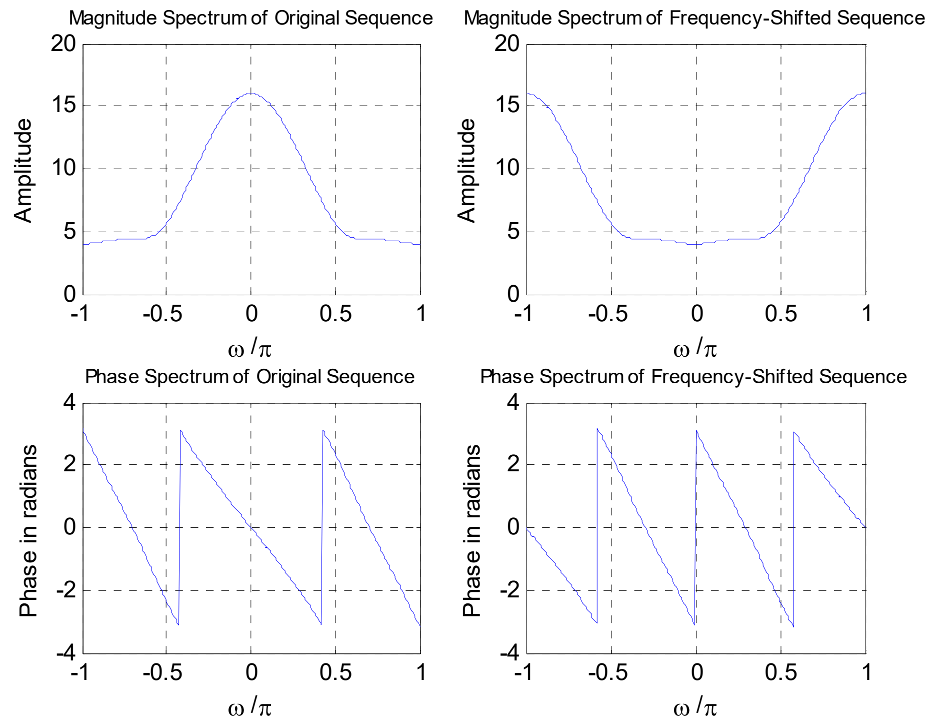
From these plots we make the following observations: In this case, the magnitude and phase spectra are shifted left by π/2 rad. As before, the frequency shifted signal is complex-valued, so the frequency shift causes a loss of the conjugate symmetry that was present in the spectrum of the original signal. NOTE: you should keep in mind that these spectra are all 2π-periodic; we are only displaying the fundamental period.

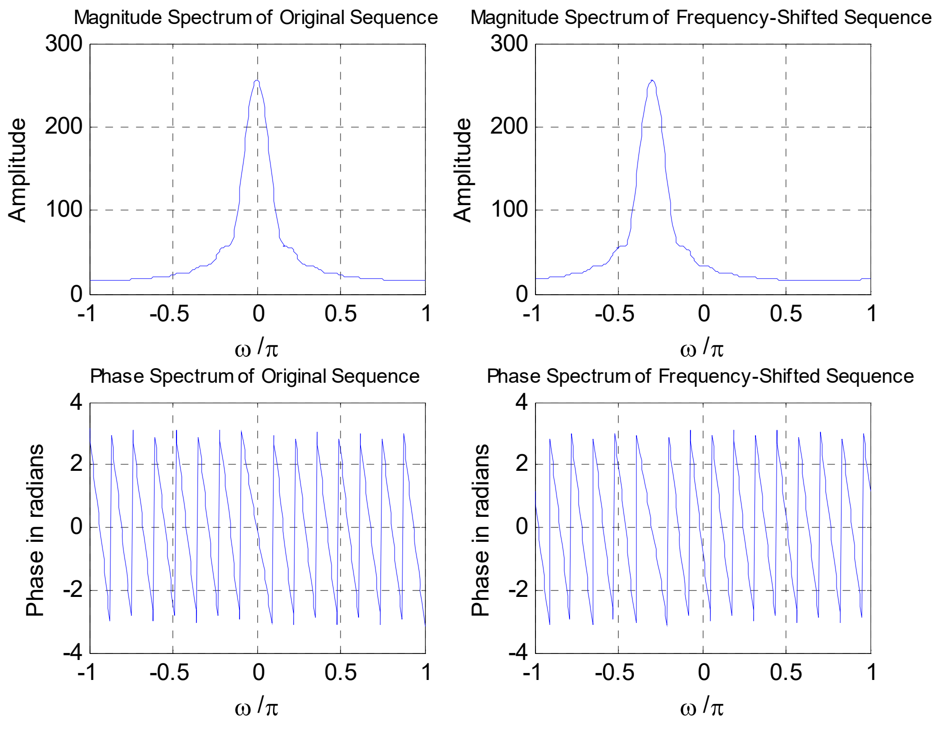
Q3.13 Program P3_3 was run for the following values of the frequency-shift and for the following values of length for the sequence –

1. Length 4, frequency shift `wo` = π.

2. Length 16, frequency shift `wo` = -0.3π.

The plots generated by running the modified program are given below:



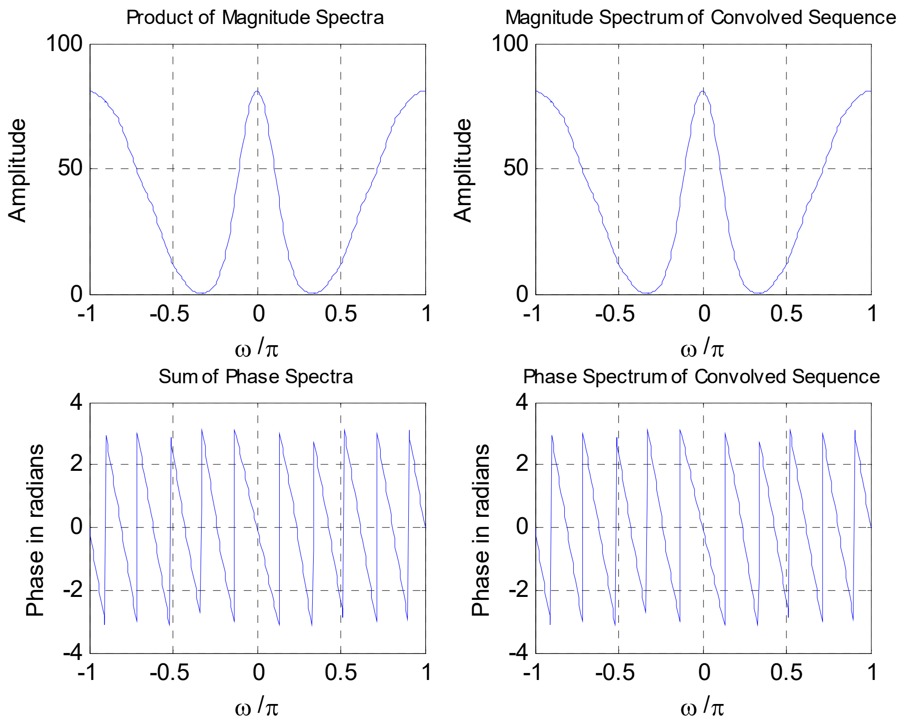


From these plots we make the following observations: The original sequences have a low pass characteristic. As before with the time shift property, a shorter length gives a broader low pass magnitude spectrum, whereas a longer length results in a low pass magnitude spectrum that is more narrow. Also, if we consider these sequences to be the impulse responses of LTI systems, then a shorter length implies a smaller delay between the input and output of the system, which corresponds to a phase spectrum with a negative slope that is less steep. Likewise, a longer length implies a longer delay between the input and output, which corresponds to a phase spectrum with a negative slope that is steeper. In both cases shown here, the effect of the frequency shift is to translate both the magnitude and phase spectrum by `wo` radians. Whereas the original sequences are real-valued and hence have conjugate symmetric spectra, the frequency shifted sequences are complex-valued and do not exhibit any inherent spectral symmetry.

Q3.14 The modified Program P3_4 created by adding appropriate comment statements, and adding program statements for labeling the two axes of each plot being generated by the program is given below:

```
% Program P3_4B
% Convolution Property of DTFT
clf;
w = -pi:2*pi/255:pi; % freqency vector for evaluating DTFT
x1 = [1 3 5 7 9 11 13 15 17]; % first sequence
x2 = [1 -2 3 -2 1]; % second sequence
y = conv(x1,x2); % time domain convolution of x1 and x2
h1 = freqz(x1, 1, w); % DTFT of sequence x1
h2 = freqz(x2, 1, w); % DTFT of sequence x2
% hp is the pointwise product of the two DTFT's
hp = h1.*h2;
% h3 is the DTFT of the time domain convolution;
%   it should be the same as hp
h3 = freqz(y,1,w);
% plot the magnitude of the product of the two original spectra
subplot(2,2,1)
plot(w/pi,abs(hp));grid
title('Product of Magnitude Spectra','FontSize',8)
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the magnitude spectrum of the time domain convolution
subplot(2,2,2)
plot(w/pi,abs(h3));grid
title('Magnitude Spectrum of Convolved Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the phase of the product of the two original spectra
subplot(2,2,3)
plot(w/pi,angle(hp));grid
title('Sum of Phase Spectra','FontSize',8)
xlabel('\omega /\pi');
ylabel('Phase in radians');
% plot the phase spectrum of the time domain convolution
subplot(2,2,4)
plot(w/pi,angle(h3));grid
title('Phase Spectrum of Convolved Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Phase in radians');
```

Q3.15 The plots generated by running the modified program are given below:



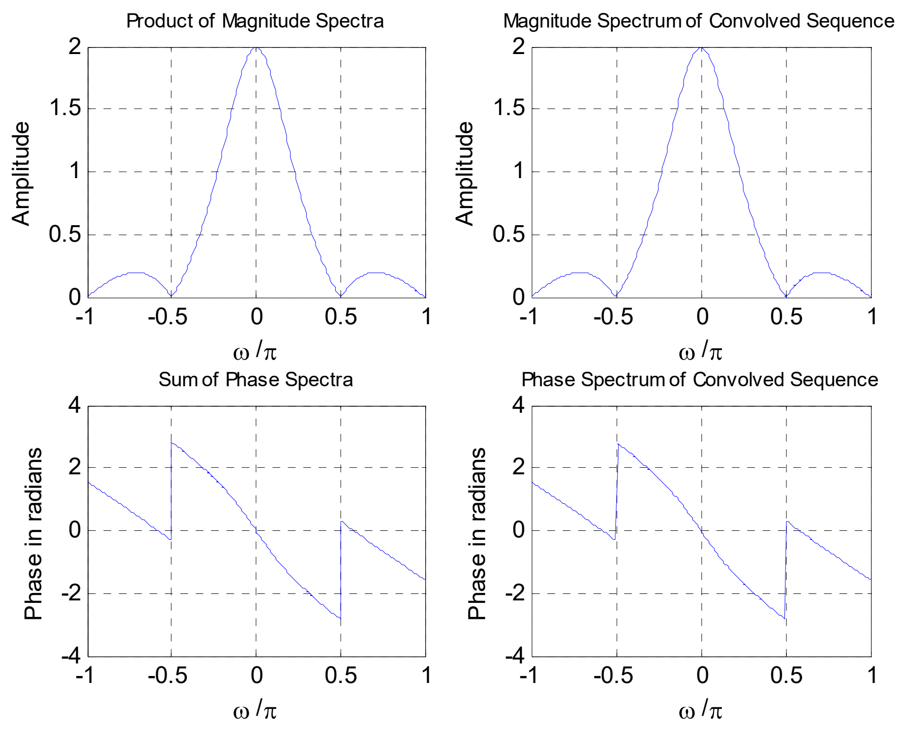
From these plots we make the following observations: The DTFT magnitude and phase spectra obtained by performing pointwise multiplication of the two DTFT's of the original sequences are identical to those obtained by performing time domain convolution of the two original sequences; this verifies the convolution property of the DTFT.

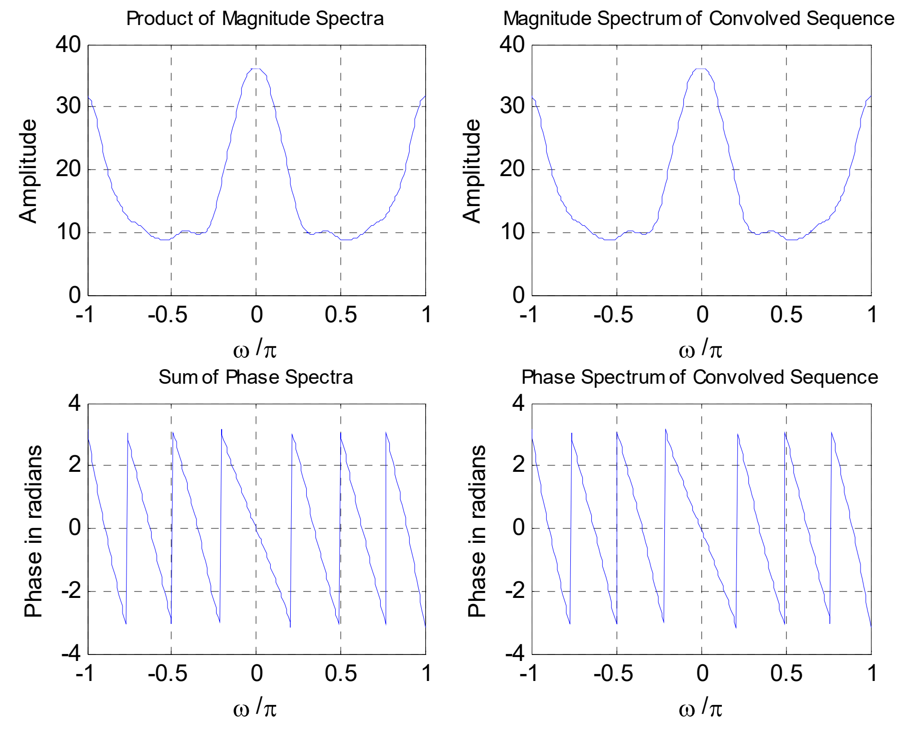
Q3.16 Program P3_4 was run for the following two different sets of sequences of varying lengths –

1. Length of `x1` = 8; `x1[n]` = $(1/2)^n$ for $0 \le n \le 7$;
   Length of `x2` = 4; `x2[n]` = [0.25 0.25 0.25 0.25]

2. Length of `x1` = 16; `x1[n]` = $(-3/4)^n$ for $0 \le n \le 15$;
   Length of `x2` = 8; `x2[n]` = [1 3 5 7 9 11 13 15]

The plots generated by running the modified program are given below:



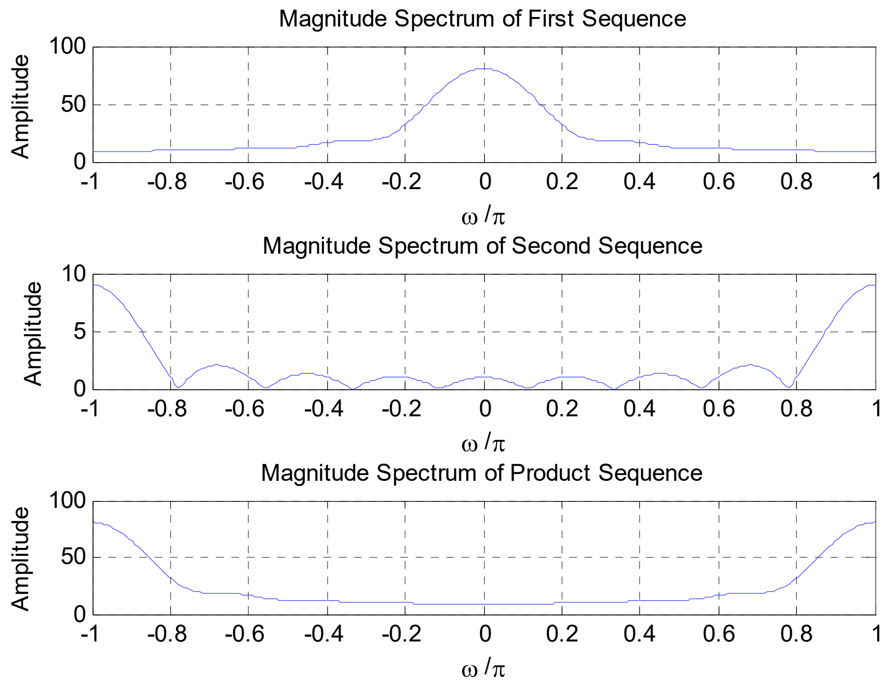


From these plots we make the following observations: The convolution property of the DTFT is again verified in both cases. In each case, the DTFT magnitude and phase obtained by taking the pointwise products of the DTFT's of the two original sequences are identical to the magnitude and phase spectra obtained from the DTFT of the time domain convolution of the two original sequences.

Q3.17 The modified Program P3_5 created by adding appropriate comment statements, and adding program statements for labeling the two axes of each plot being generated by the program is given below:

```
% Program P3_5B
% Modulation Property of DTFT
clf;
w = -pi:2*pi/255:pi; % freqency vector for evaluating DTFT
x1 = [1 3 5 7 9 11 13 15 17]; % first sequence
x2 = [1 -1 1 -1 1 -1 1 -1 1]; % second sequence
% y is the time domain pointwise product of x1 and x2
y = x1.*x2;
h1 = freqz(x1, 1, w); % DTFT of sequence x1
h2 = freqz(x2, 1, w); % DTFT of sequence x2
h3 = freqz(y,1,w);    % DTFT of sequence y
% plot the magnitude spectrum of x1
subplot(3,1,1)
plot(w/pi,abs(h1));grid
title('Magnitude Spectrum of First Sequence')
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the magnitude spectrum of x2
subplot(3,1,2)
plot(w/pi,abs(h2));grid
title('Magnitude Spectrum of Second Sequence')
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the magnitude spectrum of y
%   it should be 1/2pi times the convolution of the DTFT's
%   of the two original sequences.
subplot(3,1,3)
plot(w/pi,abs(h3));grid
title('Magnitude Spectrum of Product Sequence')
xlabel('\omega /\pi');
ylabel('Amplitude');
```

Q3.18 The plots generated by running the modified program are given below:



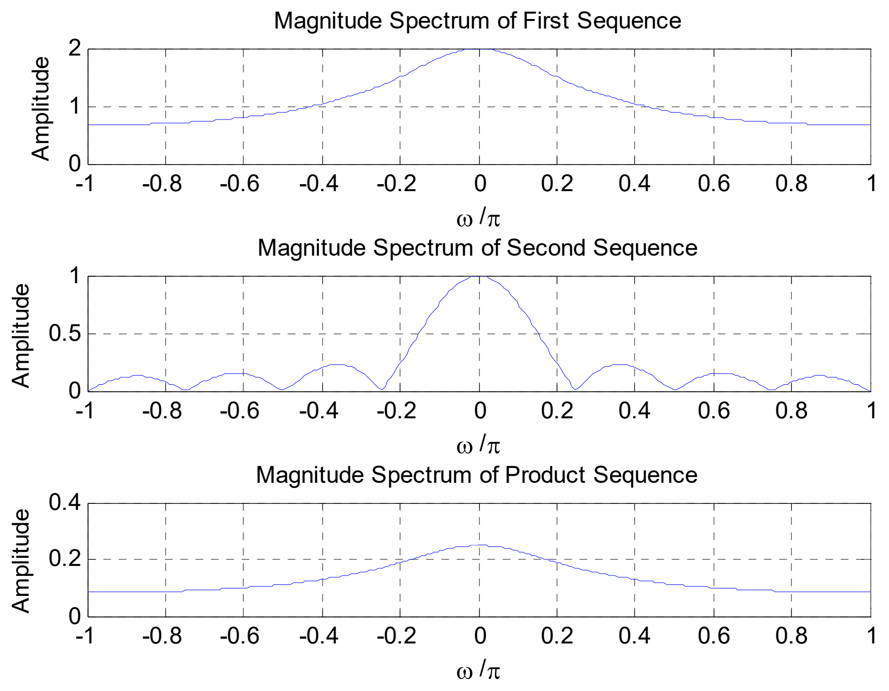
From these plots we make the following observations: The DTFT of the product sequence `y` is $1/2\pi$ times the convolution of the DTFT's of the two sequences `x1` and `x2`, as expected. The low-pass mainlobe of the DTFT of `x1` combines with the high-pass mainlobe of the DTFT of `x2` to produce a high-pass mainlobe centered at $\pm\pi$ in the magnitude spectrum of the product signal. The low-pass mainlobe of the DTFT of `x1` combines with the low-pass sidelobes of the DTFT of `x2` to produce a low-pass smooth region of relatively lower gain centered at DC in the magnitude spectrum of the product signal.

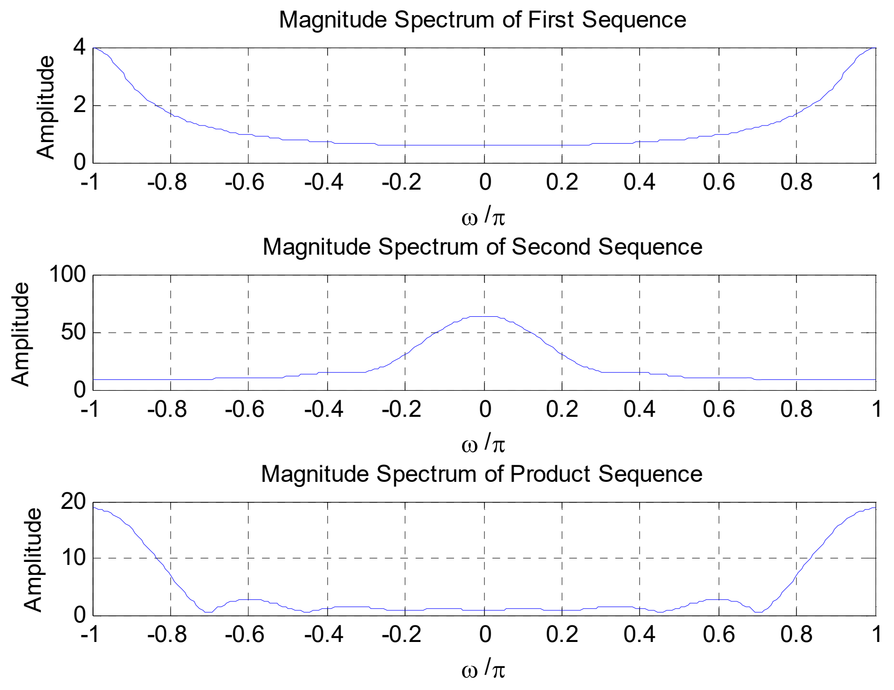
Q3.19 Program P3_5 was run for the following two different sets of sequences of varying lengths –

1. Length of `x1` = 8; `x1[n]` = $(1/2)^n$ for $0 \le n \le 7$;
   Length of `x2` = 8; `x2[n]`= [1/8 1/8 1/8 1/8 1/8 1/8 1/8 1/8]

2. Length of `x1` = 16; `x1[n]` = $(-3/4)^n$ for $0 \le n \le 15$;
   Length of `x2` = 16; `x2[n]`= [1 3 5 7 9 11 13 15 0 0 0 0 0 0 0 0]

The plots generated by running the modified program are given below:





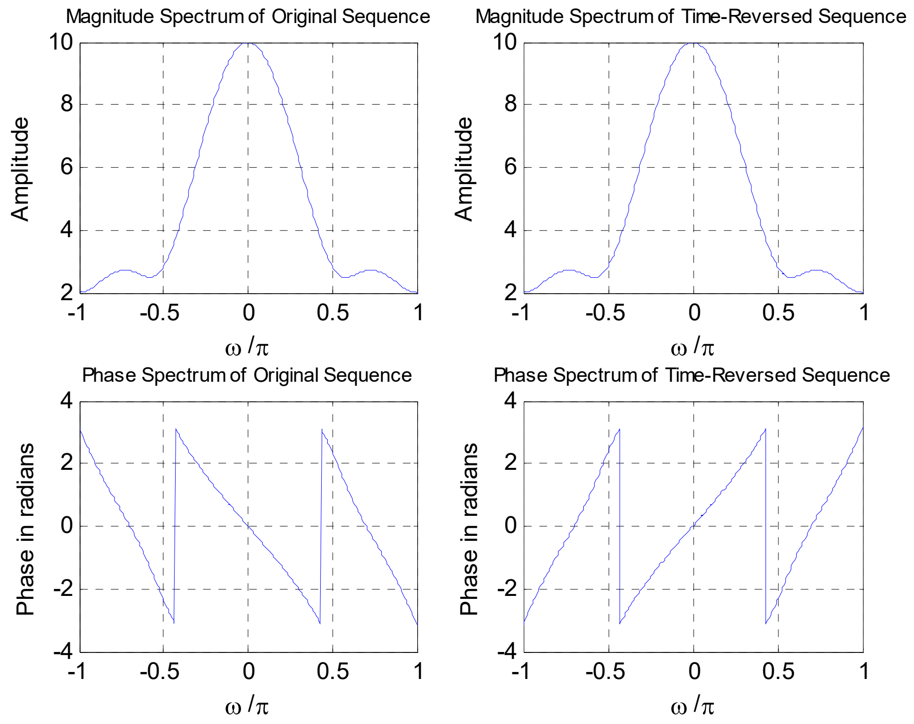
From these plots we make the following observations: In the first example, both `x1` and `x2` are low pass sequences. Moreover, the DTFT of `x2` is a sync pulse. Taking the product of these two sequences produces a new sequence `y` that is also low pass in character. The magnitude spectrum of `y` has a shape that is very similar to that of `x1`, but with some averaging. The spectral magnitude of `y` is reduced compared to that of `x1`, primarily due to the division by 2π that is inherent in the DTFT frequency convolution (time modulation) property. In the second example, `x1` is high pass while `x2` is low pass. As before in Q3.18, we see that magnitude spectrum of `y` is essentially high pass in character as a result of the low pass mainlobe of the DTFT of `x2` being convolved with the high pass mainlobe of `x1`.

Q3.20 The modified Program P3_6 created by adding appropriate comment statements, and adding program statements for labeling the two axes of each plot being generated by the program is given below:

```
% Program P3_6B
% Time Reversal Property of DTFT
clf;
w = -pi:2*pi/255:pi; % freqency vector for evaluating DTFT
% original ramp sequence
%  note: num is nonzero for 0 <= n <= 3.
num = [1 2 3 4];
L = length(num)-1;
h1 = freqz(num, 1, w); % DTFT of original ramp sequence
% h2 contains the sample values of h1 in reverse order, but
%   it is NOT the time reversed version of h1.  The time
%   reversed version must be nonzero for -3 <= n <= 0.  However,
%   h2 is nonzero for 0 <= n <= 3.  So, to get the time reversed
%   version of h1, we still need to time SHIFT h2 to the left.
%   This is accomplished in the frequency domain using the time
%   shift property of the DTFT.  Thus, h3, which IS the time
%   reversed version of h1, is obtained by multiplying h2 times
%   a linear phase term to accomplish the required time shift.
h2 = freqz(fliplr(num), 1, w);
h3 = exp(w*L*i).*h2;
% plot the magnitude spectrum of the original ramp sequence
subplot(2,2,1)
plot(w/pi,abs(h1));grid
title('Magnitude Spectrum of Original Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the magnitude spectrum of the time reversed ramp sequence
subplot(2,2,2)
plot(w/pi,abs(h3));grid
title('Magnitude Spectrum of Time-Reversed Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Amplitude');
% plot the phase spectrum of the original ramp sequence
subplot(2,2,3)
plot(w/pi,angle(h1));grid
title('Phase Spectrum of Original Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Phase in radians');
% plot the phase spectrum of the time reversed ramp sequence
subplot(2,2,4)
plot(w/pi,angle(h3));grid
title('Phase Spectrum of Time-Reversed Sequence','FontSize',8)
xlabel('\omega /\pi');
ylabel('Phase in radians');
```

The program implements the time-reversal operation as follows – The original ramp sequence is nonzero for $0 \leq n \leq 3$. A new sequence is formed by using `fliplr`; this new sequence contains the samples of the original ramp sequence in time reversed order. However, the new sequence is still nonzero for $0 \leq n \leq 3$, whereas the time reversed ramp sequence must be nonzero for $-3 \leq n \leq 0$. This required left shift in time is accomplished in the frequency domain using the time shift property of the DTFT as follows. First, `freqz` is called to set `h2` equal to the DTFT of the new sequence obtained from calling `fliplr` on the original ramp sequence. Finally, `h3` is set equal to the DTFT of the time reversed ramp by multiplying `h2` times a linear phase term to implement the required left shift in the time domain.

Q3.21 The plots generated by running the modified program are given below:



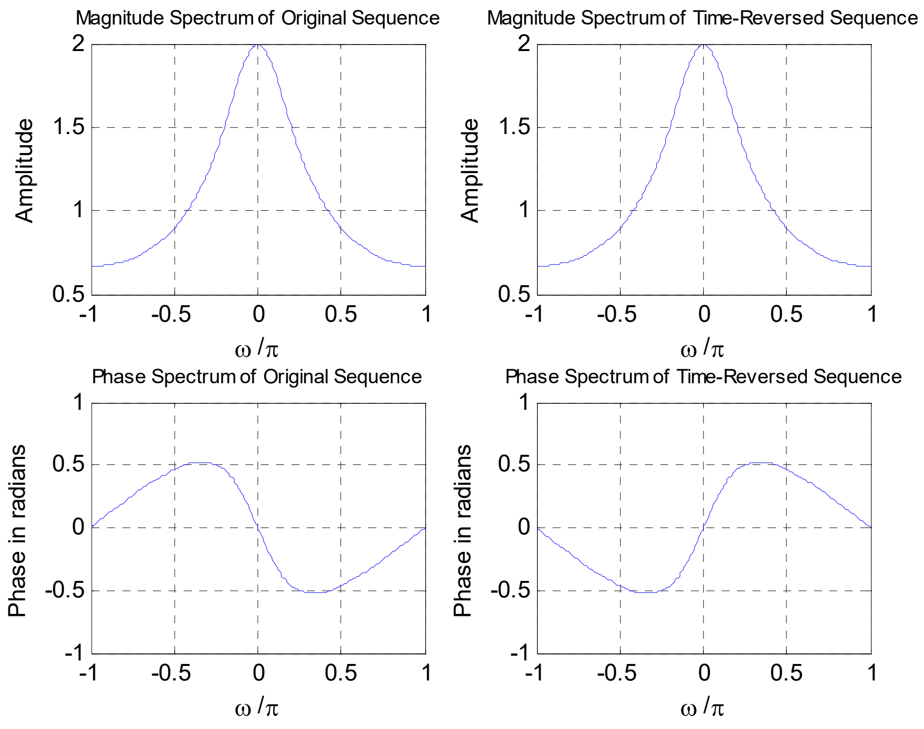
From these plots we make the following observations: Both the original and time reversed ramp sequences are real-valued. Therefore, they have conjugate symmetric DTFT's. For both sequences, this implies that the magnitude spectrum is even symmetric and the phase spectrum is odd symmetric. Now, the DTFT of the time reversed ramp (`h3`) is equal to a *frequency reversed* version of the DTFT of the original sequence (`h1`). Since "flipping" an even function has no net effect, we see in the graphs above that both the original and time reversed sequences have identical magnitude spectra. However, since the phase spectra are odd symmetric, the phase spectrum of the time reversed sequence is a frequency reversed version of the phase spectrum of the original sequence.

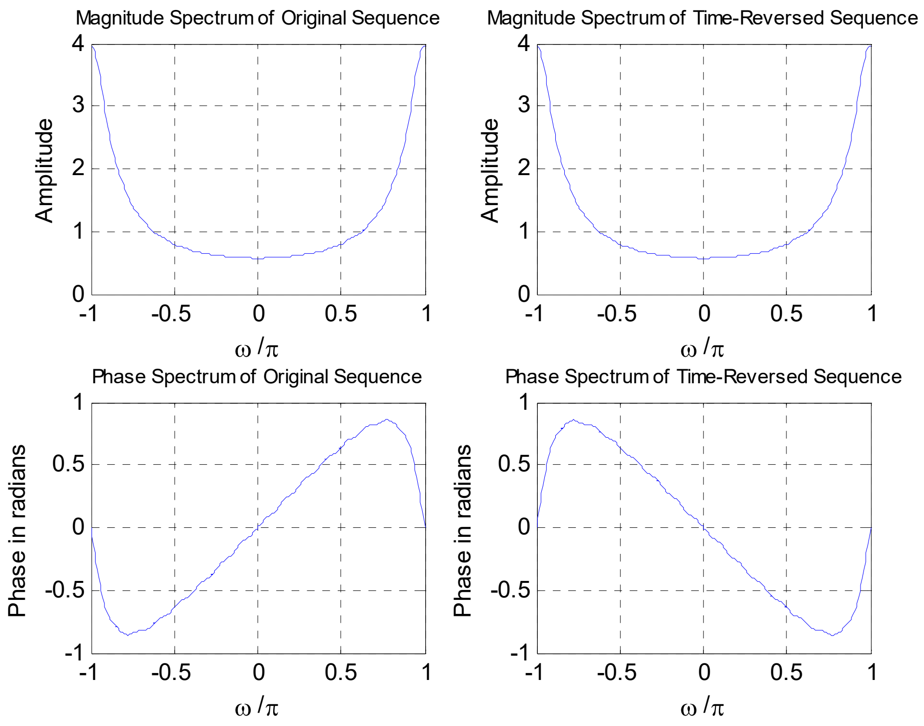
Q3.22 Program P3_6 was run for the following two different sets of sequences of varying lengths –

1. Length of `num` = 8; `num` = $(1/2)^n$ for $0 \le n \le 7$

2. Length of `num` = 16; `num` = $(-3/4)^n$ for $0 \le n \le 15$

The plots generated by running the modified program are given below:





From these plots we make the following observations: As before in Q3.20, we see that the DTFT of the time reversed sequence is a frequency reversed version of the DTFT of the original sequence. In particular, because these are real-valued sequences with even magnitude spectra and odd phase spectra, the magnitude spectra of the original and time reversed sequences are the same. However, the phase spectrum of the time reversed sequence is a frequency reversed version of the phase spectrum of the original sequence.

3.2 DISCRETE FOURIER TRANSFORM

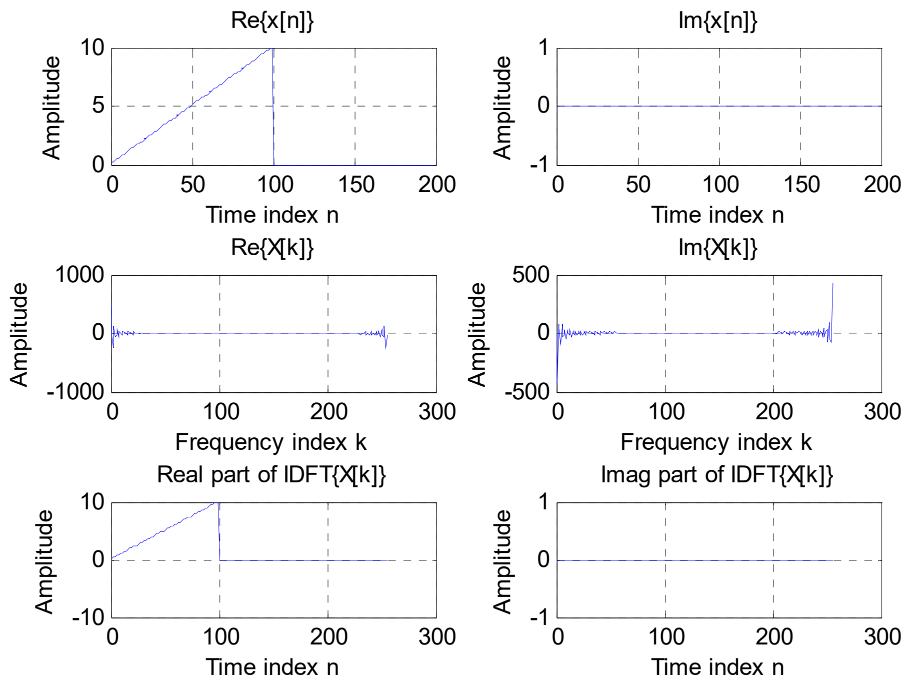
Project 3.3 DFT and IDFT Computations

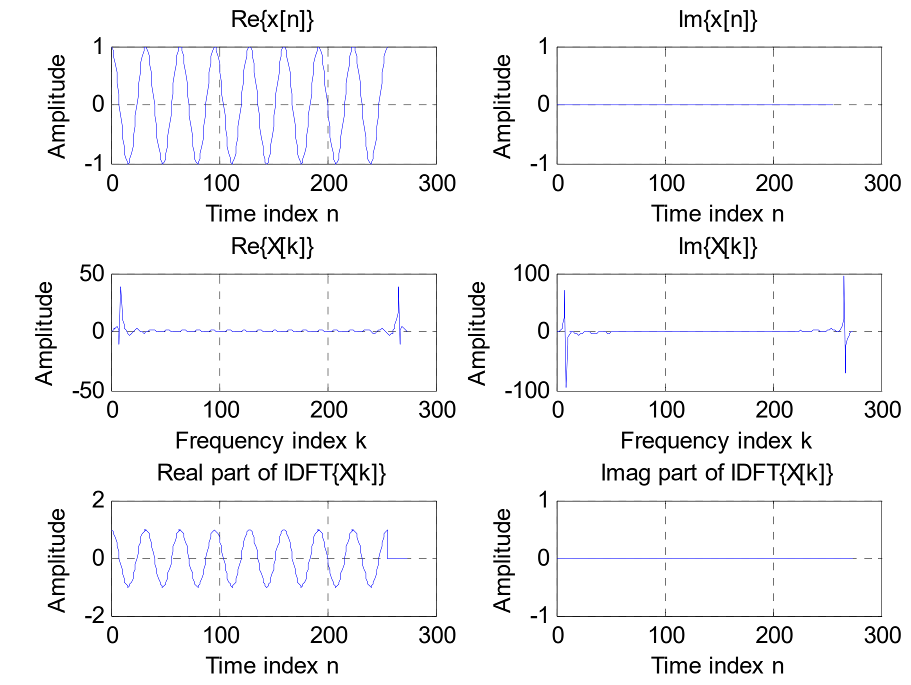
Answers:

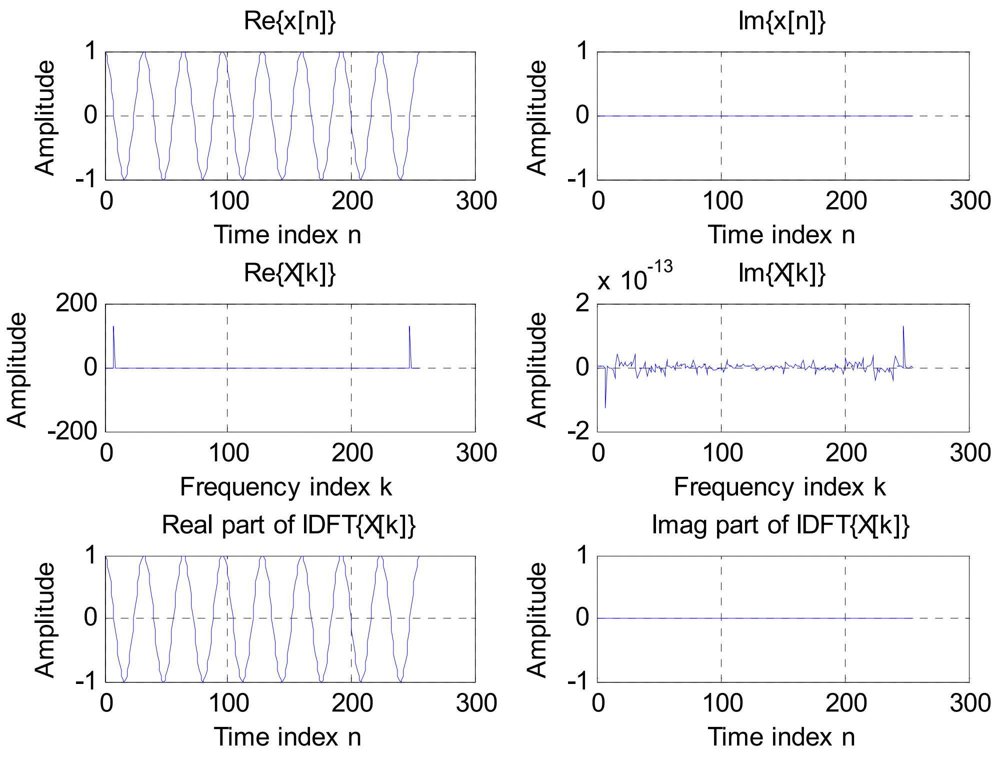
Q3.23 The MATLAB program to compute and plot the L-point DFT X[k] of a length-N sequence x[n] with L $\geq$ N and then to compute and plot the IDFT of X[k] is given below:

```
% Program P3_3DFT
% Compute and plot the L-point DFT of an N-point signal, L >= N.
clf;
%Initialize
N=200;  % length of signal
L=256;  % length of DFT
nn = [0:N-1];
kk = [0:L-1];
% the signal x
xR = [0.1*(1:100) zeros(1,N-100)]; % real part
xI = [zeros(1,N)]; % imag part
x = xR + i*xI;
% DFT
XF = fft(x,L);
% plot xR and xI
subplot(3,2,1);grid;
plot(nn,xR);grid;
title('Re\{x[n]\}');
xlabel('Time index n');
ylabel('Amplitude');
subplot(3,2,2);
plot(nn,xI);grid;
title('Im\{x[n]\}');
xlabel('Time index n');
ylabel('Amplitude');
% plot real and imag parts of DFT
subplot(3,2,3);
plot(kk,real(XF));grid;
title('Re\{X[k]\}');
xlabel('Frequency index k');
ylabel('Amplitude');
subplot(3,2,4);
plot(kk,imag(XF));grid;
title('Im\{X[k]\}');
xlabel('Frequency index k');
ylabel('Amplitude');
% IDFT
xx = ifft(XF,L);
% plot real and imaginary parts of the IDFT
subplot(3,2,5);
plot(kk,real(xx));grid;
title('Real part of IDFT\{X[k]\}');
xlabel('Time index n');
ylabel('Amplitude');
subplot(3,2,6);
plot(kk,imag(xx));grid;
title('Imag part of IDFT\{X[k]\}');
xlabel('Time index n');
ylabel('Amplitude');
```

The DFT and the IDFT pairs generated by running the program for sequences of different lengths `N` and for different values of the DFT length `L` are shown below:







From these plots we make the following observations: In the first example, the signal is a real-valued ramp of length 100 zero padded on the right for a total signal length of N=200. The length of the DFT is L=256. We see that the 256-point DFT is conjugate symmetric as expected. The signal obtained from the IDFT has a length of L=256, but is otherwise identical to the original signal up to roundoff.

The second example is a real-valued cosine with frequency $\omega = \pi/16$ and length N=256. The length of the DFT is L=275. The signal would be a sum of two DFT basis functions if N and L were equal. However, since this is not the case, we see that there are more than two nonzero samples in the DFT. Moreover, since the zero padded L-point version of the signal does not have even symmetry, there are nonzero frequency samples in both the real and imaginary parts of the DFT.

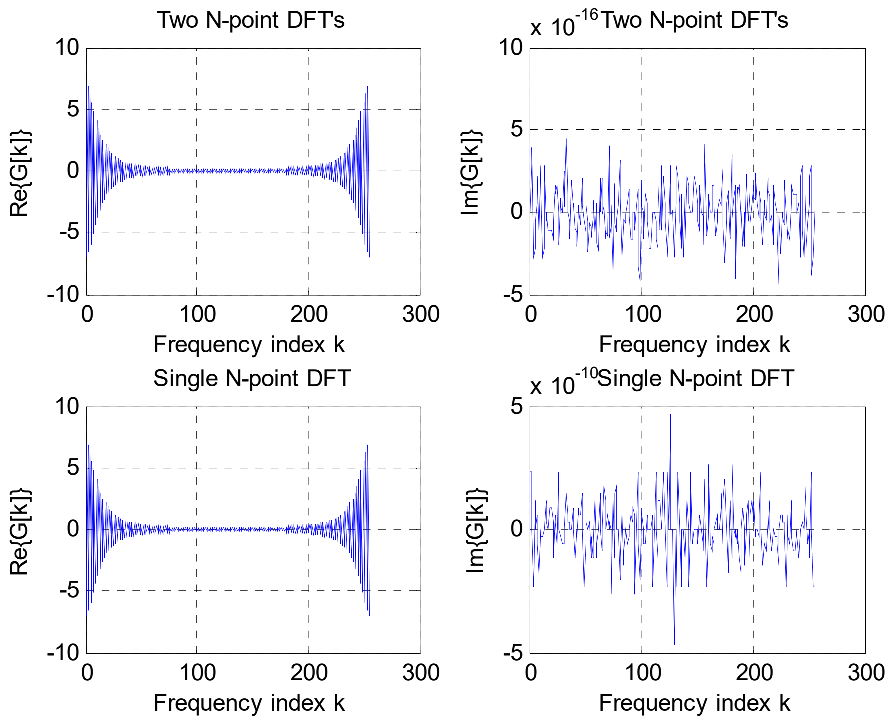
In the third example, the signal is the same as in the second example. However, in this case we have N=L. So there is no zero padding and the signal that is passed to the FFT routine is real and even. Consequently, the DFT is also real and even up to roundoff. Moreover, in this case the signal is a simple sum of two DFT basis functions. Therefore, we see only two nonzero samples in the DFT array (to within roundoff).

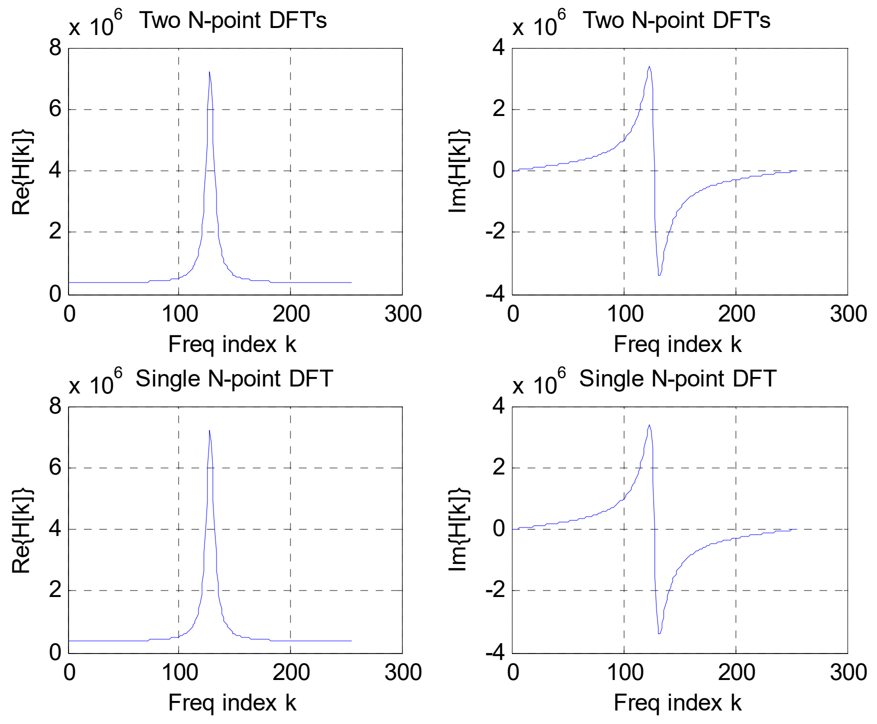
Q3.24 The MATLAB program to compute the N-point DFT of two length-N real sequences using a single N-point DFT and compare the result by computing directly the two N-point DFTs is given below:

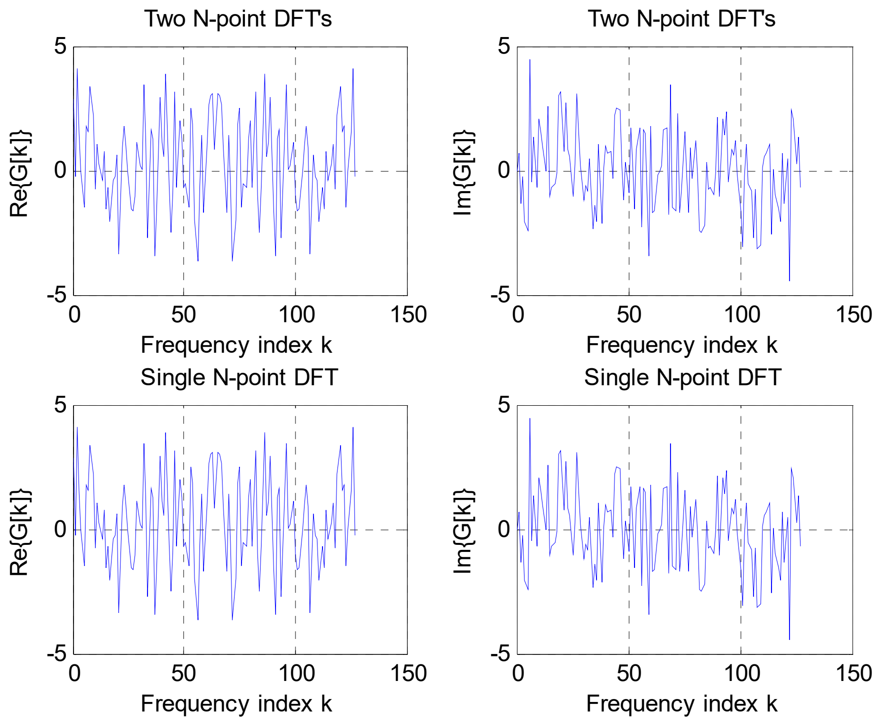
```
% Program Q3_24
% Use a single N-point DFT to compute simultaneously the N-point
%   DFT's of two real-valued N-point sequences.
clf;
%Initialize
N=256;  % length of signal
nn = [0:N-1];
ntime = [-N/2:N/2-1];
g = (0.75).^abs(ntime); % signal g
h = (-0.9).^ntime; % signal h
% DFT's of g and h
GF = fft(g);
HF = fft(h);
x = g + i*h; % the composite signal x
% DFT of composite signal
XF = fft(x);
% DFT of g derived from composite DFT XF
XFstar = conj(XF);
XFstarmod = [XFstar(1) fliplr(XFstar(2:N))];
GF2 = 0.5*(XF + XFstarmod);
HF2 = -i*0.5*(XF - XFstarmod);

abs(max(GF-GF2))
abs(max(HF-HF2))

% plot real and imag parts of direct computation of GF
figure(1);clf;
subplot(2,2,1);grid;
plot(nn,real(GF));grid;
title('Two N-point DFT''s');
xlabel('Frequency index k');
ylabel('Re\{G[k]\}');
subplot(2,2,2);
plot(nn,imag(GF));grid;
title('Two N-point DFT''s');
xlabel('Frequency index k');
ylabel('Im\{G[k]\}');
% plot real and imag parts of composite computation of GF
subplot(2,2,3);grid;
plot(nn,real(GF2));grid;
title('Single N-point DFT');
xlabel('Frequency index k');
ylabel('Re\{G[k]\}');
subplot(2,2,4);
plot(nn,imag(GF2));grid;
title('Single N-point DFT');
xlabel('Frequency index k');
ylabel('Im\{G[k]\}');

% plot real and imag parts of direct computation of HF
figure(2);clf;
```

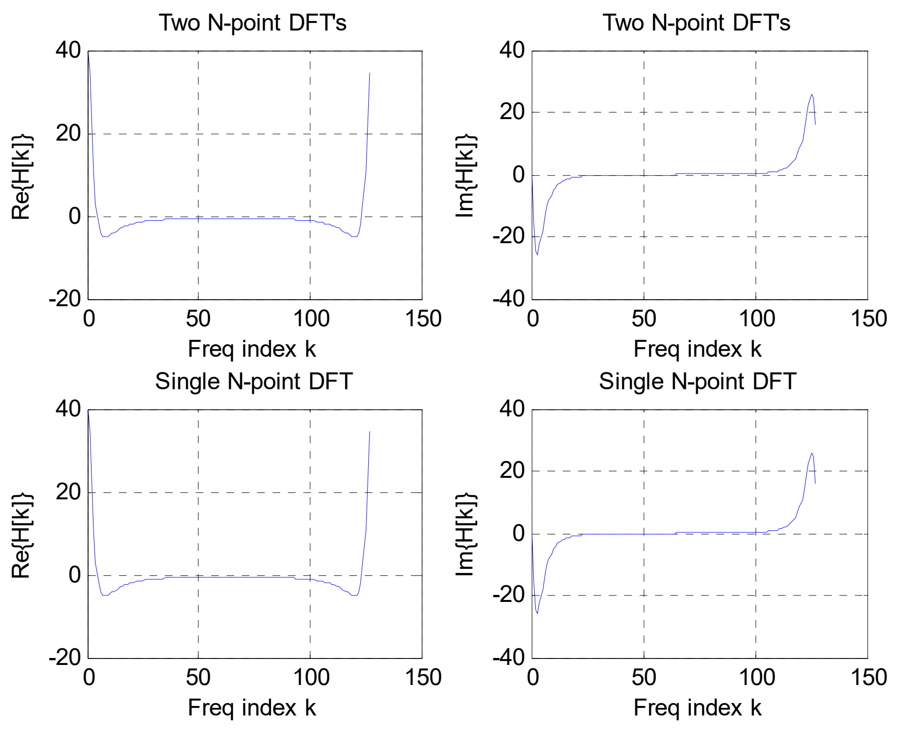
```
subplot(2,2,1);grid;
plot(nn,real(HF));grid;
title('Two N-point DFT''s');
xlabel('Freq index k');
ylabel('Re\{H[k]\}');
subplot(2,2,2);
plot(nn,imag(HF));grid;
title('Two N-point DFT''s');
xlabel('Freq index k');
ylabel('Im\{H[k]\}');
% plot real and imag parts of composite computation of HF
subplot(2,2,3);grid;
plot(nn,real(HF2));grid;
title('Single N-point DFT');
xlabel('Freq index k');
ylabel('Re\{H[k]\}');
subplot(2,2,4);
plot(nn,imag(HF2));grid;
title('Single N-point DFT');
xlabel('Freq index k');
ylabel('Im\{H[k]\}');
```

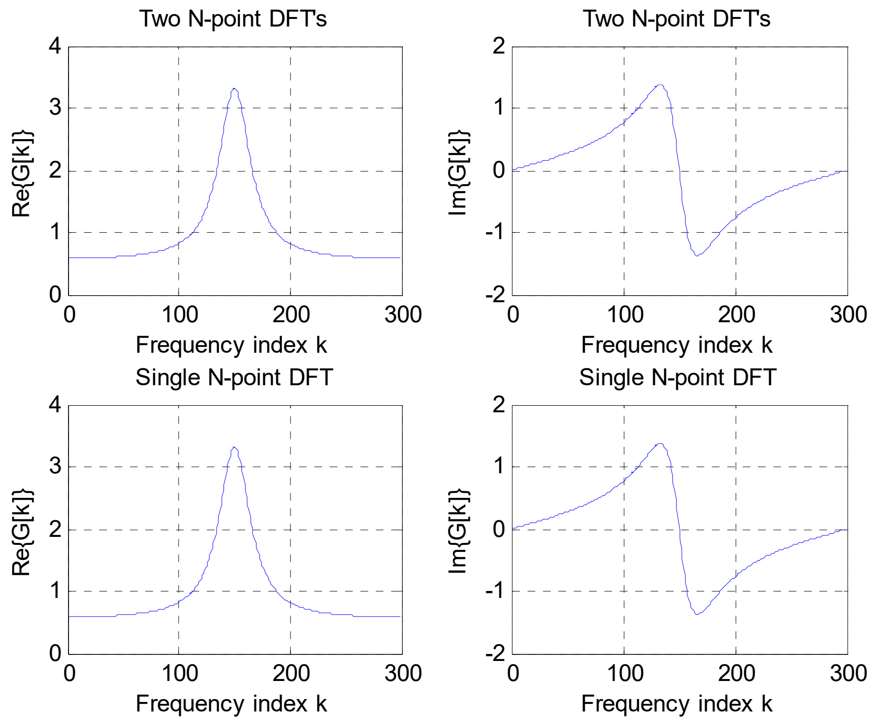
The DFTs generated by running the program for sequences of different lengths N are shown below:

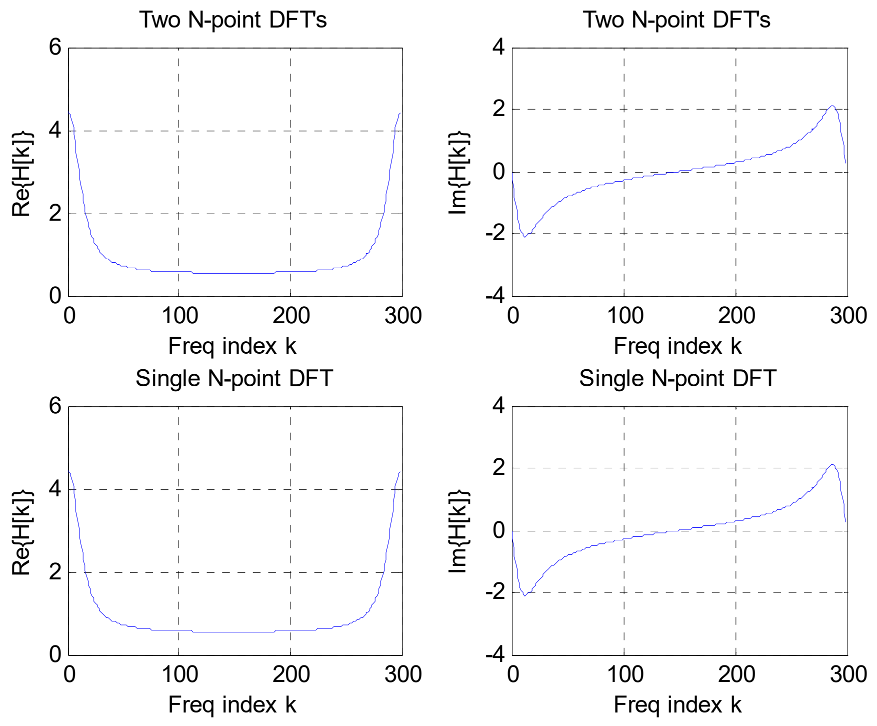



x 10^6 Two N-point DFT's
Re{H[k]}
Freq index k
x 10^6 Two N-point DFT's
Im{H[k]}
Freq index k
x 10^6 Single N-point DFT
Re{H[k]}
Freq index k
x 10^6 Single N-point DFT
Im{H[k]}
Freq index k


Two N-point DFT's
Re{G[k]}
Frequency index k
Two N-point DFT's
Im{G[k]}
Frequency index k
Single N-point DFT
Re{G[k]}
Frequency index k
Single N-point DFT
Im{G[k]}
Frequency index k


Two N-point DFT's
Re{H[k]}
Freq index k
Two N-point DFT's
Im{H[k]}
Freq index k
Single N-point DFT
Re{H[k]}
Freq index k
Single N-point DFT
Im{H[k]}
Freq index k


Two N-point DFT's
Re{G[k]}
Frequency index k
Two N-point DFT's
Im{G[k]}
Frequency index k
Single N-point DFT
Re{G[k]}
Frequency index k
Single N-point DFT
Im{G[k]}
Frequency index k



From these plots we make the following observations: In the first example, the length is N=256. The sequence g[n] is real and even and the sequence h[n] is real but has no inherent symmetry. The DFT G[k] is real and even to within numerical roundoff; it is important to realize in the plots that Im{G[k]} is zero to within floating point precision.

In the second example, the length is N=128. The sequence g[n] is a sample function of an IID stochastic process uniformly distributed between +0.8 and -0.8. The sequence h[n] is the product of a deterministic ramp and a deterministic decaying exponential.

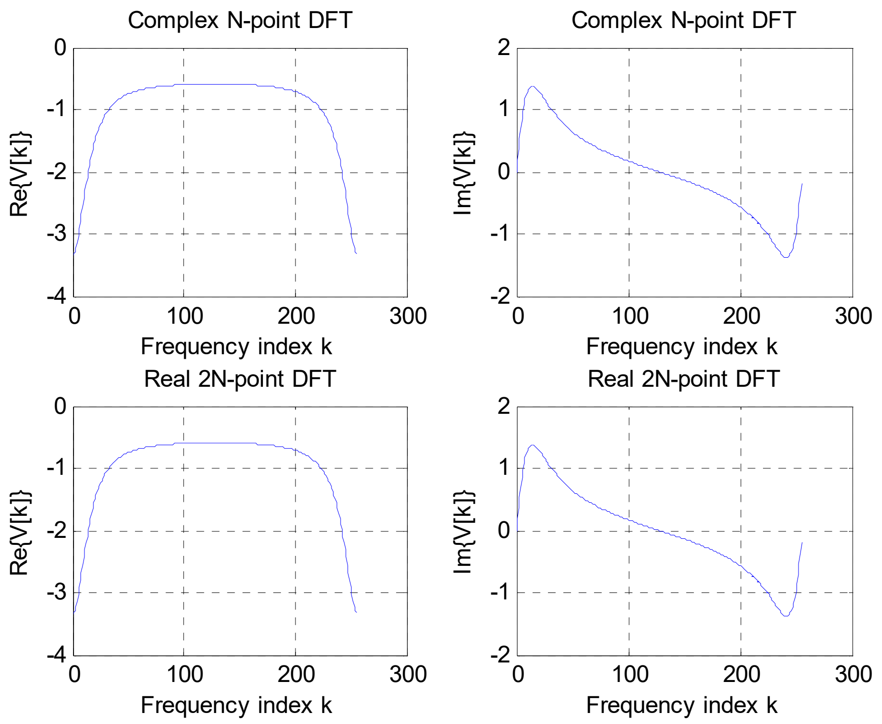
In the third example, the length is N=300. The sequence g[n] is an alternating and decaying exponential, while sequence h[n] is the product of a monotonically decaying exponential and a cosine.

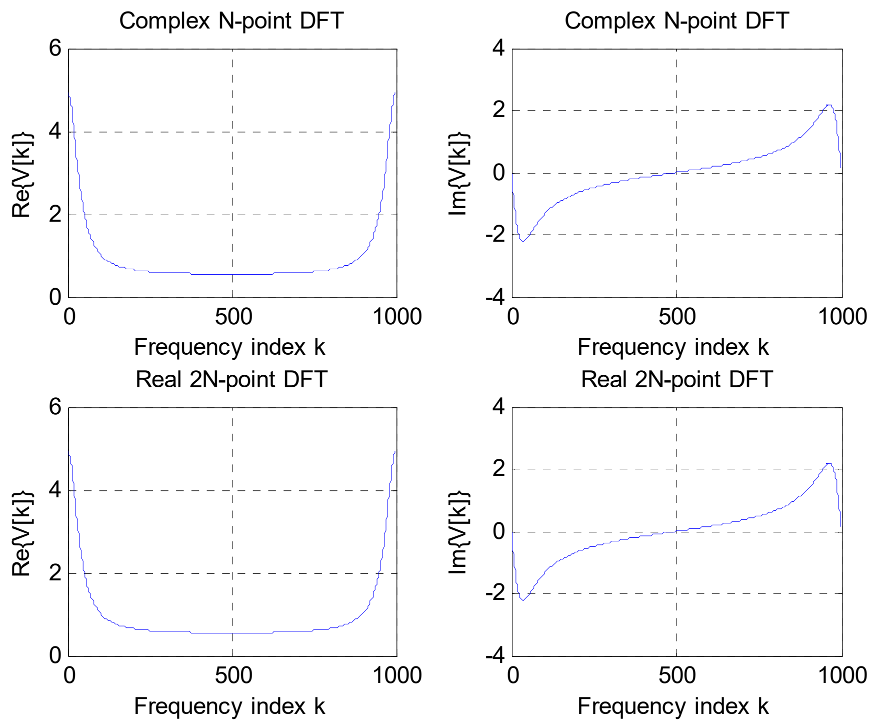
In all three of these examples, we see that the results obtained by computing G[k] and H[k] with a single N-point DFT or directly with two N-point DFT's are identical up to numerical roundoff.

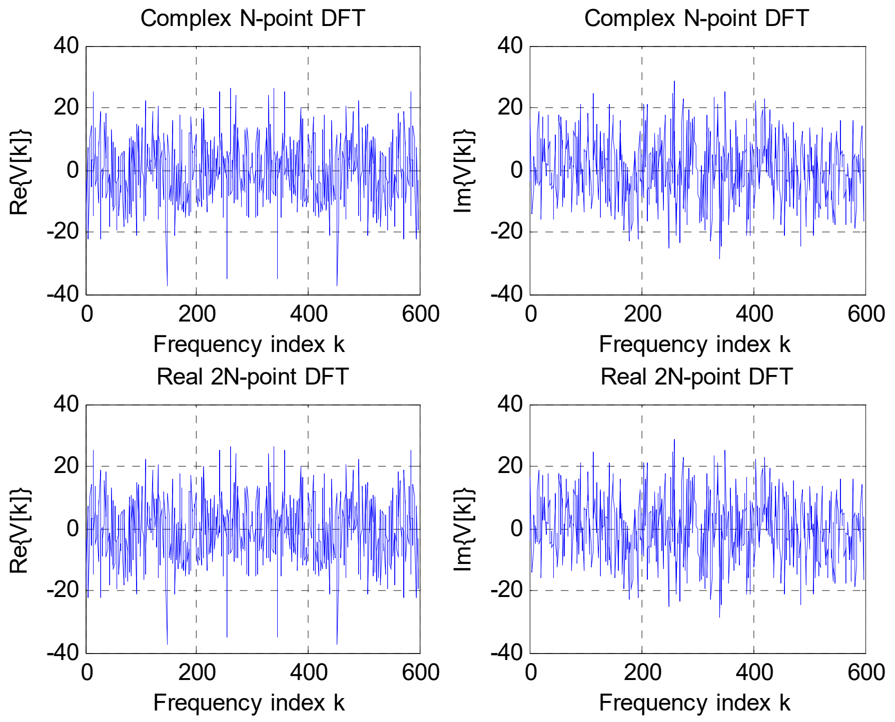
Q3.25 The MATLAB program to compute the $2N$-point DFT of a length-$2N$ real sequence using a single N-point DFT and compare the result by computing directly the $2N$-point DFT is shown below:

```
% Program Q3_25A
% Use a single N-point complex-valued DFT to compute the 2N-point
%   DFT of a 2N-point real-valued sequence.
%
clf;
%Initialize constants
N = 128;           % length of the complex-valued DFT
TwoN = 2*N;        % length of the real-valued sequence
W2N = exp(-i*pi/N);
k = [0:TwoN-1];
% create 2N-point signal v[n]
v = (-0.7.^k);
% create two N-point signals
g = downsample(v,2);    % g[n] = v[2n]
h = downsample(v,2,1); % h[n] = v[2n+1]
% N-point complex-valued composite signal x[n]
x = g + i*h;
% Use one N-point complex DFT to compute simultaneously
%   G[k] and H[k] as in Q3.24.
XF = fft(x);     % N-point complex DFT of x[n]
% DFT's G[k] and H[k] derived from composite DFT X[k]
XFstar = conj(XF);
XFstarmod = [XFstar(1) fliplr(XFstar(2:N))];
GF = 0.5*(XF + XFstarmod);
HF = -i*0.5*(XF - XFstarmod);
% 2N-point DFT V[k]
VF = [GF GF] + (W2N.^k).*[HF HF];
% For Comparison, compute directly the 2N-point DFT V[k]
VF2 = fft(v);
% Print Sanity Check
abs(max(VF-VF2))
% plot real and imag parts of V[k] computed by complex N-point DFT
subplot(2,2,1);
plot(k,real(VF));grid;
title('Complex N-point DFT');
xlabel('Frequency index k');
ylabel('Re\{V[k]\}');
subplot(2,2,2);
plot(k,imag(VF));grid;
title('Complex N-point DFT');
xlabel('Frequency index k');
ylabel('Im\{V[k]\}');
% plot real and imag parts of V[k] computed by 2N-point DFT
subplot(2,2,3);
plot(k,real(VF2));grid;
title('Real 2N-point DFT');
xlabel('Frequency index k');
ylabel('Re\{V[k]\}');
subplot(2,2,4);
plot(k,imag(VF2));grid;
title('Real 2N-point DFT');
xlabel('Frequency index k');
ylabel('Im\{V[k]\}');
```

The DFTs generated by running the program for sequences of different lengths $2N$ are shown below:







From these plots we make the following observations: In the first example, the length of `v[n]` is 256 and `V[k]` is computed using a complex valued 128-point DFT. The signal is $v[n] = (-0.7)^n$. In the second example, the length of `v[n]` is 1000 and `V[k]` is computed using a complex valued 500-point DFT. The signal is $v[n] = 0.8^n \cos(\pi n/25)$. In the third example, the length of `v[n]` is 600 and `V[k]` is computed using a complex valued 600-point DFT. In each case, the results obtained by the 2*N*-point real DFT and by the *N*-point complex DFT are identical up to numerical roundoff.

Project 3.4 DFT Properties

Answers:

Q3.26 The purpose of the command `rem` in the function `circshift` is – `rem(x,y)` is the remainder after `x` is divided by `y`.

Q3.27 The function `circshift` operates as follows: The input sequence `x` is circularly shifted left by `M` positions. If `M` > 0, then `circshift` removes the leftmost `M` elements from the vector `x` and appends them on the right side of the remaining elements to obtain the circularly shifted sequence. If If `M` < 0, then `circshift` first complements M by the length of x, i.e., the rightmost length(x)-M samples are removed from x and appended on the right of the remaining M samples to obtain the circularly shifted sequence.

Q3.28 The purpose of the operator ~= in the function `circonv` is – This is the binary relational NOT EQUAL operator. A ~= B returns the value 1 if A and B are unequal and the value 0 if A and B are equal.

Q3.29 The function `circonv` operates as follows: The input is two vectors x1 and x2 of equal length L. To understand how circonv works, it is useful to think in terms of the periodic extension of x2. Let x2p be the infinite-length periodic extension of x2. Conceptually, the routine time reverses x2p and sets x2tr equal to elements 1 through L of the time reversed version of x2p. Elements 1 through L of the output vector y are then obtained by taking the inner product between x1 and a length L vector sh obtained by circularly shifting right the time reversed vector x2tr. For the output sample y[n], $1 \leq n \leq L$, the amount of the right circular shift is n-1 positions.

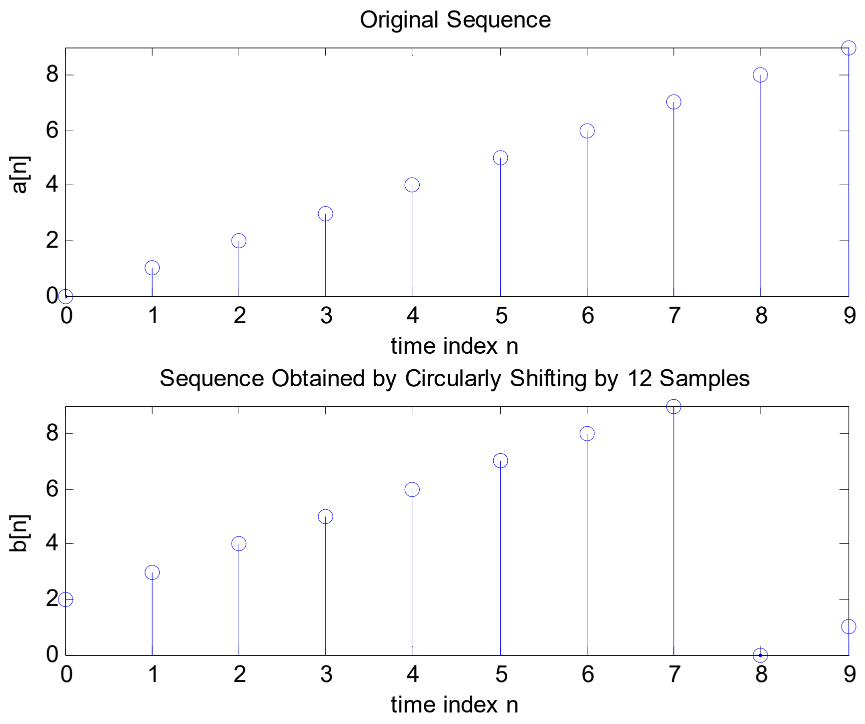
Q3.30 The modified Program P3_7 created by adding appropriate comment statements, and adding program statements for labeling each plot being generated by the program is given below:

```
% Program P3_7B
% Illustration of Circular Shift of a Sequence
clf;
% initialize shift amount M
M = 6;
% initialize sequence a to be shifted
a = [0 1 2 3 4 5 6 7 8 9];
b = circshift(a,M); % perform the circular shift
L = length(a)-1;
% plot the original sequence a and the circularly shifted sequence b
n = 0:L;
subplot(2,1,1);
stem(n,a);axis([0,L,min(a),max(a)]);
title('Original Sequence');
xlabel('time index n');
ylabel('a[n]');
subplot(2,1,2);
stem(n,b);axis([0,L,min(a),max(a)]);
title(['Sequence Obtained by Circularly Shifting by ',num2str(M),'
Samples']);
xlabel('time index n');
ylabel('b[n]');
```

The parameter determining the amount of time-shifting is - M

If the amount of time-shift is greater than the sequence length then – The circular shift actually implemented is rem(M,length(a)) positions left, which is equivalent to circularly shifting by M positions (more than once around) and also to shifting left by M the periodic extension of the sequence.

Q3.31 The plots generated by running the modified program are given below:



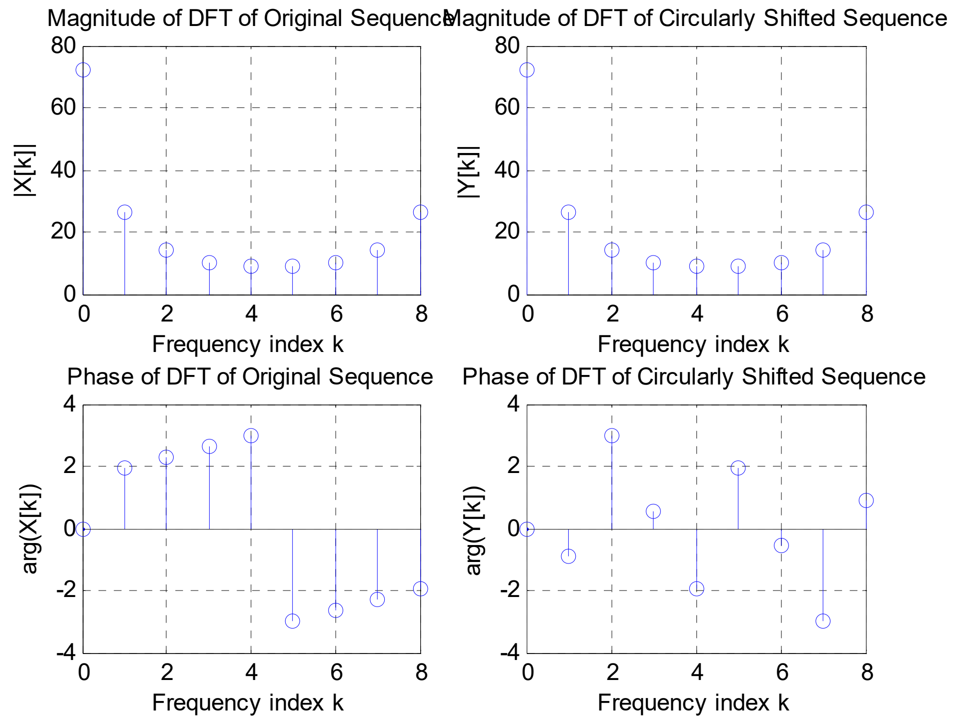
From these plots we make the following observations: Here, the length of the sequence is 10 samples and we have M=12. This may be interpreted alternatively as a circular shift left by 12 positions (more than once around), as a circular shift left by 12-10 = 2 positions, or as a linear shift left by 2 or by 12 of the periodic extension of the sequence.

Q3.32 The modified Program P3_8 created by adding appropriate comment statements, and adding program statements for labeling each plot being generated by the program is given below:

```
% Program P3_8B
% Circular Time-Shifting Property of DFT
clf;
x = [0 2 4 6 8 10 12 14 16]; % original sequence x
N = length(x)-1; n = 0:N;     % time index vector
% set y equal to the circular shift left of x
y = circshift(x,5);
XF = fft(x);           % DFT of x
YF = fft(y);           % DFT of y
subplot(2,2,1);
% plot the spectral magnitudes of the original and shifted sequences
stem(n,abs(XF));grid;
title('Magnitude of DFT of Original Sequence');
xlabel('Frequency index k');
ylabel('|X[k]|');
subplot(2,2,2);
stem(n,abs(YF));grid;
title('Magnitude of DFT of Circularly Shifted Sequence');
xlabel('Frequency index k');
ylabel('|Y[k]|');
% plot the spectral phases of the original and shifted sequences
subplot(2,2,3);
stem(n,angle(XF));grid;
title('Phase of DFT of Original Sequence');
xlabel('Frequency index k');
ylabel('arg(X[k])');
subplot(2,2,4);
stem(n,angle(YF));grid;
title('Phase of DFT of Circularly Shifted Sequence');
xlabel('Frequency index k');
ylabel('arg(Y[k])');
```

The amount of time-shift is - hard coded in this program at 5 samples to the left.

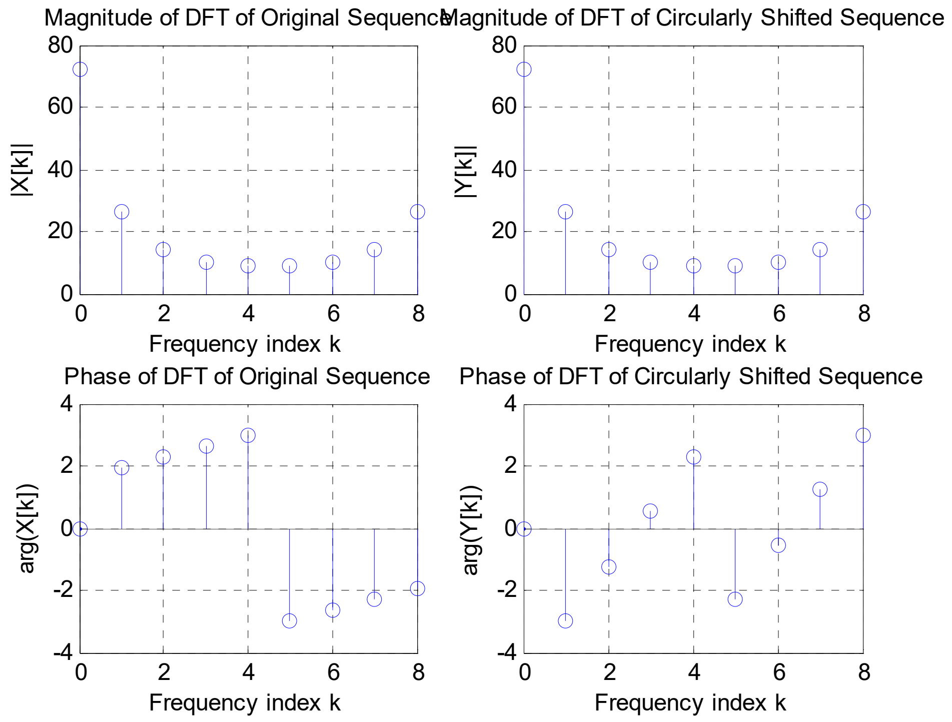
Q3.33 The plots generated by running the modified program are given below:



From these plots we make the following observations: The length of the sequence is N=8 and the time shift is an advance by five samples to the left. The phase term introduced by this time shift is $W_N^{kn_0} = W_N^{-k5} = e^{jk10\pi/8} = e^{jk5\pi/4}$. This is a substantial shift that dramatically increases the slope of the spectral phase. Whereas the original phase function has only one branch cut, there are five branch cuts in the spectral phase of the shifted signal.

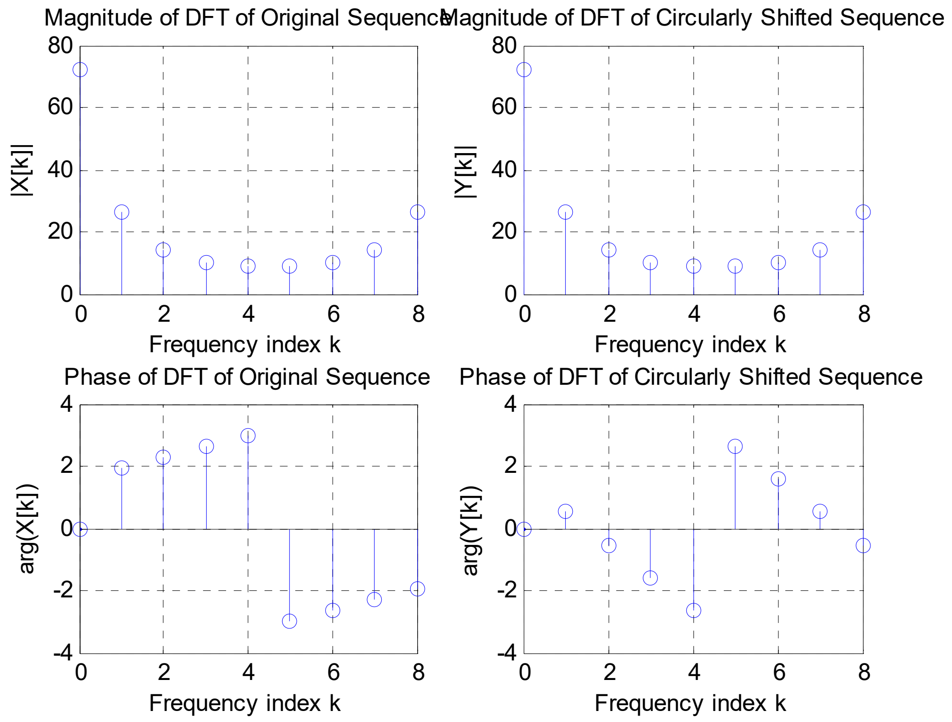
Q3.34 The plots generated by running the modified program for the following two different amounts of time-shifts, with the amount of shift indicated, are shown below:

M=2



Magnitude of DFT of Original Sequence
Magnitude of DFT of Circularly Shifted Sequence
|X[k]|
|Y[k]|
Frequency index k
Phase of DFT of Original Sequence
Phase of DFT of Circularly Shifted Sequence
arg(X[k])
arg(Y[k])
Frequency index k

M=-2

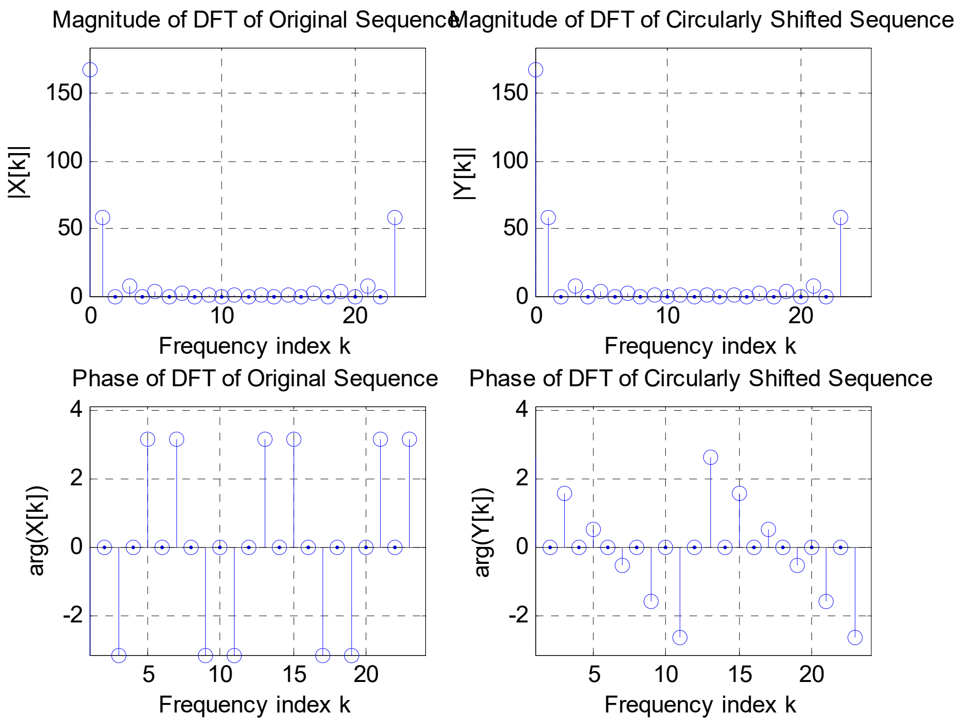


Magnitude of DFT of Original Sequence
Magnitude of DFT of Circularly Shifted Sequence
|X[k]|
|Y[k]|
Frequency index k
Phase of DFT of Original Sequence
Phase of DFT of Circularly Shifted Sequence
arg(X[k])
arg(Y[k])
Frequency index k

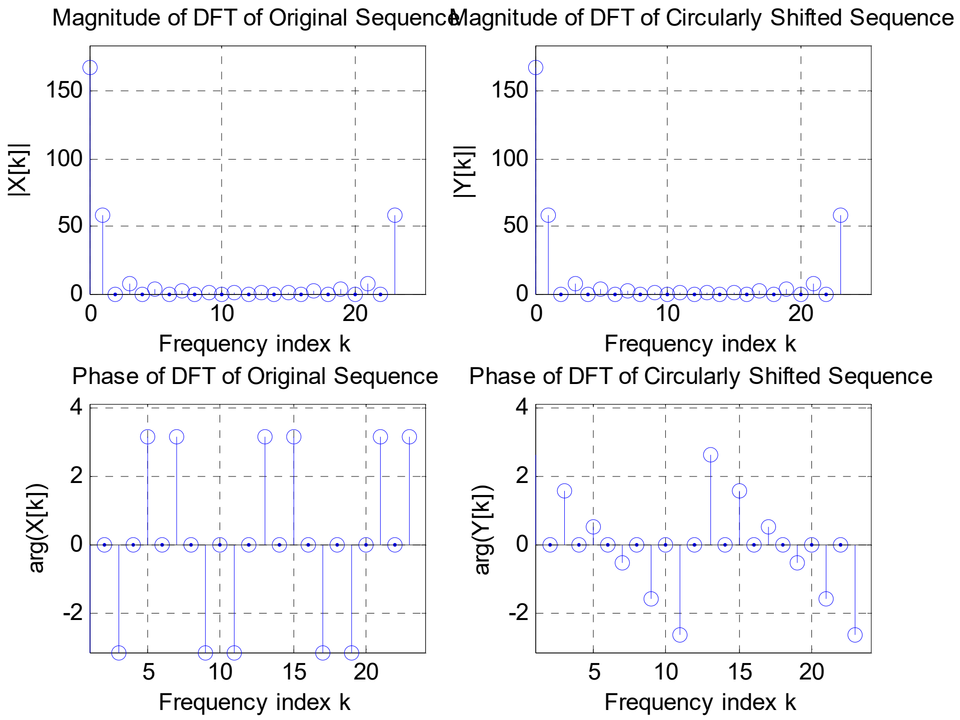
From these plots we make the following observations: In all cases, the spectral magnitude is not affected by the shift. For the first example, the time shift is a circular shift left by 2 samples. This introduces an increased slope to the spectral phase that is significantly less than what we saw in Q3.33. In the second example, the shift is circular shift right by 2 samples (M=-2). This cancels the positive slope seen in the spectral phase of the original sequences and results in a moderate negative slope.

Q3.35 The plots generated by running the modified program for the following two different sequences of different lengths, with the lengths indicated, are shown below:

Length = 24



Length = 16



From these plots we make the following observations: In the first example, the sequence is real and periodically (circularly) even, so the phase takes only the two values zero and π. The shift is a circular shift to the right by 2 (M=-2), which is seen to induce a negative slope to the phase. In the second example, the signal is given by $(-0.75)^n$ and the shift is again M=-2 which again introduces a negative slope in the phase.

Q3.36 A copy of Program P3_9 is given below along with the plots generated by running this program:

```
% Program P3_9
% Circular Convolution Property of DFT
g1 = [1 2 3 4 5 6]; g2 = [1 -2 3 3 -2 1];
ycir = circonv(g1,g2);
disp('Result of circular convolution = ');disp(ycir)
G1 = fft(g1); G2 = fft(g2);
yc = real(ifft(G1.*G2));
disp('Result of IDFT of the DFT products = ');disp(yc)
```

```
Result of circular convolution =
    12    28    14     0    16    14

Result of IDFT of the DFT products =
    12    28    14     0    16    14
```

From these plots we make the following observations: The DFT of a circular convolution is the pointwise products of the DFT's.

Q3.37 Program P3_9 was run again for the following two different sets of equal-length sequences:

The plots generated are shown below:

```
Result of circular convolution =

     9     2    -7     6     9     4

Result of IDFT of the DFT products =

    9.0000    2.0000   -7.0000    6.0000    9.0000    4.0000
```

From these plots we make the following observations: The circular convolution property of the DFT seems to hold.

Q3.38 A copy of Program P3_10 is given below along with the plots generated by running this program:

```
% Program P3_10
% Linear Convolution via Circular Convolution
g1 = [1 2 3 4 5];g2 = [2 2 0 1 1];
g1e = [g1 zeros(1,length(g2)-1)];
g2e = [g2 zeros(1,length(g1)-1)];
ylin = circconv(g1e,g2e);
disp('Linear convolution via circular convolution = ');disp(ylin);
y = conv(g1, g2);
disp('Direct linear convolution = ');disp(y)
```

```
Linear convolution via circular convolution =

     2     6    10    15    21    15     7     9     5

Direct linear convolution =

     2     6    10    15    21    15     7     9     5
```

From these plots we make the following observations: zero padding to the appropriate length does indeed make it possible to implement linear convolution using circular convolution.

Q3.39 Program P3_10 was run again for the following two different sets of sequences of unequal lengths:

```
g1 = [3 1 4 1 5 9 2];

g2 = [1 1 1 0 0];
```

g1 = [5 4 3 2 1 0];

g2 = [-2 1 2 3 4];

The plots generated are shown below:

```
Linear convolution via circular convolution =
 3    4    8    6   10   15   16   11    2    0    0

Direct linear convolution =
 3    4    8    6   10   15   16   11    2    0    0


Linear convolution via circular convolution =
 -10   -3    8   22   38   30   20   11    4    0

Direct linear convolution =
 -10   -3    8   22   38   30   20   11    4    0
```

From these plots we make the following observations: You can implement the linear convolution of two sequences by zero padding them to the sum of their lengths less one and then invoking circular convolution on the zero padded sequences.

Q3.40 The MATLAB program to develop the linear convolution of two sequences via the DFT of each is given below:

```
% Program Q3_40
% Linear Convolution via Circular Convolution
g1 = [1 2 3 4 5];
g2 = [2 2 0 1 1];
g1e = [g1 zeros(1,length(g2)-1)];
g2e = [g2 zeros(1,length(g1)-1)];
G1EF = fft(g1e);
G2EF = fft(g2e);
ylin = real(ifft(G1EF.*G2EF));
disp('Linear convolution via DFT = ');disp(ylin);
```

The plots generated by running this program for the sequences of Q3.38 are shown below:

```
Linear convolution via DFT =

    2.0000    6.0000   10.0000   15.0000   21.0000   15.0000    7.0000    9.0000
```

From these plots we make the following observations: The result is the same as before in Q3.38; in other words, it works as advertised.

The plots generated by running this program for the sequences of Q3.39 are shown below:

```
Linear convolution via DFT =

Columns 1 through 9

3.0000    4.0000    8.0000    6.0000   10.0000   15.0000   16.0000   11.0000    2.0000

Columns 10 through 11

0.0000    0.0000


Linear convolution via DFT =

Columns 1 through 9

 -10.0000   -3.0000    8.0000   22.0000   38.0000   30.0000   20.0000   11.0000    4.0000

Column 10

  -0.0000
```

From these plots we make the following observations: The results are the same as those that were obtained before when the DFT was not used.

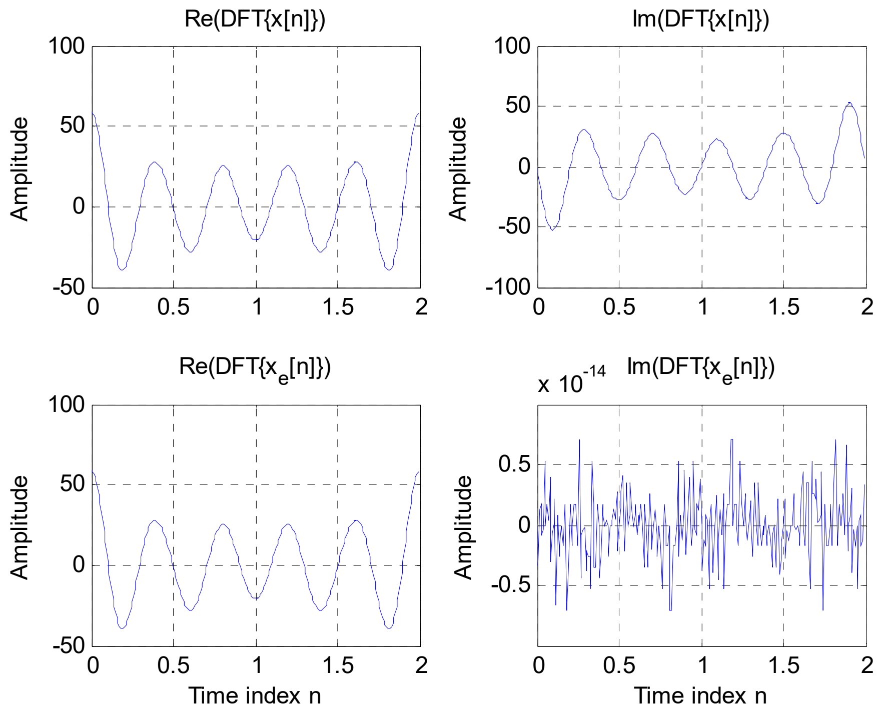
Q3.41 A copy of Program P3_11 is given below:

```
% Program P3_11
% Relations between the DFTs of the Periodic Even
% and Odd Parts of a Real Sequence
x = [1 2 4 2 6 32 6 4 2 zeros(1,247)];
x1 = [x(1) x(256:-1:2)];
xe = 0.5 *(x + x1);
XF = fft(x);
XEF = fft(xe);
clf;
k = 0:255;
subplot(2,2,1);
plot(k/128,real(XF)); grid;
ylabel('Amplitude');
title('Re(DFT\{x[n]\})');
subplot(2,2,2);
plot(k/128,imag(XF)); grid;
ylabel('Amplitude');
title('Im(DFT\{x[n]\})');
subplot(2,2,3);
plot(k/128,real(XEF)); grid;
xlabel('Time index n');ylabel('Amplitude');
title('Re(DFT\{x_{e}[n]\})');
subplot(2,2,4);
plot(k/128,imag(XEF)); grid;
```

```
xlabel('Time index n');ylabel('Amplitude');
title('Im(DFT\{x_{e}[n]\})');
```

The relation between the sequence `x1[n]` and `x[n]` is – x1[n] is a periodically time reversed version of x[n].

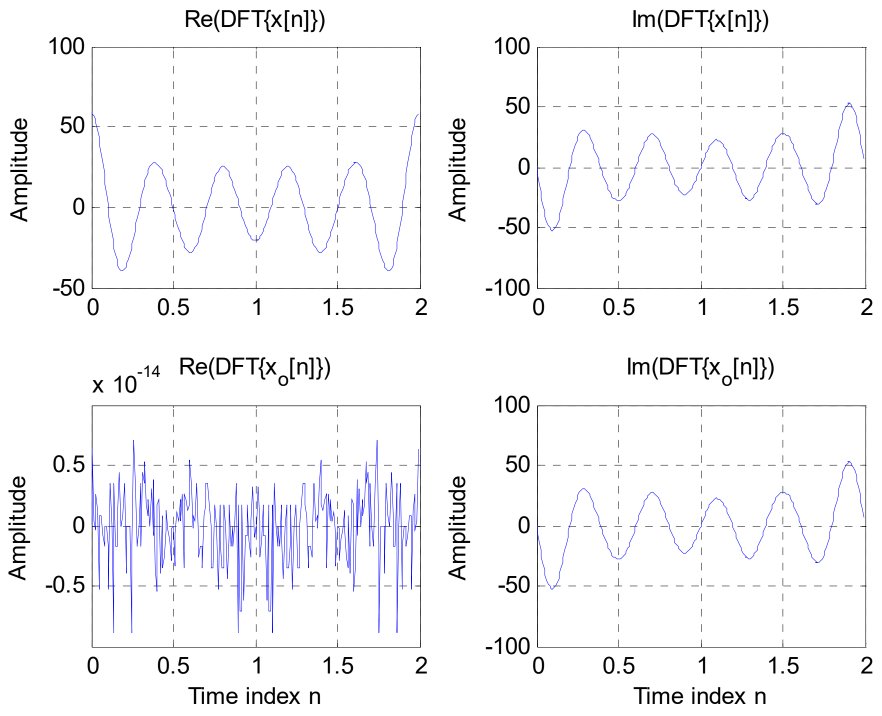
Q3.42 The plots generated by running Program P3_11 are given below:



The imaginary part of XEF is equal to zero to within floating point precision. This result can be explained as follows: The real part of the transform of x[n] is the transform of the periodically even part of x[n]. Therefore, the DFT of the periodically even part of x[n] has a real part that is precisely the real part of X[k] and an imaginary part that is zero.

Q3.43 The required modifications to Program P3_11 to verify the relation between the DFT of the periodic odd part and the imaginary part of XEF are given below along with the plots generated by running this program:

```
% Program P3_11B
% Relations between the DFTs of the Periodic Even
% and Odd Parts of a Real Sequence
x = [1 2 4 2 6 32 6 4 2 zeros(1,247)];
x1 = [x(1) x(256:-1:2)];
xo = 0.5 *(x - x1);
XF = fft(x);
XOF = fft(xo);
clf;
k = 0:255;
subplot(2,2,1);
plot(k/128,real(XF)); grid;
ylabel('Amplitude');
title('Re(DFT\{x[n]\})');
subplot(2,2,2);
plot(k/128,imag(XF)); grid;
ylabel('Amplitude');
title('Im(DFT\{x[n]\})');
subplot(2,2,3);
plot(k/128,real(XOF)); grid;
xlabel('Time index n');ylabel('Amplitude');
title('Re(DFT\{x_{o}[n]\})');
subplot(2,2,4);
plot(k/128,imag(XOF)); grid;
xlabel('Time index n');ylabel('Amplitude');
title('Im(DFT\{x_{o}[n]\})');
```



From these plots we make the following observations: The DFT of the periodically odd part of x[n] is precisely the imaginary part of the DFT of x[n]. Therefore, the DFT of the periodically odd part of x[n] has a real part that is zero to within floating point precision and an imaginary part that is precisely the imaginary part of the DFT of x[n].

Q3.44 A copy of Program P3_12 is given below:

```
% Program P3_12
% Parseval's Relation
x = [(1:128) (128:-1:1)];
XF = fft(x);
a = sum(x.*x)
b = round(sum(abs(XF).^2)/256)
```

The values for `a` and `b` we get by running this program are –

a =1414528

b =1414528

Q3.45 The required modifications to Program P3_11 are given below:

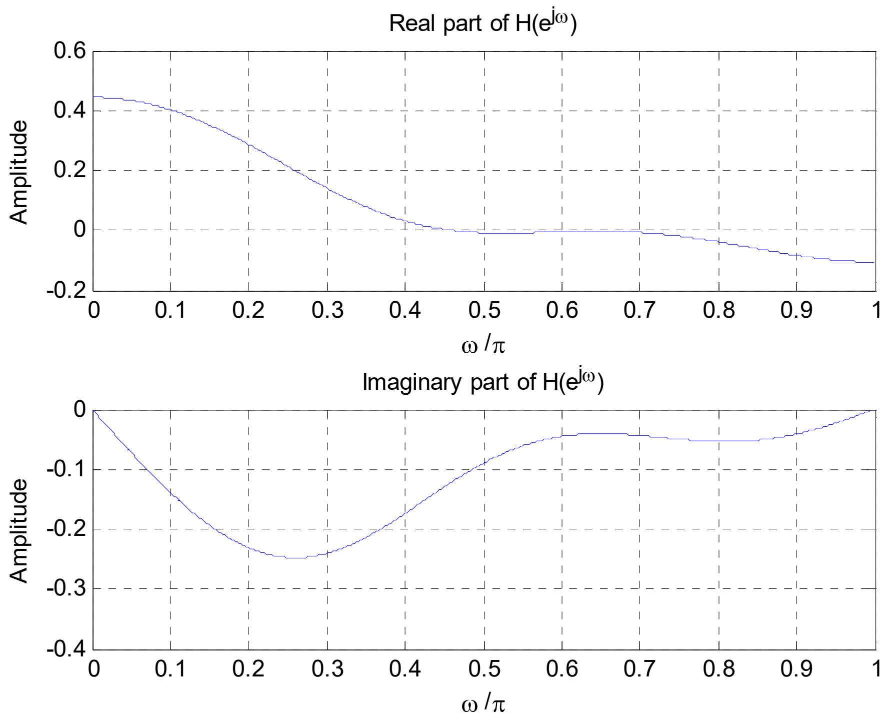
```
% Program P3_12B
% Parseval's Relation
x = [(1:128) (128:-1:1)];
XF = fft(x);
a = sum(x.*x)
b = round(sum(XF.*conj(XF))/256)
```

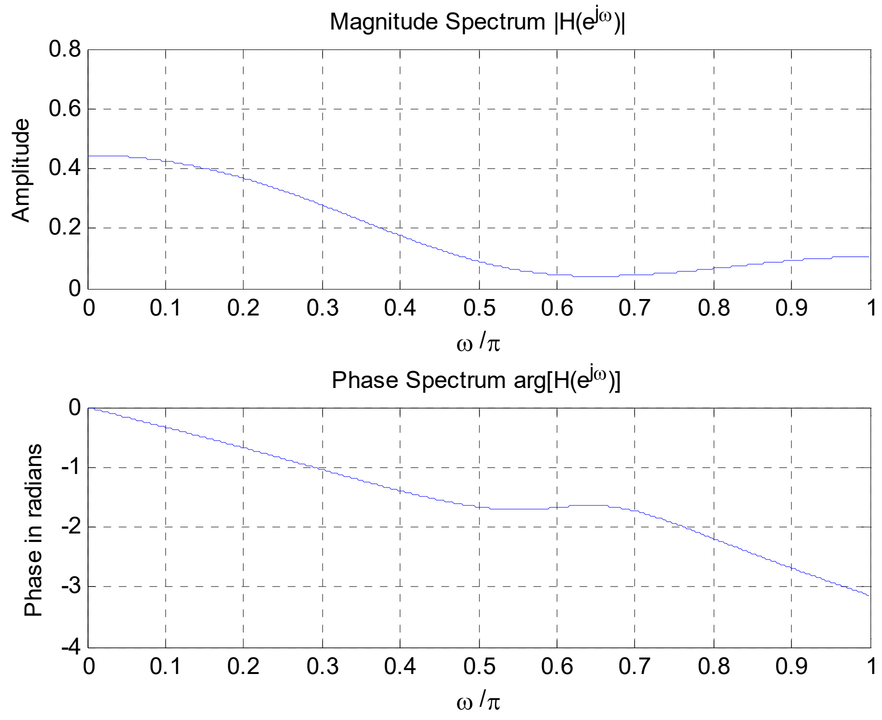
3.3 z-TRANSFORM

Project 3.5 Analysis of z-Transforms

Answers:

Q3.46 The frequency response of the z-transform obtained using Program P3_1 is plotted below:





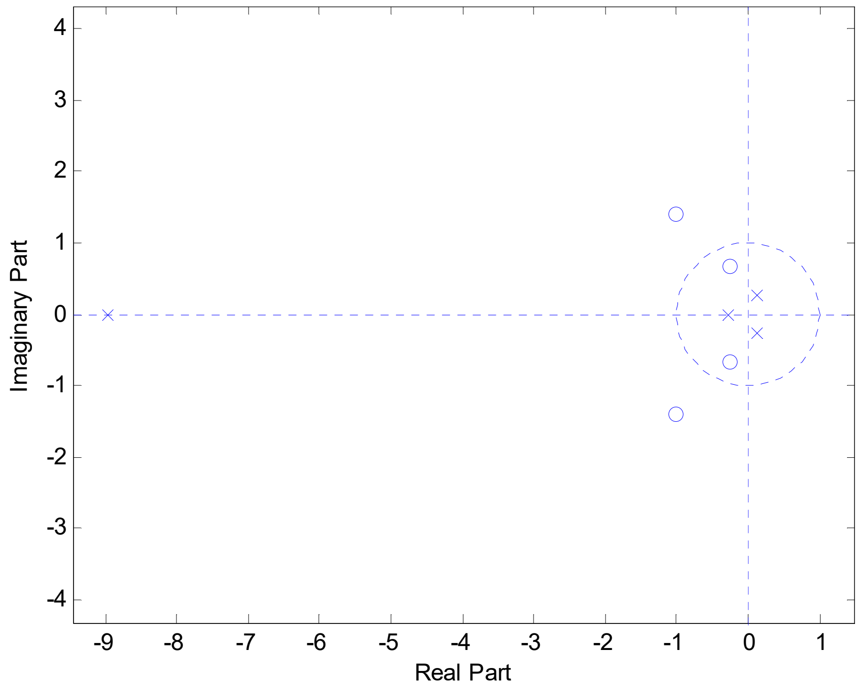
Q3.47 The MATLAB program to compute and display the poles and zeros, to compute and display the factored form, and to generate the pole-zero plot of a rational z-transform is given below:

```
% Program Q3_47
% Given numerator and denominator coefficient vectors for G(z),
%   - compute and display poles and zeros
%   - compute and display factored form of G(z)
%   - generate pole-zero plot
% NOTE: the lab book says to use tf2zp.  For a rational function
%   in z^-1, it's better to use tf2zpk.
clf;
%  initialize
num = [2  5 9 5 3];
den = [5 45 2 1 1];
% compute poles and zeros and display
[z p k] = tf2zpk(num,den);
disp('Zeros:');
disp(z);
disp('Poles:');
disp(p);
input('Hit <return> to continue...');
% compute and display factored form of G(z)
[sos k] = zp2sos(z,p,k)
input('Hit <return> to continue...');
% generate pole-zero plot
zplane(z,p);
```

Using this program we obtain the following results on the z-transform `G(z)` of Q3.46:

```
Zeros:
  -1.0000 + 1.4142i
  -1.0000 - 1.4142i
  -0.2500 + 0.6614i
  -0.2500 - 0.6614i


Poles:
  -8.9576
  -0.2718
   0.1147 + 0.2627i
   0.1147 - 0.2627i

sos =
    1.0000    2.0000    3.0000    1.0000    9.2293    2.4344
    1.0000    0.5000    0.5000    1.0000   -0.2293    0.0822


k =
    0.4000
```

$$G(z) = 0.4\frac{1+2z^{-1}+3z^{-2}}{1+9.2293z^{-1}+2.4344z^{-2}}\frac{1+0.5z^{-1}+0.5z^{-2}}{1-0.2293z^{-1}+0.0822z^{-2}}$$



Q3.48 From the pole-zero plot generated in Question Q3.47, the number of regions of convergence (ROC) of `G(z)` are - FOUR. note: the magnitude of the complex conjugate poles inside the unit circle is 0.2866.

All possible ROCs of this z-transform are sketched below:

$R_1 :\ |z| < 0.2718$ (left-sided, not stable)

$R_2 :\ 0.2718 < |z| < 0.2866$ (two-sided, not stable)

$R_3 :\ 0.2866 < |z| < 8.9576$ (two-sided, stable)

$R_4 :\ |z| > 8.9576$ (right-sided, not stable)

From the pole-zero plot it can be seen that the DTFT – You cannot tell if the DTFT exists from the pole zero plot alone. In order to know this, the region of convergence must be specified. The DTFT does exist for the sequence obtained by using the ROC R_3 shown above. This would be a stable system with a two-sided impulse response.

Q3.49 The MATLAB program to compute and display the rational z-transform from its zeros, poles and gain constant is given below:

```
% Program Q3_49
% Given the poles and zeros of G(z), compute and display the rational
% z-transform.
clf;
%  initialize
z = [0.3 2.5 -0.2+i*0.4 -0.2-i*0.4]';
p = [0.5 -0.75 0.6+i*0.7 0.6-i*0.7]';
k = 3.9;
% find numerator and denominator polynomial coefficients
[num den] = zp2tf(z,p,k)
```

The rational form of a z-transform with the given poles, zeros, and gain is found to be –

```
num =
   3.9000  -9.3600  -0.6630  -1.0140   0.5850

den =
   1.0000  -0.9500   0.1750   0.6625  -0.3187
```

$$G(z) = \frac{3.9 - 9.36z^{-1} - 0.663z^{-2} - 1.014z^{-3} + 0.585z^{-4}}{1 - 0.95z^{-1} + 0.175z^{-2} + 0.6625z^{-3} - 0.3187z^{-4}}$$

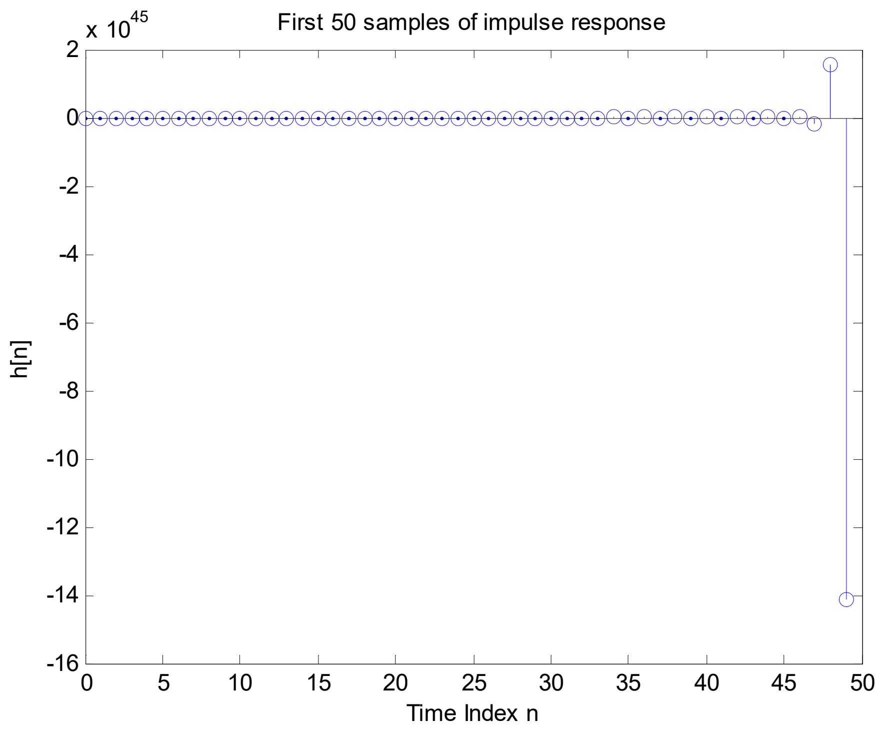
Project 3.6   Inverse z-Transform

Answers:

Q3.50 The MATLAB program to compute the first L samples of the inverse of a rational z-transform is given below:

```
% Program Q3_50
% Given numerator and denominator coefficient vectors for G(z),
% find and plot the first L samples of the impulse response, where
% the parameter L is input by the user.
%
clf;
%  initialize
num = [2  5 9 5 3];
den = [5 45 2 1 1];
% Query user for parameter L
L = input('Enter the number of samples L: ');
% find impulse response
[g t] = impz(num,den,L);
%plot the impulse response
stem(t,g);
title(['First ',num2str(L),' samples of impulse response']);
xlabel('Time Index n');
ylabel('h[n]');
```

The plot of the first 50 samples of the inverse of G(z) of Q3.46 obtained using this program is sketched below:



Q3.51 The MATLAB program to determine the partial-fraction expansion of a rational z-transform is given below:

```
% Program Q3_51
% Given numerator and denominator coefficient vectors for G(z),
% find and plot the first L samples of the impulse response, where
% the parameter L is input by the user.
%
clf;
%  initialize
num = [2  5 9 5 3];
den = [5 45 2 1 1];
% partial fraction expansion
[r p k] = residuez(num,den)
```

The partial-fraction expansion of `G(z)` of Q3.46 obtained using this program is shown below:

```
r =
   0.3109
  -1.0254 - 0.3547i
  -1.0254 + 0.3547i
  -0.8601
p =
  -8.9576
   0.1147 + 0.2627i
   0.1147 - 0.2627i
  -0.2718
k =
     3
```

$$G(z)=\frac{0.3109}{1+8.9576z^{-1}}+\frac{-1.0254-0.3547j}{1-(0.1147+0.2627j)z^{-1}}+\frac{-1.0254+0.3547j}{1-(0.1147-0.2627j)z^{-1}}-\frac{0.8601}{1+0.2718z^{-1}}+3$$

From the above partial-fraction expansion we arrive at the inverse z-transform `g[n]` as shown below:

Three of the terms are straightforward to invert from the *z*-transform table on page 110 of the *Oppenheim & Schafer* textbook:

$$\frac{0.3109}{1+8.9576z^{-1}} \xleftrightarrow{Z} 0.3109(-8.9576)^{n}u[n]$$

$$\frac{0.8601}{1+0.2718z^{-1}} \xleftrightarrow{Z} 0.8601(-0.2718)^{n}u[n]$$

$$3 \xleftrightarrow{Z} 3\delta[n]$$

There remains the term

$$\frac{-1.0254-0.3547j}{1-(0.1147+0.2627j)z^{-1}}+\frac{-1.0254+0.3547j}{1-(0.1147-0.2627j)z^{-1}}. \tag{1.1}$$

Let $a=-1.0254+0.3547j$ and $b=0.1147+0.2627j$. To save writing, let $a_R=\mathrm{Re}[a]$, $a_I=\mathrm{Im}[a]$, $b_R=\mathrm{Re}[b]$, and $b_I=\mathrm{Im}[b]$. The term (1.1) may then be written as

$$\frac{a^*}{1-bz^{-1}}+\frac{a}{1-b^*z^{-1}}. \tag{1.2}$$

After some algebra, (1.2) can be simplified to

$$\begin{aligned}\frac{a^*}{1-bz^{-1}}+\frac{a}{1-b^*z^{-1}} &= \frac{2a_R-\left(2a_Rb_R-2a_Ib_I\right)z^{-1}}{1-2b_Rz^{-1}+|b|^2\,z^{-2}}\\ &=\frac{2a_R-2a_Rb_Rz^{-1}}{1-2b_Rz^{-1}+|b|^2\,z^{-2}}+\frac{2a_Ib_Iz^{-1}}{1-2b_Rz^{-1}+|b|^2\,z^{-2}}\\ &=2a_R\frac{1-b_Rz^{-1}}{1-2b_Rz^{-1}+|b|^2\,z^{-2}}+2a_I\frac{b_Iz^{-1}}{1-2b_Rz^{-1}+|b|^2\,z^{-2}}.\end{aligned} \tag{1.3}$$

Making the associations $r\equiv|b|$ and $\omega_0\equiv\arg b$, we obtain $b_R=r\cos(\omega_0)$ and $b_I=r\sin(\omega_0)$, whereupon (1.3) may be written as

$$\frac{a^*}{1-bz^{-1}}+\frac{a}{1-b^*z^{-1}}=2a_R\frac{1-r\cos(\omega_0)z^{-1}}{1-2r\cos(\omega_0)z^{-1}+r^2z^{-2}}+2a_I\frac{r\sin(\omega_0)z^{-1}}{1-2r\cos(\omega_0)z^{-1}+r^2z^{-2}}. \tag{1.4}$$

These are the entries on lines 11 and 12 of the z-transform table on page 110 of the text. Therefore, the inverse z-transform of (1.4) is given by

$$2a_Rr^n\cos(\omega_0n)u[n]+2a_Ir^n\sin(\omega_0n)u[n]. \tag{1.5}$$

Plugging back into (1.5) the definitions of a, b, r, and ω_0, we have for the inverse z-transform of the term (1.1)

$$-2.0508(0.2866)^n\cos(1.1592n)u[n]+0.7094(0.2866)^n\sin(1.1592n)u[n].$$

All together, the required inverse z-transform is given by

$$g[n] = 0.3109(-8.9576)^n u[n] - 2.0508(0.2866)^n \cos(1.1592n)u[n] \\ + 0.7094(0.2866)^n \sin(1.1592n)u[n] - 0.8601(-0.2718)^n u[n] + 3\delta[n]$$

An alternate but equivalent solution may be obtained by inverting the two terms in (1.1) individually. Let

$$\begin{aligned} \alpha &= 0.1147 + 0.2627j \\ r &= |\alpha| = 0.2866 \\ \omega &= \arg\alpha = 1.1591 \\ \beta &= -1.0254 - 0.3547j \\ k &= |\beta| = 1.0850 \\ \theta &= \arg\beta = -2.8086 \end{aligned}$$

Then

$$\begin{aligned} (1.1) &= \frac{\beta}{1-\alpha z^{-1}} + \frac{\beta^*}{1-\alpha^* z^{-1}} \\ &\xleftrightarrow{Z} \beta\alpha^n u[n] + \beta^*\left(\alpha^*\right)^n u[n] \\ &= ke^{j\theta} r^n e^{j\omega n} u[n] + ke^{-j\theta} r^n e^{-j\omega n} u[n] \\ &= kr^n e^{j(\omega n+\theta)} u[n] + kr^n e^{-j(\omega n+\theta)} u[n] \\ &= kr^n \left[e^{j(\omega n+\theta)} + e^{-j(\omega n+\theta)} \right] u[n] \\ &= 2kr^n \cos\left(\omega n + \theta\right) u[n] \\ &= 2.1700(0.2866)^n \cos\left(1.1591n - 2.8086\right) u[n]. \end{aligned}$$

This results in the alternate but equivalent solution

$$g[n] = 0.3109(-8.9576)^n u[n] + 2.1700(0.2866)^n \cos(1.1592n - 2.8086)u[n] - 0.8601(-0.2718)^n u[n] + 3\delta[n]$$

Date: **12 October 2015** **Signature: HAVLICEK**