ECE 513 HMW5

Problem 1:

Solve:

(a) $X(k) = \sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi kn}{N}} \quad k=0,1,\cdots,N-1$

$= \sum_{n=0}^{N-1} x(n) \left(\cos\frac{2\pi kn}{N} - j\sin\frac{2\pi kn}{N}\right)$

due to the face that $x(n)$ is a real sequence,

then $Re[X(k)] = \sum_{n=0}^{N-1} x(n) \cos\frac{2\pi kn}{N}$

the DFT of $Xe(n) = \frac{1}{2}\sum_{n=0}^{N-1} x(n) e^{-j\frac{2\pi kn}{2N}} + \frac{1}{2}\sum_{-N+1}^{n=0} x(-n) e^{-j\frac{2\pi kn}{2N}} \cdots (1)$

let $-n=m$, $\frac{1}{2}\sum_{m=0}^{N-1} x(m) e^{j\frac{2\pi km}{2N}} \rightarrow \frac{1}{2}\sum_{n=0}^{N-1} x(n) e^{j\frac{2\pi kn}{2N}}$

So (1) is equal to $\frac{1}{2}\sum_{n=0}^{N-1} x(n)\left(e^{-j\frac{2\pi kn}{2N}} + e^{j\frac{2\pi kn}{2N}}\right) = \sum_{n=0}^{N-1} x(n)\cos\frac{2\pi kn}{2N} = \sum_{n=0}^{N-1} x(n)\cos\frac{\pi kn}{N}$.

Therefore $Re[X(k)] \neq$ the DFT of $Xe(n)$ by comparing (2) with $Re[X(k)]$.

(b) $\hat{x}(n) = \frac{1}{N}\sum_{k=0}^{N-1} Re[X(k)] e^{j\frac{2\pi kn}{N}} \quad n=0,1,\cdots,N-1$

$= \frac{1}{N}\sum_{k=0}^{N-1}\sum_{n=0}^{N-1} x(n)\cos\frac{2\pi kn}{N} e^{j\frac{2\pi kn}{N}} \cdots (1)$

due to that $\cos\frac{2\pi kn}{N} = \frac{e^{j\frac{2\pi kn}{N}} + e^{-j\frac{2\pi kn}{N}}}{2}$

so $\cos\frac{2\pi kn}{N}\cdot e^{j\frac{2\pi kn}{N}} = \frac{e^{j\frac{4\pi kn}{N}}+1}{2} \cdots (2)$

substitute (2) into (1): $\hat{x}(n) = \frac{1}{2N}\sum_{k=0}^{N-1}\sum_{n=0}^{N-1} x(n)\left(e^{j\frac{4\pi kn}{N}}+1\right), \quad n=0,1,\cdots,N-1$

Problem 2: Solve: $N = 512, M = 98, L = N - M + 1 = 415$

(a) The number of blocks of data: $B = \lceil \frac{63412}{415} \rceil = \lceil 152.8 \rceil = 153$

(b) The number of zeros we should add to the filter: $512 - 98 = 414$

Matlab statement:

```
filter(1:98) = the ith coefficient;
filter(99:512) = 0;
DFT_filter = fft(filter, 512)
```

(c) The number of zeros that must be added to the first block of data: $512 - 415 = 97$

The number of data values that need to be used in the block of data: $L = N - M + 1 = 415$

Matlab statement:

```
data_block_1st(415:512) = 0
DFT_data_block_1st = fft(data_block_1st, 512)
```

(d) The number of output values that'll be generated for the first block of data: $N = L + M - 1 = 512$.

Matlab statement:

```
y1(n) = ifft(ft1 ft2, 512)
y(n)(1:512) = y1(n)(1:512)
```

(e) The first 415 samples of the 2nd data block will be used

The last 97 samples of the 2nd data block will be zeros.

Matlab statement:

```
data_block_2nd(1:415) = the ith data value.
data_block_2nd(415:512) = 0
DFT_data_block_2nd = fft(data_block_2nd, 512)
```

(f) The number of output values from the second $\boxed{\text{output}}$ data block should be inserted and/or added to the output arrayy:

$M - 1 = 97$

Matlab statement:

```
y(n)(416:512) = y(n)(416:512) + y2(n)(1:97)
```

Problem 3:

(a) $W_r(\omega) = \sum_{n=-\infty}^{\infty} w_r(n) e^{-j\omega n}$

$= \sum_{n=0}^{L-1} e^{-j\omega n}$

$= \frac{1}{1 - e^{-j\omega}}$

$W_n(\omega) = \sum_{n=-\infty}^{\infty} W_n(n) e^{-j\omega n} = \sum_{n=0}^{L-1} \frac{1}{2}[1 - \cos(\frac{2\pi}{L-1} n)] e^{-j\omega n}$

$e^{-j\omega n} \cos(\frac{2\pi}{L-1} n) = e^{-j\omega n} \frac{e^{j\frac{2\pi}{L-1}n} + e^{-j\frac{2\pi}{L-1}n}}{2} = \frac{1}{2}[e^{-jn(\omega - \frac{2\pi}{L-1})} + e^{-jn(\omega + \frac{2\pi}{L-1})}]$

So $\sum_{n=0}^{L-1} \cos(\frac{2\pi}{L-1} n) e^{-j\omega n} = \frac{1}{2}[\frac{1}{1 - e^{-j(\omega - \frac{2\pi}{L-1})}} + \frac{1}{1 - \boxed{e}^{-j(\omega + \frac{2\pi}{L-1})}}]$

Therefore $W_n(\omega) = \frac{1}{2}[\frac{1}{1 - e^{-j\omega}} - \frac{1}{2}[\frac{1}{1 - e^{-j(\omega - \frac{2\pi}{L-1})}} + \frac{1}{1 - e^{-j(\omega + \frac{2\pi}{L-1})}}]]$

(b)

The Matlab code is shown as follows:

```
wr_n(1:30)=1;                    % The sequence of the
                                 rectangular window
n=0:30;
wh_n=0.5*(1-cos(2*pi*n/29));     % The sequence of the Hanning
                                 window
Wr_w=fft(wr_n,400);
Wh_w=fft(wh_n,400);
% Plot the magnitude and phase of the sampled spectra of the
sequence of the rectangular window
y1_am=abs(Wr_w);
y1_ph=angle(Wr_w);
figure(1)
subplot(2,1,1)
stem(0:length(y1_am)-1,y1_am)
title('Sampled Spectrum of wr(n) for N=400 - Amplitude')
xlabel('k')
ylabel('Amplitude')
subplot(2,1,2)
stem(0:length(y1_ph)-1,y1_ph)
title('Sampled Spectrum of wr(n) for N=400 - Phase')
xlabel('k')
ylabel('Phase')
% Plot the magnitude and phase of the sampled spectra of the
sequence of the Hanning window
y2_am=abs(Wh_w);
y2_ph=angle(Wh_w);
figure(2)
subplot(2,1,1)
stem(0:length(y2_am)-1,y2_am)
title('Sampled Spectrum of wh(n) for N=400 - Amplitude')
xlabel('k')
ylabel('Amplitude')
subplot(2,1,2)
stem(0:length(y2_ph)-1,y2_ph)
title('Sampled Spectrum of wh(n) for N=400 - Phase')
xlabel('k')
ylabel('Phase')
```

Therefore, the figures of the magnitude and phase of the sampled spectra, of which the sequences are the rectangular window and the Hanning window, are shown in Fig.1 and Fig. 2 respectively.

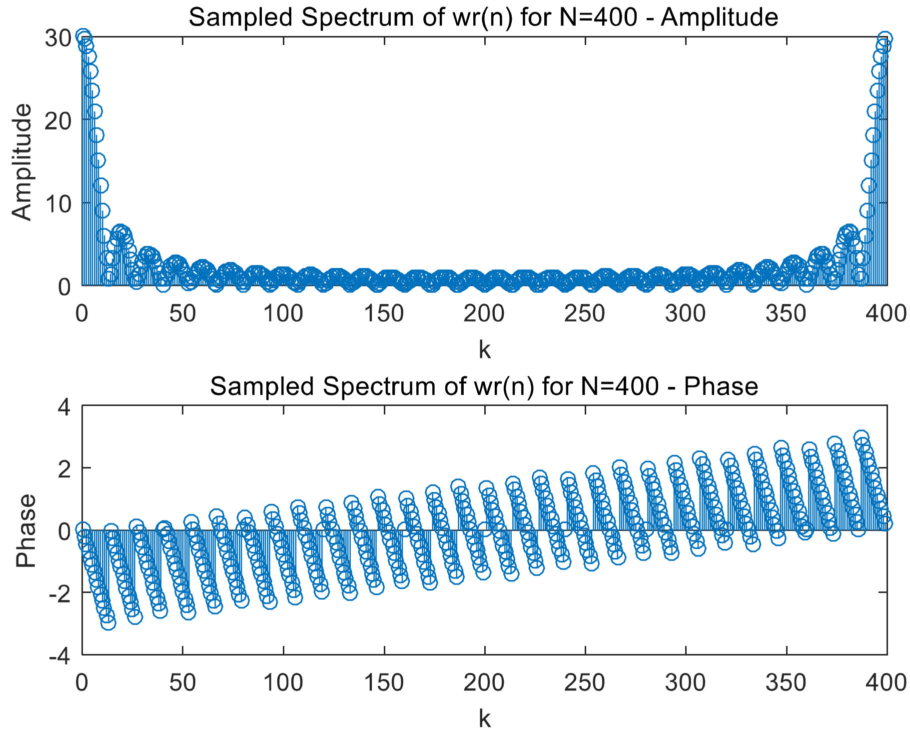

Fig. 1. The magnitude and phase of the sampled spectra of the rectangular window

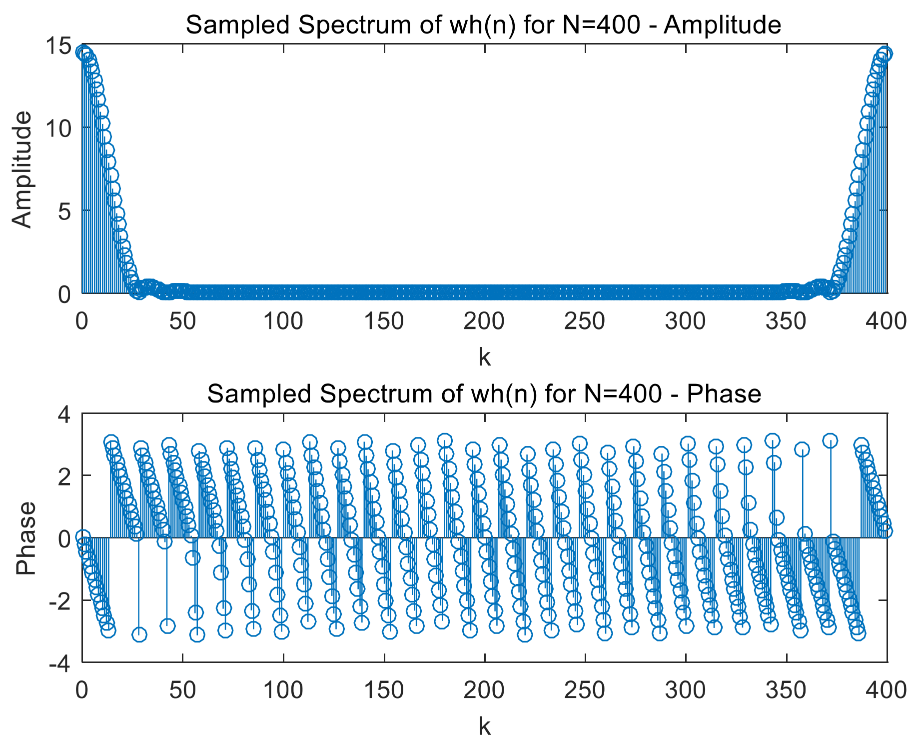

Fig. 2. The magnitude and phase of the sampled spectra of the Hanning window

(c)

The Matlab code is shown as follows:

```
% i.Use the rectangular window
n=0:49;                        % L=50
x10_n=cos(2*200*pi*n/1500)+cos(2*220*pi*n/1500)+cos(2*250*pi*n
/1500);
wr_n1(1:50)=1;
xr1=x10_n.*wr_n1;
Xr1=fft(xr1,2^14);
n=0:99;                        % L=100
x11_n=cos(2*200*pi*n/1500)+cos(2*220*pi*n/1500)+cos(2*250*pi*n
/1500);
wr_n2(1:100)=1;
xr2=x11_n.*wr_n2;
Xr2=fft(xr2,2^14);
n=0:149;                       % L=150
x12_n=cos(2*200*pi*n/1500)+cos(2*220*pi*n/1500)+cos(2*250*pi*n
/1500);
wr_n3(1:150)=1;
xr3=x12_n.*wr_n3;
Xr3=fft(xr3,2^14);
w=0:2^14-1;
figure(3)
subplot(3,1,1)                 % Plot the magnitude spectrum for
                               Xr1(k),L=50
plot(w,Xr1)
title('The magnitude spectrum for Xr1(k) L=50')
xlabel('k')
ylabel('Magnitude')
subplot(3,1,2)                 % Plot the magnitude spectrum for
                               Xr2(k),L=100
plot(w,Xr2)
title('The magnitude spectrum for Xr2(k) L=100')
xlabel('k')
ylabel('Magnitude')
subplot(3,1,3)                 % Plot the magnitude spectrum for
                               Xr3(k),L=150
plot(w,Xr3)
title('The magnitude spectrum for Xr3(k) L=150')
xlabel('k')
ylabel('Magnitude')
% ii.Use the Hanning window
n=0:49;                        % L=50
x20_n=cos(2*200*pi*n/1500)+cos(2*220*pi*n/1500)+cos(2*250*pi*n
```

```
/1500);
wh_n=0.5*(1-cos(2*pi*n/49));
xh1=x20_n.*wh_n;
Xh1=fft(xh1,2^14);
n=0:99;                          % L=100
x21_n=cos(2*200*pi*n/1500)+cos(2*220*pi*n/1500)+cos(2*250*pi*n
/1500);
wh_n=0.5*(1-cos(2*pi*n/99));
xh2=x21_n.*wh_n;
Xh2=fft(xh2,2^14);
n=0:149;                         % L=150
x22_n=cos(2*200*pi*n/1500)+cos(2*220*pi*n/1500)+cos(2*250*pi*n
/1500);
wh_n=0.5*(1-cos(2*pi*n/149));
xh3=x22_n.*wh_n;
Xh3=fft(xh3,2^14);
w=0:2^14-1;
figure(4)
subplot(3,1,1)                   % Plot the magnitude spectrum for
                                 Xh1(k),L=50
plot(w,Xh1)
title('The magnitude spectrum for Xh1(k) L=50')
xlabel('k')
ylabel('Magnitude')
subplot(3,1,2)                   % Plot the magnitude spectrum for
                                 Xh2(k),L=100
plot(w,Xh2)
title('The magnitude spectrum for Xh2(k) L=100')
xlabel('k')
ylabel('Magnitude')
subplot(3,1,3)                   % Plot the magnitude spectrum for
                                 Xh3(k),L=150
plot(w,Xh3)
title('The magnitude spectrum for Xh3(k) L=150')
xlabel('k')
ylabel('Magnitude')
```

Hence, the figures of the magnitude spectrum for each $X_{ri}(k)$ are shown in Fig. 3. When L=100 and 150, I can resolve all spectral lines.

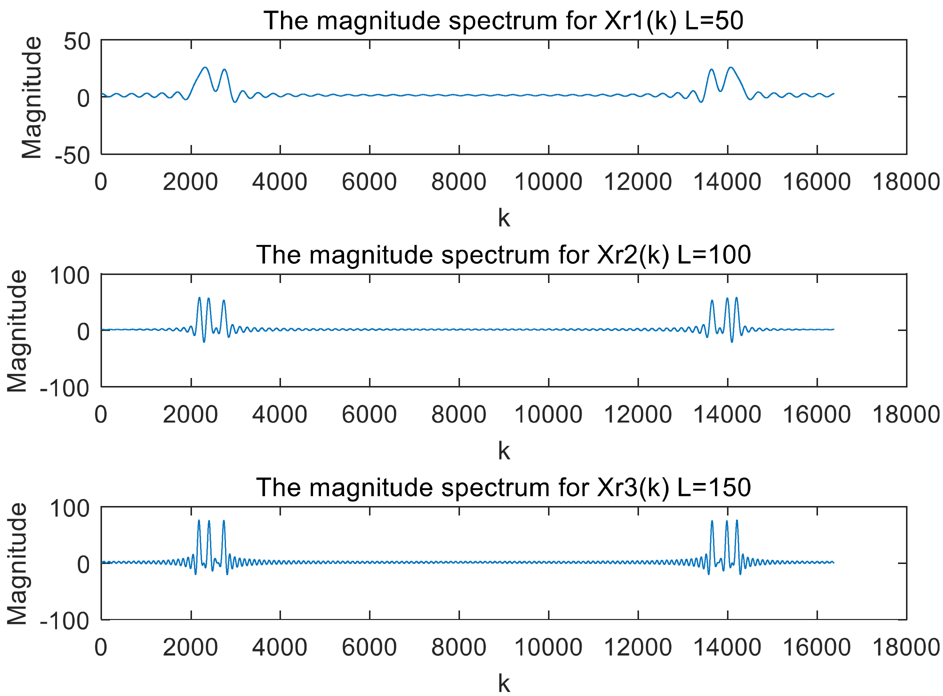

Fig. 3. The magnitude spectrum for each Xri

Moreover, the figures of the magnitude spectrum for each $X_{hi}(k)$ are shown in Fig. 4. When L=100 and 150, I can resolve all spectral lines.

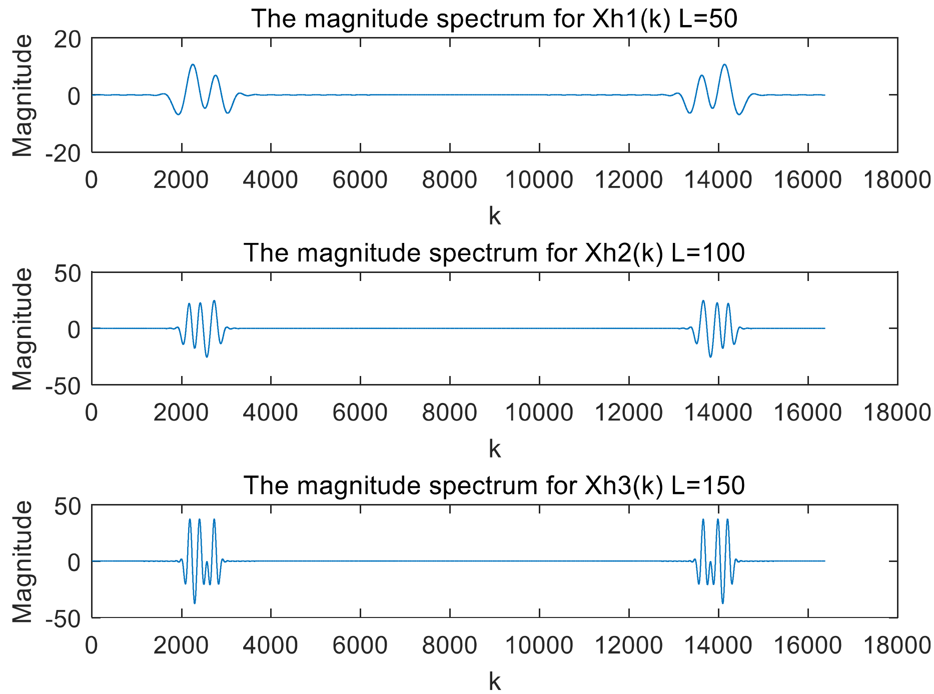

Fig. 4. The magnitude spectrum for each Xhi

4. Solve:

(a)

The Matlab code is shown as follows:

```
figure(5)
y1=sampdata;
stem(0:length(y1)-1,y1)
title('A sample input sequence')
xlabel('Sample Number')
ylabel('Amplitude')
```

The figure of the input sequence for this problem is shown in Fig. 5:

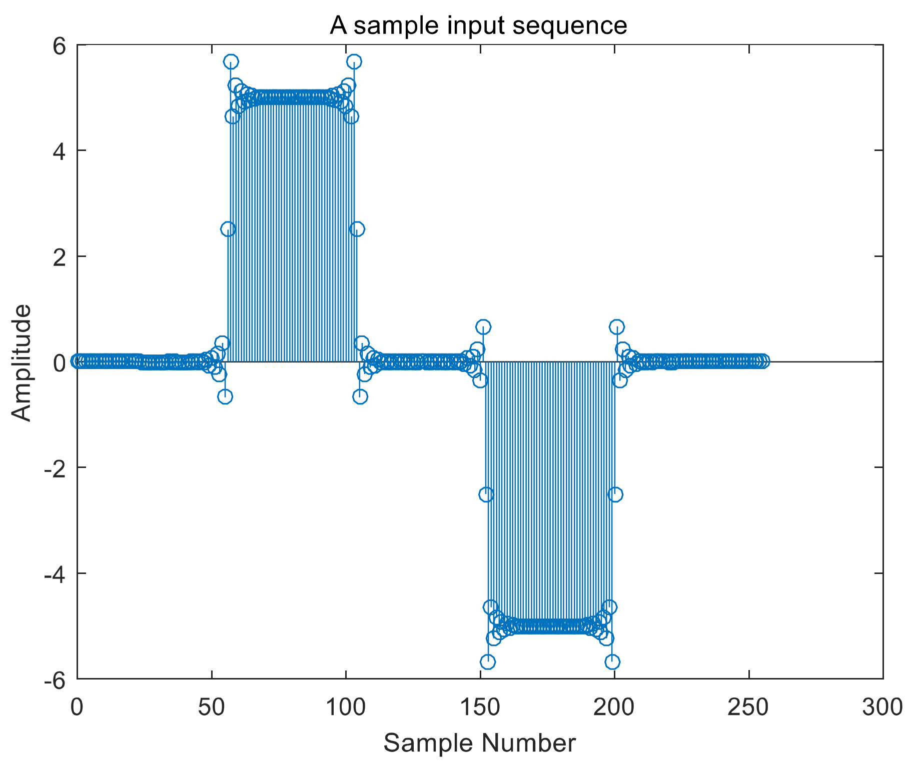

Fig. 5. The sample input sequence

(b)

Use the following Matlab code to compute the DCT for this sequence.

```
X_dct=dct(y1);
Mag_dct=abs(X_dct);
figure(6)
stem(0:length(Mag_dct)-1,Mag_dct)
title('The magnitudes of the DCT for the test sequence')
xlabel('Frequency (Radians)')
ylabel('Amplitude')
```

The figure of the magnitudes of the DCT for the test sequence is shown in Fig. 6:

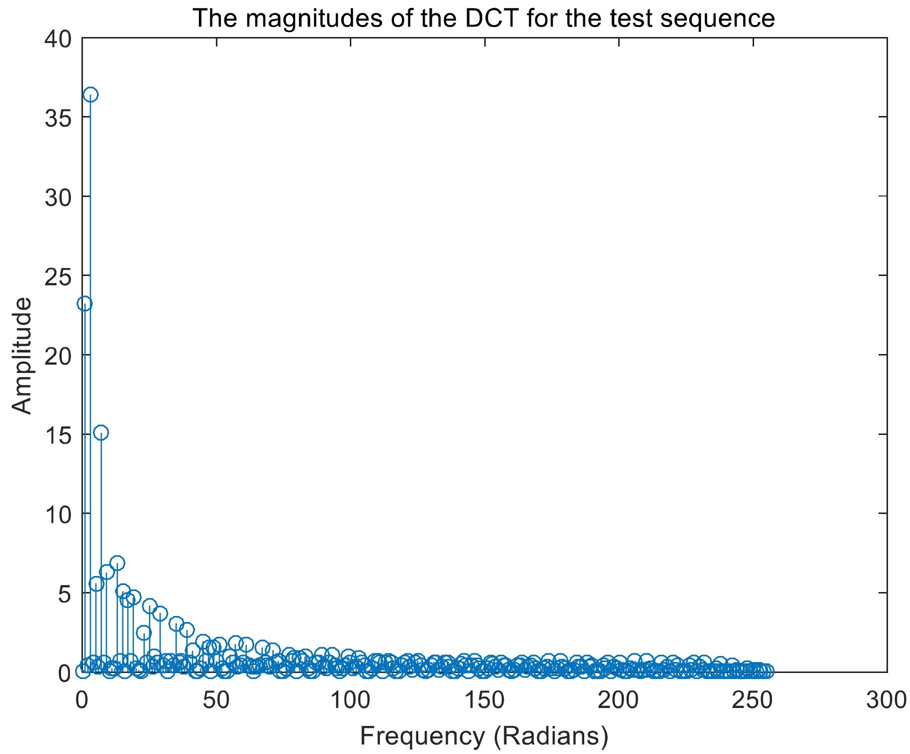

Fig. 6. The magnitudes of the DCT for the test sequence

(c)

Use the Matlab routine **fft** to compute the DCT for this sequence based on the method shown in class. The Matlab code is shown as follows:

```
n=length(y1);
y2=y1;
for j = n+1:1:2*n          % Extend the original sequence to
                            make it even symmetric
    y2(j)=y1(2*n+1-j);
end
Y2=fft(y2,length(y2));     % Use a 2N point DFT to compute Y2
for k = 1:1:length(Y2)     % Multiply Y2 by relevant complex
                            exponential
    Y2(k)=Y2(k)*exp(1)^(-1i*2*pi*k/(4*n));
end
Y2(n+1:2*n)=[];             % Extract the first N values
Mag_Y2=abs(Y2);
Abs_error=abs(Mag_dct-Mag_Y2);
figure(7)
stem(0:length(Abs_error)-1,Abs_error)
title('The absolute error between the implementation and the
result from b')
xlabel('Frequency (Radians)')
```

```
ylabel('Absolute error')
figure(8)
stem(0:length(Mag_Y2)-1,Mag_Y2)
title('The magnitudes of the DCT for the test sequence')
xlabel('Frequency (Radians)')
ylabel('Amplitude')
```

The figures of the magnitudes of my implementation, and the absolute error between my implementation and results obtained in part b are shown in Fig. 7 and Fig. 8 respectively.

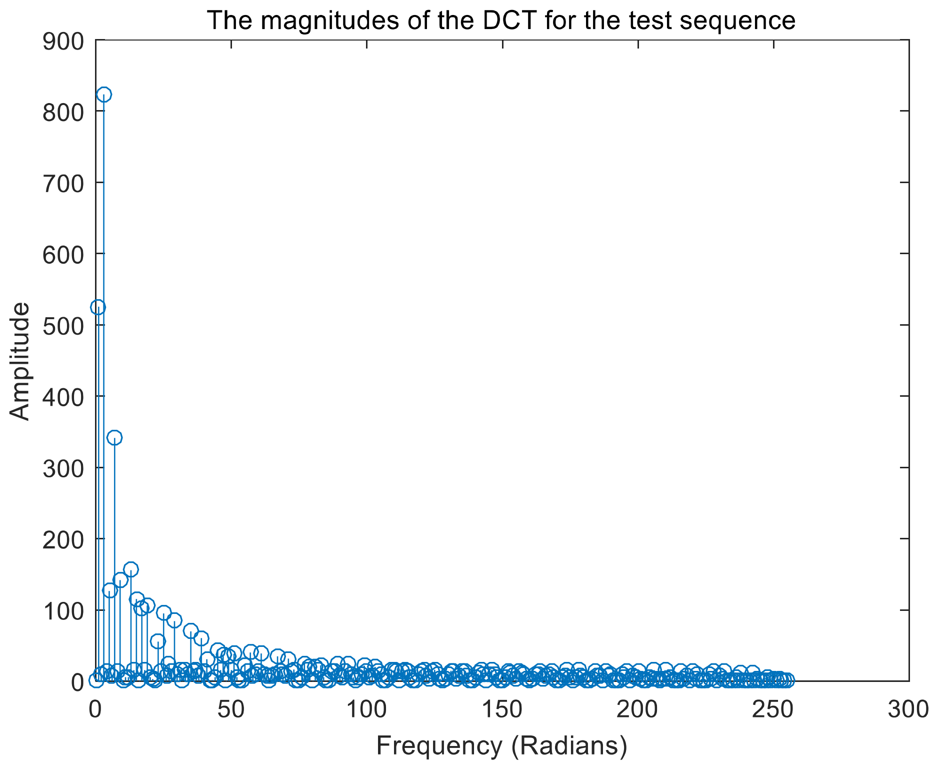

Fig. 7. The magnitudes of my implementation

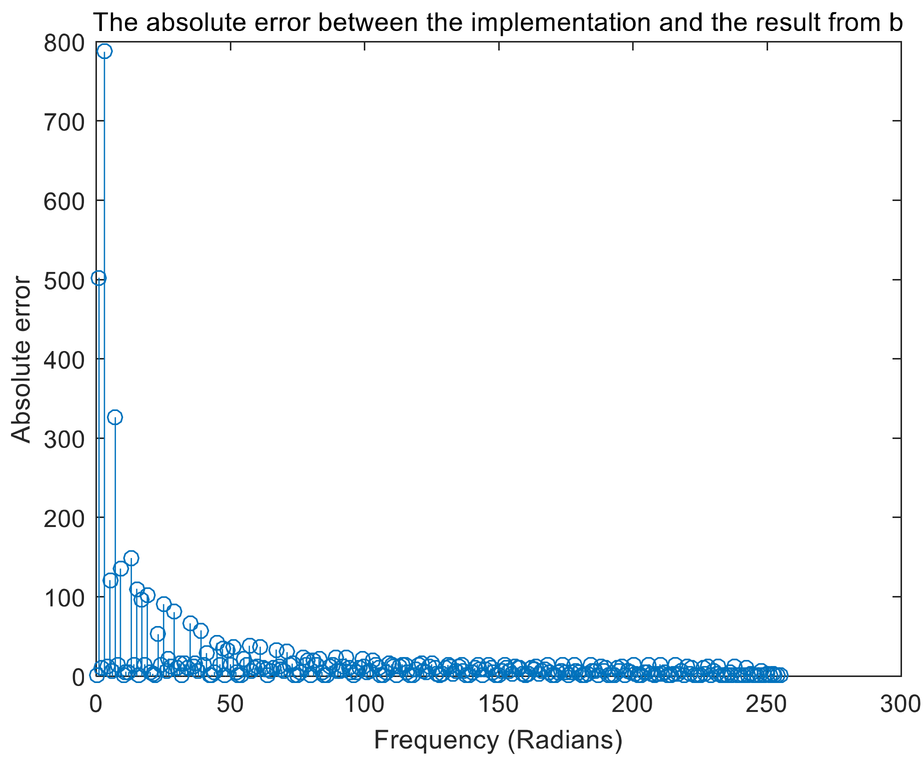

Fig. 8. The absolute error between the implementation and the result from b

As is clearly demonstrated, the absolute error is definitely big so that I have to doubt that there is something wrong with my implementation. Fortunately, with the help of the Matlab documentation for the function **dct**, I know how should I make some necessary corrections.

(d)

Description

y = dct(x) returns the unitary discrete cosine transform of x,

$$y(k) = w(k)\sum_{n=1}^{N} x(n)\cos\left(\frac{\pi}{2N}(2n-1)(k-1)\right), \quad k = 1, 2, \ldots, N,$$

where

$$w(k) = \begin{cases} \frac{1}{\sqrt{N}}, & k = 1, \\ \sqrt{\frac{2}{N}}, & 2 \le k \le N, \end{cases}$$

After looking at the Matlab documentation for the function **dct** carefully, I make some necessary corrections in my implementation of the DCT to make the results match. The Matlab code is shown as follows:

```
Mag_Y3=Mag_Y2;
Mag_Y3(1)=Mag_Y2(1)/sqrt(n);        % Change each coefficient
                                      according to the
                                      documentation
for k = 2:1:length(Y2)
    Mag_Y3(k)=Mag_Y2(k)/sqrt(n/2);
end
figure(9)
Abs_error=abs(Mag_dct-Mag_Y3);
stem(0:length(Abs_error)-1,Abs_error)
title('The absolute error between the implementation and the
result from b')
xlabel('Frequency (Radians)')
ylabel('Absolute error')
figure(10)
stem(0:length(Mag_Y3)-1,Mag_Y3)
title('The magnitudes of the DCT for the test sequence')
xlabel('Frequency (Radians)')
ylabel('Amplitude')
```

With the help of some corrections, the figures of the magnitudes of my implementation, and the absolute error between my implementation and results obtained in part b are shown in Fig. 9 and Fig. 10 respectively.

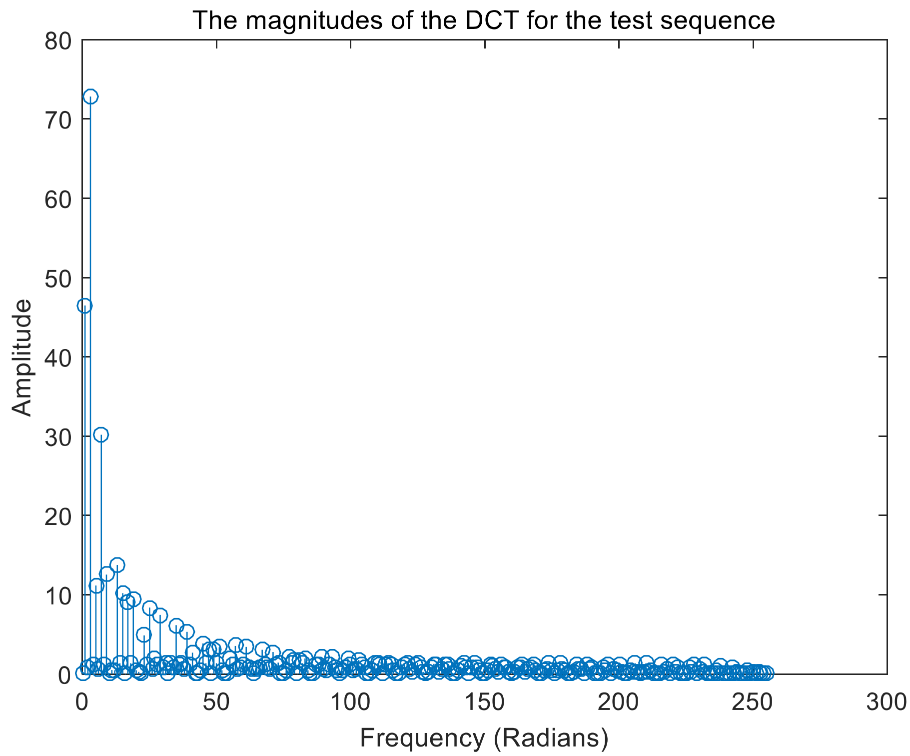

Fig. 9. The magnitudes of my implementation after corrections

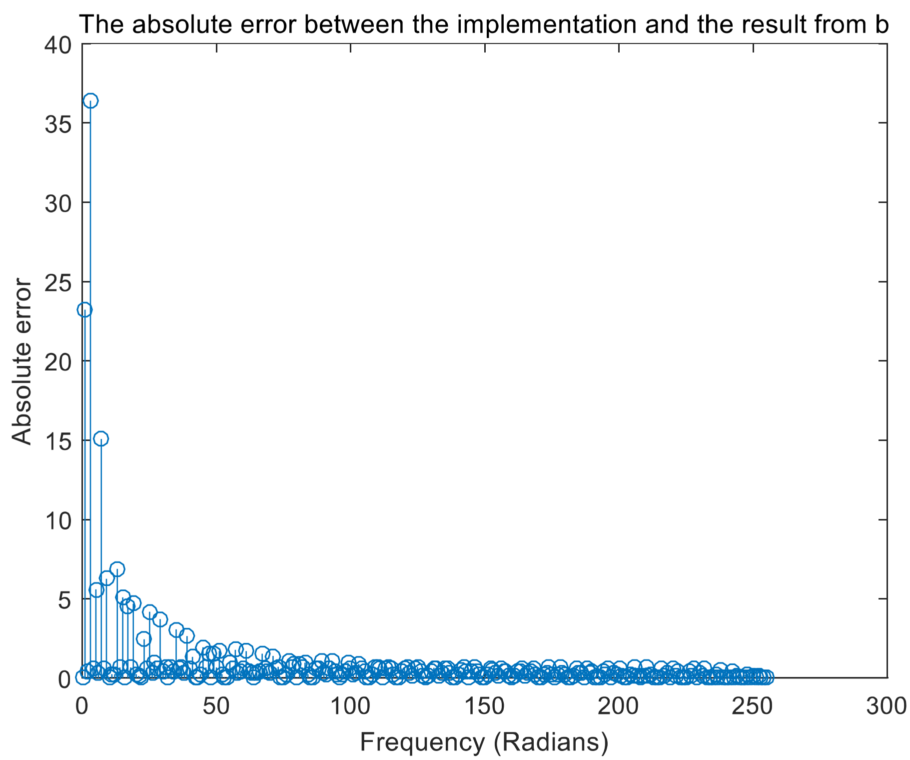

Fig. 10. The absolute error between the implementation and the result from b after corrections

As is clearly demonstrated, the absolute error is definitely smaller than part (c). Even if there are some errors occurring probably due to the precision of my computer, yet I can still safely reach a conclusion that the result of my modified implementation is the same as the one from the Matlab routine **dct**. The algorithm itself is correct enough.