Table of Contents

lab4.m

```
clear
delete(allchild(0));
w = linspace(-pi, pi, 11);
x = sequence([1 4 3 -2 6], -1);
%x = sequence([1 5 2 -1 4 1], -2);
```

Problem #1: Even, Odd

```
test_lab4('even(x)');
test_lab4('odd(x)');
test_lab4('trim(plus(even(x), odd(x)))');
```

```
_____________
even(x): sequence O.K.
Your answer:

z =

  sequence with properties:

      data: [3 -1 2 4 2 -1 3]
    offset: -3


_____________
odd(x): sequence O.K.
Your answer:

z =

  sequence with properties:

      data: [-3 1 -1 0 1 -1 3]
    offset: -3


_____________
trim(plus(even(x), odd(x))): sequence O.K.
Your answer:
```

```
z =

  sequence with properties:

      data: [1 4 3 -2 6]
    offset: -1
```

Problem #2: DTFT

```
x = sequence([1 1 1], -1);
test_lab4('dtft(x, w)');

% Simple impulse Caution! check your answer for this.
% It should be a sequence.
x = sequence(1, 0);
test_lab4('dtft(x, w)');

%x = sequence([1 4 3 -2 6], -1)
x = sequence([1 3 -1 -4 1], -2);
test_lab4('dtft(x, w)');

%x = sequence([1 4 3 -2 6], -1)
x = sequence([1+j 0 1-j], -1);
test_lab4('dtft(x, w)-dtft(conj(flip(x)), w)');
```

```
_____________
dtft(x, w): data O.K.
Your answer:

z =

  Columns 1 through 7

   -1.0000   -0.6180    0.3820    1.6180    2.6180    3.0000    2.6180

  Columns 8 through 11

    1.6180    0.3820   -0.6180   -1.0000

_____________
dtft(x, w): data O.K.
Your answer:

z =

     1     1     1     1     1     1     1     1     1     1     1

_____________
dtft(x, w): data O.K.
Your answer:
```

```
z =

  Columns 1 through 4

   2.0000 - 0.0000i   0.4271 - 4.1145i  -2.3090 - 6.6574i  -2.9271 -
 6.6574i

  Columns 5 through 8

  -1.1910 - 4.1145i   0.0000 + 0.0000i  -1.1910 + 4.1145i  -2.9271 +
 6.6574i

  Columns 9 through 11

  -2.3090 + 6.6574i   0.4271 + 4.1145i   2.0000 + 0.0000i
```

```
dtft(x, w)-dtft(conj(flip(x)), w): data O.K.
Your answer:

z =

     0     0     0     0     0     0     0     0     0     0     0
```

Problem #3: Real and Imaginary

```
x = sequence([1 1 1 1 1], -1);
test_lab4('dtft2(x, w)');

%x = sequence([1 4 3 -2 6], -1);
x = sequence([1 2 2 -1 2 1], -2);
test_lab4('dtft2(x, w)');
```

```
dtft2(x, w): data O.K.
Your answer:

z =

  prop with properties:

    real: [1×11 double]
    imag: [1×11 double]
```

```
dtft2(x, w): data O.K.
Your answer:

z =

  prop with properties:
```

```
    real: [3 2.4271 0.0729 -0.9271 3.4271 7 3.4271 -0.9271 0.0729
 2.4271 3]
    imag: [1×11 double]
```

Problem #4: Magnitude and Phase

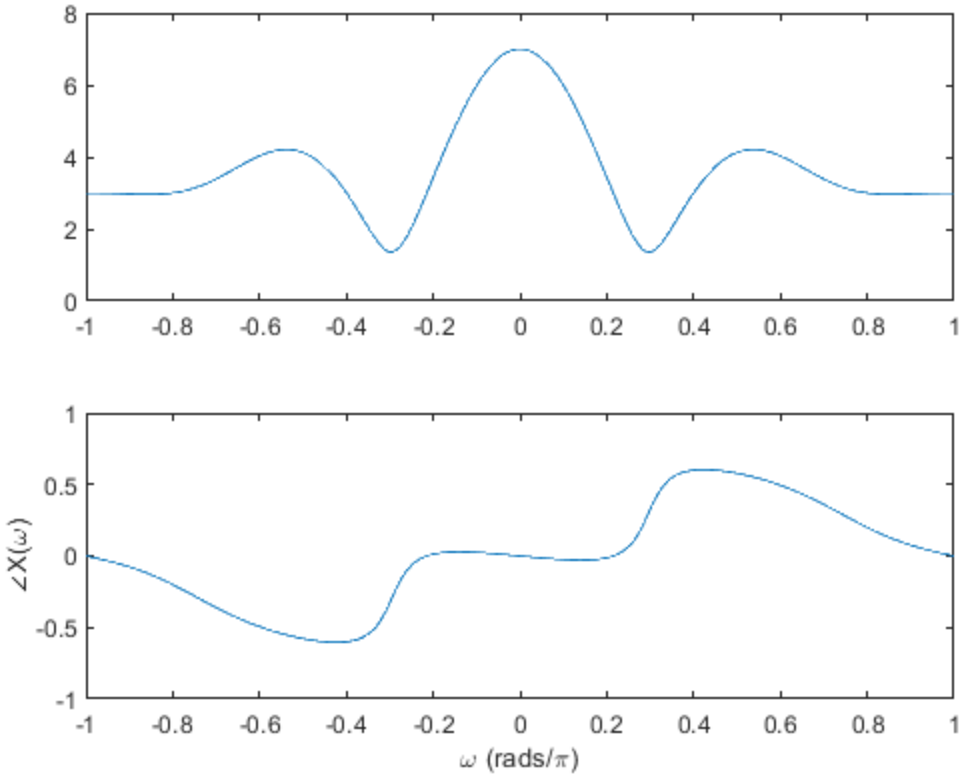
```
test_lab4('mag_phase(dtft2(x, w))');
```

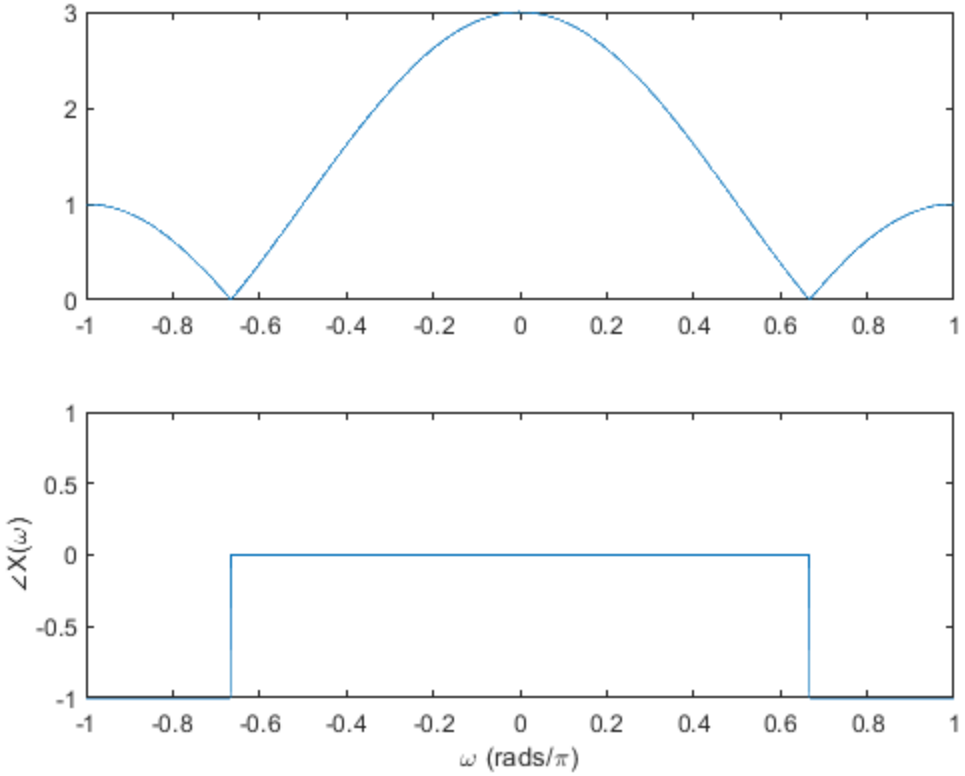
```
_____________
mag_phase(dtft2(x, w)): data O.K.
Your answer:

z =

  m with properties:

      mag: [3 3.0000 4.0294 3.0000 3.4299 7 3.4299 3.0000 4.0294
 3.0000 3]
    phase: [1×11 double]
```

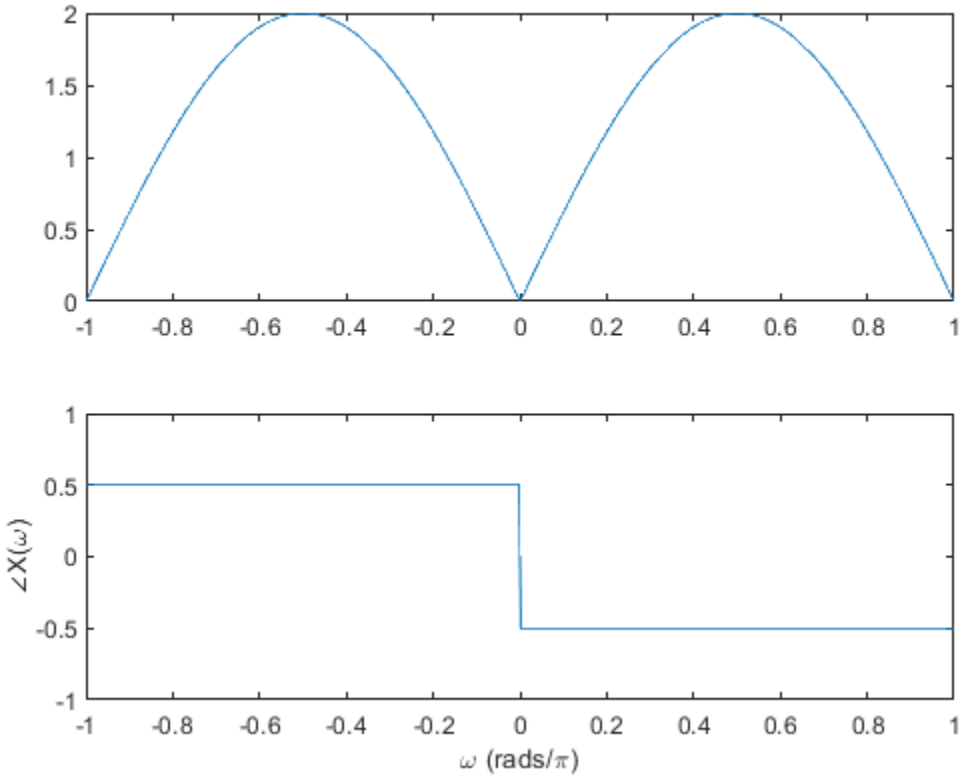
Problem #5 Plotting

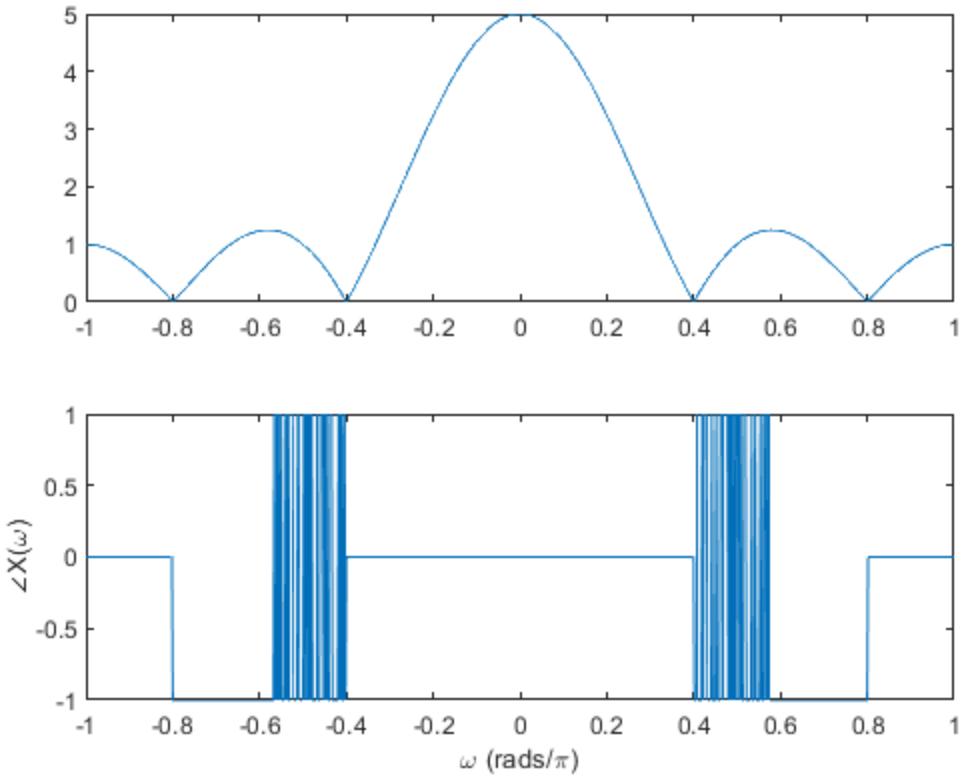
```
w = linspace(-pi, pi, 1001);
plot_magph(x, w);

% This is a purely real and even function.
% What can you say about the phase?
% Specifically why is it either 0 or pi?
x = sequence([1 1 1], -1);
set(figure, 'Color', 'w');
plot_magph(x, w);

% This is a purely real and odd function.
% What can you say about the phase?
% Specifically why is it either +pi/2 or -pi/2?
x = sequence([-1 0 1], -1);
set(figure, 'Color', 'w');
plot_magph(x, w);

% Here are a series of pulse functions.
% What happens to the magnitude of the transform as the pulse gets
 broader?
% You may note that the phase 'chatters' between +pi and -pi at some
 values of w.
% This doesn't look nice and it's confusing. How could you fix this in
 your plot_magph
% program so that the phase doesn't chatter? No biggie if you can't.
% (Hint: it has something to do with a very small imaginary part...).

x = sequence(ones(1, 5), -2);
```

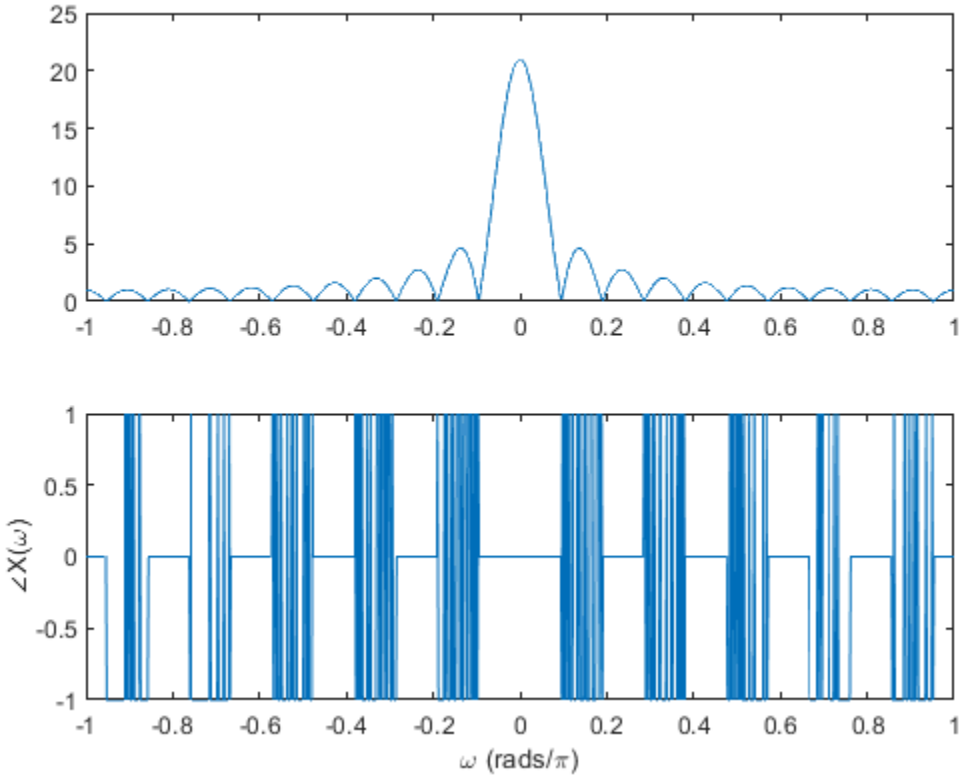
```
set(figure, 'Color', 'w');
plot_magph(x, w)

x = sequence(ones(1, 21), -10);
set(figure, 'Color', 'w');
plot_magph(x, w)

x = sequence(ones(1, 101), -50);
set(figure, 'Color', 'w');
plot_magph(x, w)
```

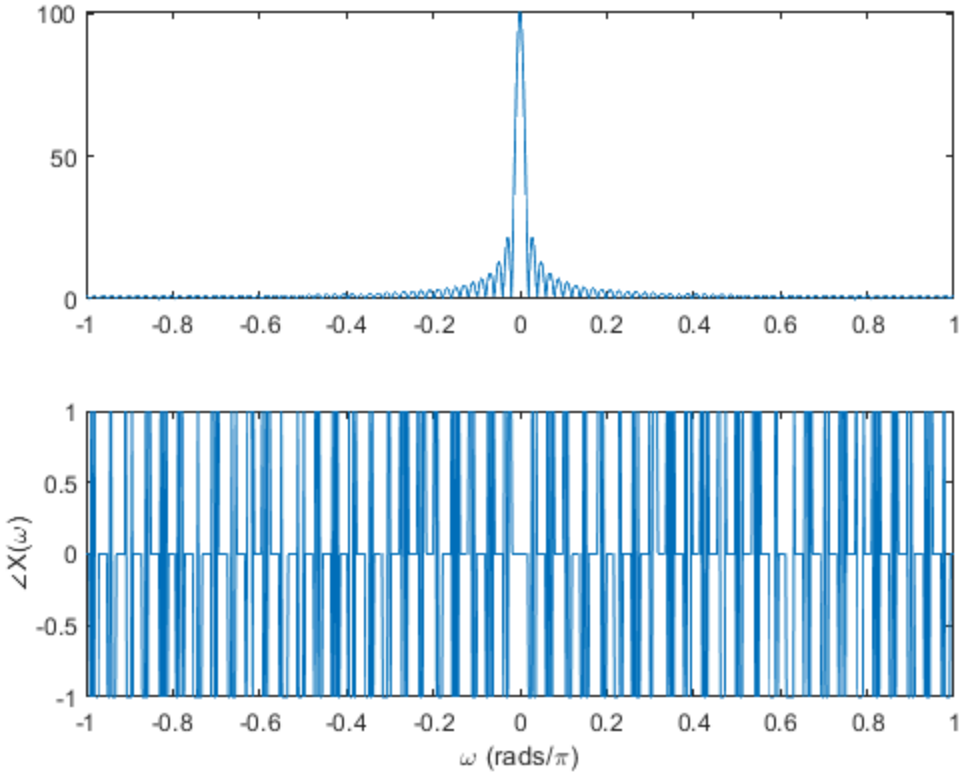



3
2
1
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
1
0.5
0
-0.5
-1
∠X(ω)
ω (rads/π)


2
1.5
1
0.5
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
1
0.5
0
-0.5
-1
∠X(ω)
ω (rads/π)


5
4
3
2
1
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
1
0.5
0
-0.5
-1
∠X(ω)
ω (rads/π)


25
20
15
10
5
0
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
1
0.5
0
-0.5
-1
∠X(ω)
ω (rads/π)



Print programs

```
disp(' ')
disp('--- dtft.m --------------------------------')
type('dtft')
disp('--- dtft2.m -------------------------------')
type('dtft2')
disp('--- mag_phase.m ---------------------------')
type('mag_phase')
disp('--- plot_magph.m --------------------------')
type('plot_magph')


--- dtft.m --------------------------------

function y = dtft(x,w)
 assert(isa(x,'sequence'),'DTFT error, x is not a sequence',class(x));
 %needed to be able to use sequence properties
 Output = zeros(1,length(w));
 for k = 1:length(w)
  for n = x.offset:length(x.data)+x.offset - 1
   Output(k) = Output(k) + (x.data(n-x.offset+1)*exp(-1j * w(k) * n));
  end
 end
 y = Output;
end
```

```
--- dtft2.m ------------------------------

function y = dtft2(x,w)
 assert(isa(x,'sequence'),'DTFT2 error, x is not a
 sequence',class(x)); %needed to be able to use sequence properties
 R = zeros(1,length(w));
 I = zeros(1,length(w));
 for k = 1:length(w)
  for n = x.offset:length(x.data)+x.offset - 1
   R(k) = R(k) + (x.data(n-x.offset + 1)*cos(w(k)*n));
   I(k) = I(k) + (x.data(n-x.offset + 1)*sin(w(k)*n));
  end
 end
 I = I * -1;
 y = prop(R,I); %to a class with only definitions for real and
 imaginary numbers
end
--- mag_phase.m --------------------------

function y = mag_phase(x)
 assert(isa(x,'prop'),'mag_phase error, x is not a prop',class(x));
 %needed to be able to use real and imaginary properties
 for i=1:length(x.real)
  mag(i) = sqrt((x.real(i)^2) + (x.imag(i)^2));
  ph(i) = atan2(x.imag(i),x.real(i));
 end
 y = m(mag,ph); %to a class with only definitions for magnitude and
 phase numbers
end
--- plot_magph.m -------------------------

function plot_magph(x,w)
 assert(isa(x,'sequence'),'plot_magph error, x is not a
 sequence',class(x));%needed to be able to use magnitude and phase
 properties
 z = dtft2(x,w);
 y = mag_phase(z);
 subplot(2, 1, 1); %top plot
 plot(w/pi,y.mag);
 subplot(2, 1, 2); %bottom plot
 plot(w/pi,y.phase/pi);
 ylim([-1, 1]);
 xlabel('\omega (rads/\pi)');
 ylabel('\angleX(\omega)');
end
```

Published with MATLAB® R2020a