Xiaobin Zhuang - 300519184

ECEN220 Lab 4

26/09/2020

Objectives

The primary object of this lab was to practice the DTFT and inverse DTFT in MATLAB.

Methodology

1.

1a) I saved a dtft() function into a file called dtft.m. Shown in figure 1.

Lab4pt1.m | Lab4pt2.m | dtft.m | invdtft.m

```
function [X] = dtft( x, n, omega )
V=exp(-j*omega*n'); %the V matrix
X=V*x; % the DTFT
% Computes the DTFT
% X = dtft( x, n, omega )
% where X = DTFT at each value of omega % x =
% omega = vector of frequency samples
end
```

fig. 1

1b) I computed and plot the DTFT of x, which shown in figure 2.

The reasonable number M of frequency samples was 500 - 1000. If M was too small, there would be low magnitude and phase resolution; if M was too high, the resolution would be high, however it would take longer time to calculate.

```
%% 1b)
clear all
omega0 = -2*pi; %first frequency sample
omegaM = 2*pi;  %last frequency sample
M = 1000;
k = (0:M)';
omega = omega0 + (omegaM - omega0)*k/M;

x = [-3,2,6,6,4]';
n = [0:length(x)-1]'-1;
X = dtft( x, n, omega )

subplot(311)
stem(n,x)
title('x[n]')
xlabel('n')
ylabel('x[n]')
subplot(312)
plot(omega,abs(X))
xlim([-2*pi 2*pi])
title('Magnitude')
xlabel('w')
```

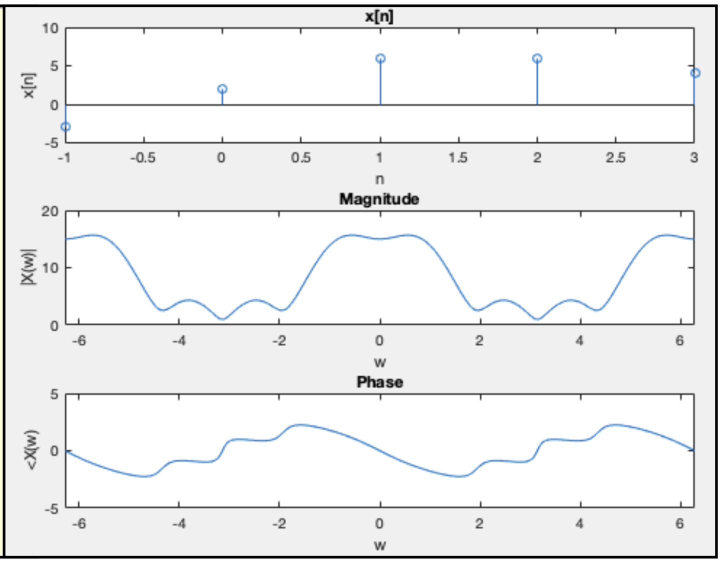

fig. 2

1c) by running the code shown in figure 3, the location of n = 0 has shift from 2nd to 4th. The magnitude plot remain unchanged, however the phase has been changed. Shown in figure 3.

```
clear all
omega0 = -2*pi; %first frequency sample
omegaM = 2*pi;  %last frequency sample
M = 1000;
k = (0:M)';
omega = omega0 + (omegaM - omega0)*k/M;

x = [-3,2,6,6,4]';
n = [0:length(x)-1]'-3; %change the location of n=(
X = dtft( x, n, omega )

subplot(311)
stem(n,x)
title('x[n]')
xlabel('n')
ylabel('x[n]')
subplot(312)
plot(omega,abs(X))
xlim([-2*pi 2*pi])
title('Magnitude')
xlabel('w')
ylabel('|X(w)|')
```

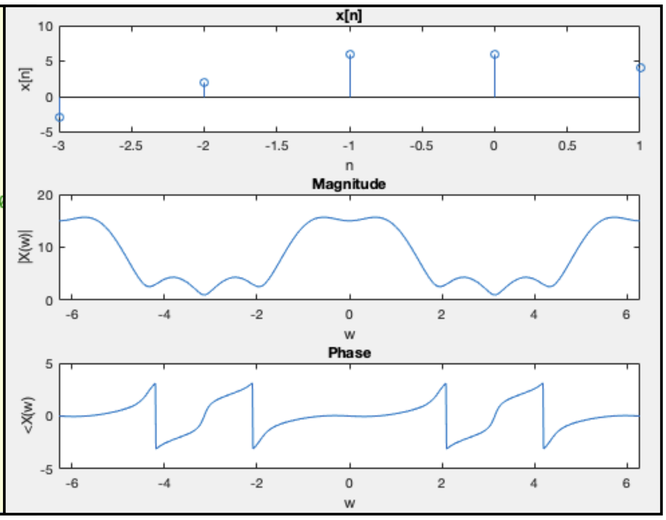

fig. 3

2.

2a) I defined a function called invdtft() and saved it in a file called invdtft.m. Shown in figure 4.

Lab4pt1.m | Lab4pt2.m | dtft.m | invdtft.m | Lab4pt3.m

```
function [x] = invdtft( X, n, omega )
V=exp(-j*omega*n'); %the V matrix
x = V\X; % the inverse DTFT
% Computes the DTFT
% X = dtft( x, n, omega )
% where X = DTFT at each value of omega % x = samples
% omega = vector of frequency samples
end
```

fig. 4

The difference between this function and dtft() is that the position of X and x was changed. Because they were opposite calculation.

2b) I applied invdtft() function to the Fourier transform X(w) calculated in 1b). (Figure 5)

```
%% 2b)
clear all
omega0 = -2*pi; %first frequency sample
omegaM = 2*pi;  %last frequency sample
M = 1000;
k = (0:M)';
omega = omega0 + (omegaM - omega0)*k/M;

x = [-3,2,6,6,4]';
n = [0:length(x)-1]'-1;
X = dtft( x, n, omega ) %DTFT
x1 = invdtft( X, n, omega ) %INVDTFT

stem(n,x1)
title('x[n]')
xlabel('n')
ylabel('x[n]')
```

fig. 5

Then got the graph shown in figure 6.

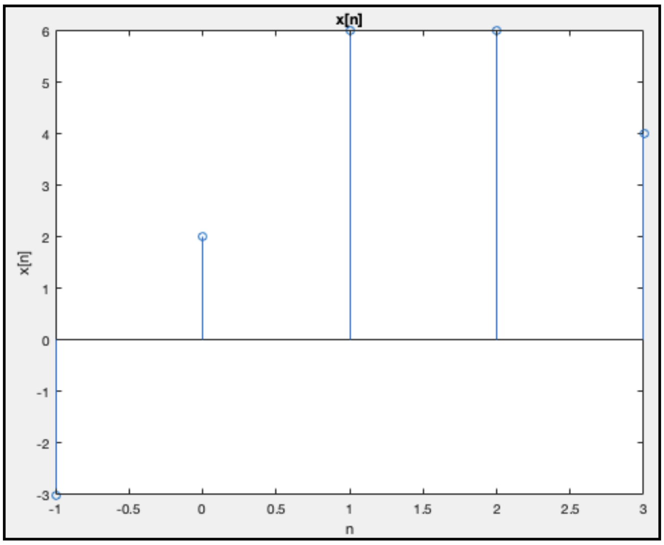

fig. 6

It matched the original sequence x = [-3,2,6,6,4]. The information displayed in command window told us that there was no imaginary components. (Figure 7)

```
Command Window
x1 =

  -3.0000 + 0.0000i
   2.0000 + 0.0000i
   6.0000 + 0.0000i
   6.0000 - 0.0000i
   4.0000 + 0.0000i

Warning: Using only the real component of complex data.
fx >>
```

fig. 7

The imaginary components are significant for Fourier transform because it tells us the phase shift of Fourier transform.

3.

3a) executing the code below (figure 8), I import the data from ‘Temperature.txt’, then calculate DTFT of it.

```
% 3a)
clear all
data=importdata('Temperature.txt');
n=data(:,1);
x=data(:,2);

omega0 = -pi; %first frequency sample
omegaM = pi;  %last frequency sample
M = 1000;
k = (0:M)';
omega = omega0 + (omegaM - omega0)*k/M;

X = dtft( x, n, omega )
```

fig. 8

Plot the magnitude and phase of X on the frequency range -pi to pi. (Figure 9)

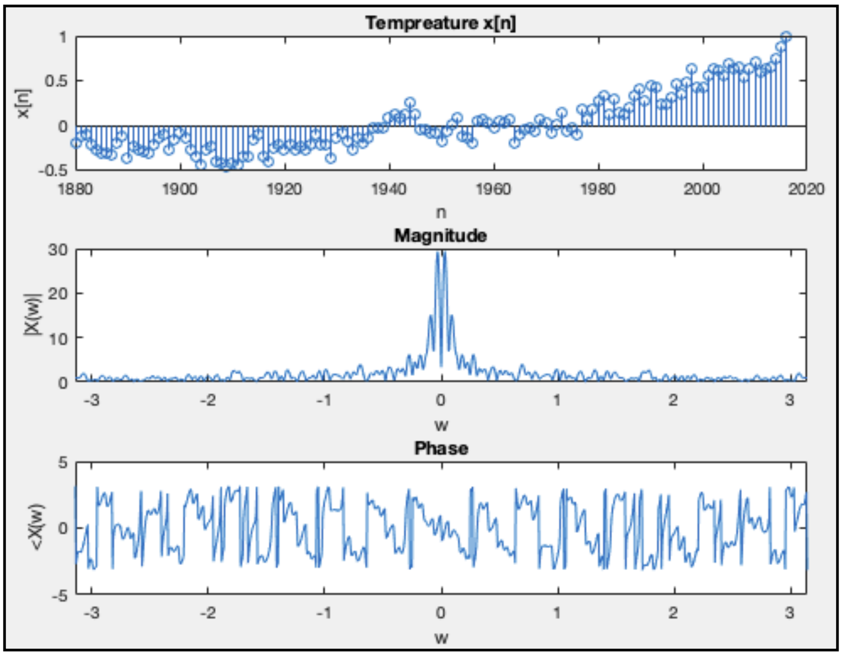

fig. 9

3b) applying the code below(figure 10), saved the values into an array Hav[ ].

```
% 3b)
m = 5;
H = (1/m)*exp(-j*omega*(m-1)/2).*(sin(omega*m/2)/sin(omega/2));
if  omega(:,1) == 0
    H = 1;
end

Hav = H(:,1) %store first column into array Hav[]
```

fig .10

I used if statement to judge that when omega = 0, H(w) = 1. However, it seems not working… So the plot shown below (figure 11) would be the magnitude an phase of H(w) for omega = otherwise. (When m = 5)

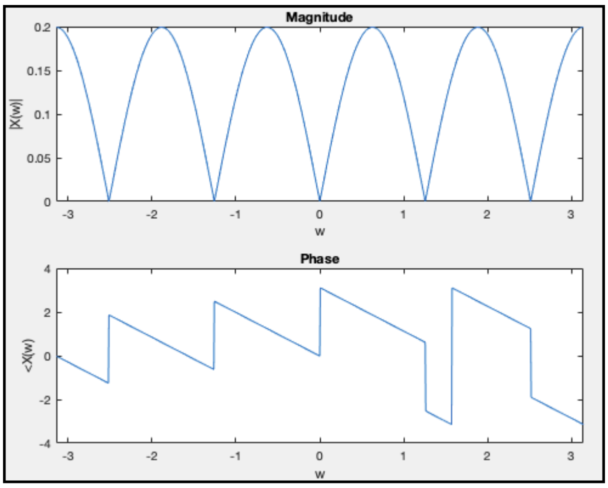

fig .11

3c) running the code below, used convolution property to find Y(w). Then find inverse DTFT of Y and X by using the function in 2a). Finally, plot them. (Figure 13)

```
%3c)
Y = Hav.*X;
y1 = invdtft( Y, n, omega ) %INVDTFT of Y
x1 = invdtft( X, n, omega ) %INVDTFT of X
figure()
plot(n,x1,n,y1)
legend('x1','y1')
title('Compare x and y')
xlabel('n')
ylabel('x1,y1')
```

fig .12

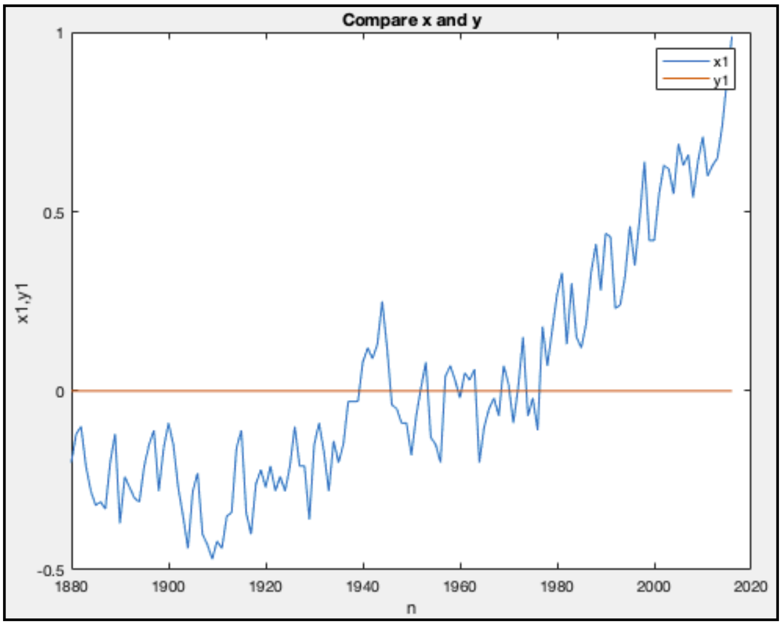

fig .13

The plot of y was not right because I didn't create the value of H(w) = 1when omega = 0. So all of the signals has been filtered. So the value of y1 is 0.

By changing the value of m, the resolution of Fourier transform of H would be changed. The bigger value of m, the higher resolution of H(w) could get.