CSULB: Fall 2022

EE 386 Lab

Professor Duc Tran

Lab 2:

Discrete-Time Fourier Transform

By

Flores, Bryan

Due on: 10 / 04 / 22

Introduction

This lab focused on using computer-aided design to obtain results of the Discrete-time Fourier transform. Overall the basis of the lab dealt with the mathematical calculations (absolutely summable) transform expression (X(ejw)) of a given finite and discrete sequence(xn). The details of the mathematical theory will for the most part be left for the reader to know, but specific and repeating expressions will be addressed further on.

Since the makeup of the resultant Fourier transfer response is a function of (e^jw): a complex expression, we had the opportunity to plot the Magnitude, Real Part, Angle Part, and Imaginary Parts of our results

There were certain tasks accomplished in this lab: the first was to create MATLAB function 'dtft(x,n,w)' that will process the discrete-time Fourier transform of a sequence. Another task was to use the transform to prove the 'sample shift property' of the sequence to transform pairs.

Procedure

The whole of lab was done using MATLAB in order to quickly process and plot the resultant complex sequences. For most of the tasked we gave 4 plots that showed the 4 parts of the response (Magnitude, Real Angle and Imaginary).

For task one we were given a basic outline for a function in order to create one of our own. But overall the basis of the function was described in a pre-lab example that used the theoretical summation based definition of the Discrete Time Fourier transform, and with some manipulations of the step changes and clever use of MATLAB's vector arithmetic, the function was fairly straightforward.

The following task asked us to non-explicitly plot the transform then prove it using MATLAB functions we created.

Then next task was to rove the shifting property on the discrete time Fourier transform from a given sinusoidal discrete time sequence.

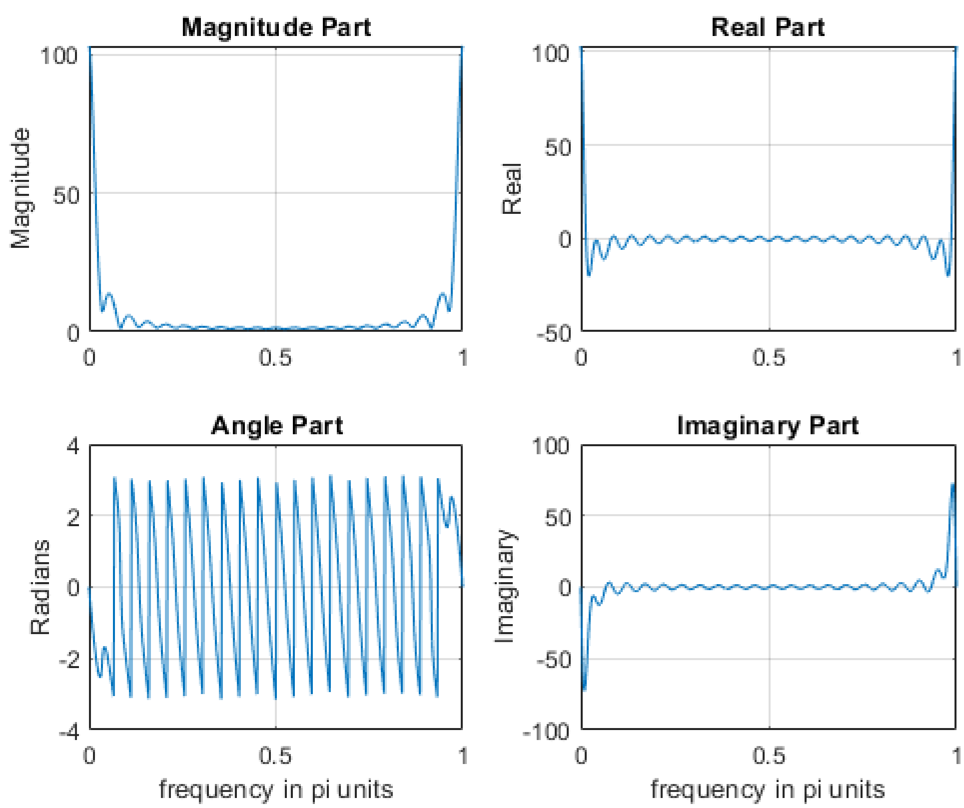
Results and Discussion

Task 1) The following is my function for the Discrete Time Fourier Transform (dtft) function.

```
function [X] = dtft(x,n,k)
% Computes discrete-time Fourier transform
% of a given seqeunce 'x',
% M is usually  = 500, but here we will take the set w range as input
% and use the last most value as our M value
%%%%
% w = eval range [0,pi], or [-pi, pi]
% then wk = (pi/M)*k
% where k = 0,1,2 ... M
%%%%
M = k(end);
```

```
% w = (abs(w(end)-w(1))/500)*k % if you arent able to chnage input w
% consider range of w input
% w = (w(end)/500)*k; % or have the input w = 2*pi and use (w/500)*k

X = x*(exp(-1i*pi/M)).^(n'*k);

end
```

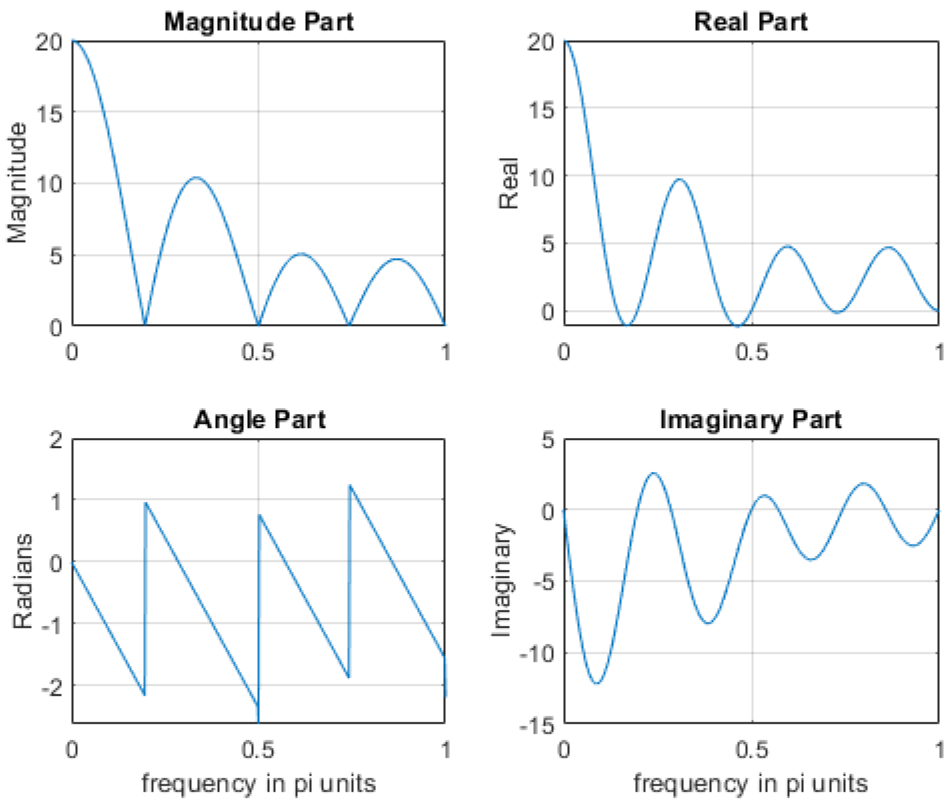
In the subsidiary parts of the task we were asked to plot the DTFT of two sequences. For part a we were given a sequence x(n) containing a product of a an exponential and a sum with a shifted step and asked to find the transform across -pi to pi. Since the given function contained a mix of step functions it was easier to simplify the sequence using the MATLAB ‘ones()’ function and then find the dtft without using the created function.



In

In the second part we are given a sequence of set values. In this case and on the same interval it was much easier to use the created function.

For both cases we plotted the 4 aspects of the complex transform.



Task 2) In Task 2 we were given a sequence x(n) = {1, 2, 2, 1}, at a zero point on the first term. Then it is stated the x2(n) is equal to x1(n) from 0<= n <= 3 and equal to x1(n-4) from 4 <= n <= 7 and zero otherwise.

For this task we were asked to express the DTFT of x2(n) in terms of the DTFT of x1(n) ithout explicitly solving for the dtft of x1(n). The following is the work.

2a) $x_1(n) = \{1, 2, 2, 1\}$

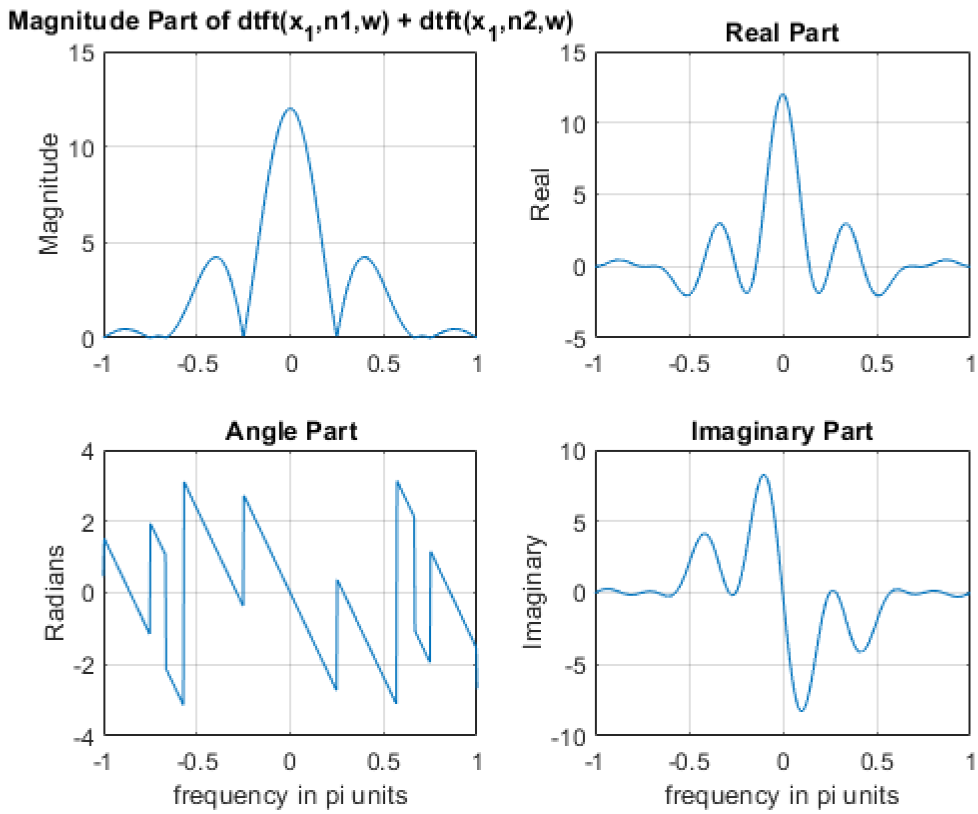
$$x_2(n) = \begin{cases} x_1(n) & 0 \le n \le 3 \\ x_1(x-4), & 4 \le n \le 7 \\ 0, & \text{otherwise} \end{cases}$$

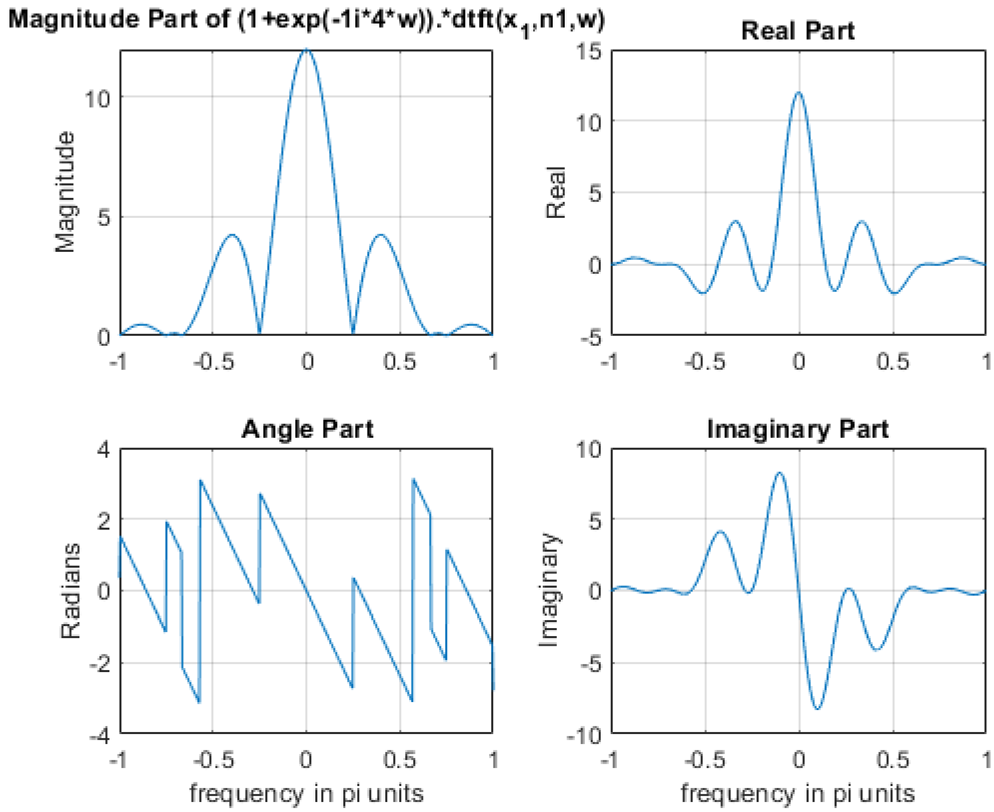
a) $X_2(e^{jw}) = X_1(e^{jw}) + e^{-j4w} X_1(e^{jw})$ for $0 \le n \le 7$

$X_2(e^{jw}) = (1 + e^{-j4w}) X_1(e^{jw})$ i.e. $0 \le K \le \infty$

The second part of task 2 was to confirm the result using MATLAB and plotting the respective DTFT's of x2(n).

The method done here was to use the dtft function on x1(n) for the 2 different ranges and then sum them into a new variable to represent the dtft of x2(n) and plot the results. Then rom there use MATLAB to plot the reduced result determined in part a: X = (1+e^-j4w)*X1(e^jw). Then plot the results.





The results as expected did match and we accomplished the confirmation.

Task 3) Task 3 asked for us to verify the sample shift property of discrete time fourier transforms, using a random sequence uniformly distributed between [0,1], 0 <= n <= 10 and where y(n) = x(n-2).

```
y = sigshift(x,n,n0);
Y1 = dtft(y,n,k);
% prove y1 = y2 .. where y2 = exp(-jn0w)*X(exp(jw))
Y2 = exp(-1i*(pi/5000)*n0)*dtft(x,n,k);
% Y2 is DTFT => exp(-jn0w)*X(exp^jw) ...
% i.e. exp(-jn0w) * dtft of preshifted seq -> x

%%%%%%%%%%%%% Verification
error = max(abs(Y1-Y2)) % Difference
```

error =

0.0080

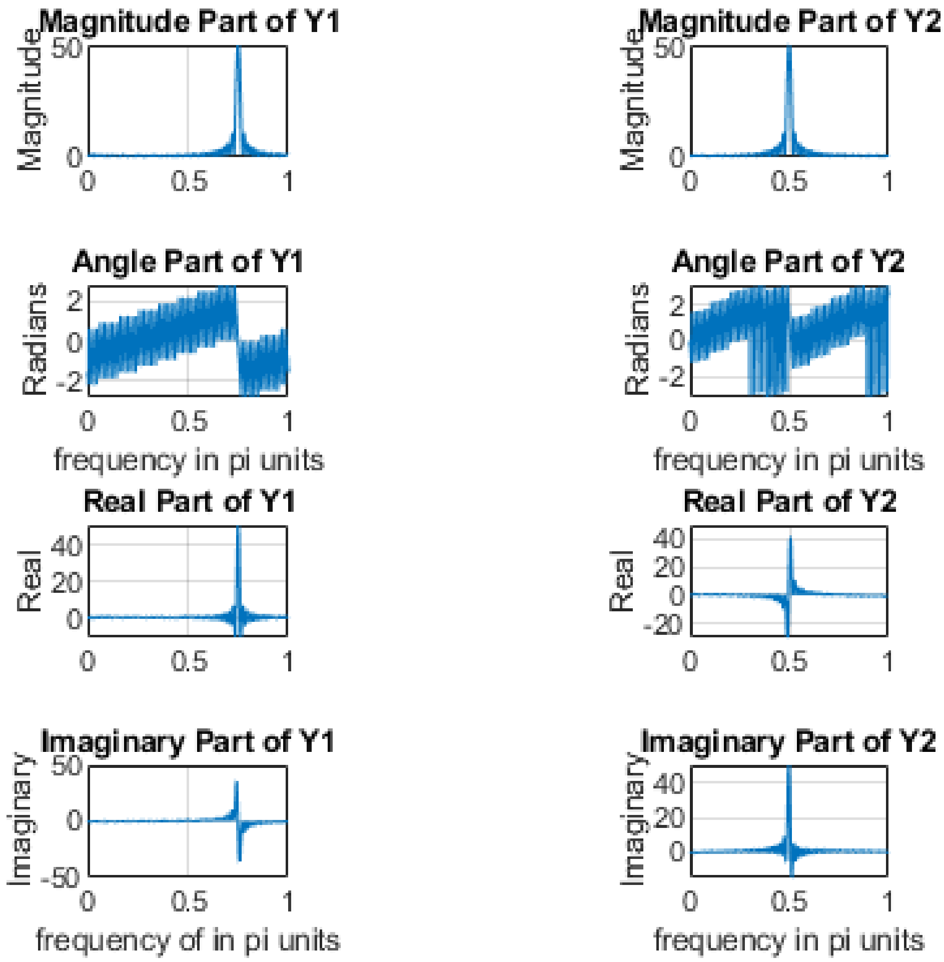
The latter MATLAB shows the methodology. We created (not shown – see Appendix) the random sequence then used the ‘sigshift’ function to shift the sequence before finding its transform. This result we called Y1 and used it to compare with the expected DTFT tansform. The expected transform should be exp^(-jn0w)*X(exp^(jw)), so we found the dtft of the random and multiplied it with exp^(-jn0w) and called this Y2.

Finally we used compared the total result values to get the error ‘error = max(abs(Y1-Y2))’. To which our result was practically zero (.0080), proving that both expressions are the dtft of a shifted sequence.

Task 4) Given a sequence x(n) = cos(pi*n/2) and DTFT y(n) = exp^(j*pi*n/4)*x(n). Similar to the previous task we now intended to prove the Modulation property i.e. given

exp^(j*n*w0)*x(n) -> DTFT -> X(exp^(j(w-w0)))

We used the same methodology and plotted the results as a better comparison of the results.



Again we approached the left side and found the dtft (called Y1)then compared it with he plot of the right side (Y2).

The tricky part had to do with keeping track of the exp^(j(w-w0) DTFT solution.

Conclusion

The lab gave a good introduction into the plotting representation of the Discrete Time Fourier Transform (DTFT) and the transform pair. It of course is also a great exercise in MATLAB as a computer aided design to help us quickly process tricky series expansion, which are core to solving the transform.

Significantly is the fact that the dtft function is fairly straight forward; in itself it is simply multiplying a complex exponential to a sequence with a change in range that matches the new frequency (w) range.

All the tasks went well, and the goal was to verify expected results. Most went well; to note Task 4 was particularly interesting. In verifying the modulation property, where the prompt asked us to plot the results and compare rather than check the error. Out of curiosity the error was at 0.008 (i.e. 0.8% error) between the two results. That's a comparably high error when the previous task asked to verify shifting and the results were so low it reached MATLAB's limits. It was in the plotting where we can see the similarity and contrast the plots to see the effect of the shifting.

Appendix

Full MATLAB code

Chapter 1 EE 386 DTFT

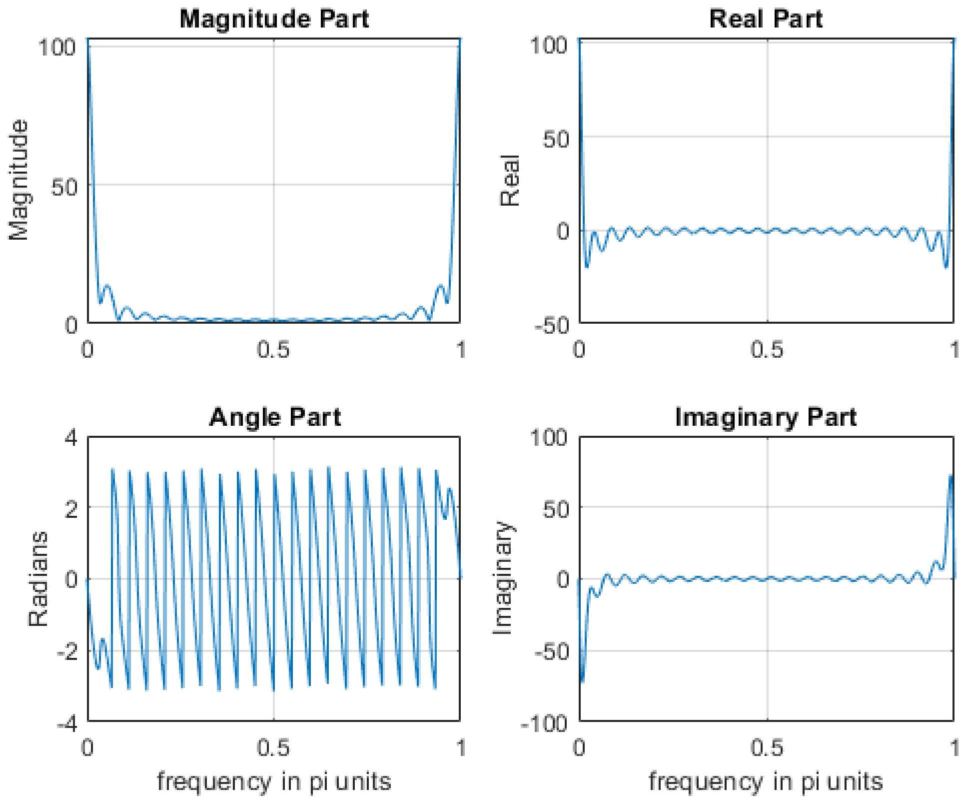
TASK 1

1 Using the matrix-vector multiplication approach discussed in section 2.1 of this lab, write a Matlab function to compute the DTFT of a finite-duration sequence.

Task 1a

over − ≤ ≤ . Plot DTFT magnitude and angle graphs. () = (0.9)^* [() − (− 21)]

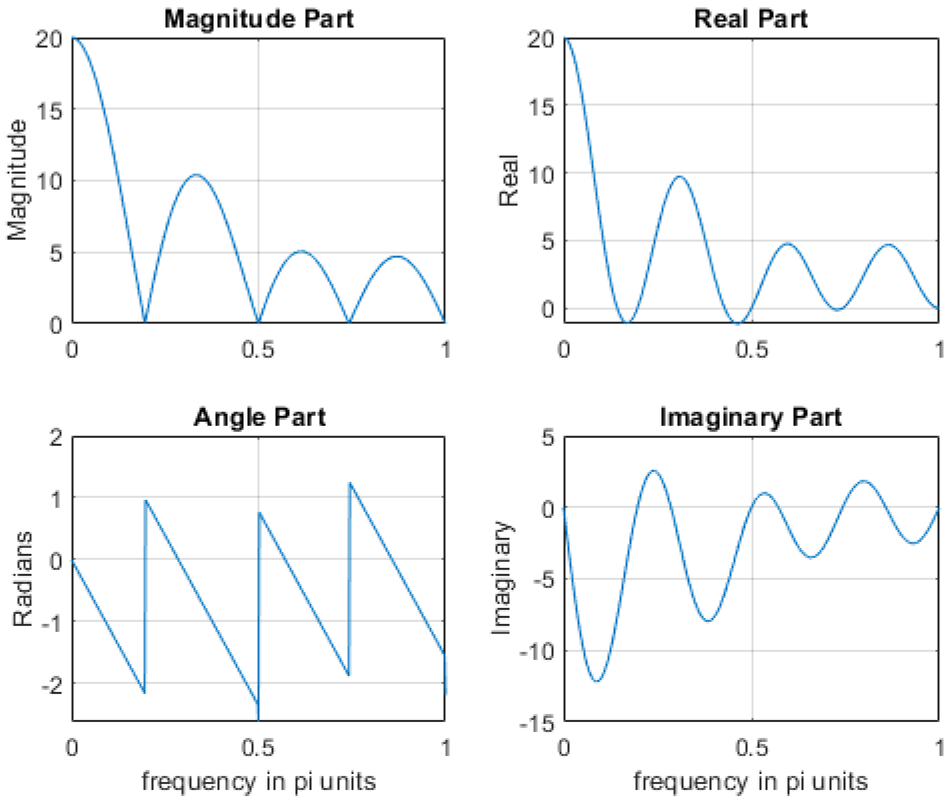
```
w = [0:1:500]*(2*pi)/500; % [0, pi] axis divided into 501 points.
a1 = 0.9.*ones(1,501).*exp(-1i*w);
X1a = (21*a1.^21 - 22*a1.^22 + a1 )./ (1-a1).^2;

magX1a = abs(X1a); angX1a = angle(X1a); realX1a = real(X1a); imagX1a = imag(X1a);

figure(1)
subplot(2,2,1); plot(w/(2*pi),magX1a); grid
title('Magnitude Part'); ylabel('Magnitude')
subplot(2,2,3); plot(w/(2*pi),angX1a); grid
xlabel('frequency in pi units'); title('Angle Part'); ylabel('Radians')
subplot(2,2,2); plot(w/(2*pi),realX1a); grid
title('Real Part'); ylabel('Real')
subplot(2,2,4); plot(w/(2*pi),imagX1a); grid
xlabel('frequency in pi units'); title('Imaginary Part'); ylabel('Imaginary')
```



Task 1b

over − ≤ ≤ . Plot DTFT magnitude and angle graphs of () = {, 3, 2, 1, 1, 2, 3, 4}

```
n = 0:7;
w = [0:1:500]*(2*pi)/500;
x1b = [4 3 2 1 1 2 3 4];
X1b = dtft(x1b,n,w);

% might want to put all this junk in the function too
magX1b = abs(X1b); angX1b = angle(X1b); realX1b = real(X1b); imagX1b = imag(X1b);

figure(2)
subplot(2,2,1); plot(w/(2*pi),magX1b); grid
title('Magnitude Part'); ylabel('Magnitude')
subplot(2,2,3); plot(w/(2*pi),angX1b); grid
xlabel('frequency in pi units'); title('Angle Part'); ylabel('Radians')
subplot(2,2,2); plot(w/(2*pi),realX1b); grid
title('Real Part'); ylabel('Real')
subplot(2,2,4); plot(w/(2*pi),imagX1b); grid
xlabel('frequency in pi units'); title('Imaginary Part'); ylabel('Imaginary')
```



Task 2

```
%Let 1() = { , 2, 2, 1}. A new sequence 2() is formed using:
%   x2(n) = {x1(n), 0<= n <=3} , {x1(n-4), 4<= n <=7}, {0, otherwise}
```

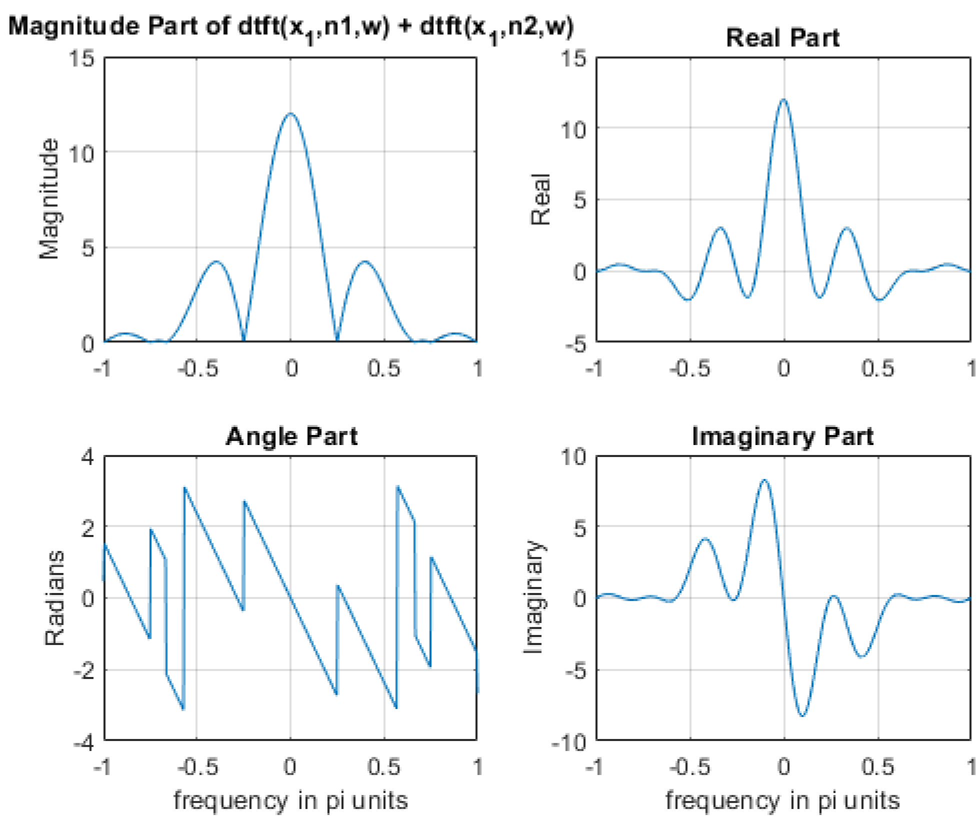
Part A done on paper

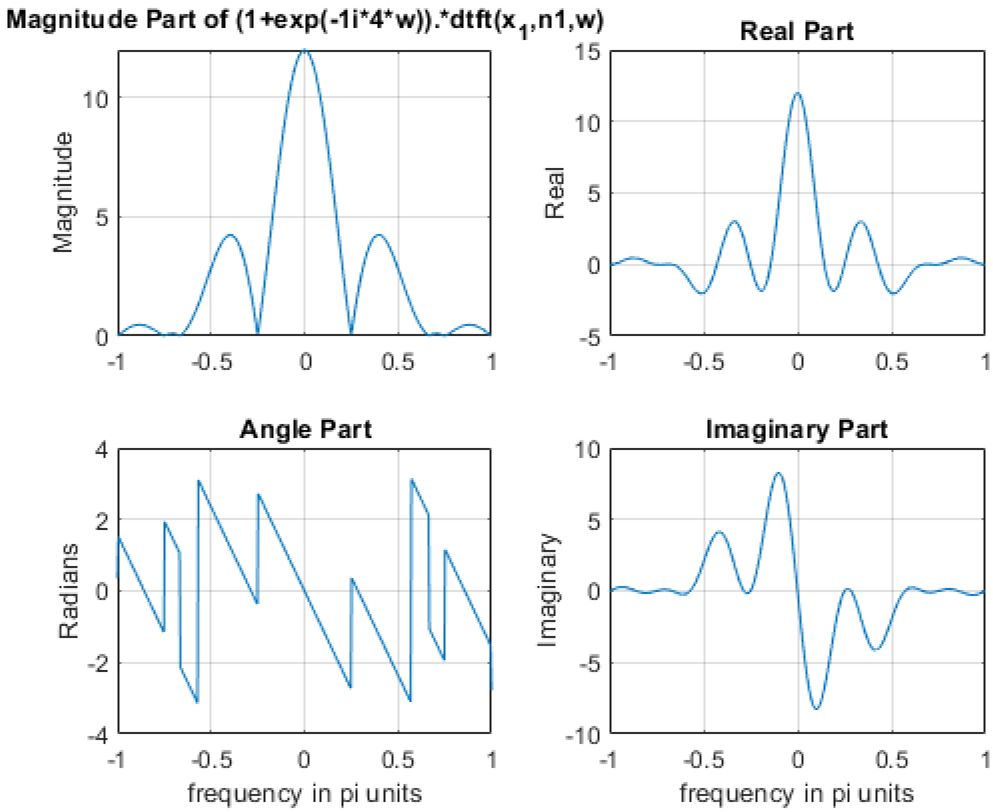
Task 2B

Express X2(e^jw) in terms of X1(e^jw) without explicitly computing X1(e^jw)

```
x_1 = [1 2 2 1]; % seqeunce given
n1 = 0:3; % intervals
n2 = 4:7;
w = [-250:1:250]*(2*pi)/500;

X1_A = dtft(x_1,n1,w); X1_B = dtft(x_1,n2,w);
X2 = X1_A + X1_B;

magX2 = abs(X2); angX2 = angle(X2); realX2 = real(X2); imagX2 = imag(X2);

figure(3)
subplot(2,2,1); plot(w/(pi),magX2); grid % < -- note I changed so it looks like 0 to 2pi
title('Magnitude Part of dtft(x_1,n1,w) + dtft(x_1,n2,w)'); ylabel('Magnitude')
subplot(2,2,2); plot(w/(pi),realX2); grid
title('Real Part'); ylabel('Real')
```

```
subplot(2,2,3); plot(w/(pi),angX2); grid
title('Magnitude Part'); ylabel('Magnitude')
xlabel('frequency in pi units'); title('Angle Part'); ylabel('Radians')
subplot(2,2,4); plot(w/(pi),imagX2); grid
xlabel('frequency in pi units'); title('Imaginary Part'); ylabel('Imaginary')

%%%% now plotting the result from part a

X3 = (1+exp(-1i*4*w)).*dtft(x_1,n1,w);
magX3 = abs(X3); angX3 = angle(X3); realX3 = real(X3); imagX3 = imag(X3);

figure(4)
subplot(2,2,1); plot(w/(pi),magX3); grid % < -- note I changed so it looks like 0 to 2pi
title('Magnitude Part of (1+exp(-1i*4*w)).*dtft(x_1,n1,w)'); ylabel('Magnitude')
subplot(2,2,2); plot(w/(pi),realX3); grid
title('Real Part'); ylabel('Real')
subplot(2,2,3); plot(w/(pi),angX3); grid
title('Magnitude Part'); ylabel('Magnitude')
xlabel('frequency in pi units'); title('Angle Part'); ylabel('Radians')
subplot(2,2,4); plot(w/(pi),imagX3); grid
xlabel('frequency in pi units'); title('Imaginary Part'); ylabel('Imaginary')
```





TASK 3

Let () be a random sequence uniformly distributed between [0, 1] over 0 ≤ ≤ 10, and let () = (− 2). Using these two sequences, write a Matlab script similar to Example 3 to verify the sample shift property.
SamplShft: x(n-n0) => DTFT => exp(-jn0w)*X(exp(jw))

```
s = rng;
a = rand(0,11);
rng(s);
x = rand(1,11); % contains exactly the same values as u1

n = 0:10;
k = 0:5000;
w = (pi/5000)*k; % <----- only needed for plotting

n0 = 2;

y = sigshift(x,n,n0);
Y1 = dtft(y,n,k);
% prove y1 = y2 .. where y2 = exp(-jn0w)*X(exp(jw))
Y2 = exp(-1i*(pi/5000)*n0)*dtft(x,n,k);
% Y2 is DTFT => exp(-jn0w)*X(exp^jw) ...
% i.e. exp(-jn0w) * dtft of preshifted seq -> x
```

```
%%%%%%%%%%%%% Verification
error = max(abs(Y1-Y2)) % Difference
```

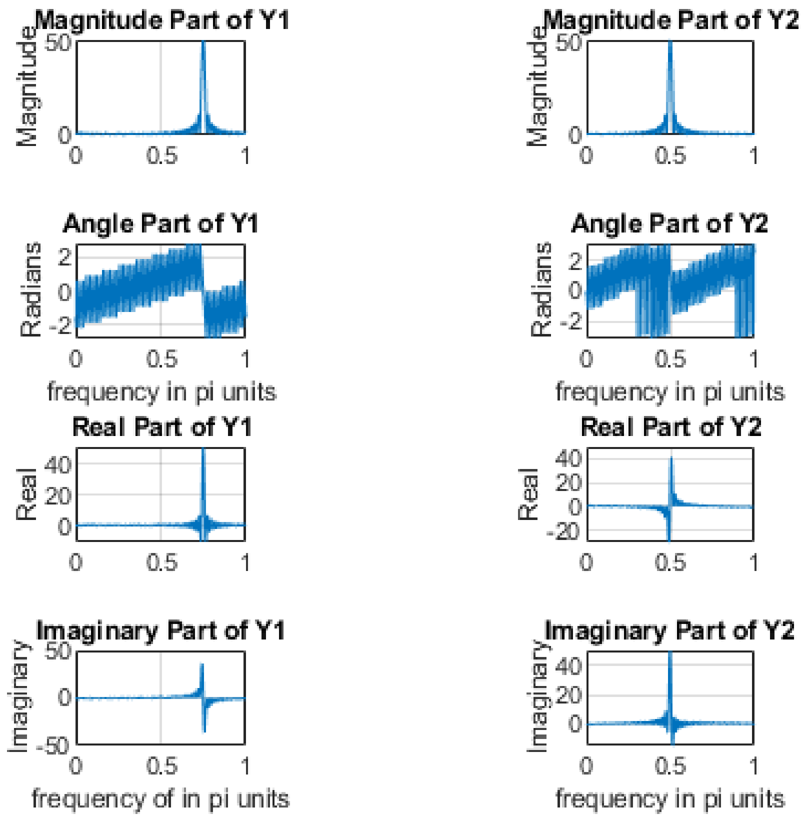
```
error =

    0.0080
```

TASK 4

prove dtft(yn) = DTFT(-w0)

```
n = 0:100;
k = 0:500;
w0 = pi/4;
xn = cos(pi*n/2);
yn = exp(1i*w0*n).*xn;
%  dtft(yn)
Y1 = dtft(yn,n,k);
% vs X(exp(j*(w-w0)))
Y2 = xn*(exp(-1i*((pi/500)))).^((n-w0)'*k);
% or it might be : a -w0 next to the pi/500
% i.e. Y2 = xn*(exp(-1i*((pi/500)-w0))).^(n'*k);

% theory: the angle is the best comparison ... answers aren't very similar

error = max(abs(Y1-Y2)) % Difference

magy1 = abs(Y1); angy1 = angle(Y1); realy1 = real(Y1); imagy1 = imag(Y1);
magy2 = abs(Y2); angy2 = angle(Y2); realy2 = real(Y2); imagy2 = imag(Y2);
figure(5)
subplot(4,4,1); plot(k/500,magy1); grid
title('Magnitude Part of Y1'); ylabel('Magnitude')
subplot(4,4,3); plot(k/500,magy2); grid
title('Magnitude Part of Y2'); ylabel('Magnitude')

subplot(4,4,5); plot(k/500,angy1); grid
xlabel('frequency in pi units'); title('Angle Part of Y1'); ylabel('Radians')
subplot(4,4,7); plot(k/500,angy2); grid
xlabel('frequency in pi units'); title('Angle Part of Y2'); ylabel('Radians')

subplot(4,4,9); plot(k/500,realy1); grid
title('Real Part of Y1'); ylabel('Real')
subplot(4,4,11); plot(k/500,realy2); grid
title('Real Part of Y2'); ylabel('Real')

subplot(4,4,13); plot(k/500,imagy1); grid
xlabel('frequency of in pi units'); title('Imaginary Part of Y1'); ylabel('Imaginary')
subplot(4,4,15); plot(k/500,imagy2); grid
xlabel('frequency in pi units'); title('Imaginary Part of Y2'); ylabel('Imaginary')
```

```
error =

   50.0574
```



Published with MATLAB® R2021b