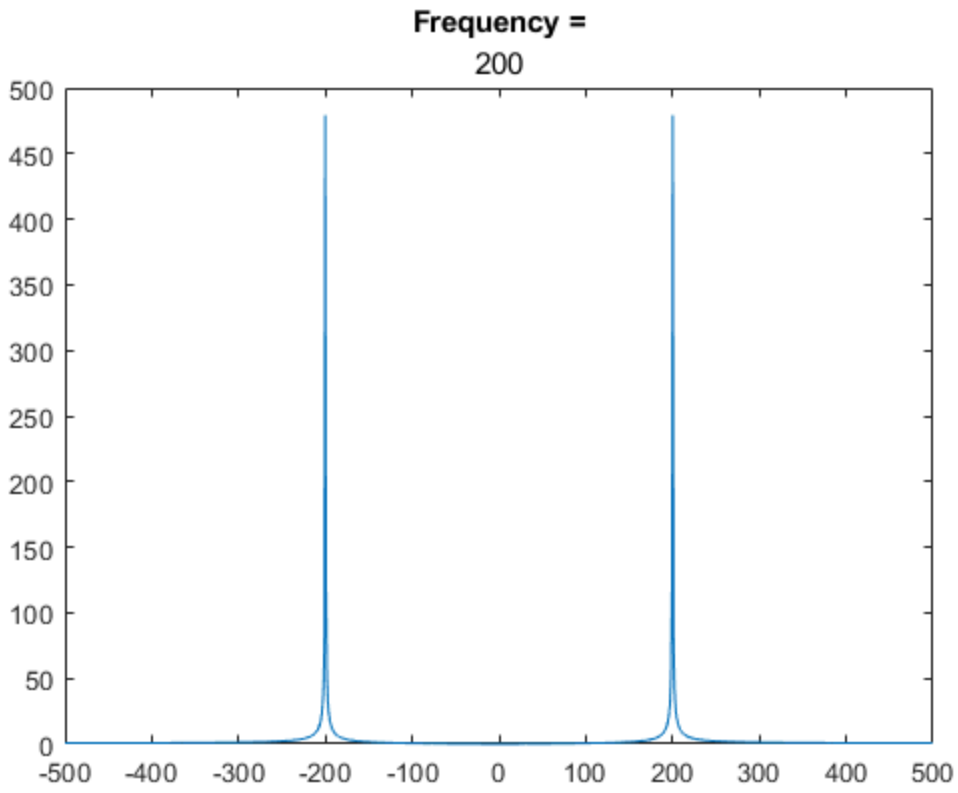
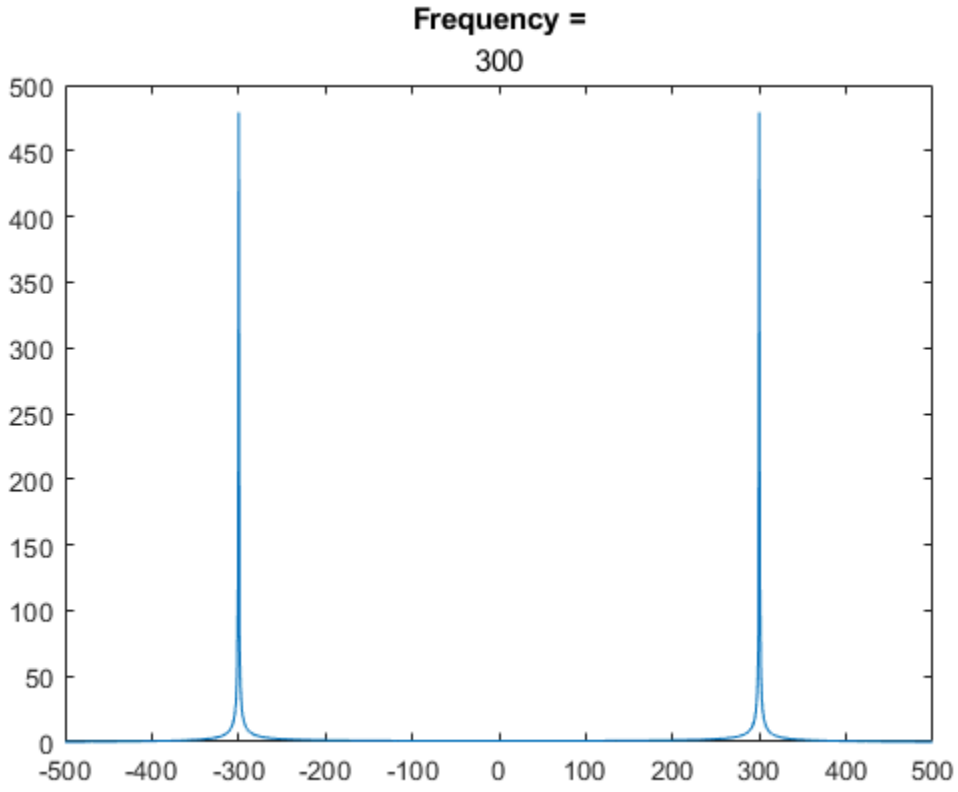
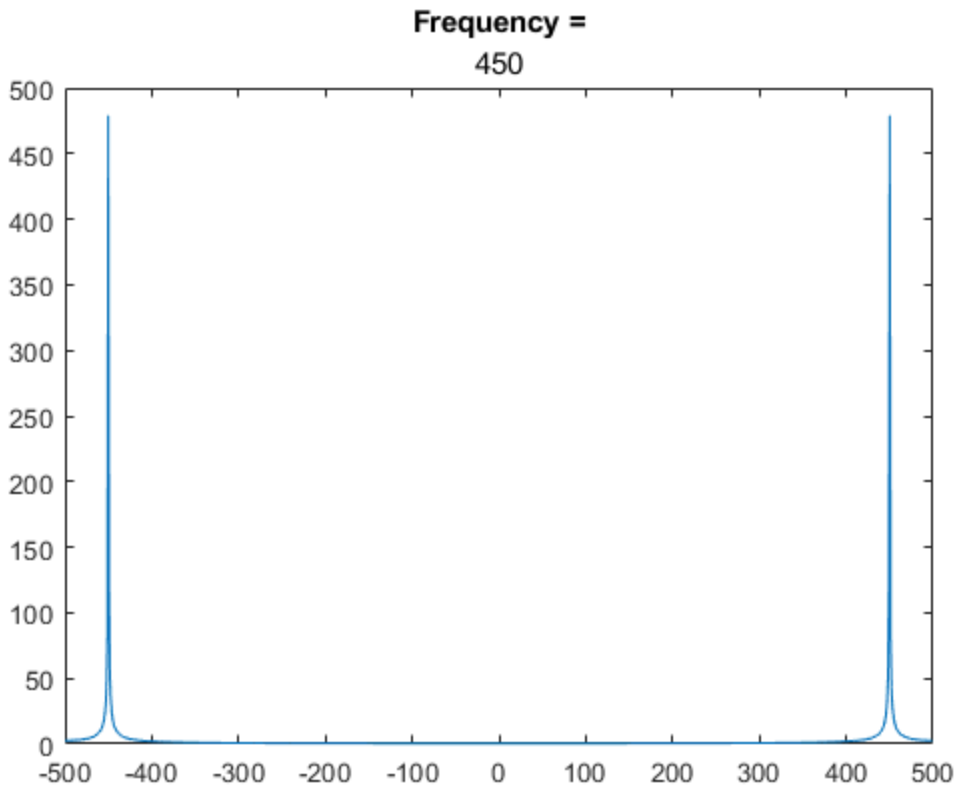
Table of Contents

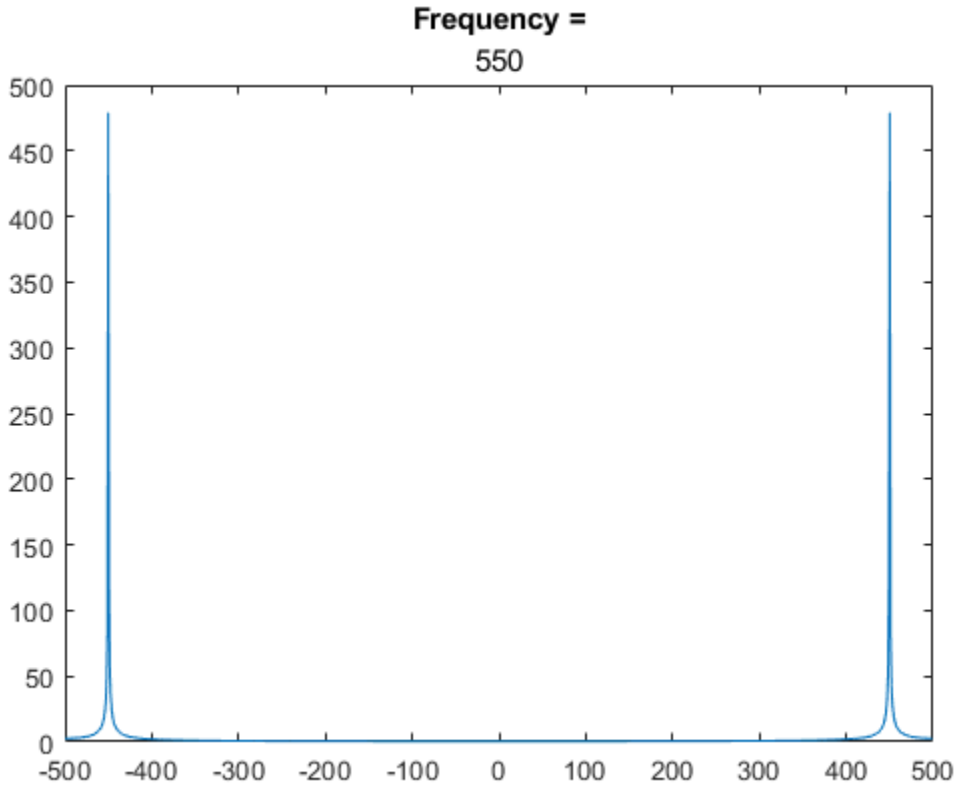
Exercise 7.5

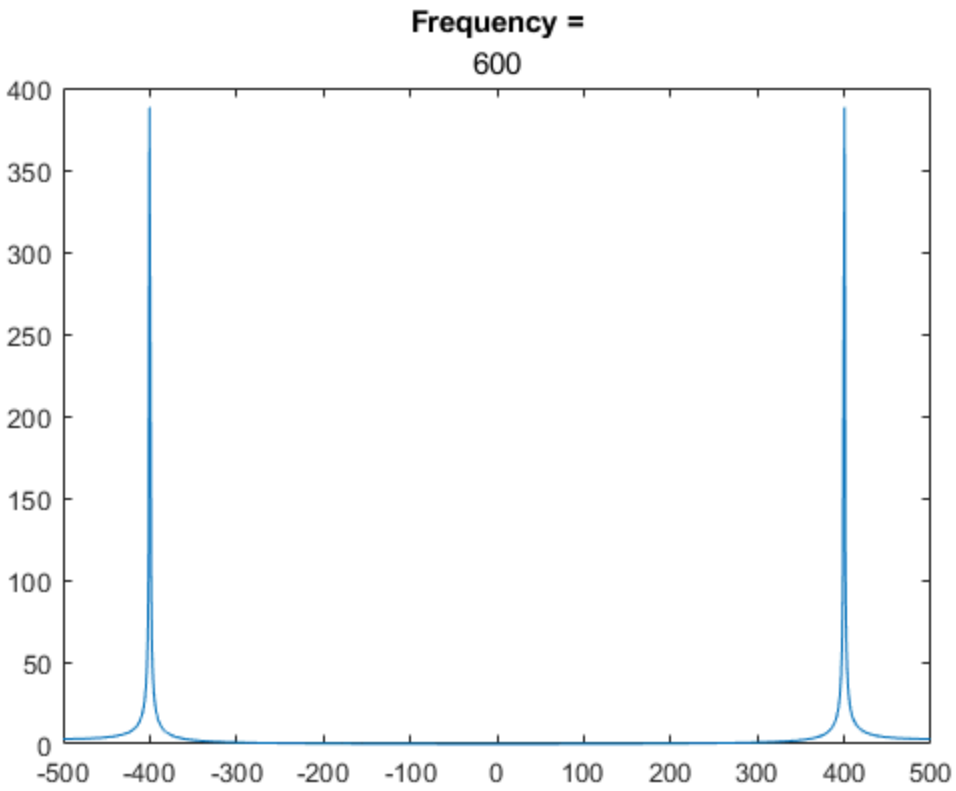
```
f = [200, 300, 450, 550, 600, 800, 2200];
Ts=1/1000; time=10.0; % freq, sampling interval, time
t=Ts:Ts:time ; % define a time vector
for i=1:7
w=sin(2*pi*f(i)*t); % define the sinusoid
N=2^10; % size of analysis window
ssf =(-N/2:N/2-1)/(Ts*N); % frequency vector
fw = fft(w(1:N)); % do DFT/FFT
fws = fftshift(fw); % shift it for plotting
figure
plot (ssf ,abs(fws)) % plot magnitude spectrum
title('Frequency =', num2str(f(i)))
end
```

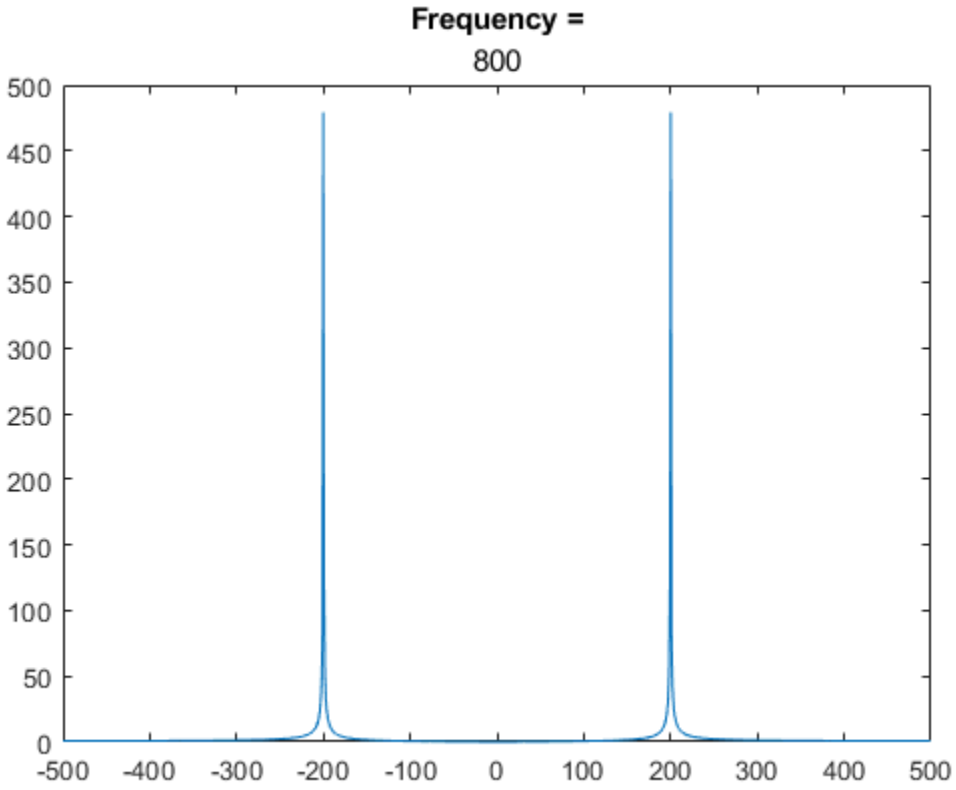

Frequency =
200
500
450
400
350
300
250
200
150
100
50
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

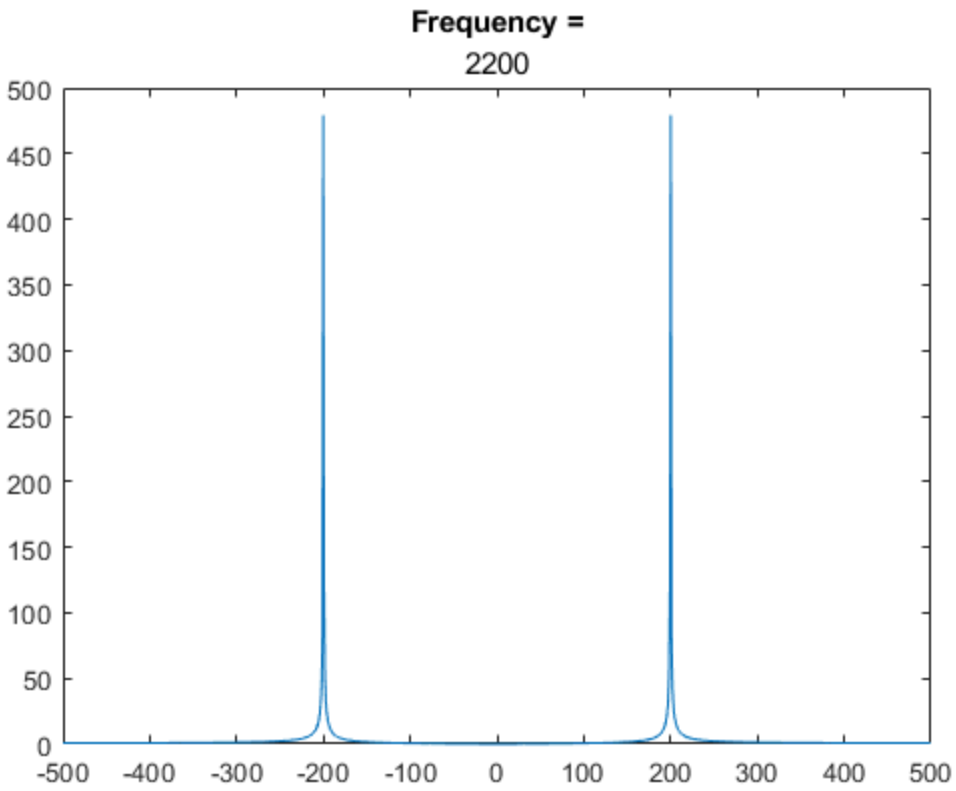

Frequency =
300
500
450
400
350
300
250
200
150
100
50
0
-500
-400
-300
-200
-100
0
100
200
300
400
500


Frequency =
450
500
450
400
350
300
250
200
150
100
50
0
-500
-400
-300
-200
-100
0
100
200
300
400
500


Frequency =
550
500
450
400
350
300
250
200
150
100
50
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

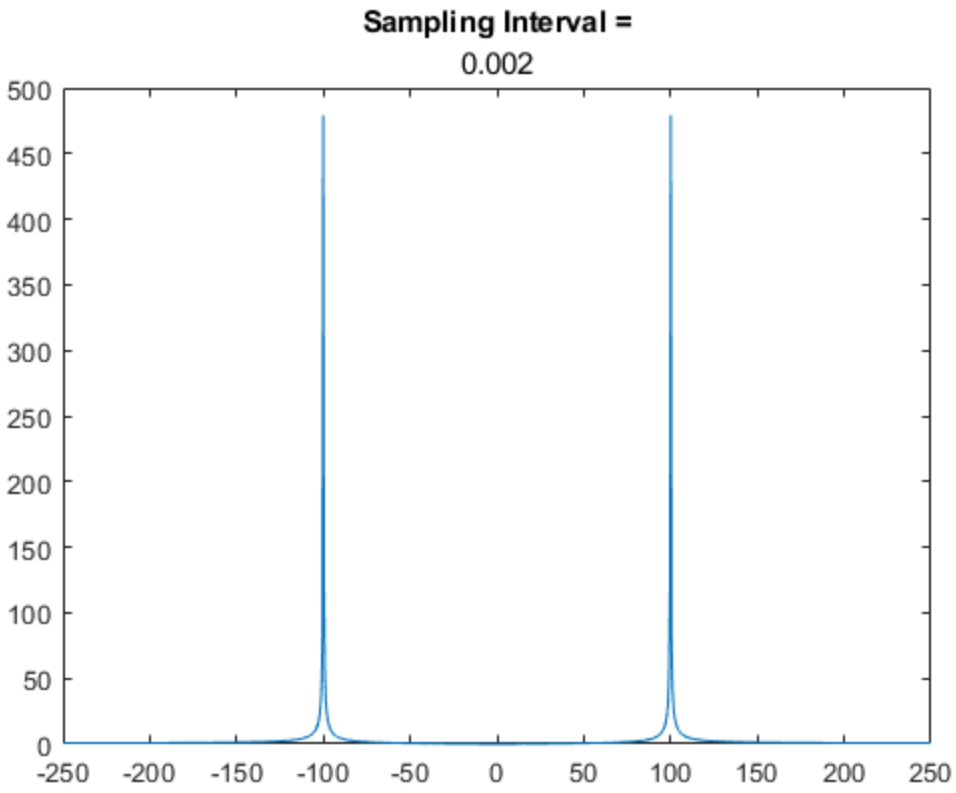

Frequency =
600
400
350
300
250
200
150
100
50
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

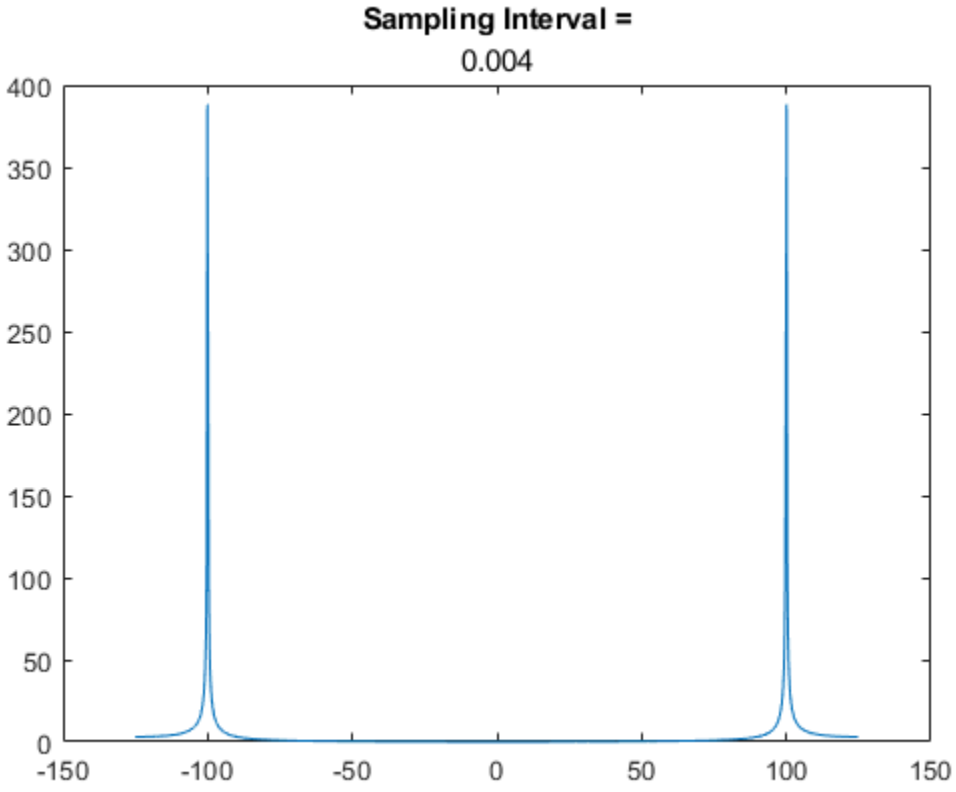

Frequency =
800
500
450
400
350
300
250
200
150
100
50
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

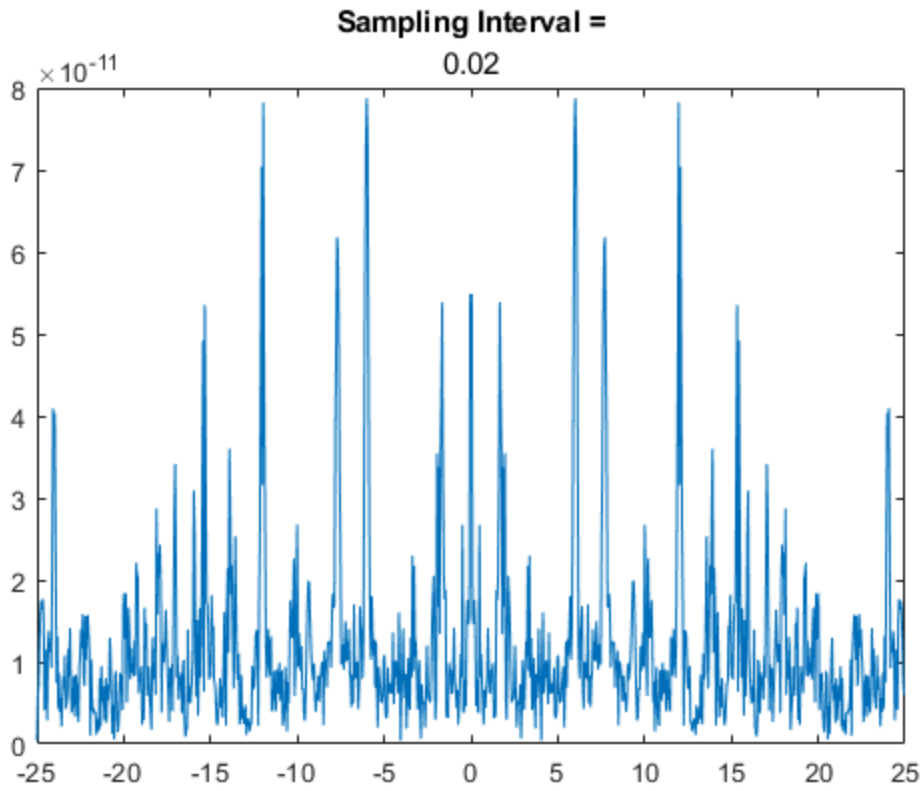


Exercise 7.5b

```
f = 100;
Ts=[1/500, 1/250, 1/50];
time=100.0; % freq, sampling interval, time
for i=1:3
t=Ts(i):Ts(i):time ; % define a time vector
w=sin(2*pi*f*t); % define the sinusoid
N=2^10; % size of analysis window
ssf =(-N/2:N/2-1)/(Ts(i)*N); % frequency vector
fw = fft(w(1:N)); % do DFT/FFT
fws = fftshift(fw); % shift it for plotting
figure
plot (ssf ,abs(fws)) % plot magnitude spectrum
title('Sampling Interval =', num2str(Ts(i)))
end
```

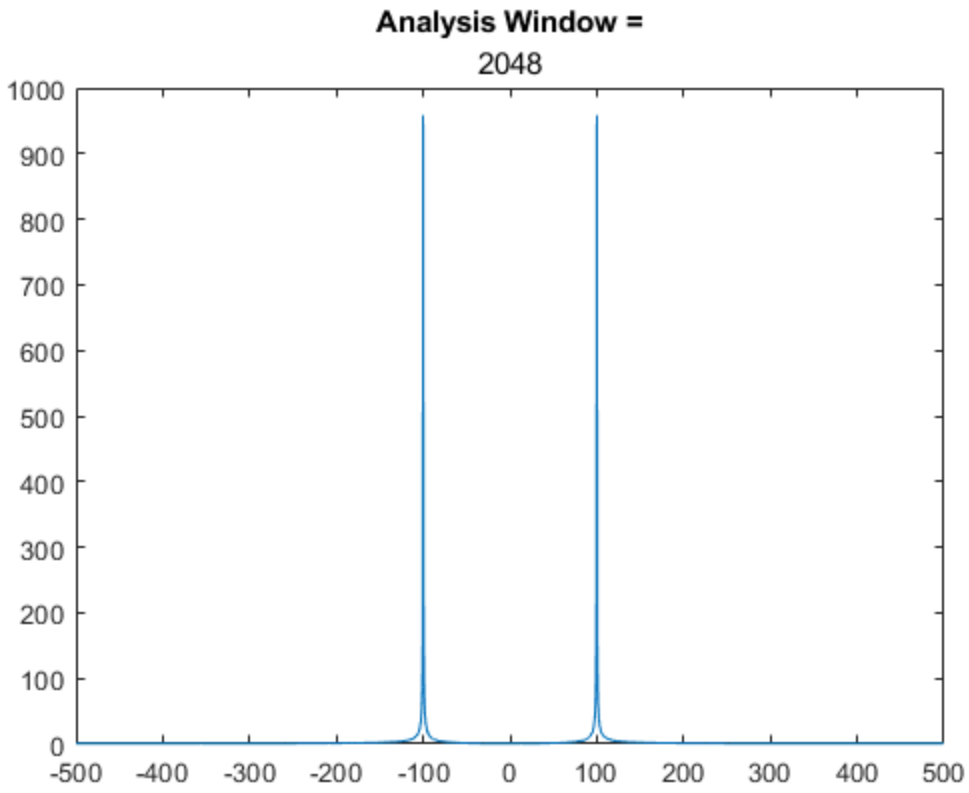

Sampling Interval =
0.002
500
450
400
350
300
250
200
150
100
50
0
-250
-200
-150
-100
-50
0
50
100
150
200
250

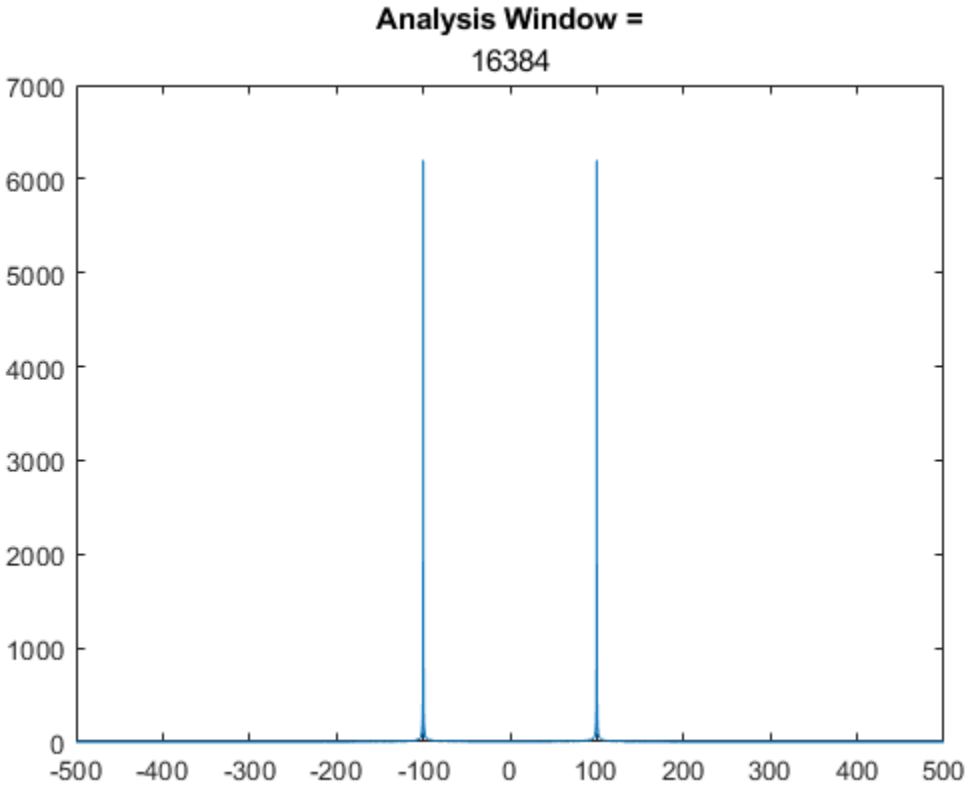

Sampling Interval =
0.004
400
350
300
250
200
150
100
50
0
-150
-100
-50
0
50
100
150

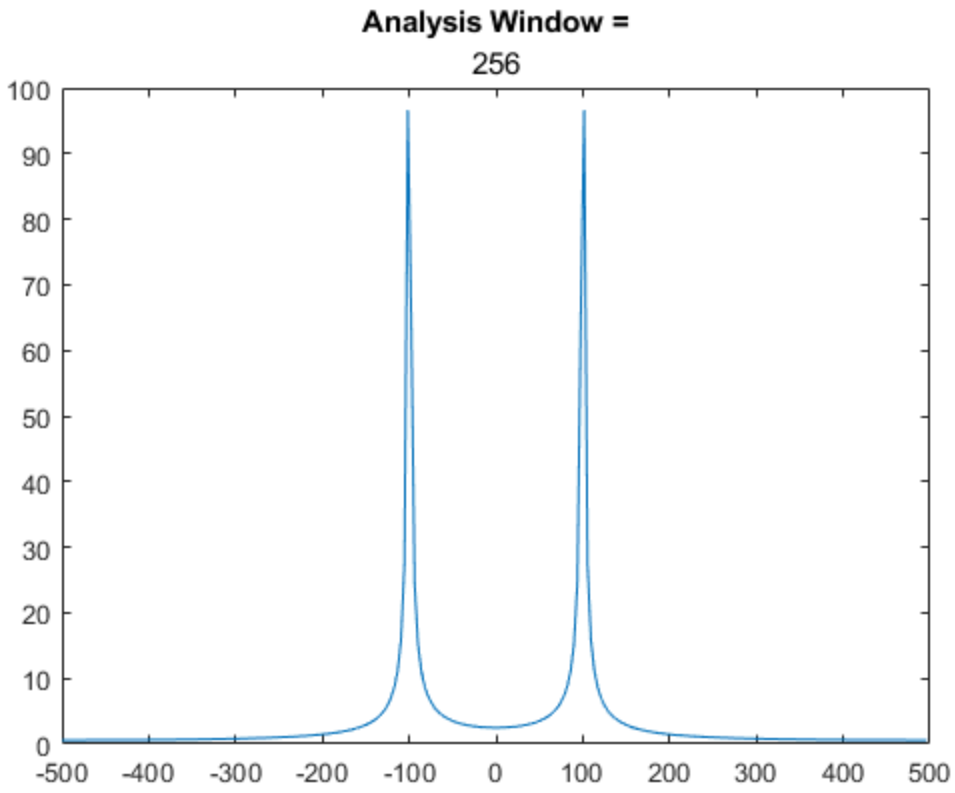


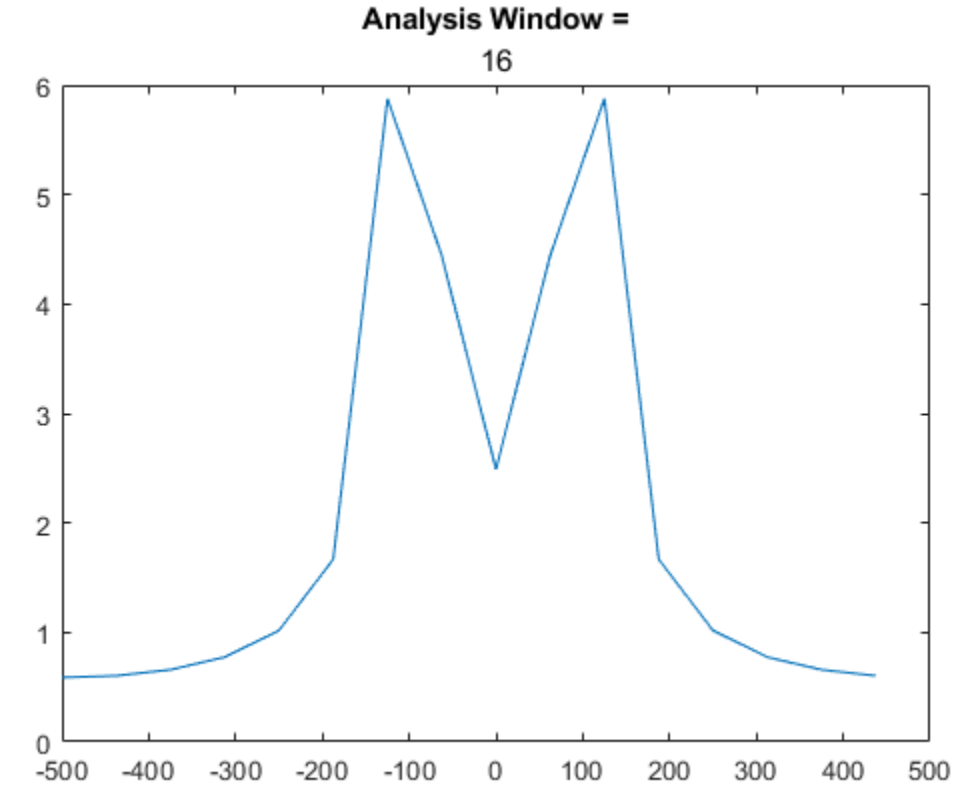
Exercise 7.5c

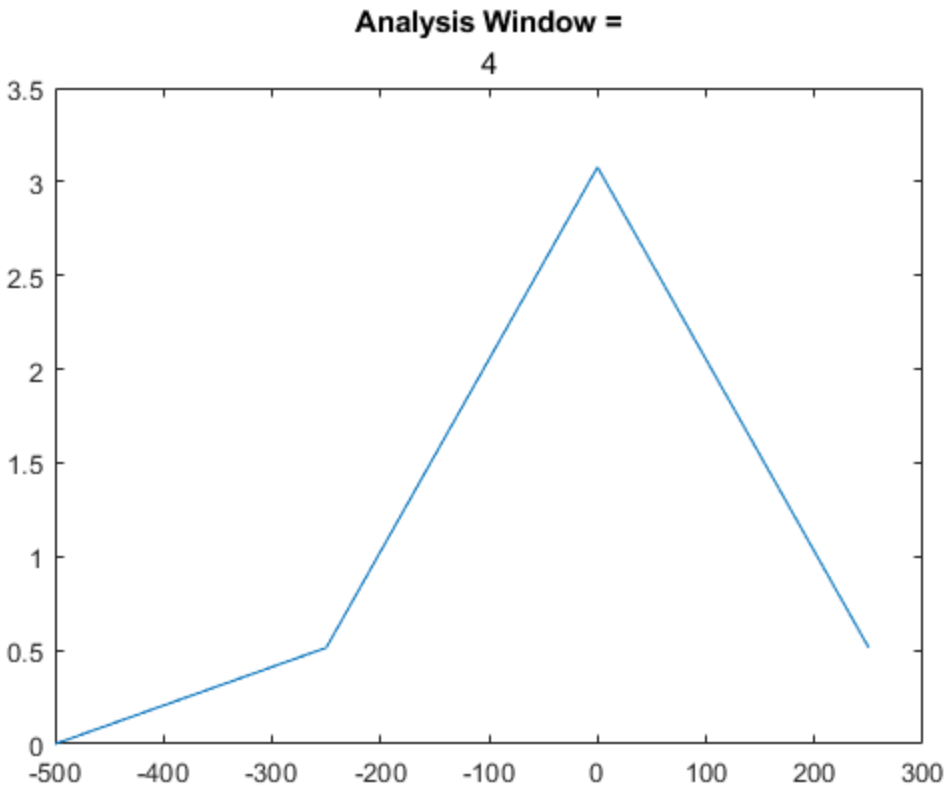
```
f = 100;
Ts = 1/1000;
time=20000.0; % freq, sampling interval, time
t=Ts:Ts:time ; % define a time vector
w=sin(2*pi*f*t); % define the sinusoid
N=[2^11, 2^14, 2^8, 2^4, 2^2, 2^20]; % size of analysis window
for i=1:6
ssf =(-N(i)/2:N(i)/2-1)/(Ts*N(i)); % frequency vector
fw = fft(w(1:N(i))); % do DFT/FFT
fws = fftshift(fw); % shift it for plotting
figure
plot (ssf ,abs(fws)) % plot magnitude spectrum
title('Analysis Window =', num2str(N(i)))
end
```

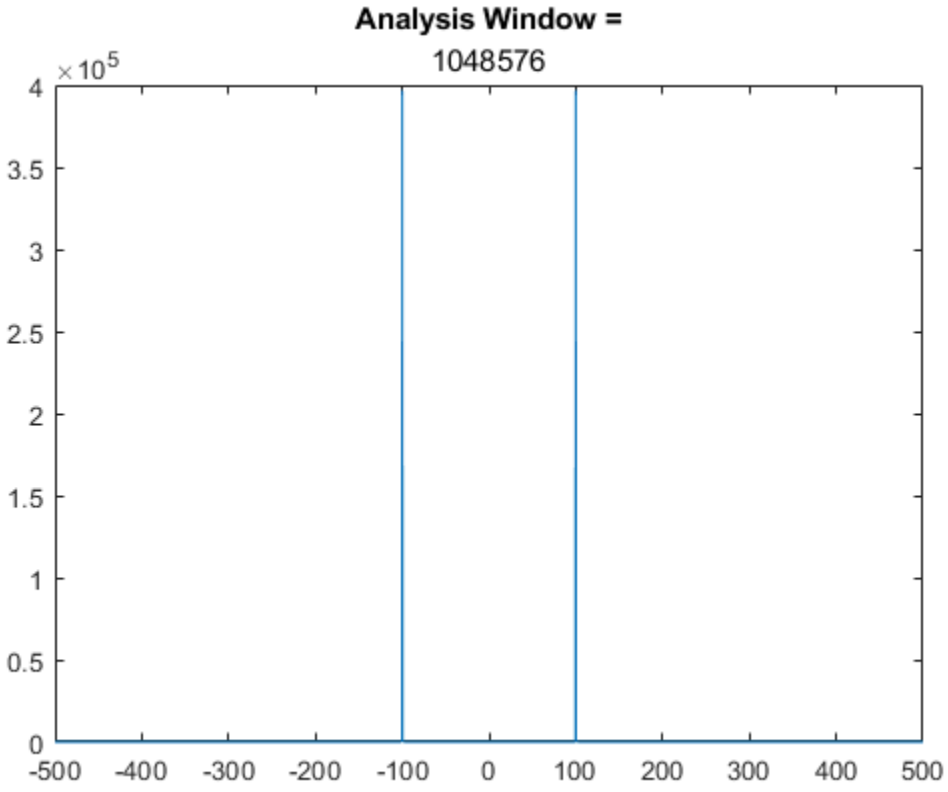

Analysis Window =
2048
1000
900
800
700
600
500
400
300
200
100
0
-500
-400
-300
-200
-100
0
100
200
300
400
500


Analysis Window =
16384
7000
6000
5000
4000
3000
2000
1000
0
-500
-400
-300
-200
-100
0
100
200
300
400
500


Analysis Window =
256
100
90
80
70
60
50
40
30
20
10
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

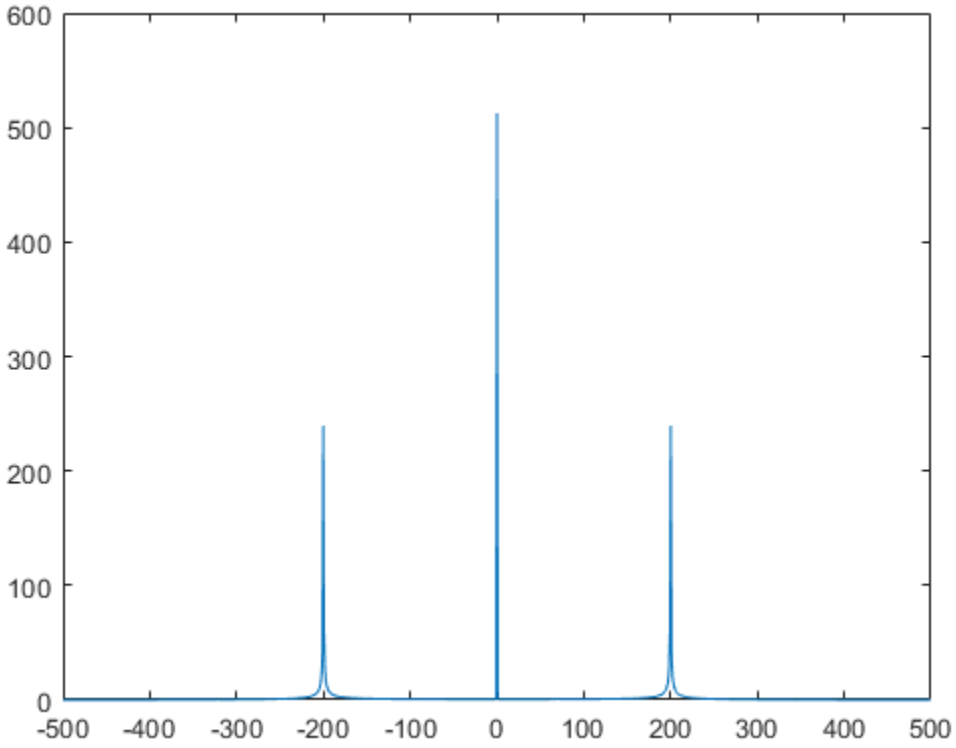

Analysis Window =
16
6
5
4
3
2
1
0
-500
-400
-300
-200
-100
0
100
200
300
400
500


Analysis Window =
4
3.5
3
2.5
2
1.5
1
0.5
0
-500
-400
-300
-200
-100
0
100
200
300


Analysis Window =
1048576
×10^5
4
3.5
3
2.5
2
1.5
1
0.5
0
-500
-400
-300
-200
-100
0
100
200
300
400
500

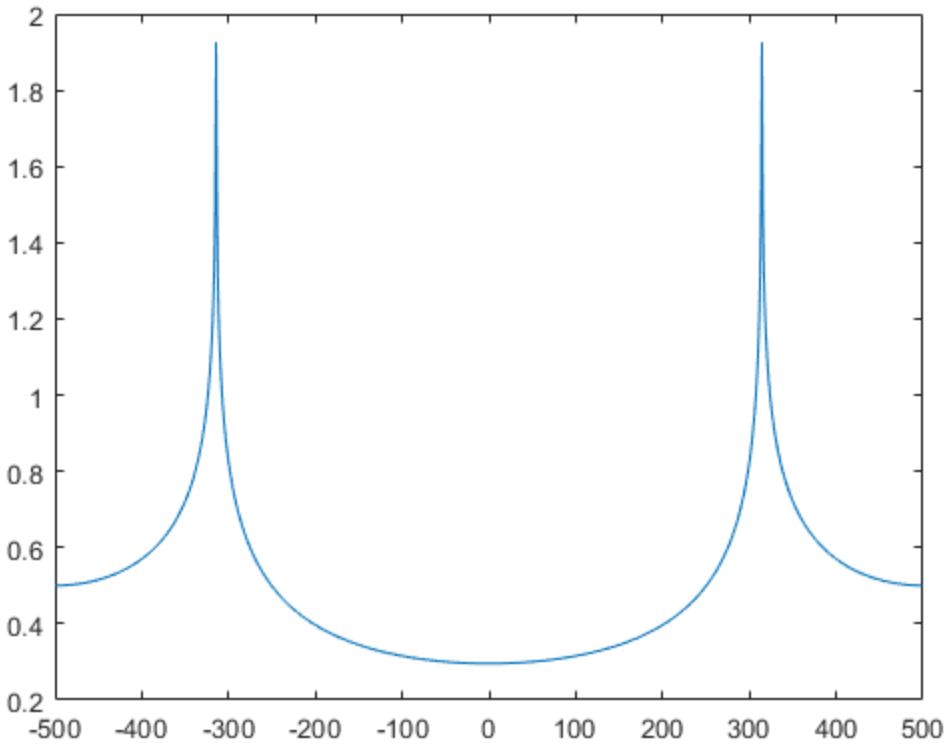
Exercise 7.6

```
f = 100;
Ts=1/1000; time=10.0; % freq, sampling interval, time
t=Ts:Ts:time ; % define a time vector
w=sin(2*pi*f*t).^2; % define the sinusoid
N=2^10; % size of analysis window
ssf =(-N/2:N/2-1)/(Ts*N); % frequency vector
fw = fft(w(1:N)); % do DFT/FFT
fws = fftshift(fw); % shift it for plotting
figure
plot (ssf ,abs(fws)) % plot magnitude spectrum
```



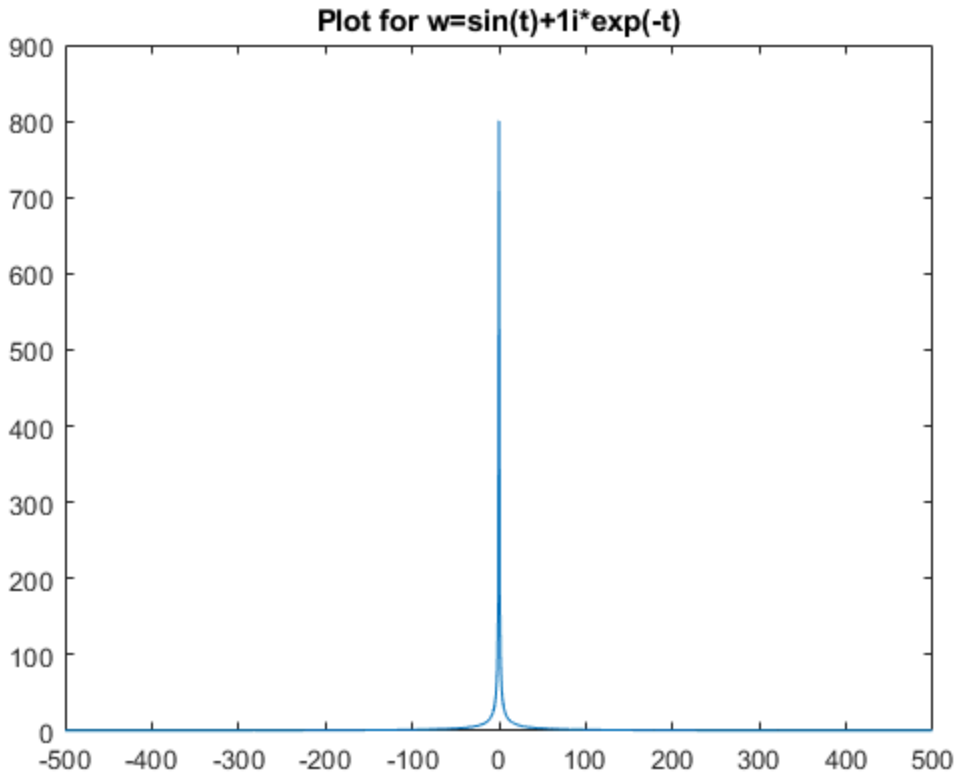
Exercise 7.7

```
f = 100;
Ts=1/1000; time=10.0; % freq, sampling interval, time
t=Ts:Ts:time ; % define a time vector
w=sinc(2*pi*f*t); % define the sinusoid
N=2^10; % size of analysis window
ssf =(-N/2:N/2-1)/(Ts*N); % frequency vector
fw = fft(w(1:N)); % do DFT/FFT
fws = fftshift(fw); % shift it for plotting
figure
plot (ssf ,abs(fws)) % plot magnitude spectrum
```



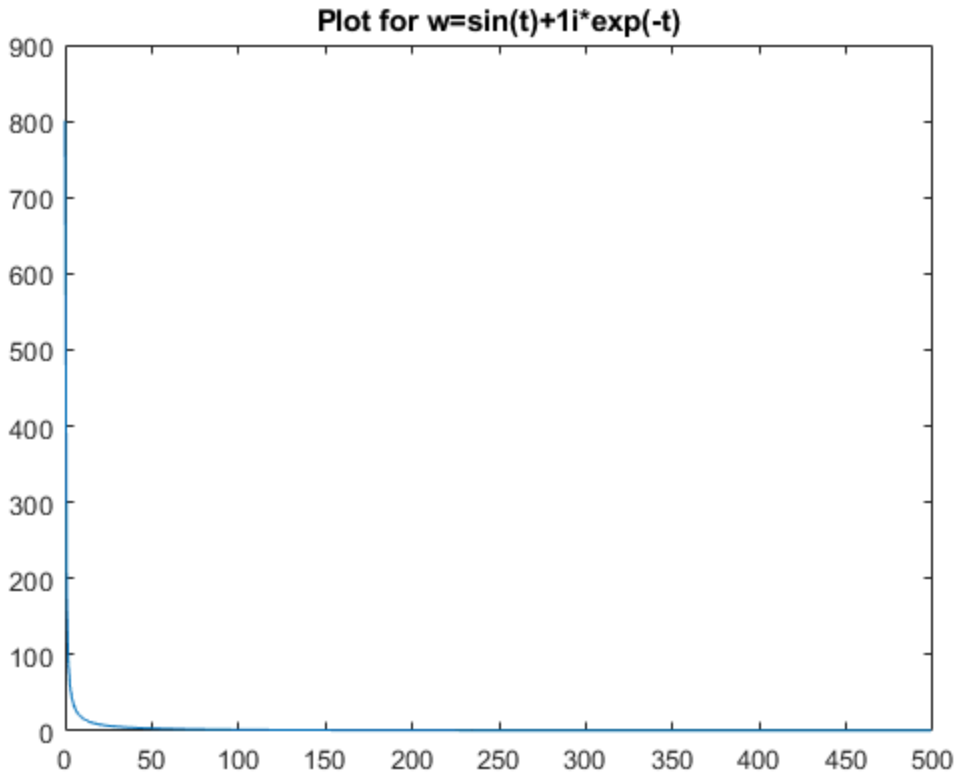
Exercise 7.8

```
f = 100;
Ts=1/1000; time=10.0; % freq, sampling interval, time
t=Ts:Ts:time ; % define a time vector
w=sin(t)+1i*exp(-t); % define the sinusoid
N=2^10; % size of analysis window
ssf =(-N/2:N/2-1)/(Ts*N); % frequency vector
fw = fft(w(1:N)); % do DFT/FFT
fws = fftshift(fw); % shift it for plotting
figure
plot (ssf ,abs(fws)) % plot magnitude spectrum
title('Plot for w=sin(t)+1i*exp(-t)')
```



Exercise 7.8.2

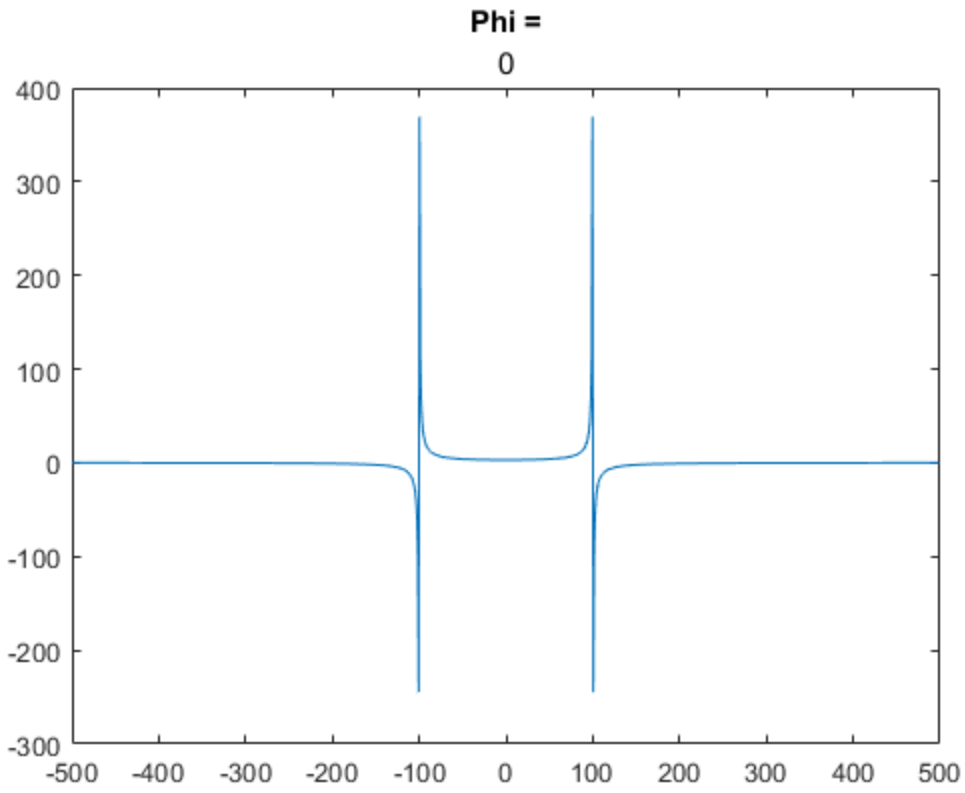
```
f =100; Ts=1/1000; time =5.0; % freq , sampling interval , time
t=Ts:Ts:time ; % define a time vector
w=sin(t)+1i*exp(-t); % define the sinusoid
N=2^10; % si z e o f a n al y si s window
ssf = (0:N/2-1)/(Ts*N); % frequency vector
fw=abs(fft(w(1:N))) ; % f i n d magnitude o f DFT/FFT
plot(ssf, fw(1:N/2)) % plot for positive freq only
title('Plot for w=sin(t)+1i*exp(-t)')
%specin2.m is the preferable way to plot the spectrum of
 sin(t)+1i*exp(-t),
%because is it symmetric at 0 on the x-axis.
```

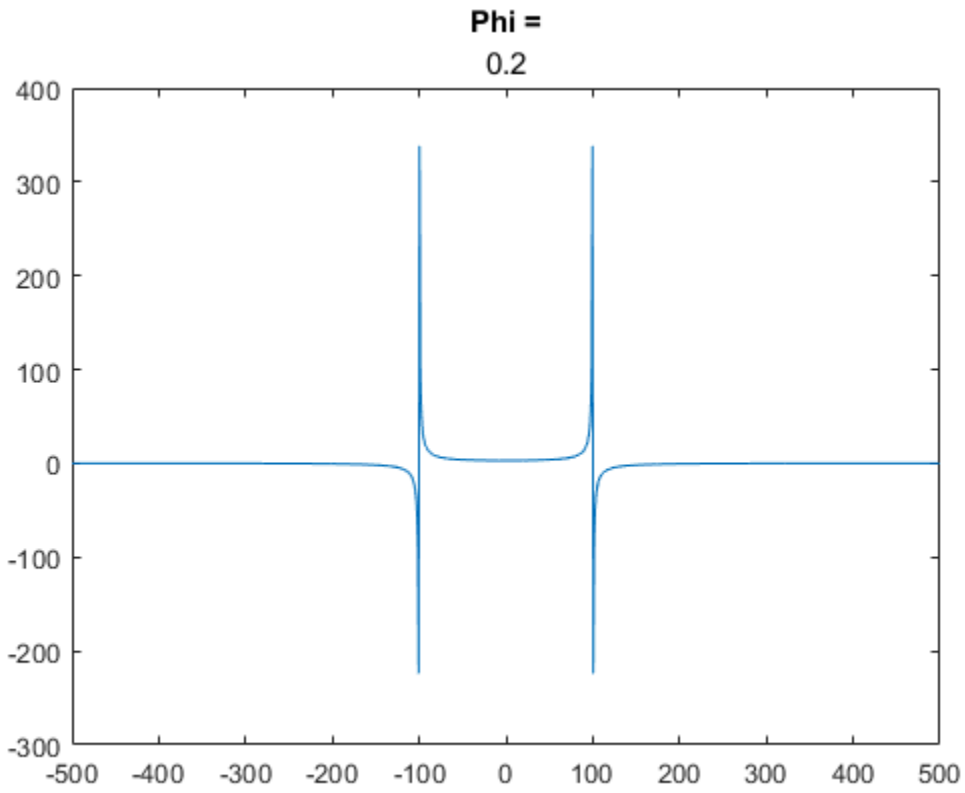


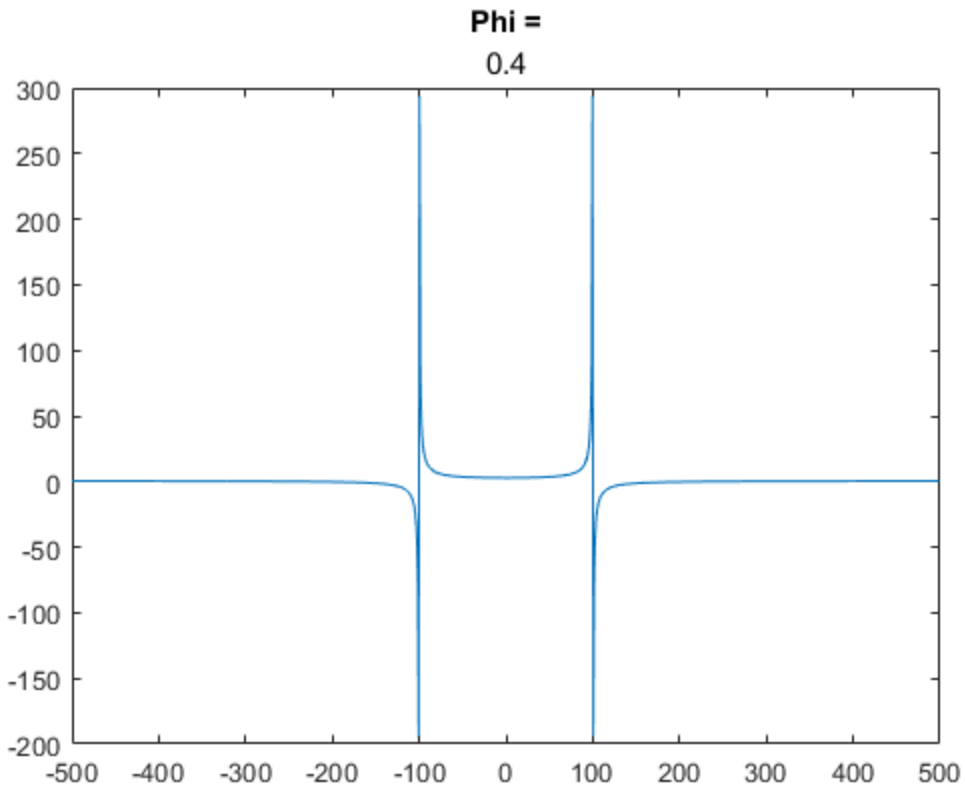
Exercise 7.9a

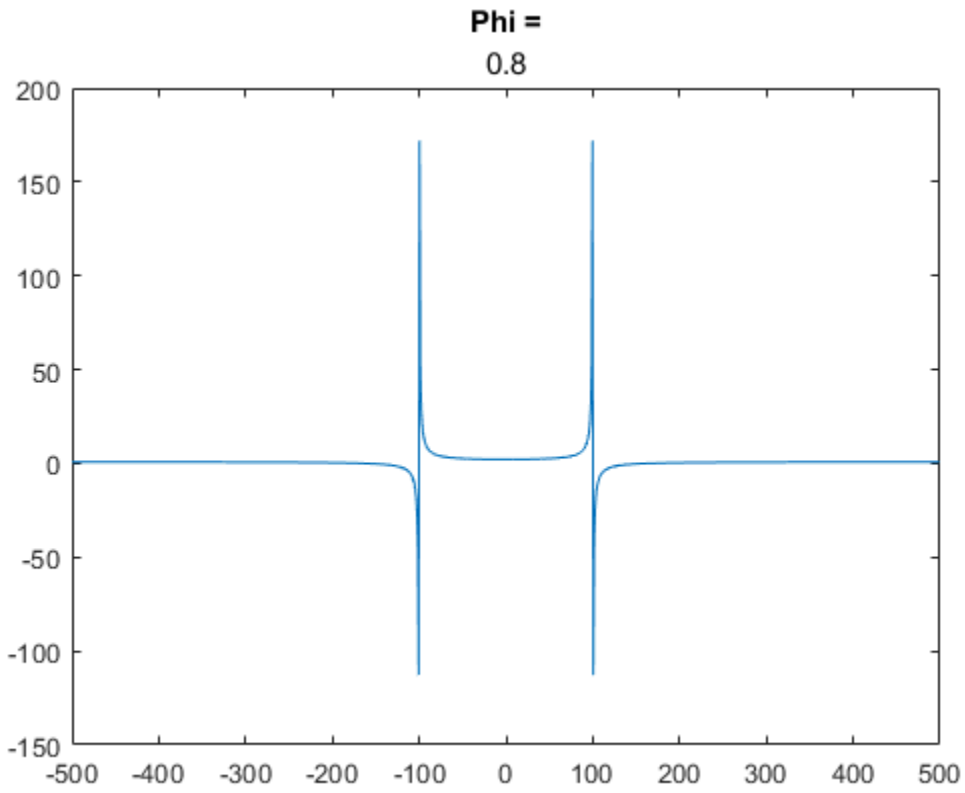
```
f =100; Ts=1/1000; time =5.0; % freq , sampling interval , time
t=Ts:Ts:time ; % define a time vector
phi=[0, 0.2, 0.4, 0.8, 1.5, 3.14];
for i=1:6
w=sin(2*pi*f*t+phi(i));
N=2^10; % size of analysis window
ssf =(-N/2:N/2-1)/(Ts*N); % frequency vector
fw = fft(w(1:N)); % do DFT/FFT
fws = fftshift(fw); % shift it for plotting
figure
plot (ssf ,fws) % plot magnitude spectrum
title('Phi =', num2str(phi(i)));
end
```

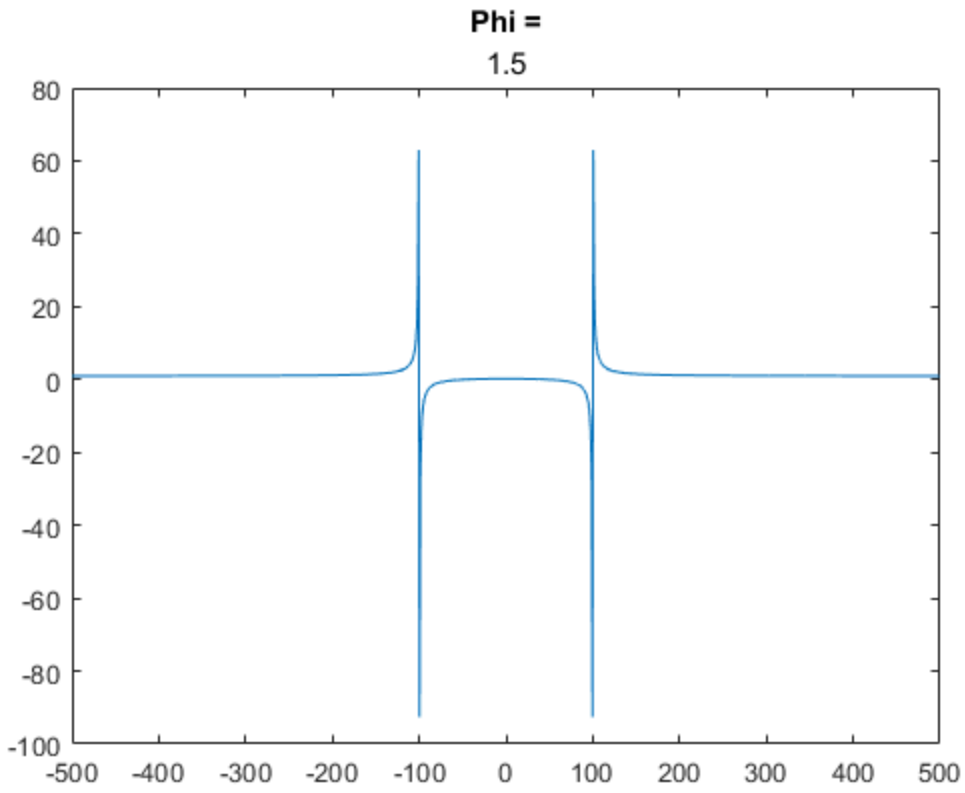
```
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
```

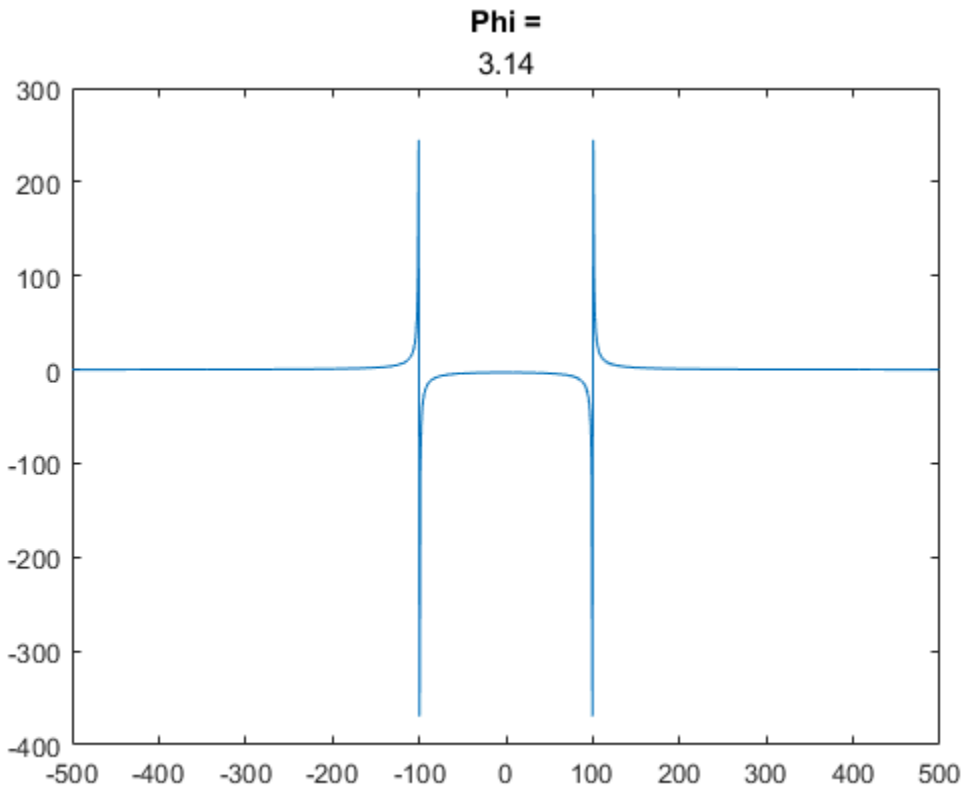

Phi =
0
400
300
200
100
0
-100
-200
-300
-500
-400
-300
-200
-100
0
100
200
300
400
500


Phi =
0.2
400
300
200
100
0
-100
-200
-300
-500
-400
-300
-200
-100
0
100
200
300
400
500


Phi =
0.4
300
250
200
150
100
50
0
-50
-100
-150
-200
-500
-400
-300
-200
-100
0
100
200
300
400
500


Phi =
0.8
200
150
100
50
0
-50
-100
-150
-500
-400
-300
-200
-100
0
100
200
300
400
500

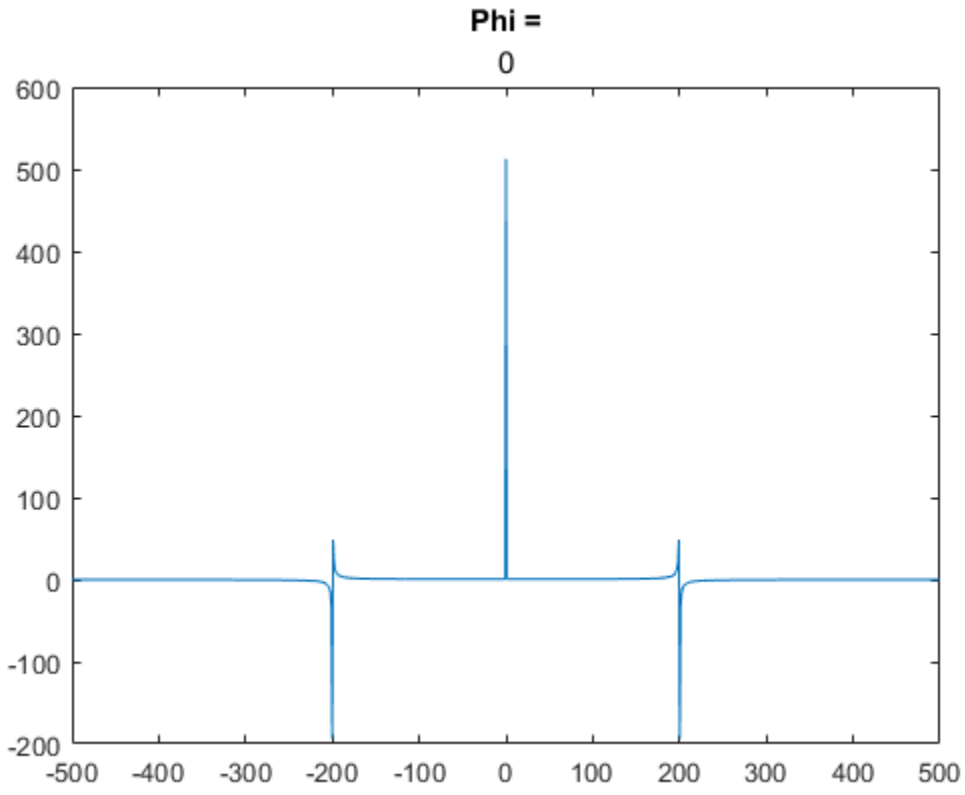

Phi =
1.5
80
60
40
20
0
-20
-40
-60
-80
-100
-500
-400
-300
-200
-100
0
100
200
300
400
500

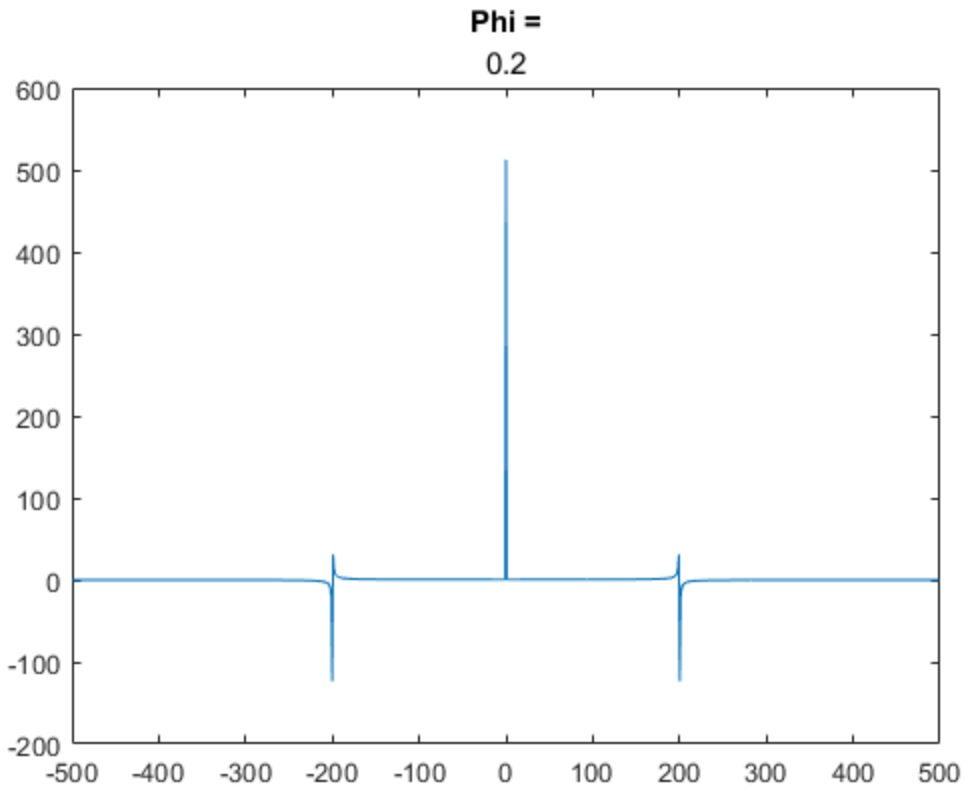

Phi =
3.14
300
200
100
0
-100
-200
-300
-400
-500
-400
-300
-200
-100
0
100
200
300
400
500

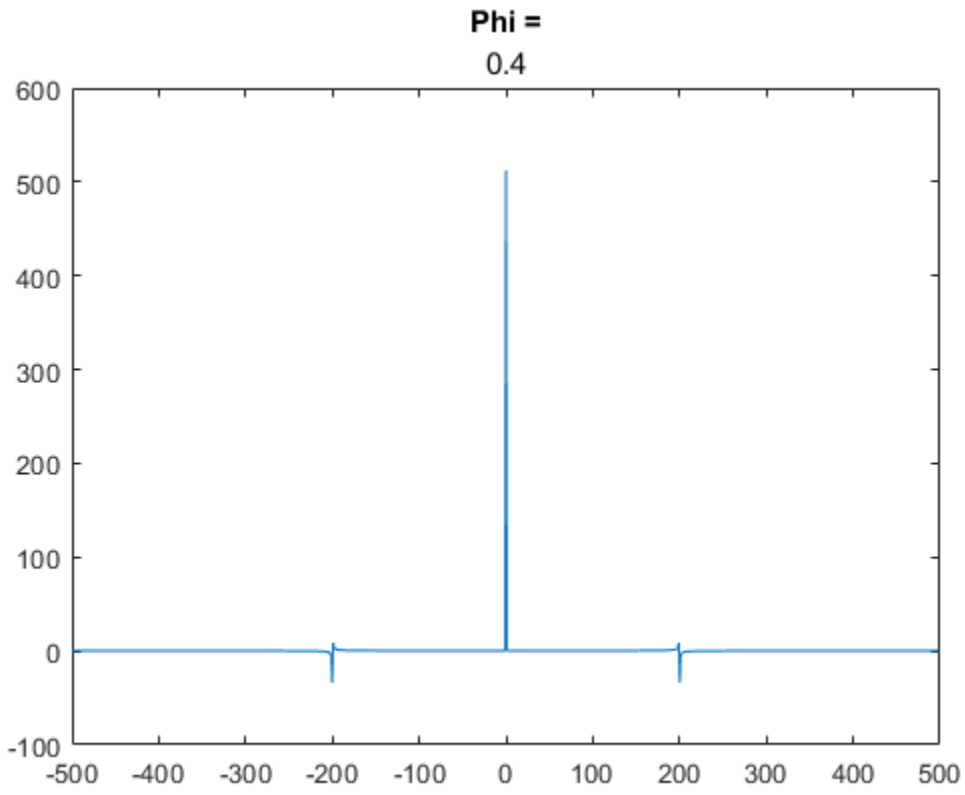
Exercise 7.9b

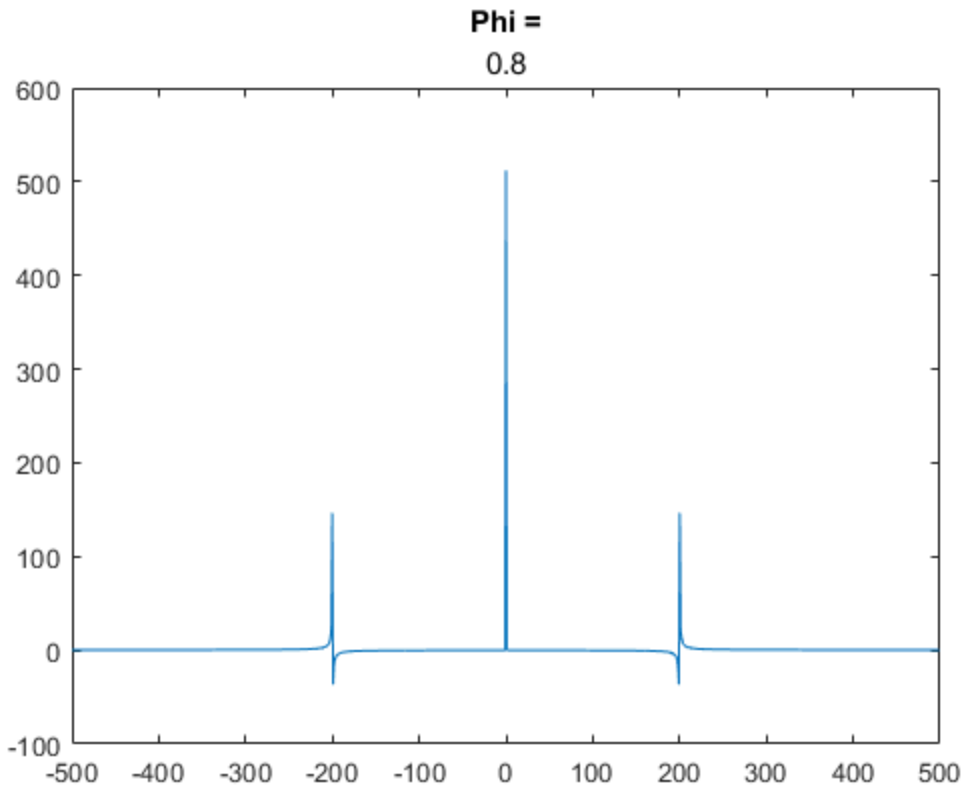
```
f =100; Ts=1/1000; time =5.0; % freq , sampling interval , time
t=Ts:Ts:time ; % define a time vector
phi=[0, 0.2, 0.4, 0.8, 1.5, 3.14];
for i=1:6
w=sin(2*pi*f*t+phi(i)).^2;
N=2^10; % size of analysis window
ssf =(-N/2:N/2-1)/(Ts*N); % frequency vector
fw = fft(w(1:N)); % do DFT/FFT
fws = fftshift(fw); % shift it for plotting
figure
plot (ssf ,fws) % plot magnitude spectrum
title('Phi =', num2str(phi(i)));
end
```

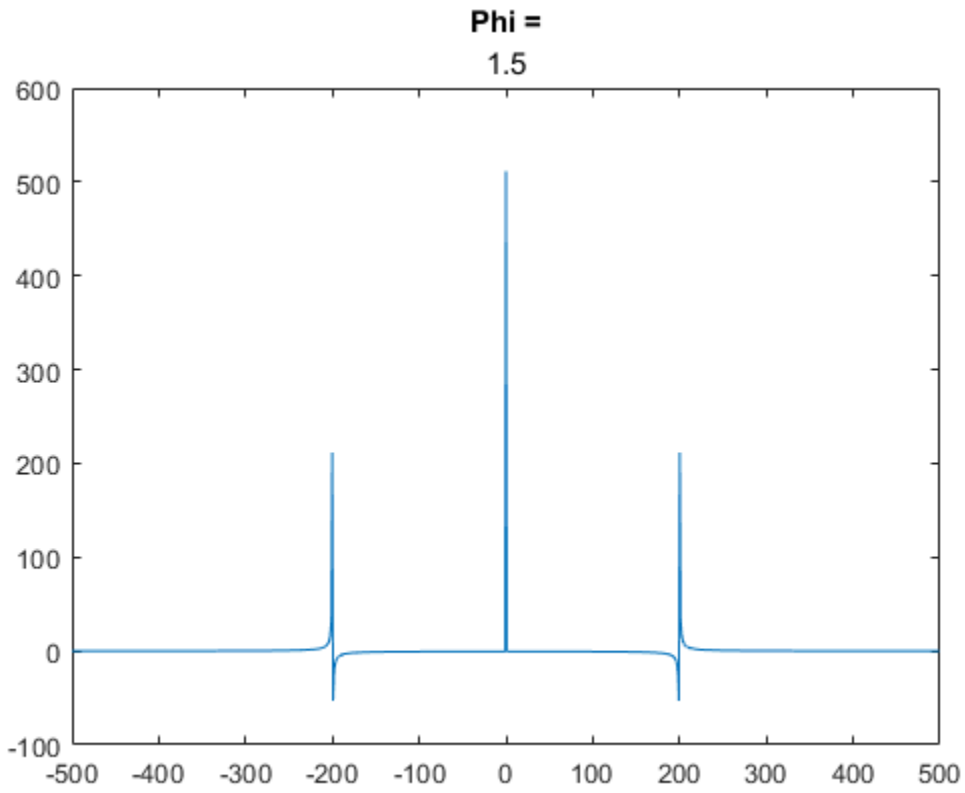
```
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
Warning: Imaginary parts of complex X and/or Y arguments ignored.
```

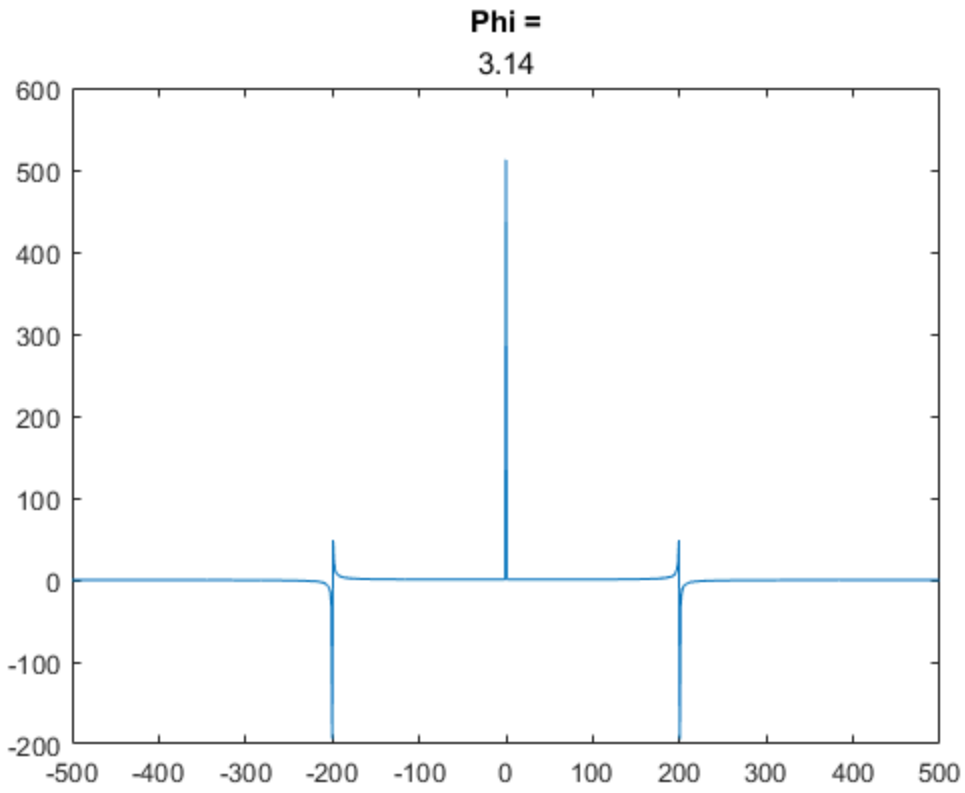



Phi =
0.2
600
500
400
300
200
100
0
-100
-200
-500
-400
-300
-200
-100
0
100
200
300
400
500


Phi =
0.4
600
500
400
300
200
100
0
-100
-500
-400
-300
-200
-100
0
100
200
300
400
500

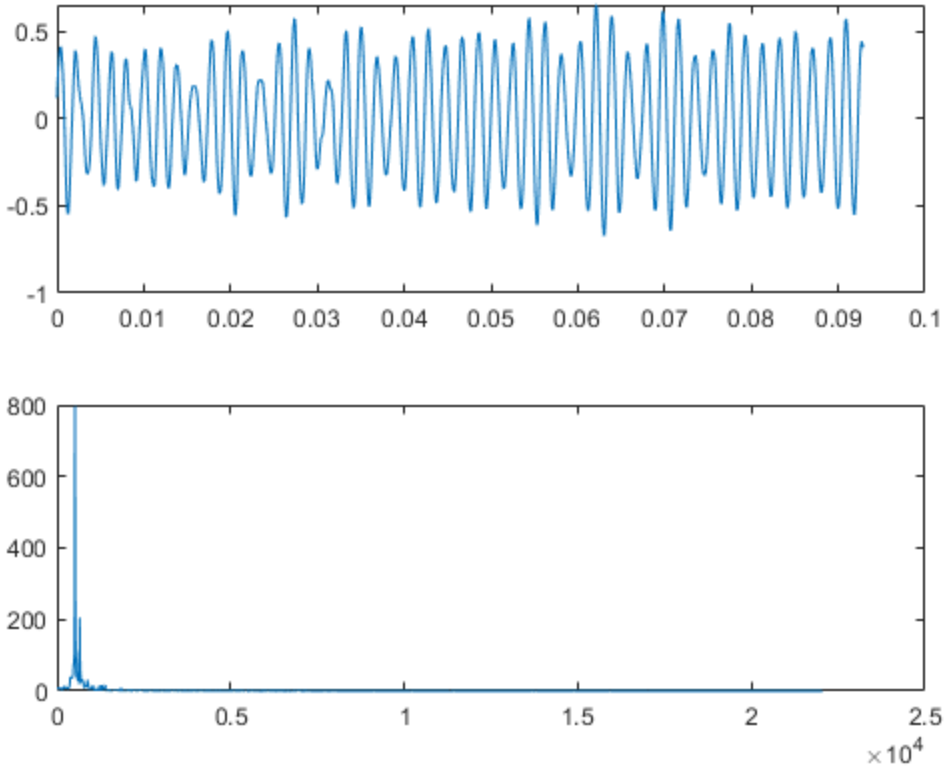

Phi =
0.8
600
500
400
300
200
100
0
-100
-500
-400
-300
-200
-100
0
100
200
300
400
500


Phi =
1.5
600
500
400
300
200
100
0
-100
-500
-400
-300
-200
-100
0
100
200
300
400
500



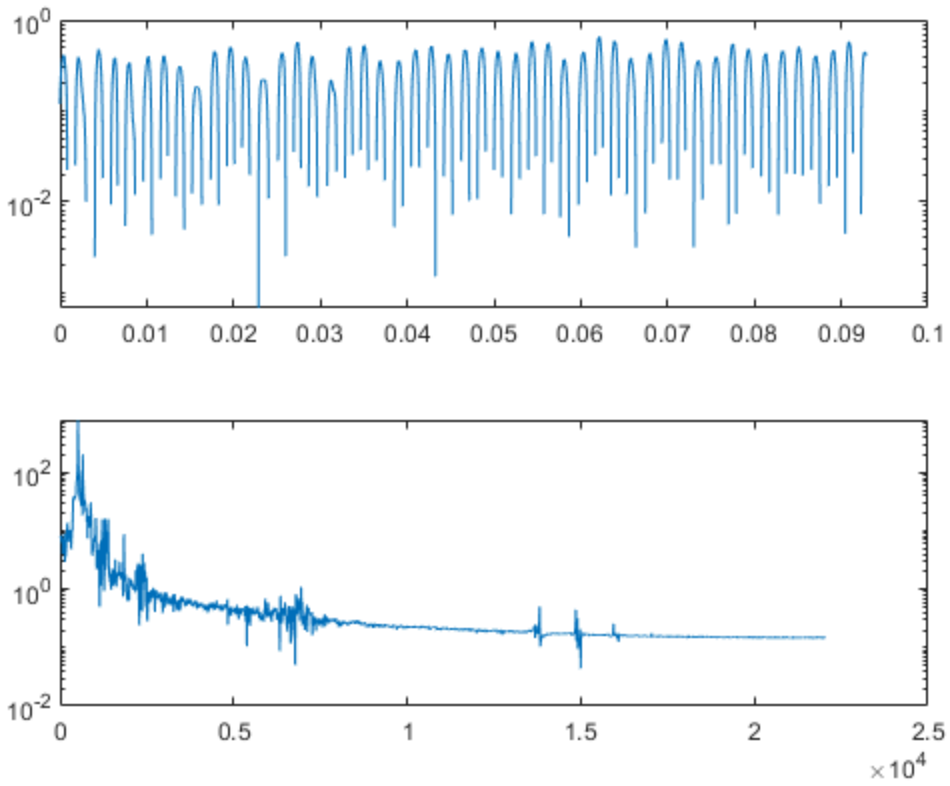
Exercise 7.10

```
filename='gong.wav'; % name o f wave f i l e
[x,sr]= audioread(filename); % read in wavefile
Ts=1/sr; % sample interval & # of samples
N=2^12; x=x(1:N)'; % length for analysis
sound(x,1/Ts) % play sound ( i f possible )
time=Ts*(0:length(x)-1); % time base for plotting
subplot(2,1,1), plot(time,x) % and pl o t top fi g u re
magx= abs(fft(x)); % t a ke FFT magnitude
ssf = (0:N/2-1)/(Ts*N); % freq base for plotting
subplot(2,1,2), plot(ssf,magx(1:N/2)) % p l o t mag spectrum
```



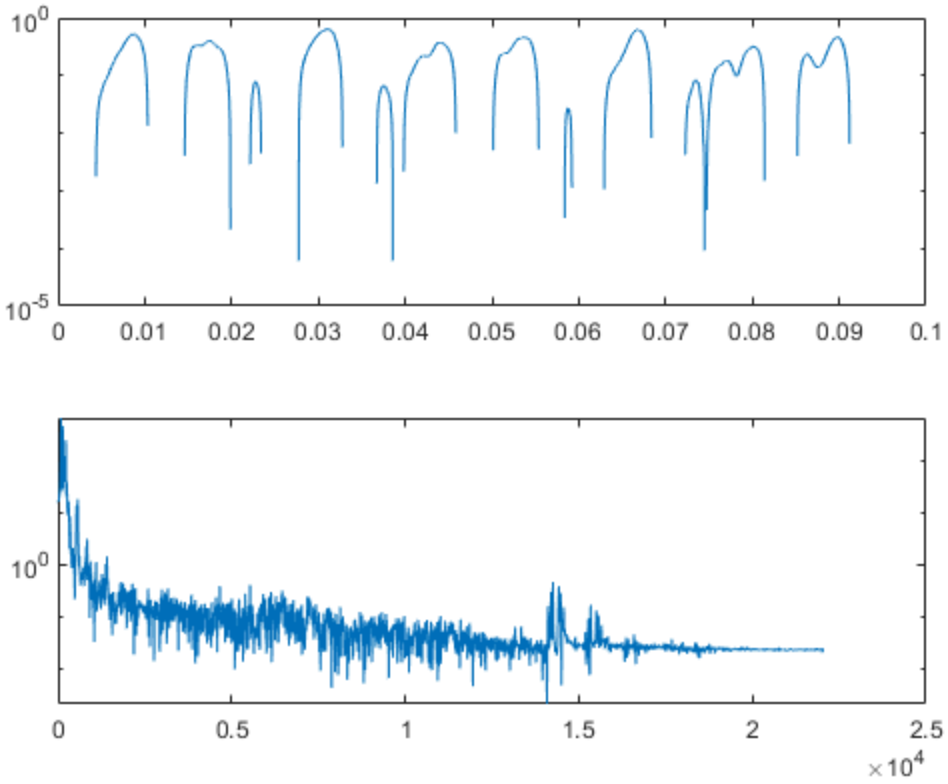
Exercise 7.11

```
filename='gong.wav'; % name o f wave f i l e
[x,sr]= audioread(filename); % read in wavefile
Ts=1/sr; % sample interval & # of samples
N=2^12; x=x(1:N)'; % length for analysis
sound(x,1/Ts) % play sound ( i f possible )
time=Ts*(0:length(x)-1); % time base for plotting
subplot(2,1,1), semilogy(time,x) % and pl o t top fi g u re
magx= abs(fft(x)); % t a ke FFT magnitude
ssf = (0:N/2-1)/(Ts*N); % freq base for plotting
subplot(2,1,2), semilogy(ssf,magx(1:N/2)) % p l o t mag spectrum
%The resulting plot seems to have more flucuations than the initial
 plot
```



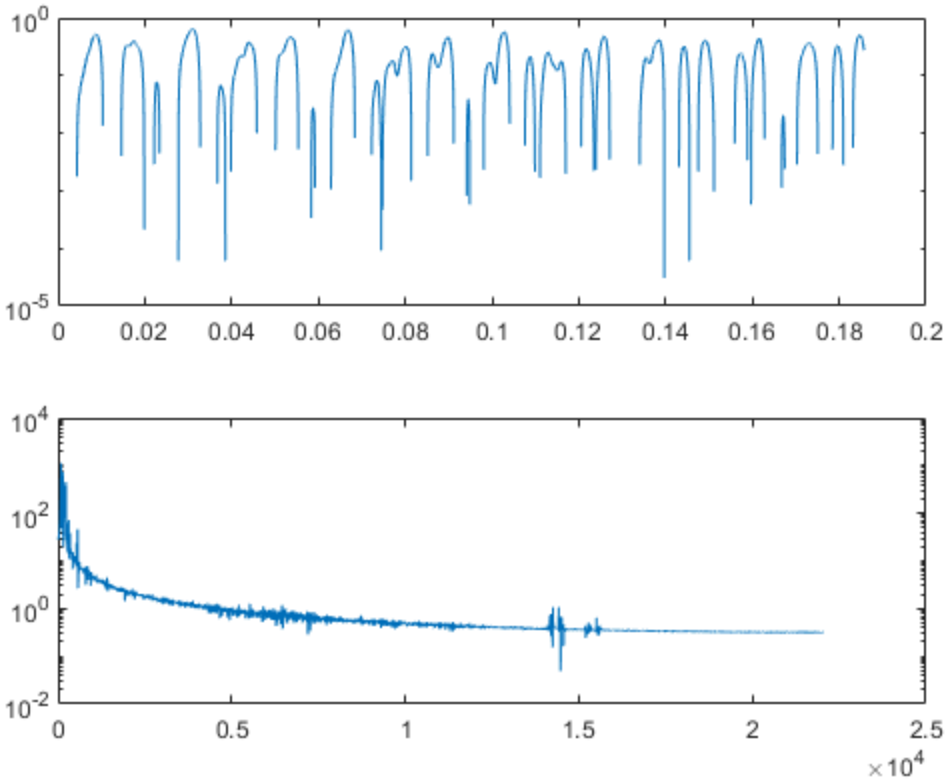
Exercise 7.12a

```
filename='gong2.wav'; % name o f wave f i l e
[x,sr]= audioread(filename); % read in wavefile
Ts=1/sr; % sample interval & # of samples
N=2^12;
x=x(1:N)'; % length for analysis
sound(x,1/Ts) % play sound ( i f possible )
time=Ts*(0:length(x)-1); % time base for plotting
subplot(2,1,1), semilogy(time,x) % and pl o t top fi g u re
magx= abs(fft(x)); % t a ke FFT magnitude
ssf = (0:N/2-1)/(Ts*N); % freq base for plotting
subplot(2,1,2), semilogy(ssf,magx(1:N/2)) % p l o t mag spectrum
```



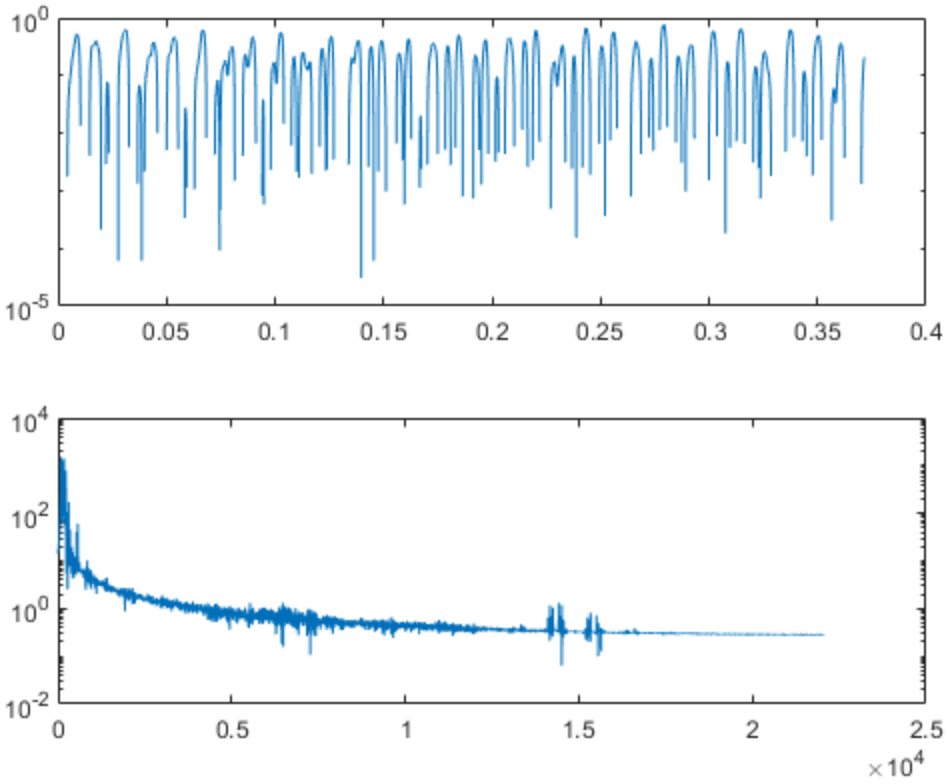
Exercise 7.12b

```
filename='gong2.wav'; % name o f wave f i l e
[x,sr]= audioread(filename); % read in wavefile
Ts=1/sr; % sample interval & # of samples
N=2^13;
x=x(1:N)'; % length for analysis
sound(x,1/Ts) % play sound ( i f possible )
time=Ts*(0:length(x)-1); % time base for plotting
subplot(2,1,1), semilogy(time,x) % and pl o t top fi g u re
magx= abs(fft(x)); % t a ke FFT magnitude
ssf = (0:N/2-1)/(Ts*N); % freq base for plotting
subplot(2,1,2), semilogy(ssf,magx(1:N/2)) % p l o t mag spectrum
```



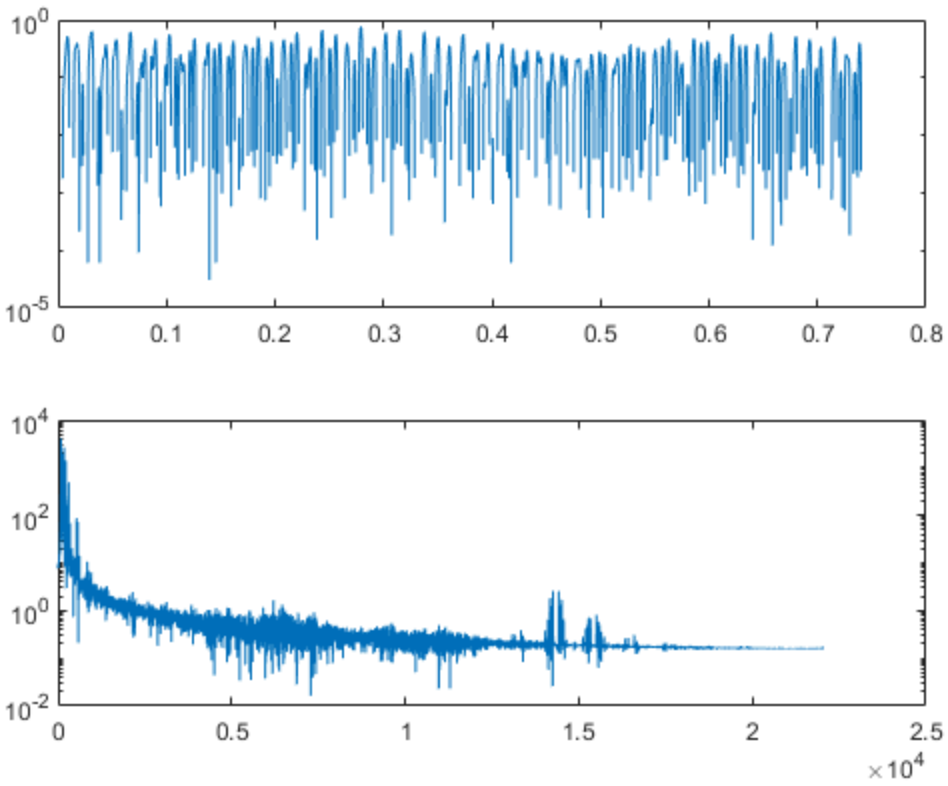
Exercise 7.12c

```
filename='gong2.wav'; % name o f wave f i l e
[x,sr]= audioread(filename); % read in wavefile
Ts=1/sr; % sample interval & # of samples
N=2^14;
x=x(1:N)'; % length for analysis
sound(x,1/Ts) % play sound ( i f possible )
time=Ts*(0:length(x)-1); % time base for plotting
subplot(2,1,1), semilogy(time,x) % and pl o t top fi g u re
magx= abs(fft(x)); % t a ke FFT magnitude
ssf = (0:N/2-1)/(Ts*N); % freq base for plotting
subplot(2,1,2), semilogy(ssf,magx(1:N/2)) % p l o t mag spectrum
```



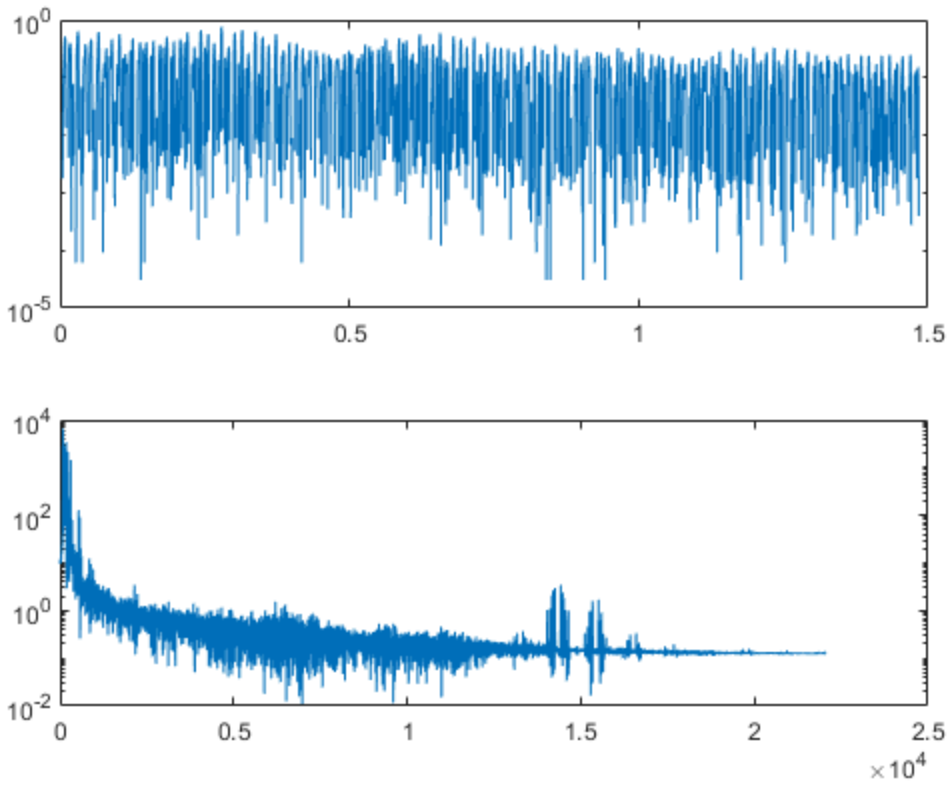
Exercise 7.12d

```
filename='gong2.wav'; % name o f wave f i l e
[x,sr]= audioread(filename); % read in wavefile
Ts=1/sr; % sample interval & # of samples
N=2^15;
x=x(1:N)'; % length for analysis
sound(x,1/Ts) % play sound ( i f possible )
time=Ts*(0:length(x)-1); % time base for plotting
subplot(2,1,1), semilogy(time,x) % and pl o t top fi g u re
magx= abs(fft(x)); % t a ke FFT magnitude
ssf = (0:N/2-1)/(Ts*N); % freq base for plotting
subplot(2,1,2), semilogy(ssf,magx(1:N/2)) % p l o t mag spectrum
```



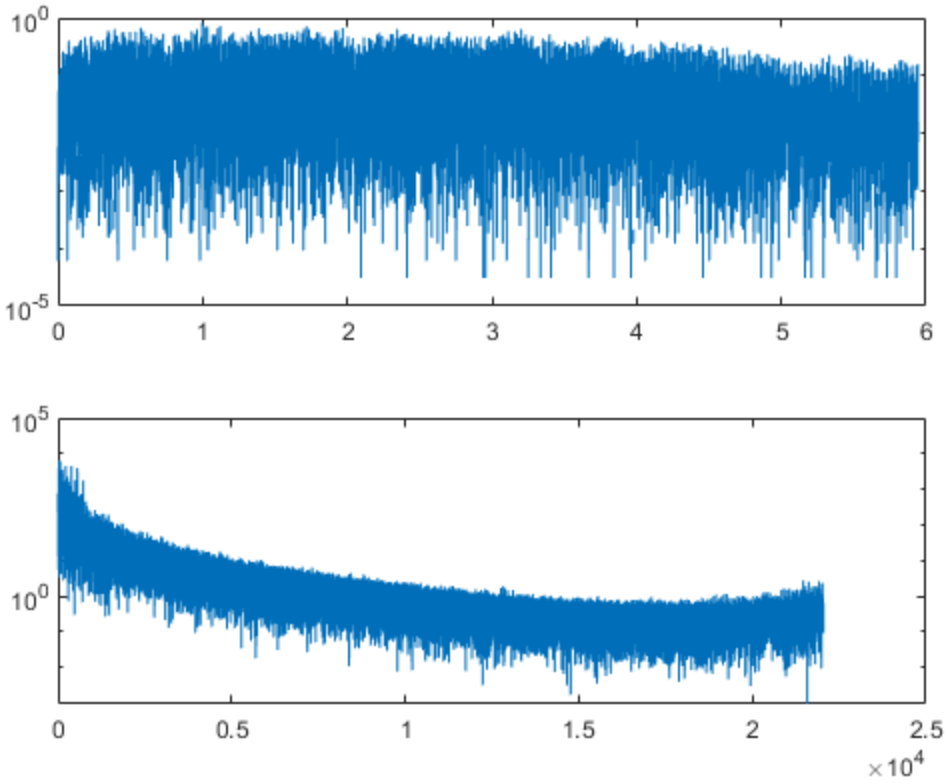
Exercise 7.12e

```
filename='gong2.wav'; % name o f wave f i l e
[x,sr]= audioread(filename); % read in wavefile
Ts=1/sr; % sample interval & # of samples
N=2^16;
x=x(1:N)'; % length for analysis
sound(x,1/Ts) % play sound ( i f possible )
time=Ts*(0:length(x)-1); % time base for plotting
subplot(2,1,1), semilogy(time,x) % and pl o t top fi g u re
magx= abs(fft(x)); % t a ke FFT magnitude
ssf = (0:N/2-1)/(Ts*N); % freq base for plotting
subplot(2,1,2), semilogy(ssf,magx(1:N/2)) % p l o t mag spectrum
```



Exercise 7.13

```
filename='mixkit-horns-of-vengeance-713.wav'; % name o f wave f i l e
[x,sr]= audioread(filename); % read in wavefile
Ts=1/sr; % sample interval & # of samples
N=2^18; x=x(1:N)'; % length for analysis
sound(x,1/Ts) % play sound (if possible)
time=Ts*(0:length(x)-1); % time base for plotting
subplot(2,1,1), semilogy(time,x) % and pl o t top fi g u re
magx= abs(fft(x)); % t a ke FFT magnitude
ssf = (0:N/2-1)/(Ts*N); % freq base for plotting
subplot(2,1,2), semilogy(ssf,magx(1:N/2)) % p l o t mag spectrum
```



Exercise 7.15

```
a=[1 -0.9]; lena=length(a)-1; % autoregressive coefficients
b = [2]; lenb=length(b); % moving average c o e f fi c i e n t s
d=randn(1 ,20); % data to f il t e r
if lena>=lenb, % dimpulse needs lena>=l enb
    h=impz (b,a) ; % impulse response of fil ter
    yfilt=filter (h,1,d); % fi l t e r x[ k] with h[ k]
end
yfilt2 = filter (b,a,d); % f i l t e r using a and b
y=zeros(lena,1); x=zeros(lenb,1); % initial states in filter
for k=1:length(d)-lenb % time#domain method
    x=[d(k); x(1:lenb-1)]; % past values of inputs
    ytim(k)=-a(2:lena+1)*y+b*x ; % directly calculate y[k]
    y=[ytim(k); y(1:lena-1)]; % past values of outputs
end
%yfilt2 and ytim are equivalent
```

Published with MATLAB® R2020b