EC 415: Homework 2

Due by Friday 03/05/2021 6:00PM

Professor David Starobinski

Michael Kremer
kremerme@bu.edu

Exercise 3.1

Use specsquare.m to investigate the relationship between the time behavior of the square wave and its spectrum. The Matlab command zoom on is often helpful for viewing details of the plots.

a. Try square waves with different frequencies: f=20, 40, 100, 300 Hz. How do the time plots change? How do the spectra change?

b. Try square waves of different lengths, time=1, 10, 100 seconds. How does the spectrum change in each case?

c. Try different sampling times, T_s=1/100, 1/10000 seconds. How does the spectrum change in each case?

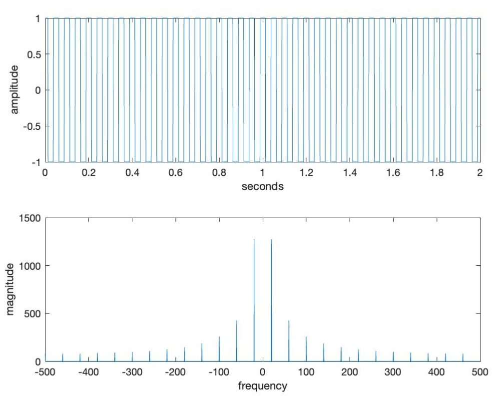
Solution

To obtain all solutions to this exercise, a copy of the provided specsquare.m was used. For each part of the exercise, the relevant variable was changed before re-running the script and generating a plot.

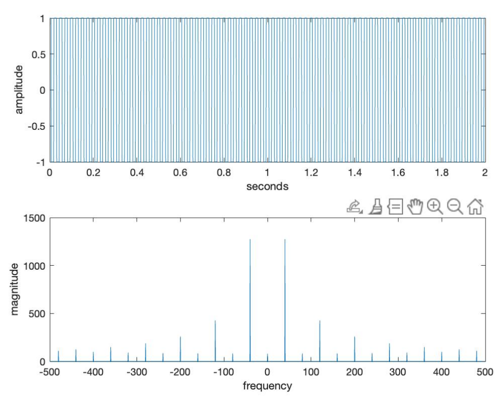
Listing 1: MATLAB code for Exercise 3.1

```
% specsquare.m plot the spectrum of a square wave
f = 10;                  % "frequency" of square wave, 10
time = 2;                % length of time, 2
Ts = 1/10000;            % time interval between samples, 1/1000
t = Ts:Ts:time;          % create a time vector
x = sign(cos(2*pi*f*t)); % square wave = sign of cos wave
plotspec(x,Ts)           % call plotspec to draw spectrum
```

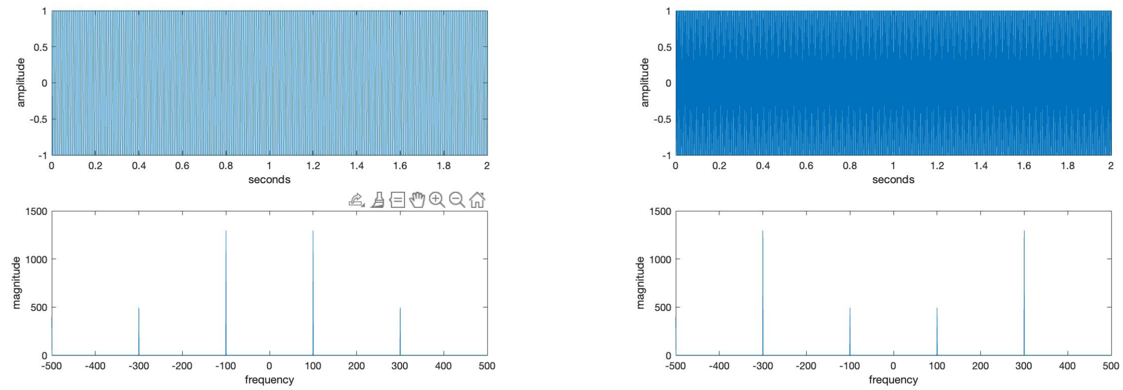
a. As the frequency increases, the density of square waves on the time plot increases while the spectra plot is extended across the frequency axis.



(a) f=20Hz



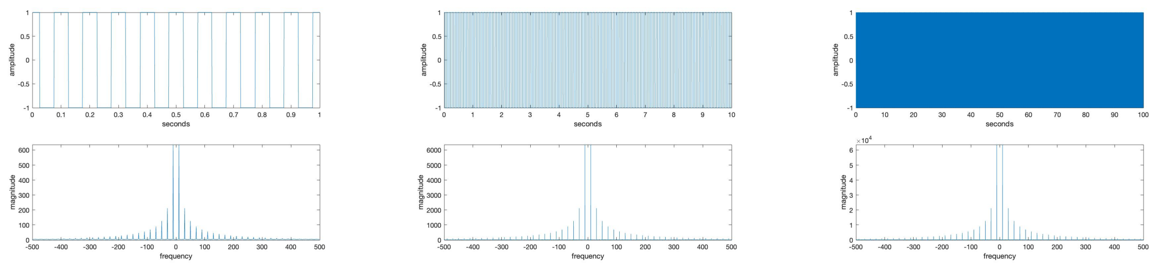
(b) f=40Hz



(a) f=100Hz (b) f=300Hz

Figure 2: square waves with different frequencies

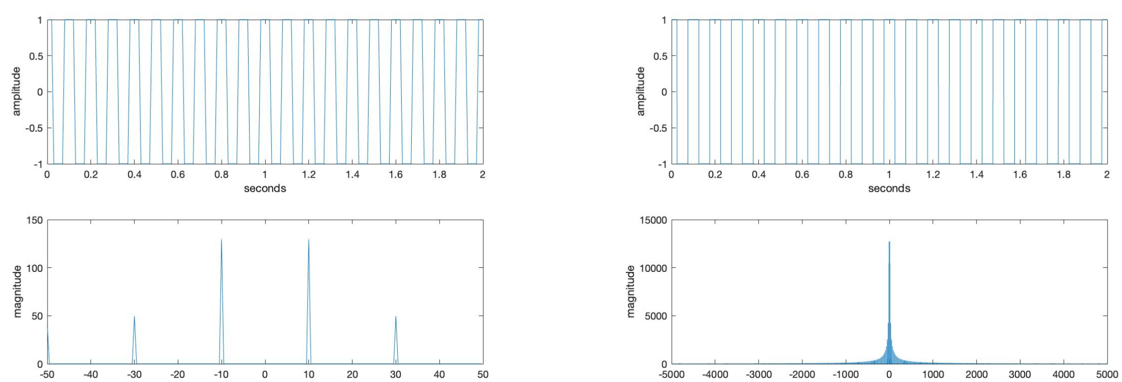
b. As the length of time increases, the magnitude of the spectra increases.



(a) t=1sec (b) t=10sec (c) t=100sec

Figure 3: square waves of different lengths

c. As the sampling time decreases, the sampling frequency increases. This means the spectrum plot is a more complete picture with more plotted spectra. However, this does not change the actual shape of the spectrum.



(a) T_s=1/100 (b) T_s=1/10000

Figure 4: different sampling times

Exercise 3.3

Mimic the code in specsquare.m to find the spectrum of:

a. An exponential pulse $s(t) = e^{-t}$ for $0 < t < 10$

b. A scaled exponential pulse $s(t) = 5e^{-t}$ for $0 < t < 10$

c. A Gaussian pulse $s(t) = e^{-t^2}$ for $-2 < t < 2$

d. A Gaussian pulse $s(t) = e^{-t^2}$ for $-20 < t < 20$

e. The sinusoids $s(t) = sin(2\pi ft + \phi)$ for $f = 20, 100, 1000$ with $\phi = 0, \pi/4, \pi/2$ and $0 < t < 10$

Solution

a. Here is the MATLAB code:

Listing 2: MATLAB code for part a

```
% 3.3 PART A
time = 10; % run fo 10 sec
Ts = 1/10000; % ms timescale
t = Ts:Ts:time; % time vector
x = exp(-t); % signal
plotspec(x, Ts) % plot spectrum
```

The resulting plot is shown in the figure below.

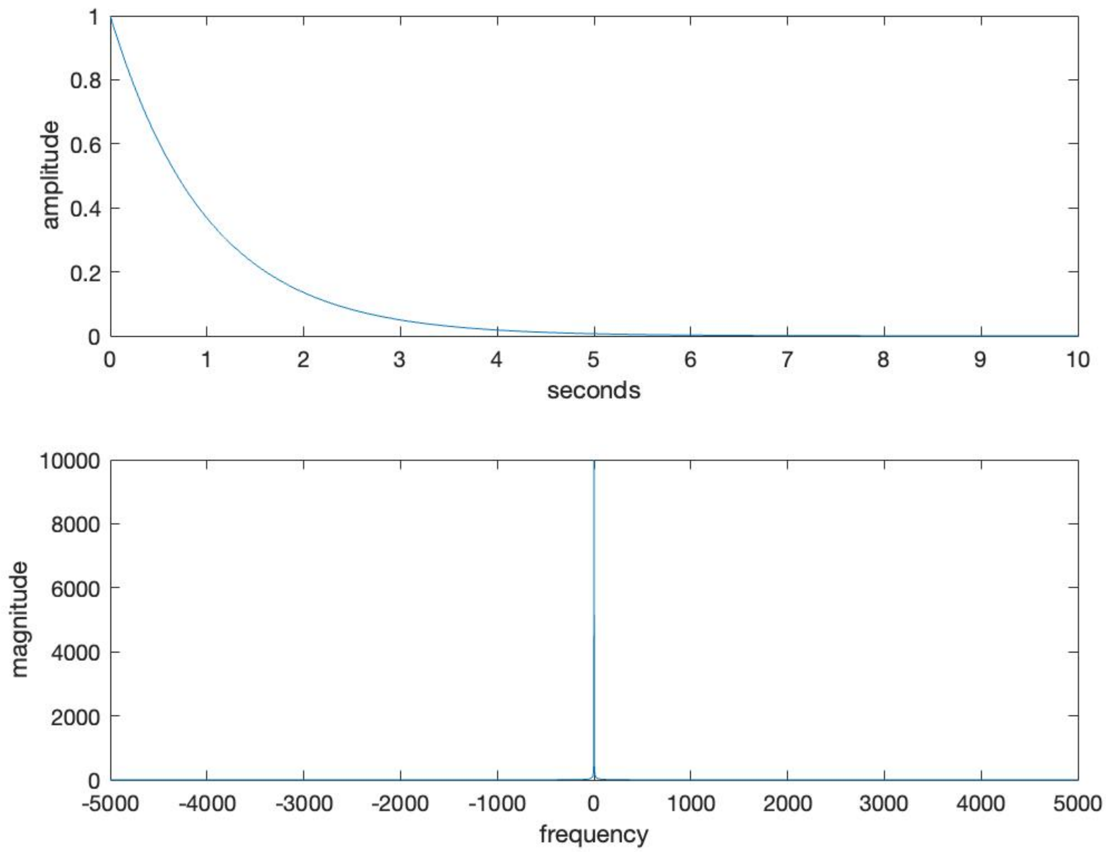


Figure 5: An exponential pulse $s(t) = e^{-t}$ for $0 < t < 10$

b. Here is the MATLAB code:

Listing 3: MATLAB code for part b

```
%p3_3b.m
time = 10; % run fo 10 sec
Ts = 1/1000; % ms timescale
t = Ts:Ts:time; % time vector
x = 5*exp(-t); % signal
plotspec(x, Ts) % plot spectrum
```

The resulting plot is shown in the figure below.

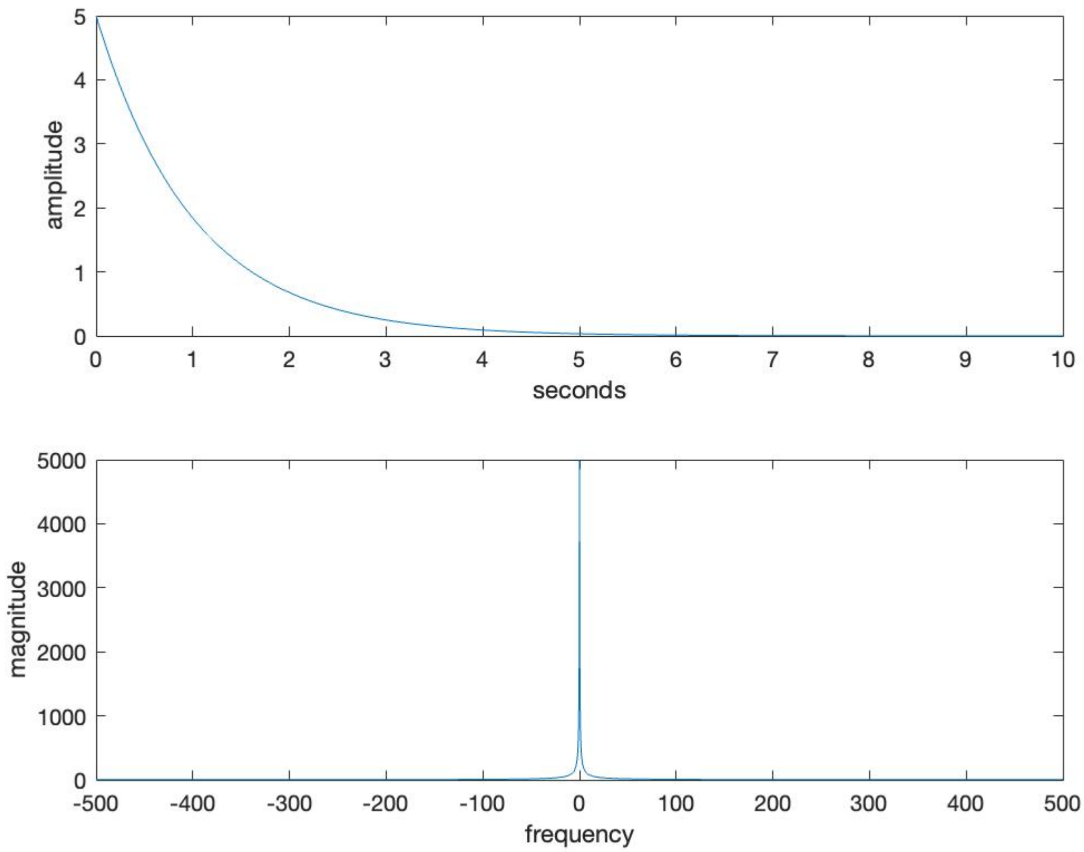


Figure 6: A scaled exponential pulse $s(t) = 5e^{-t}$ for $0 < t < 10$

c. Here is the MATLAB code:

Listing 4: MATLAB code for part c

```
%p3_3c.m
time = 2; % run fo 2 sec
Ts = 1/1000; % ms timescale
t = (0-time):Ts:time; % time vector from -time to time
x = 5*exp((-t).^2); % signal
plotspec(x, Ts) % plot spectrum
```

The resulting plot is shown in the figure below.

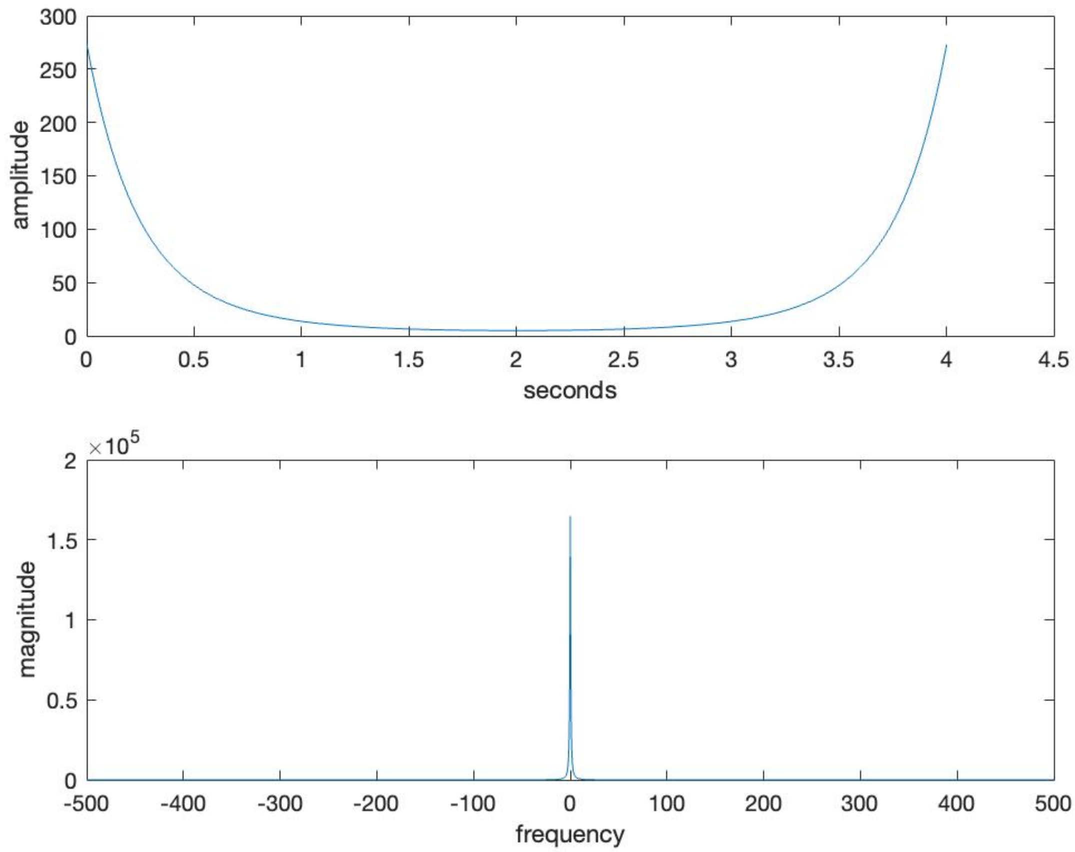


Figure 7: A Gaussian pulse $s(t) = e^{-t^2}$ for $-2 < t < 2$

d. Here is the MATLAB code:

Listing 5: MATLAB code for part d

```
%p3_3d.m
time = 20; % run fo 20 sec
Ts = 1/1000; % ms timescale
t = (0-time):Ts:time; % time vector from -time to time
x = 5*exp((-t).^2); % signal
plotspec(x, Ts) % plot spectrum
```

The resulting plot is shown in the figure below.

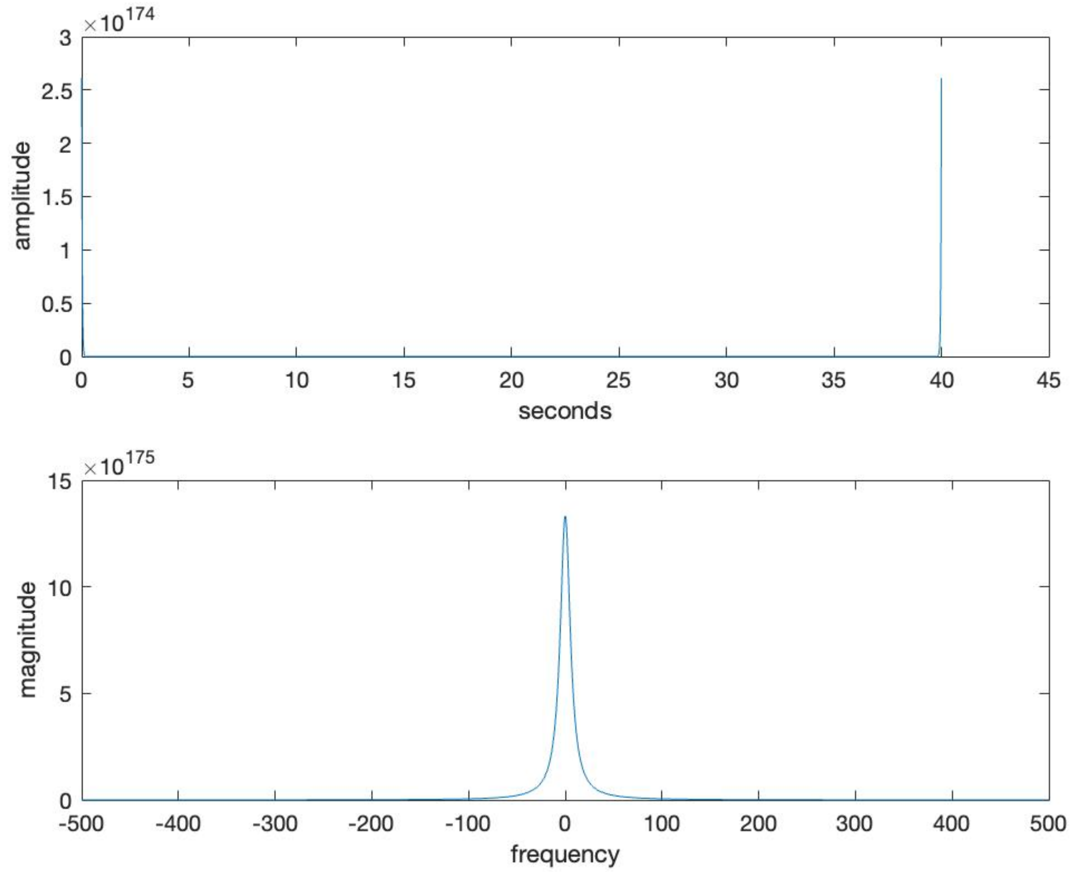


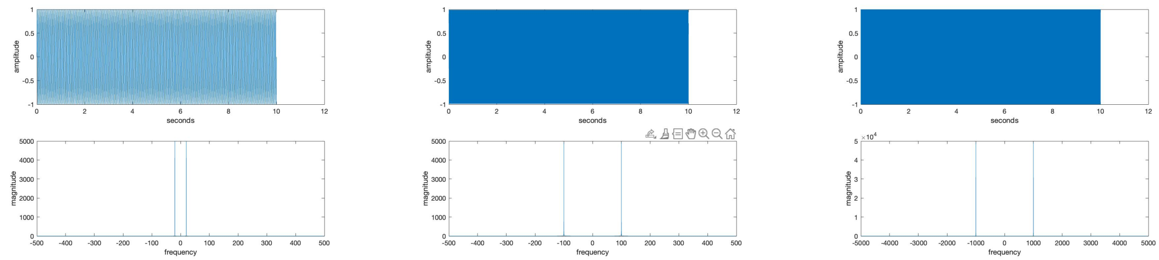
Figure 8: A Gaussian pulse $s(t) = e^{-t^2}$ for $-20 < t < 20$

e. Here is the MATLAB code:

Listing 6: MATLAB code for part e

```
%p3_3e.m
time = 10; % run fo 10 sec
Ts = 1/10000; % ms timescale
t = 0:Ts:time; % time vector
f = 20; phi = 0; % set f and phi
x = sin((2*pi*f*t) + phi); % signal
plotspec(x, Ts) % plot spectrum
```

The resulting plots are shown in the figures below.



(a) f=20Hz and ϕ=0 (b) f=100Hz and $\phi=\pi/4$ (c) f=1000Hz and $\phi=\pi/2$

Figure 9: The sinusoids $s(t) = sin(2\pi ft + \phi)$

Exercise 3.6

Mimic the code in speccos.m to find the spectrum of a cosine wave

a. For different frequencies f=1, 2, 20, 30 Hz

b. for different phases ϕ = 0, 0.1, $\pi/8$, $\pi/2$ radians

c. For different sampling rates T_s=1/10, 1/1000, 1/100000.

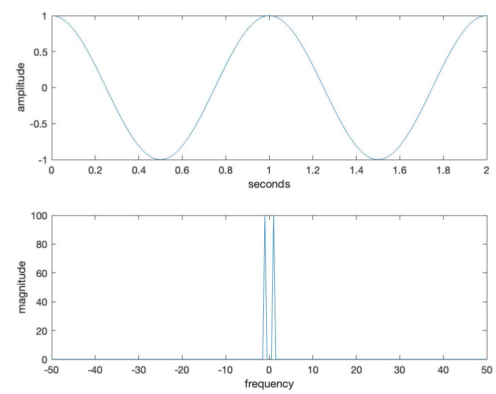
Solution

To obtain all solutions to this exercise, a copy of the provided speccos.m was used. For each part of the exercise, the relevant variable was changed before re-running the script and generating a plot.

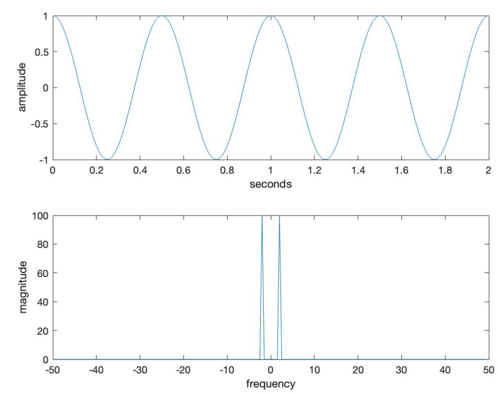
Listing 7: MATLAB code for Exercise 3.6

```
% speccos.m plot the spectrum of a cosine wave
f=10; phi=0;                % specify frequency and phase
time=2;                     % length of time
Ts=1/100;                   % time interval between samples
t=Ts:Ts:time;               % create a time vector
x=cos(2*pi*f*t+phi);        % create cos wave
plotspec(x,Ts)              % draw waveform and spectrum
```

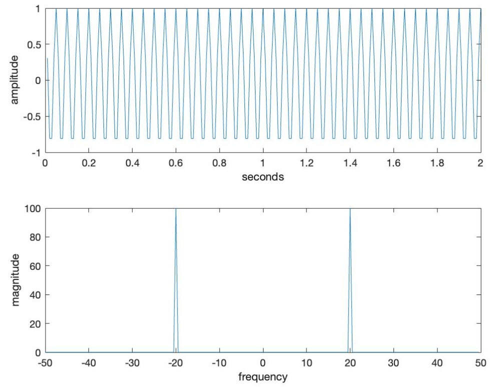
a. Because the two non-zero spectrum values of a cosine wave are at +f and -f, as f changes, the two spectra peaks located at +f and -f change with it.



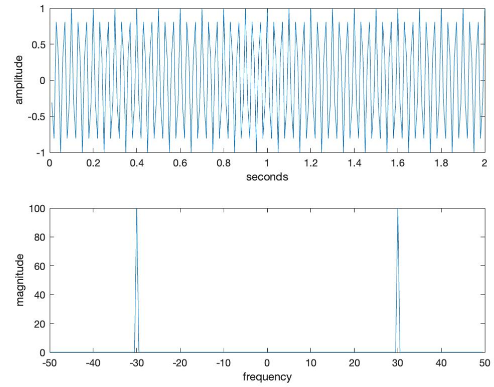
(a) f=1Hz



(b) f=2Hz



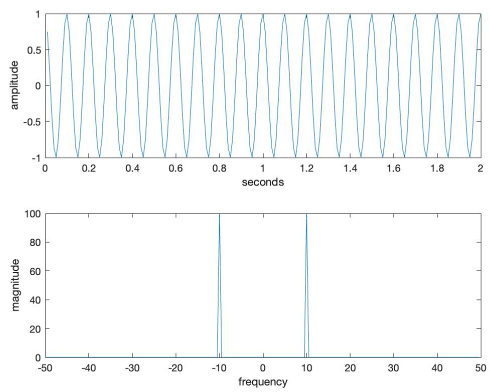
(a) f=20Hz



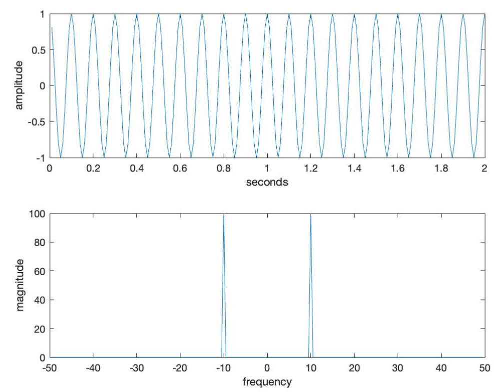
(b) f=30Hz

Figure 11: different frequencies

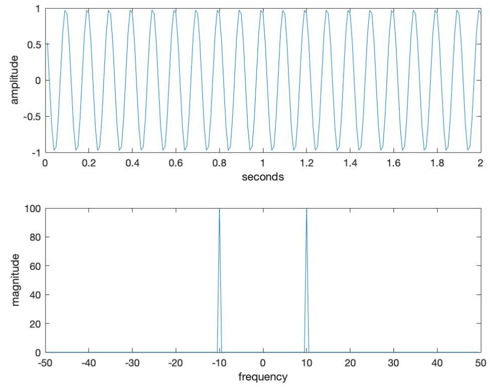
b. As the phase changes, the plot in the time domain translates left and right accordingly, but the spectrum plot does not change.



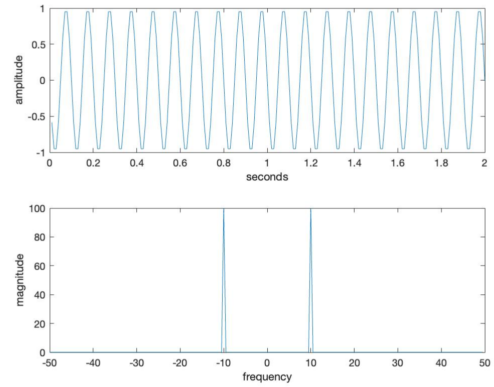
(a) phi=0



(b) phi=2Hz



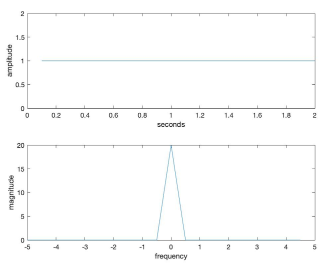
(a) *phi=pi/8*



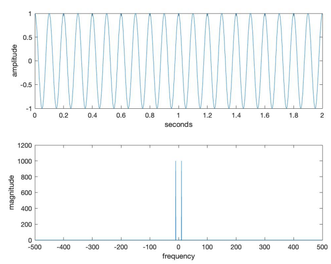
(b) *phi=pi/2*

Figure 13: different phases

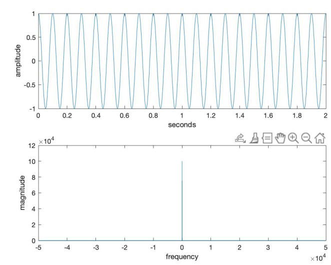
c. As you can see in the figures below, the sampling rate is important for plotting the spectrum of cosine. In the case of T_s=1/10 which is equal to 1/f, we mistakenly get a single peak spectrum plot. In the third image, there are two distinct peaks with sufficient zooming, but MATLAB was not cooperating on this one.



(a) T_s=1/10



(b) T_s=1/1000



(c) T_s=1/100000

Figure 14: different sampling

Exercise 3.9

Mimic the code in filternoise.m to create a filter that

a. Passes all frequencies above 500 Hz

b. Passes all frequencies below 3000 Hz

c. Rejects all frequencies between 1500 and 2500 Hz

Solution

a. Using this filter:

Listing 8: MATLAB code for Exercise 3.9a

```
% filternoise.m filter a noisy signal three ways
time=3;                        % length of time
Ts=1/10000;                    % time interval between samples
x=randn(1,time/Ts);            % generate noise signal
figure(1),plotspec(x,Ts)       % draw spectrum of input

freqs=[0 .1 0.11 1];
amps=[0 0 1 1];
b=firpm(100,freqs,amps);       % specify the HP filter
yhp=filter(b,1,x);             % do the filtering
figure(2),plotspec(yhp,Ts)     % plot the output spectrum
```

I was able to create this output.

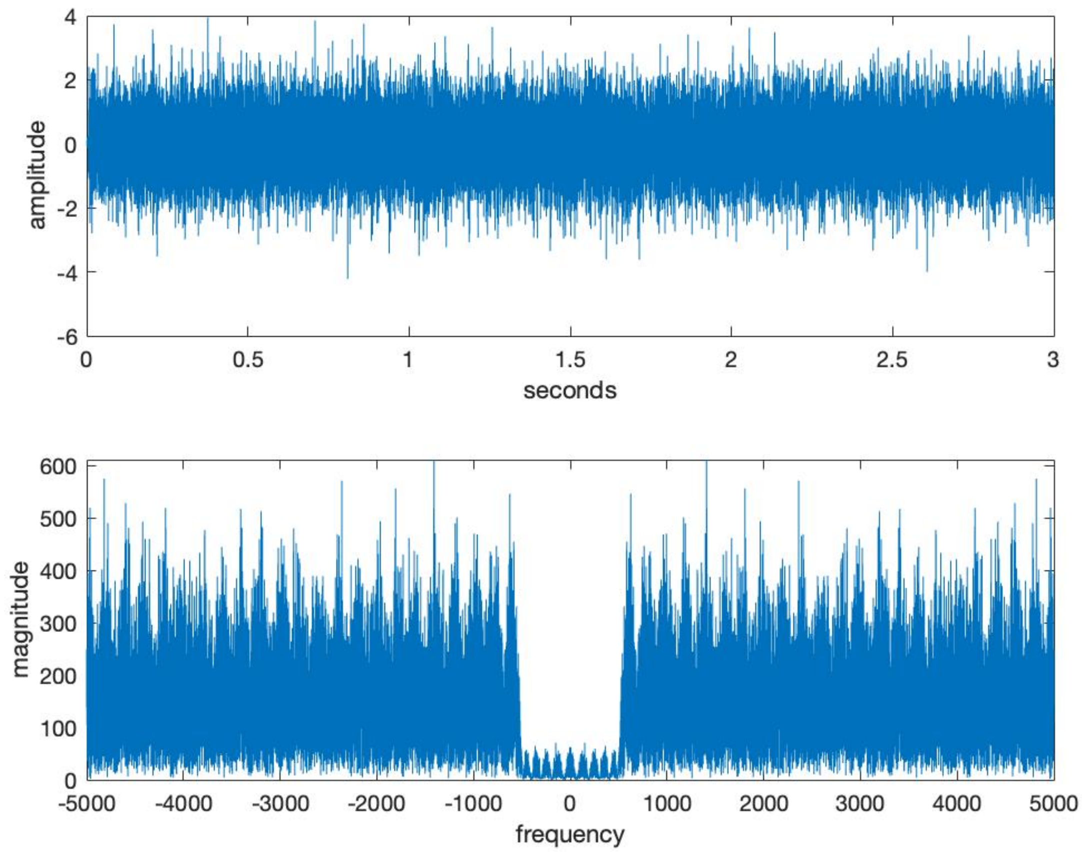


Figure 15: Passes all frequencies above 500 Hz

b. Using this filter:

Listing 9: MATLAB code for Exercise 3.9b

```
% filternoise.m filter a noisy signal three ways
time=3;                        % length of time
Ts=1/10000;                    % time interval between samples
x=randn(1,time/Ts);            % generate noise signal
figure(1),plotspec(x,Ts)       % draw spectrum of input

freqs=[0 0.6 0.61 1];
amps=[1 1 0 0];
b=firpm(100,freqs,amps);       % specify the LP filter
ylp=filter(b,1,x);             % do the filtering
figure(2),plotspec(ylp,Ts)     % plot the output spectrum
```

I was able to create this output.

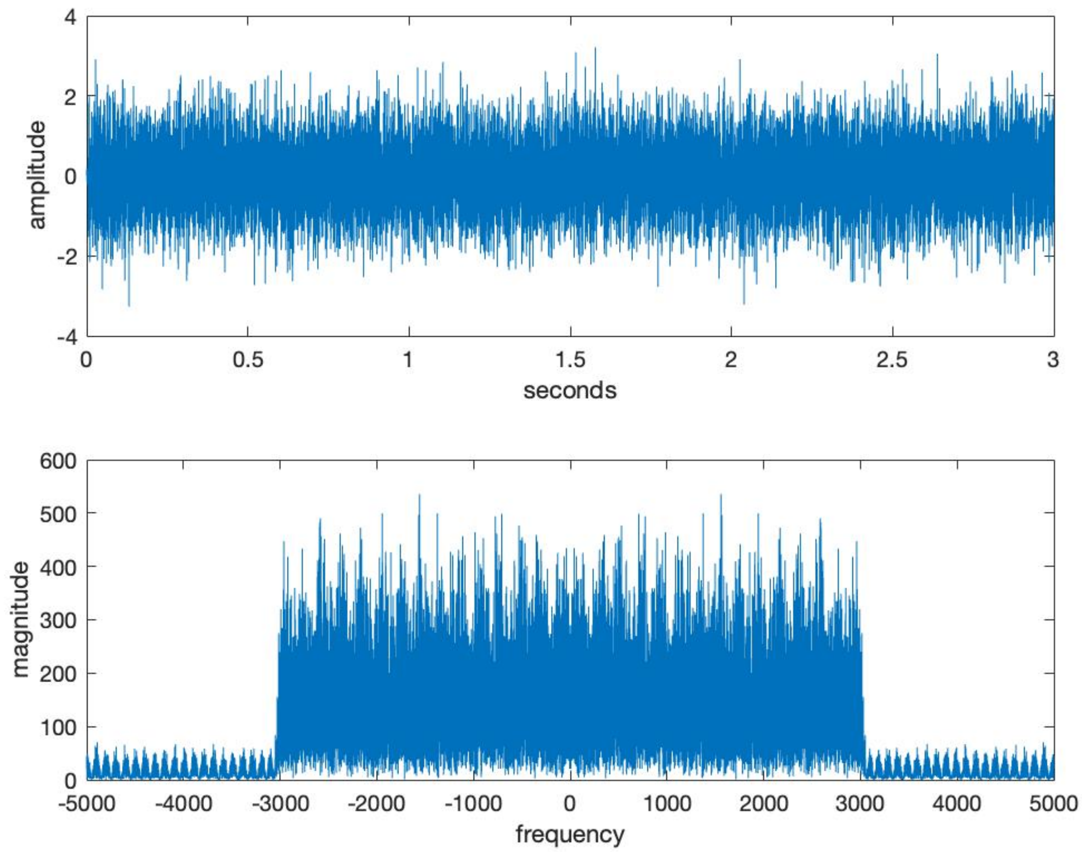


Figure 16: Passes all frequencies below 3000 Hz

c. Using this filter:

Listing 10: MATLAB code for Exercise 3.9c

```
% filternoise.m filter a noisy signal three ways
time=3;                          % length of time
Ts=1/10000;                      % time interval between samples
x=randn(1,time/Ts);              % generate noise signal
figure(1),plotspec(x,Ts)         % draw spectrum of input

freqs=[0 0.3 0.31 0.5 0.51 1];
amps=[0 0 1 1 0 0];
b=firpm(100,freqs,amps);         % BP filter
ybp=filter(b,1,x);               % do the filtering
figure(3),plotspec(ybp,Ts)       % plot the output spectrum
```

I was able to create this output.

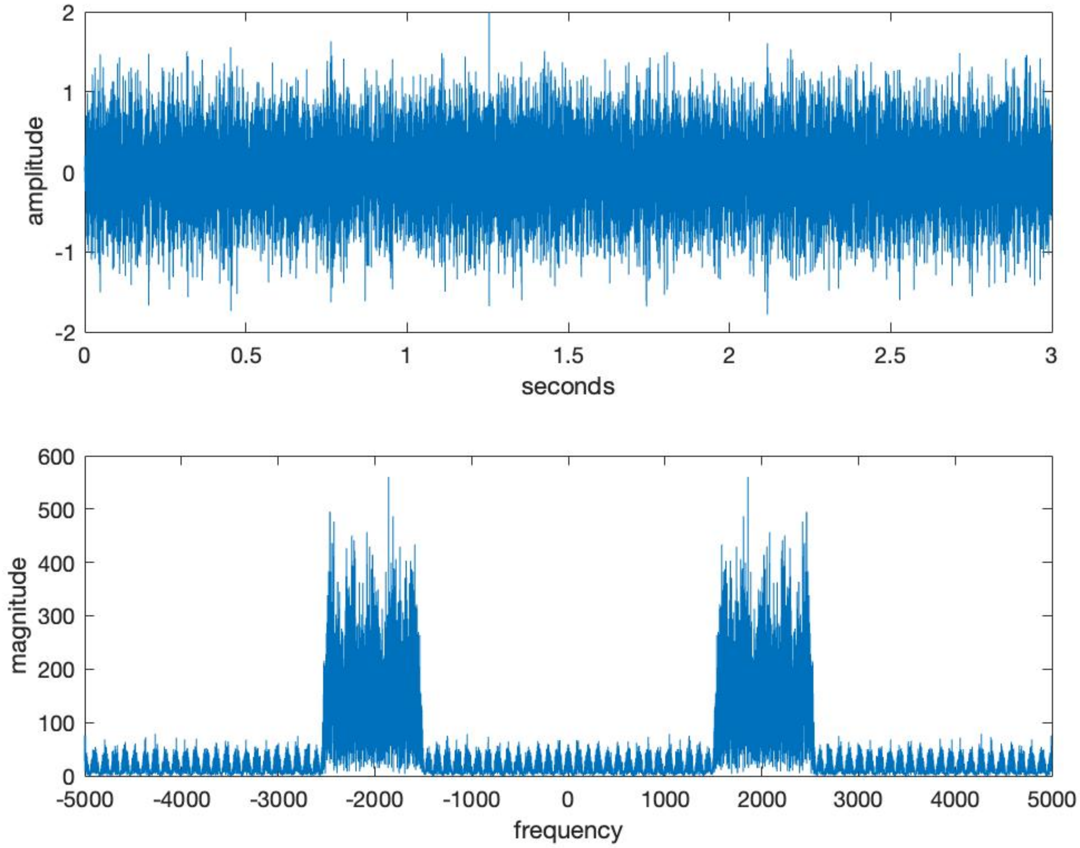


Figure 17: Rejects all frequencies between 1500 and 2500 Hz

Exercise 3.10

Change the sampling rate to T_s=1/20000. Redesign the three filters from Exercise 3.9.

Solution

a. Using this filter:

Listing 11: MATLAB code for Exercise 3.10a

```
% filternoise.m filter a noisy signal three ways
time=3;                          % length of time
Ts=1/20000;                      % time interval between samples
x=randn(1,time/Ts);              % generate noise signal
figure(1),plotspec(x,Ts)         % draw spectrum of input

freqs=[0 .05 0.051 1];
amps=[0 0 1 1];
b=firpm(100,freqs,amps);         % specify the HP filter
yhp=filter(b,1,x);               % do the filtering
figure(2),plotspec(yhp,Ts)       % plot the output spectrum
```

I was able to create this output.

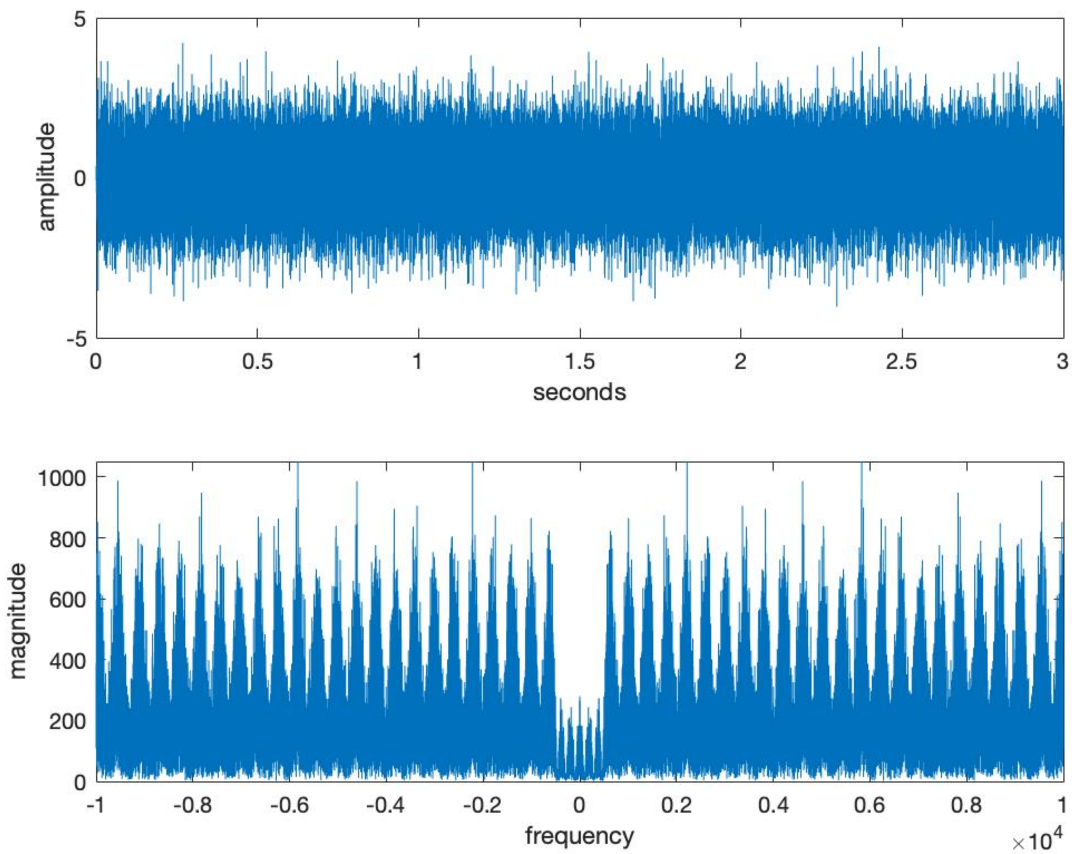


Figure 18: Passes all frequencies above 500 Hz

b. Using this filter:

Listing 12: MATLAB code for Exercise 3.10b

```
% filternoise.m filter a noisy signal three ways
time=3;                          % length of time
Ts=1/20000;                      % time interval between samples
x=randn(1,time/Ts);              % generate noise signal
figure(1),plotspec(x,Ts)         % draw spectrum of input

freqs=[0 0.3 0.31 1];
amps=[1 1 0 0];
b=firpm(100,freqs,amps);         % specify the LP filter
ylp=filter(b,1,x);               % do the filtering
figure(2),plotspec(ylp,Ts)       % plot the output spectrum
```

I was able to create this output.

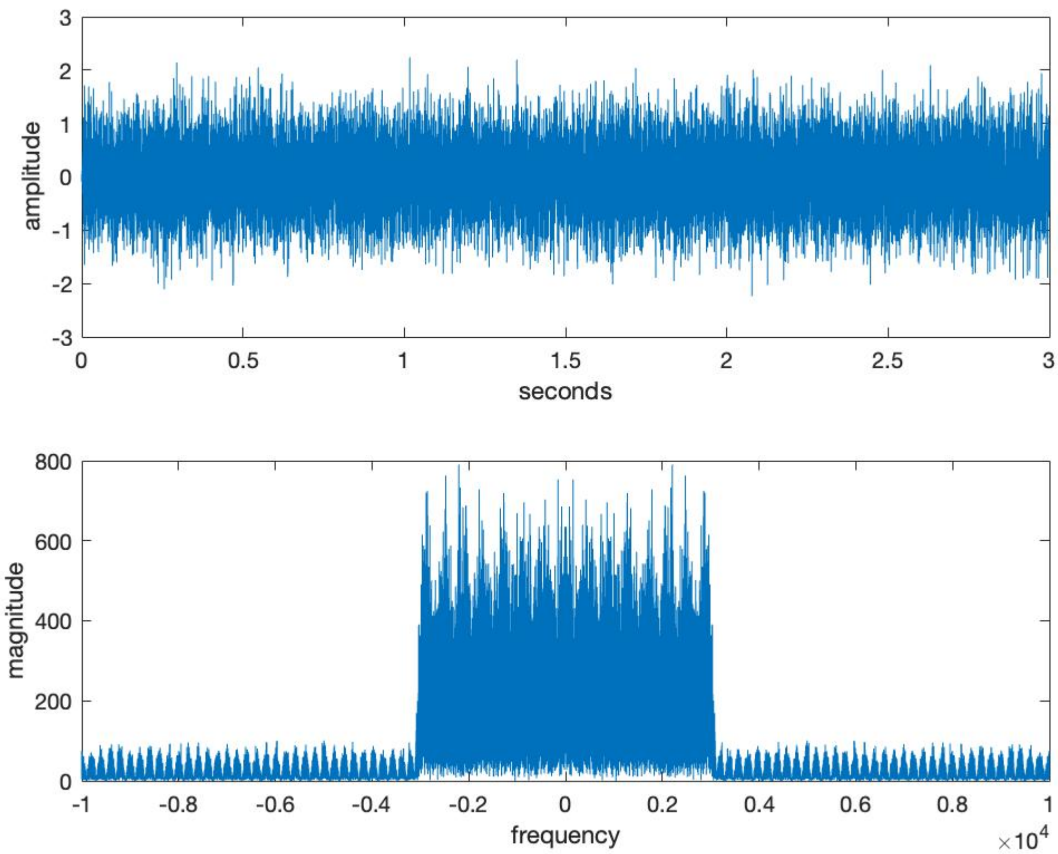


Figure 19: Passes all frequencies below 3000 Hz

c. Using this filter:

Listing 13: MATLAB code for Exercise 3.10c

```
% filternoise.m filter a noisy signal three ways
time=3;                          % length of time
Ts=1/10000;                      % time interval between samples
x=randn(1,time/Ts);              % generate noise signal
figure(1),plotspec(x,Ts)         % draw spectrum of input

freqs=[0 0.15 0.151 0.25 0.251 1];
amps=[0 0 1 1 0 0];
b=firpm(100,freqs,amps);         % BP filter
ybp=filter(b,1,x);               % do the filtering
figure(3),plotspec(ybp,Ts)       % plot the output spectrum
```

I was able to create this output.

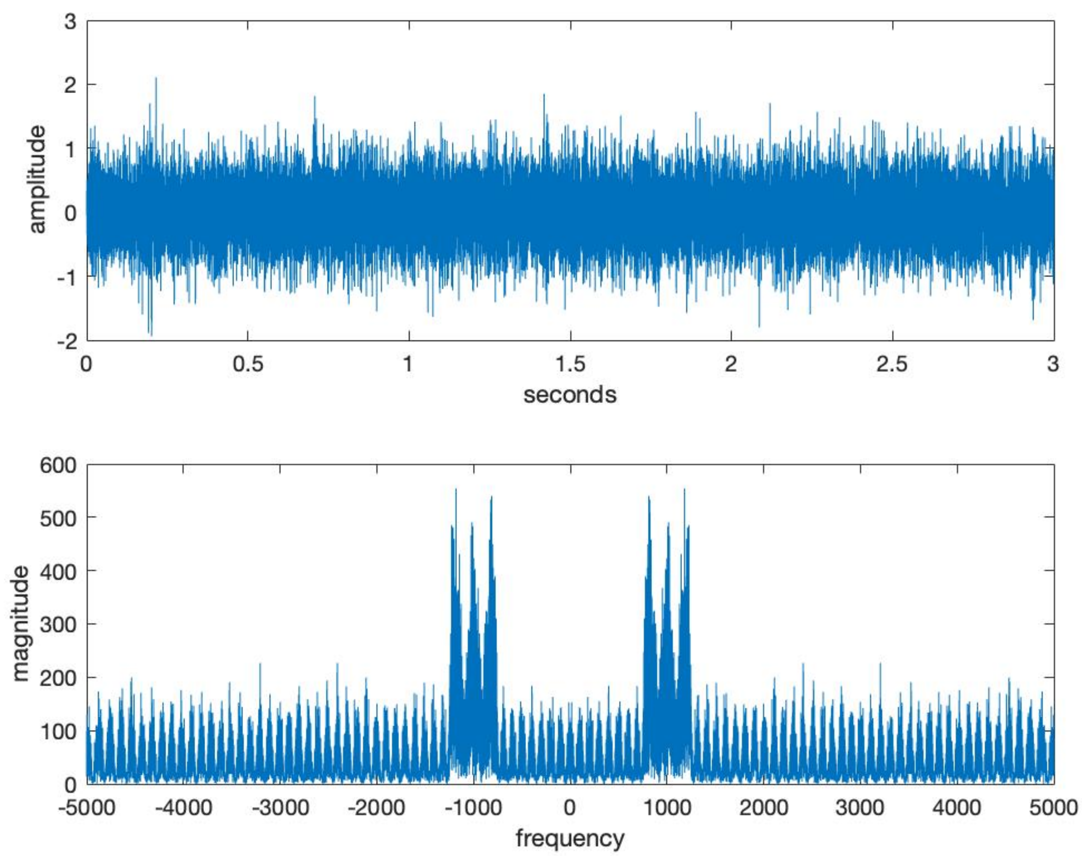


Figure 20: Rejects all frequencies between 1500 and 2500 Hz

Exercise 3.11

Let $x_1(t)$ be a cosine wave of frequency f = 800, $x_2(t)$ be a cosine wave of frequency f = 2000, and $x_3(t)$ be a cosine wave of frequency f = 4500. Let $x(t) = x_1(t) + 0.5 * x_2(t) + 2 * x_3(t)$. Use x(t) as input to each of the three filters in filternoise.m. Plot the spectra, and explain what you see.

Solution

Using this MATLAB code:

Listing 14: MATLAB code for Exercise 3.11

```
% filternoise.m filter a noisy signal three ways
time=3;                              % length of time
Ts=1/10000;                          % time interval between samples
t=Ts:Ts:time;
x1 = cos(800*t);
x2 = cos(2000*t);
x3 = cos(4500*t);
x = x1 .* x2 .* x3;
%x=randn(1,time/Ts);                  % generate noise signal
figure(1),plotspec(x,Ts)             % draw spectrum of input

freqs=[0 0.2 0.21 1];
amps=[1 1 0 0];
b=firpm(100,freqs,amps);             % specify the LP filter
ylp=filter(b,1,x);                   % do the filtering
figure(2),plotspec(ylp,Ts)           % plot the output spectrum

freqs=[0 0.24 0.26 0.5 0.51 1];
amps=[0 0 1 1 0 0];
b=firpm(100,freqs,amps);             % BP filter
ybp=filter(b,1,x);                   % do the filtering
figure(3),plotspec(ybp,Ts)           % plot the output spectrum

freqs=[0 0.74 0.76 1];
amps=[0 0 1 1];
b=firpm(100,freqs,amps);             % specify the HP filter
yhp=filter(b,1,x);                   % do the filtering
figure(4),plotspec(yhp,Ts)           % plot the output spectrum

%Here's how the figure filternoise.eps was actually drawn
N=length(x);                         % length of the signal x
t=Ts*(1:N);                          % define a time vector
ssf=(-N/2:N/2-1)/(Ts*N);             % frequency vector
fx=fftshift(fft(x(1:N)));
figure(5), subplot(4,1,1), plot(ssf,abs(fx))
xlabel('magnitude spectrum at input')
fyl=fftshift(fft(ylp(1:N)));
subplot(4,1,2), plot(ssf,abs(fyl))
xlabel('magnitude spectrum at output of low pass filter')
fybp=fftshift(fft(ybp(1:N)));
subplot(4,1,3), plot(ssf,abs(fybp))
```

```
xlabel('magnitude spectrum at output of band pass filter')
fyhp=fftshift(fft(yhp(1:N)));
subplot(4,1,4), plot(ssf,abs(fyhp))
xlabel('magnitude spectrum at output of high pass filter')
```

and looking at this generated spectrum plot:

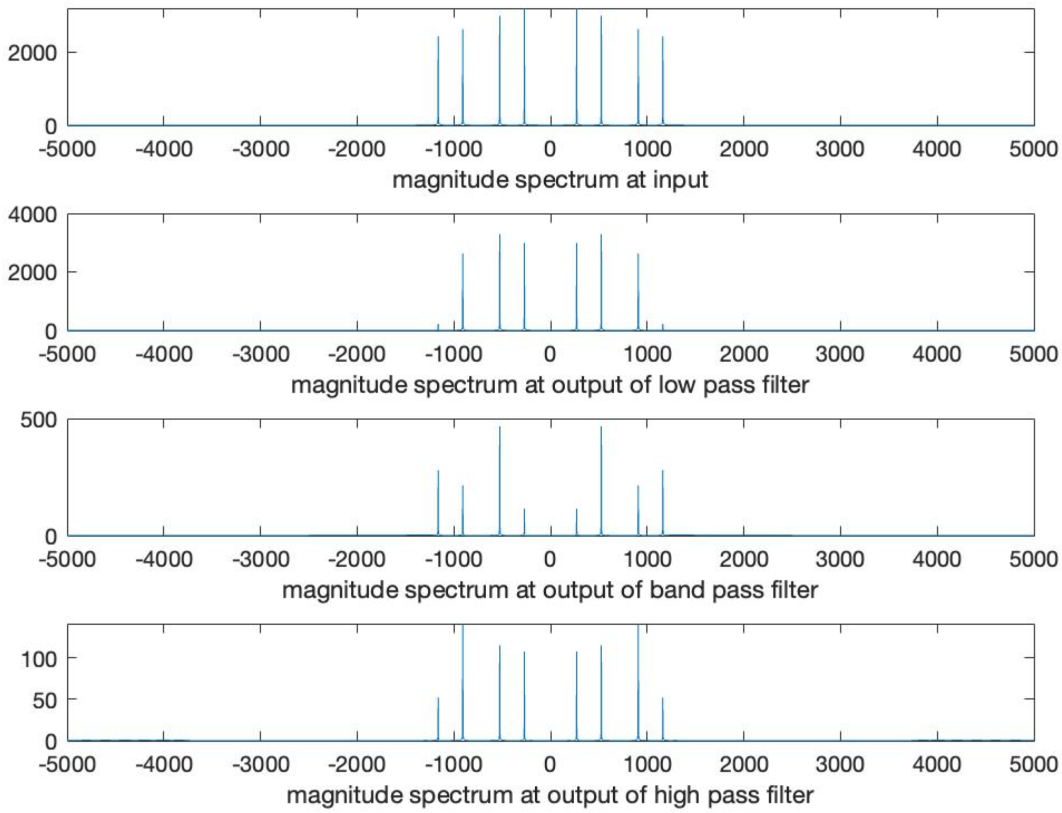


Figure 21

LOW PASS: The highest frequency spectra gets filtered out.
BAND PASS: All signals get filtered out.
HIGH PASS: All signals get filtered out.
This becomes more clear when comparing to the output of the unmodified filternoise.m below.

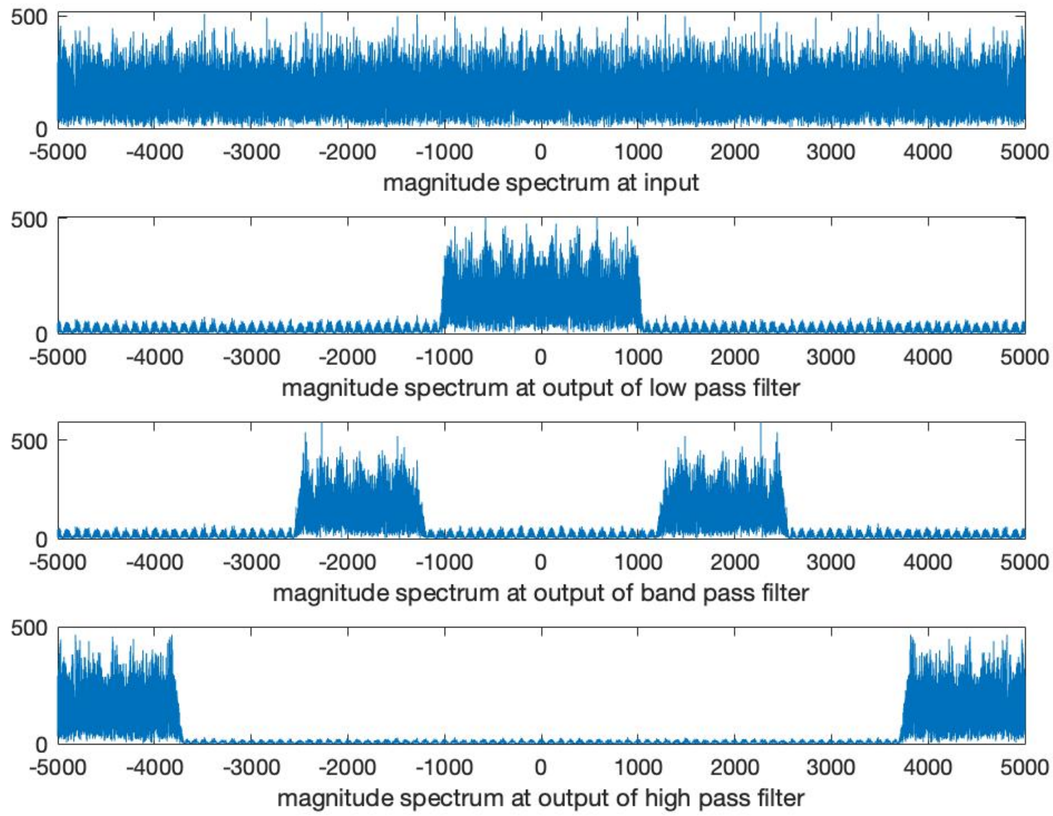


Figure 22: original filternoise.m output with noisy input

Exercise 3.26

Mimic the code in modulate.m to find the spectrum of the output y(t) of a modulator block (with modulation frequency $f_c = 1000$ Hz) when

a. The input is $x(t) = cos(2 * \pi * f_1 * t) + cos(2 * \pi * f_2 * t)$ for $f_1 = 100$ and $f_2 = 150$ Hz

b. The input is a square wave with fundamental f = 150 Hz

c. The input is a noise signal with all energy below 300 Hz

Solution

a. Using this MATLAB script:

Listing 15: MATLAB code for Exercise 3.26a

```
% modulate.m: change the frequency of the input
time=.5; Ts=1/10000;              % time and sampling interval
t=Ts:Ts:time;                     % define a 'time' vector
fc=1000; cmod=cos(2*pi*fc*t); % create cos of freq fc
f1=100; f2=150;
x=(cos(2*pi*f1*t) + cos(2*pi*f2*t)); % input is cos of freq fi
y=cmod.*x;                        % multiply input by cmod
figure(1), plotspec(cmod,Ts)  % find spectra and plot
figure(2), plotspec(x,Ts)
figure(3), plotspec(y,Ts)

%Here's how the figure was actually drawn
N=length(x);                              % length of the signal x
t=Ts*(1:N);                               % define a time vector
ssf=(-N/2:N/2-1)/(Ts*N);                  % frequency vector
fx=fftshift(fft(x(1:N)));
figure(4), subplot(3,1,1), plot(ssf,abs(fx))
xlabel('magnitude spectrum at input')
fcmod=fftshift(fft(cmod(1:N)));
subplot(3,1,2), plot(ssf,abs(fcmod))
xlabel('magnitude spectrum of the oscillator')
fy=fftshift(fft(y(1:N)));
subplot(3,1,3), plot(ssf,abs(fy))
xlabel('magnitude spectrum at output')
```

I was able to create this output.

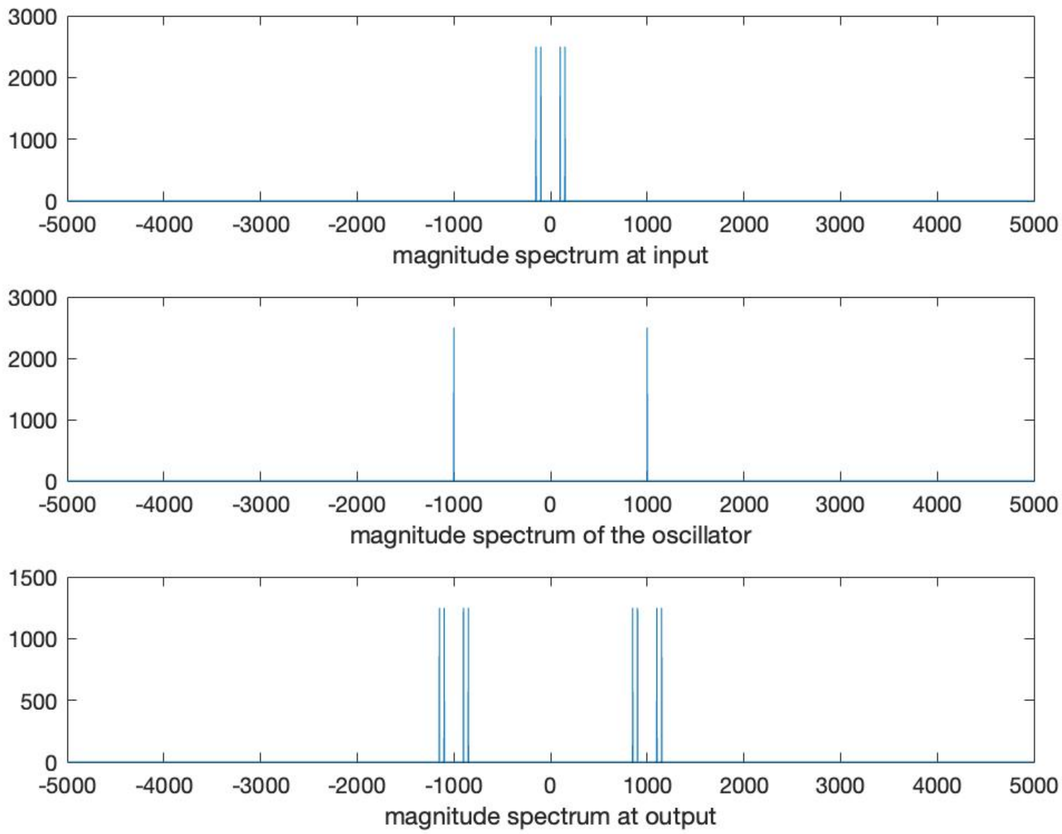


Figure 23: The input is $x(t) = cos(2 * \pi * f_1 * t) + cos(2 * \pi * f_2 * t)$ for $f_1 = 100$ and $f_2 = 150$ Hz

b. Using this MATLAB script:

Listing 16: MATLAB code for Exercise 3.26b

```
% modulate.m: change the frequency of the input
time=.5; Ts=1/10000;              % time and sampling interval
t=Ts:Ts:time;                     % define a 'time' vector
fc=1000; cmod=cos(2*pi*fc*t);     % create cos of freq fc
fi=100; x=cos(2*pi*fi*t);         % input is cos of freq fi
f = 150;                        % frequency of square wave, 150
x = sign(cos(2*pi*f*t)); % square wave = sign of cos wave
y=cmod.*x;                        % multiply input by cmod
figure(1), plotspec(cmod,Ts)      % find spectra and plot
figure(2), plotspec(x,Ts)
figure(3), plotspec(y,Ts)

%Here's how the figure was actually drawn
N=length(x);                                  % length of the signal x
t=Ts*(1:N);                                   % define a time vector
ssf=(-N/2:N/2-1)/(Ts*N);                      % frequency vector
fx=fftshift(fft(x(1:N)));
```

```
figure(4), subplot(3,1,1), plot(ssf,abs(fx))
xlabel('magnitude spectrum at input')
fcmod=fftshift(fft(cmod(1:N)));
subplot(3,1,2), plot(ssf,abs(fcmod))
xlabel('magnitude spectrum of the oscillator')
fy=fftshift(fft(y(1:N)));
subplot(3,1,3), plot(ssf,abs(fy))
xlabel('magnitude spectrum at output')
```

I was able to create this output.

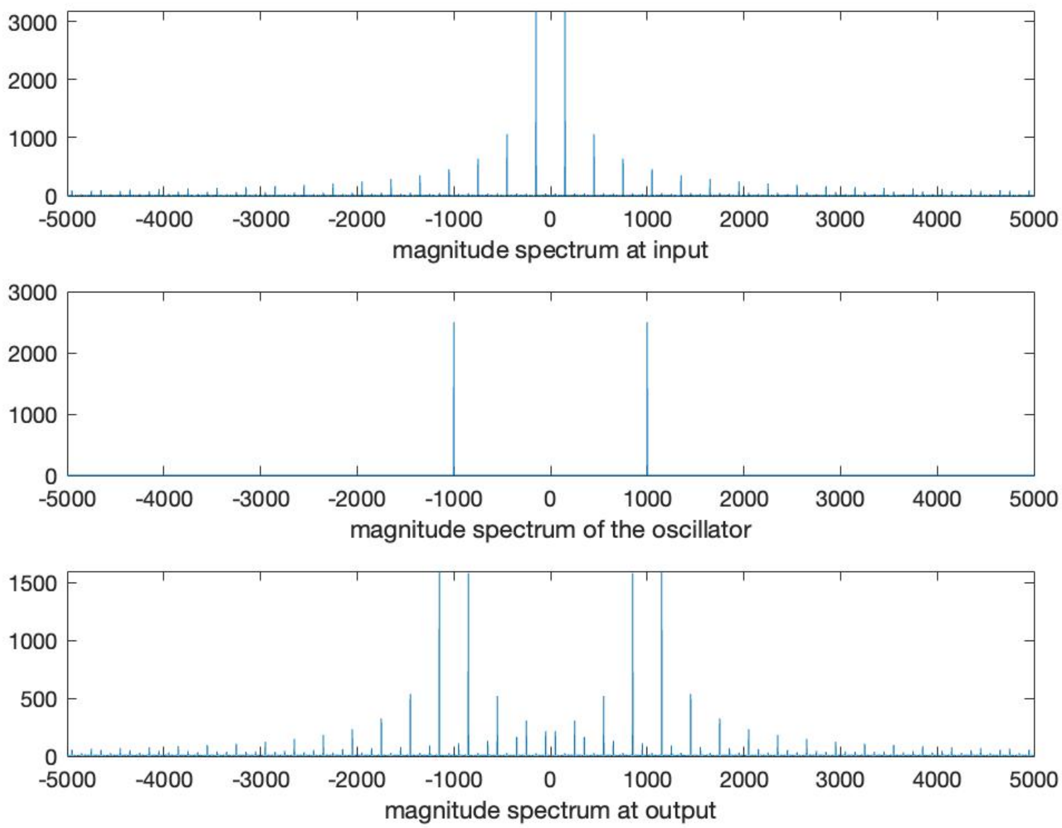


Figure 24: The input is a square wave with fundamental f = 150 Hz

c. Using this MATLAB script:

Listing 17: MATLAB code for Exercise 3.26c

```
% modulate.m: change the frequency of the input
time=.5; Ts=1/10000;              % time and sampling interval
t=Ts:Ts:time;                     % define a 'time' vector
fc=1000; cmod=cos(2*pi*fc*t);     % create cos of freq fc
rand_sig=randn(1,time/Ts);        % generate noise signal
freqs=[0 0.06 0.061 1];
```

```
amps=[1 1 0 0];
b=firpm(100,freqs,amps);        % specify the LP filter
x=filter(b,1,rand_sig);         % do the filtering
y=cmod.*x;                      % multiply input by cmod
figure(1), plotspec(cmod,Ts)    % find spectra and plot
figure(2), plotspec(x,Ts)
figure(3), plotspec(y,Ts)

%Here's how the figure was actually drawn
N=length(x);                            % length of the signal x
t=Ts*(1:N);                             % define a time vector
ssf=(-N/2:N/2-1)/(Ts*N);                % frequency vector
fx=fftshift(fft(x(1:N)));
figure(4), subplot(3,1,1), plot(ssf,abs(fx))
xlabel('magnitude spectrum at input')
fcmod=fftshift(fft(cmod(1:N)));
subplot(3,1,2), plot(ssf,abs(fcmod))
xlabel('magnitude spectrum of the oscillator')
fy=fftshift(fft(y(1:N)));
subplot(3,1,3), plot(ssf,abs(fy))
xlabel('magnitude spectrum at output')
```

I was able to create this output.

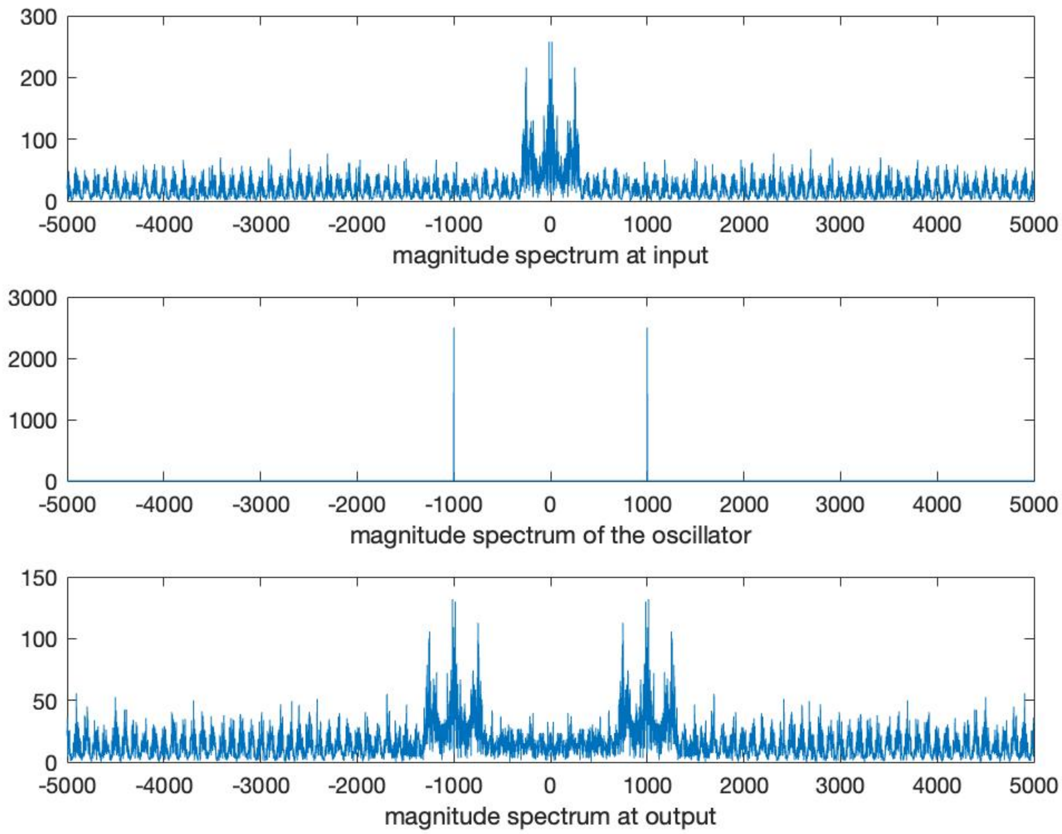


Figure 25: The input is a noise signal with all energy below 300 Hz