Table of Contents

Read in the file

```
[m, Fs] = audioread('test_message.mp3');
T = 1 / Fs; L =
length(m); t =
(0:L-1) * T;
```

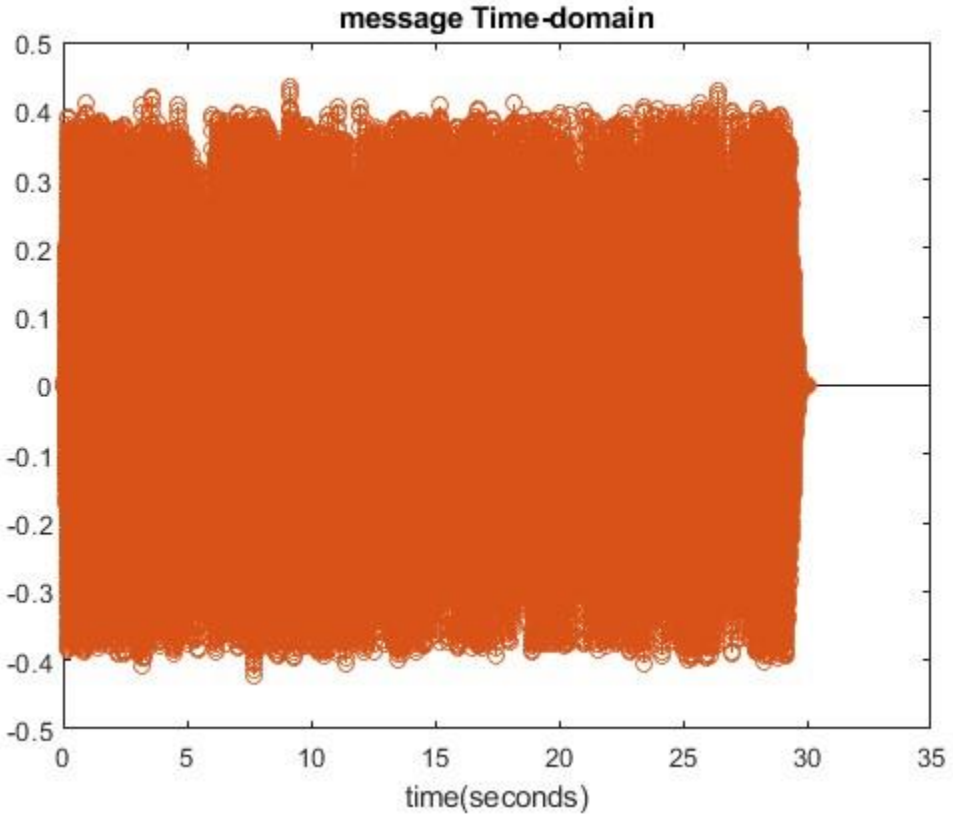
Create object for playing audio

```
pl = audioplayer(m,Fs);  % original signal
%pl.play;
```

Plot audio

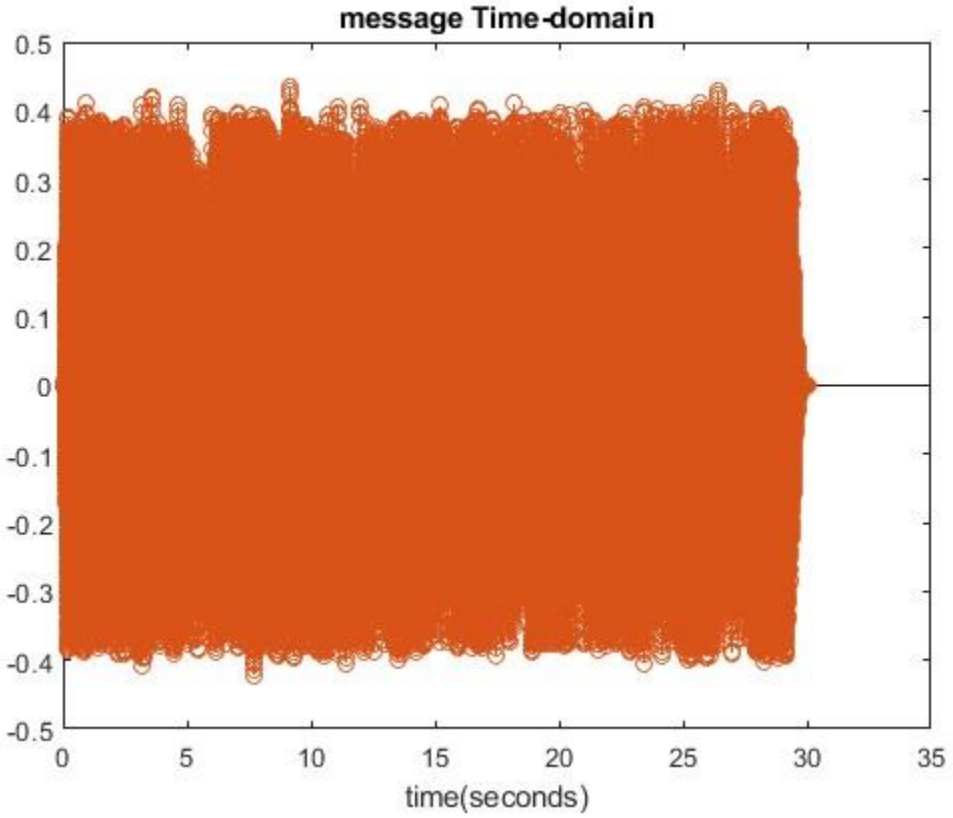
```
N = size(m, 1); figure;
stem(t, m);
```

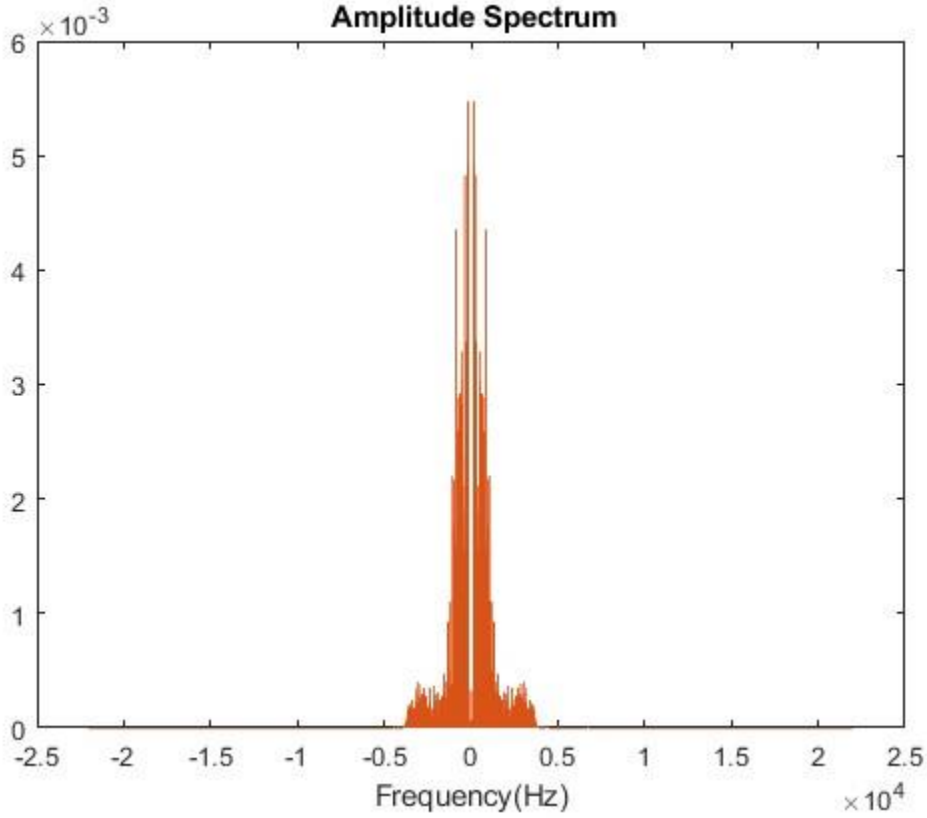
```
title('message Time-domain');  xlabel('time(seconds)');
```



Plot the spectrum

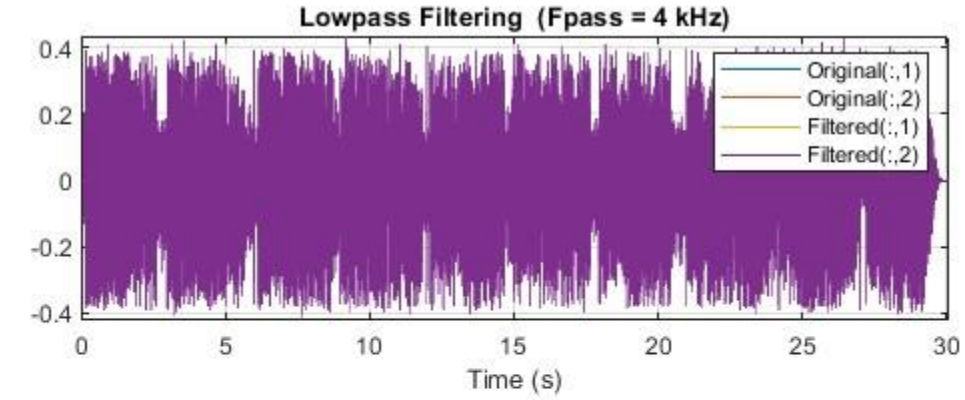
```
dm = Fs / N; w = (-(N/2):(N/2)-1)*dm; y = fft(m) / N;
% For normalizing y2 = fftshift(y); figure;  plot(w,
abs(y2));      title('      Amplitude      Spectrum');
xlabel('Frequency(Hz)');
```

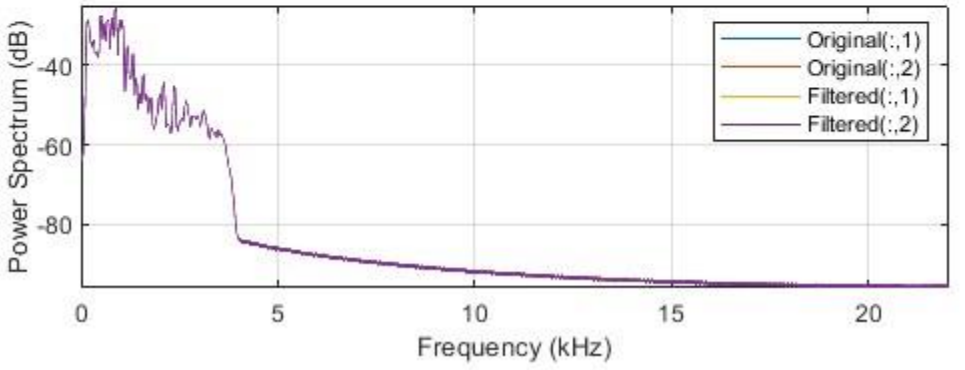

message Time-domain
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
5
10
15
20
25
30
35
time(seconds)


Amplitude Spectrum
×10-3
6
5
4
3
2
1
0
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency(Hz)
×10^4

filtering at 4K Hz (Low pass filter)

```
lowpass(m,4000,Fs); filter_m =
lowpass(m,4000,Fs);
```





the average error between the original signal and the band limited signal.

```
err = immse(m,filter_m);
disp("the error =") disp
(err)

the error =
5.2813e-09
```

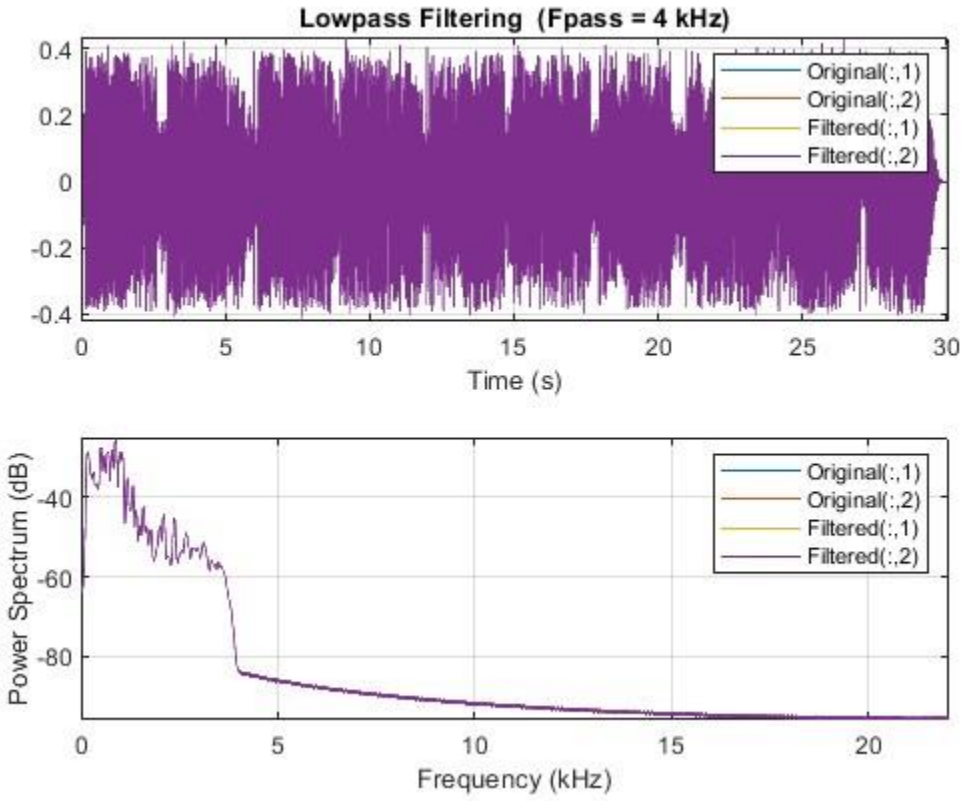
AM modulation of the recorded signal

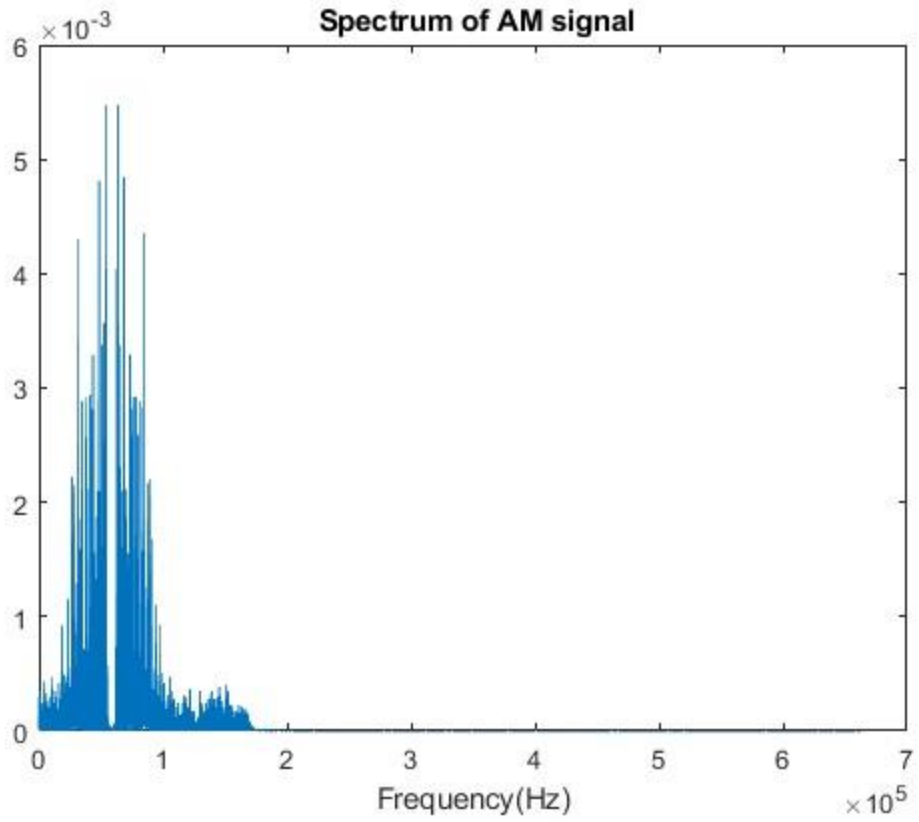
```
AM_modulate_signal = ammod(m,Fs,1000000);
```

Spectrum of modulated signal

```
spectrumAM = fft(AM_modulate_signal);
```

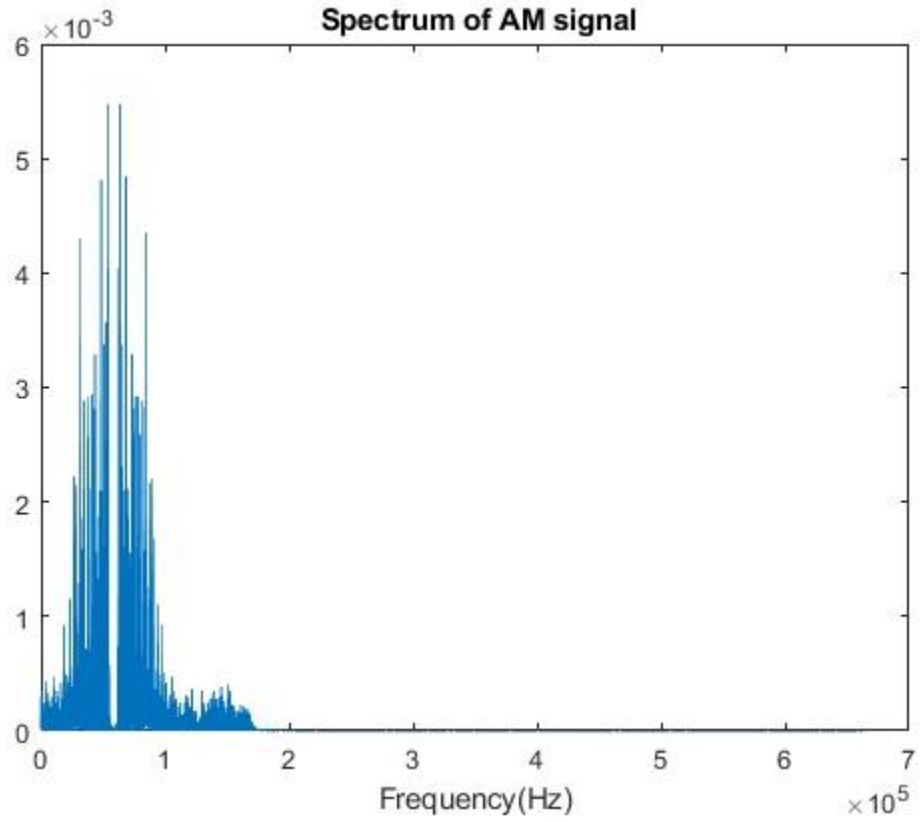
```
lengthOfSignal      =      length(m);      normalizedSpectrumAm=
abs(spectrumAM/lengthOfSignal);              frequencySpectrumAm=
normalizedSpectrumAm(1:lengthOfSignal/2+1);
frequencySpectrumAm(2:end-1)   =   2*frequencySpectrumAm(2:end-1);
figure;    plot(frequencySpectrumAm);  title('  Spectrum  of  AM
signal');  xlabel('Frequency(Hz)');
```

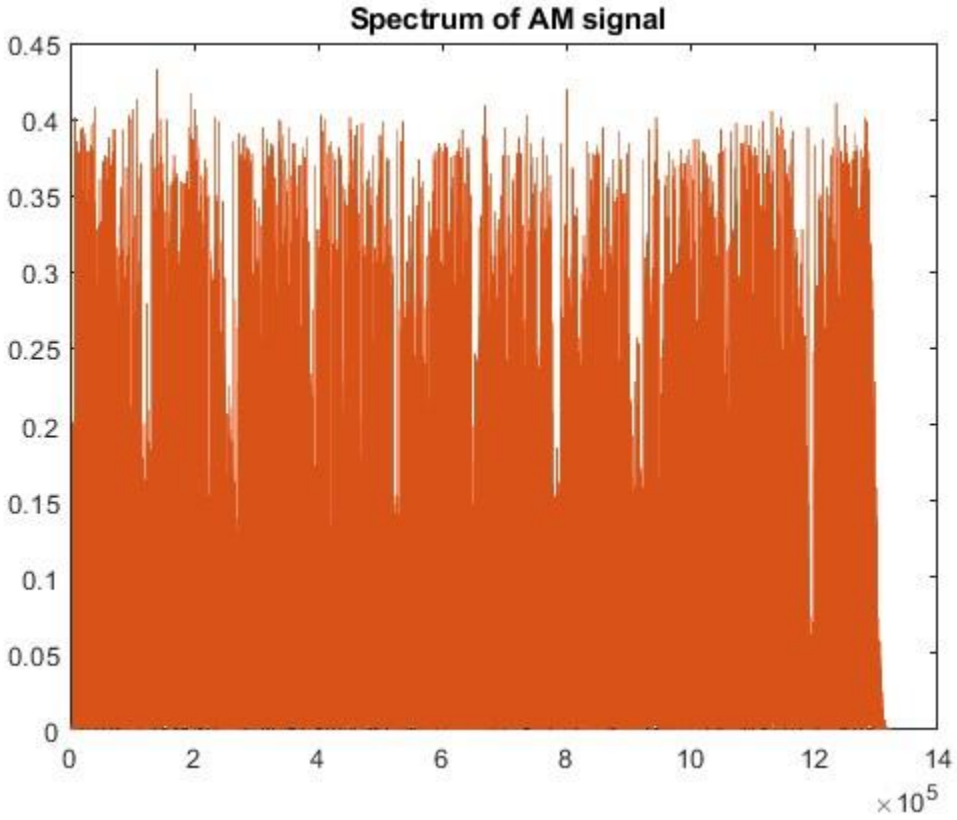




Envelope Detection based on Hilbert Transform and then FFT

```
envelope = abs (hilbert (AM_modulate_signal));
figure;  plot(envelope); title(' Spectrum of AM
signal');
```


Spectrum of AM signal
×10^-3
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
Frequency(Hz)
×10^5


Spectrum of AM signal
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
2
4
6
8
10
12
14
×10^5

the average error between the original signal and the transmitted message.

```
err = immse(m,envelope);
disp("the error =") disp
(err)
```

```
the error =
0.0423
```

signal with noise

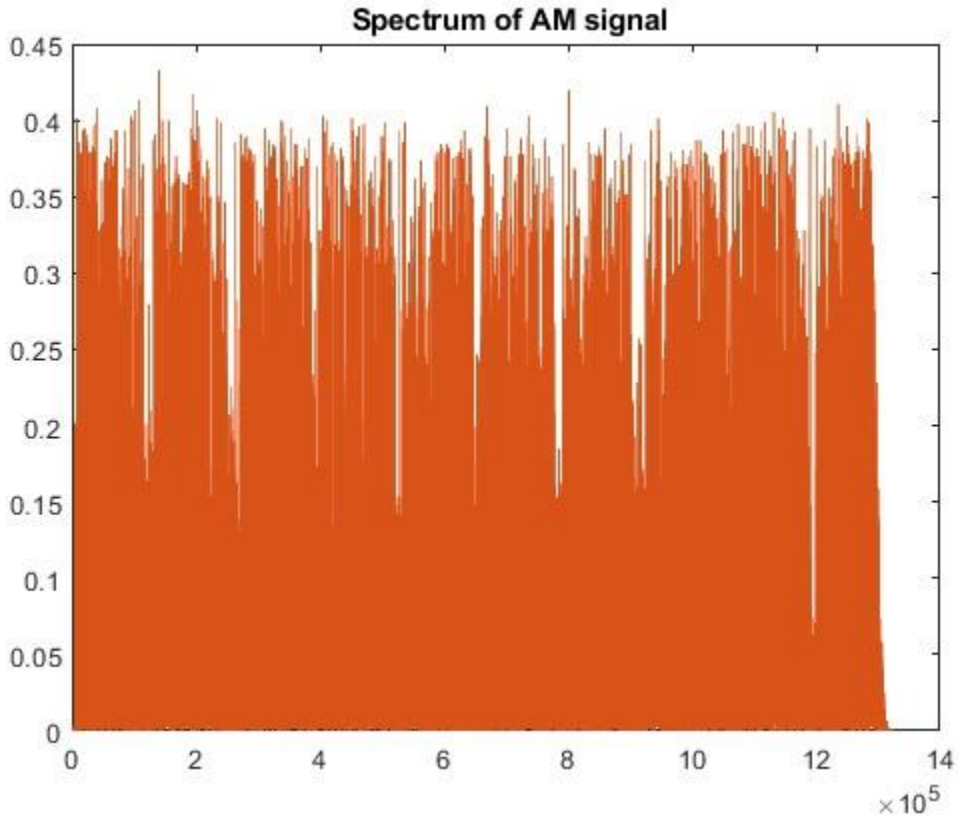
```
Noise_signal = awgn(m,10);
```

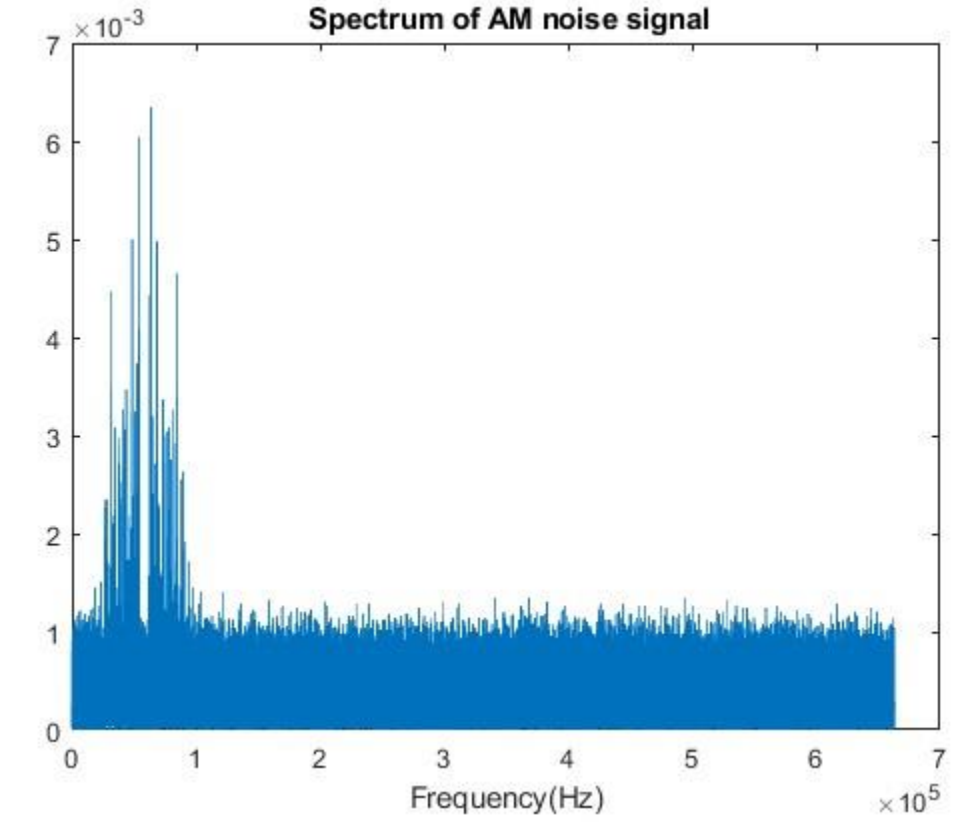
AM modulation of the recorded signal + Noise

```
AM_modulate_noise_signal = ammod(Noise_signal,Fs,1000000);
```

Spectrum of modulated signal

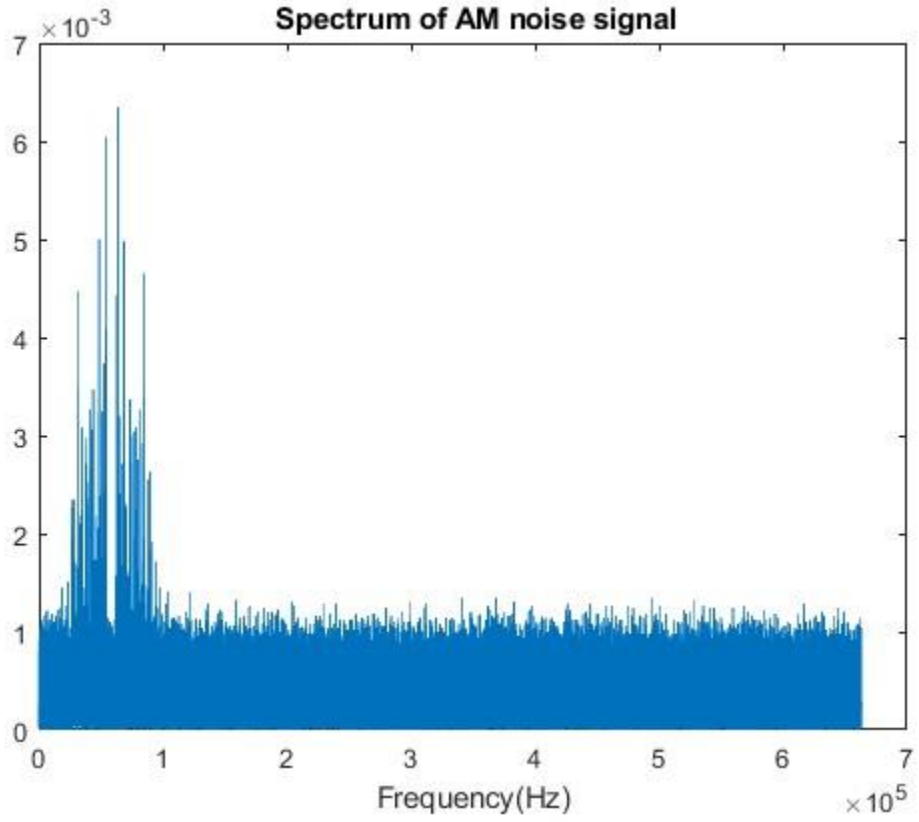
```
spectrumAM = fft(AM_modulate_noise_signal); lengthOfSignal =
length(m); normalizedSpectrumAm= abs(spectrumAM/lengthOfSignal);
frequencySpectrumAm= normalizedSpectrumAm(1:lengthOfSignal/2+1);
frequencySpectrumAm(2:end-1) = 2*frequencySpectrumAm(2:end-1);
figure; plot(frequencySpectrumAm); title(' Spectrum of AM noise
signal'); xlabel('Frequency(Hz)');
```

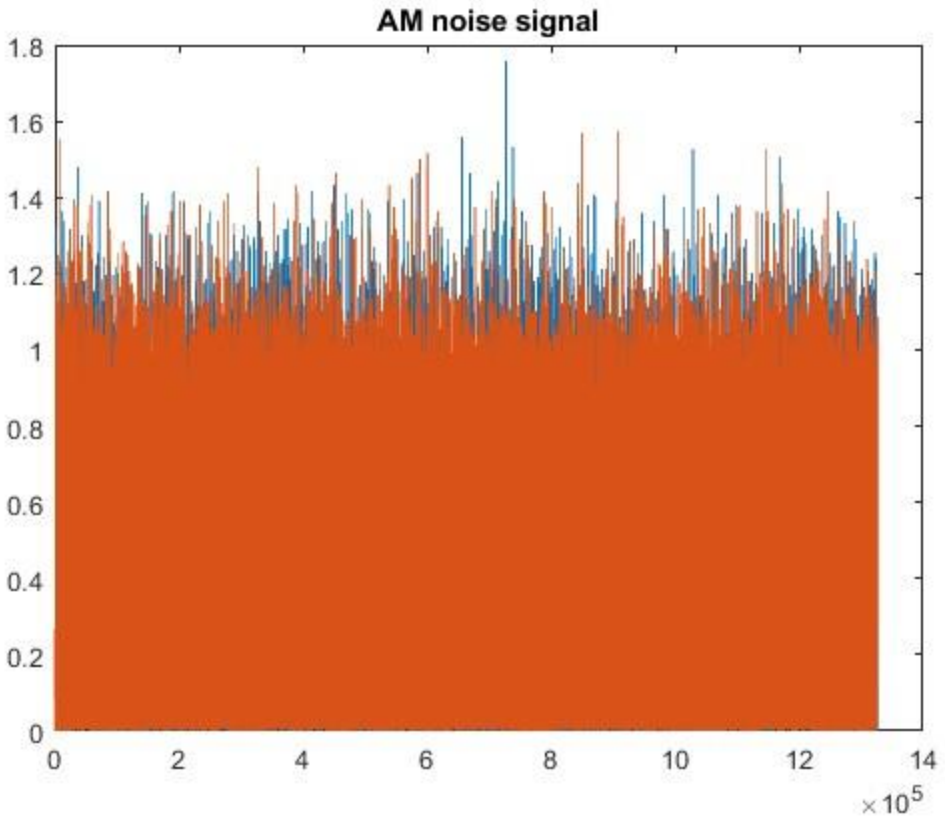

Spectrum of AM signal
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
2
4
6
8
10
12
14
×10^5


Spectrum of AM noise signal
×10^-3
7
6
5
4
3
2
1
0
0
1
2
3
4
5
6
7
Frequency(Hz)
×10^5

Envelope Detection based on Hilbert Transform and then FFT for noise signal

```
envelope = abs (hilbert (AM_modulate_noise_signal));
figure;  plot(envelope); title('AM noise signal');
```





the average error between the original signal and the transmitted message.

```
err = immse(m,envelope);
disp("the error =") disp
(err)
```

```
the error =
0.1421
```

FM modulatiom

Read in the file

```
[m, Fs] = audioread('test_message.mp3');
T = 1 / Fs; L =
length(m); t =
(0:L-1) * T;
```

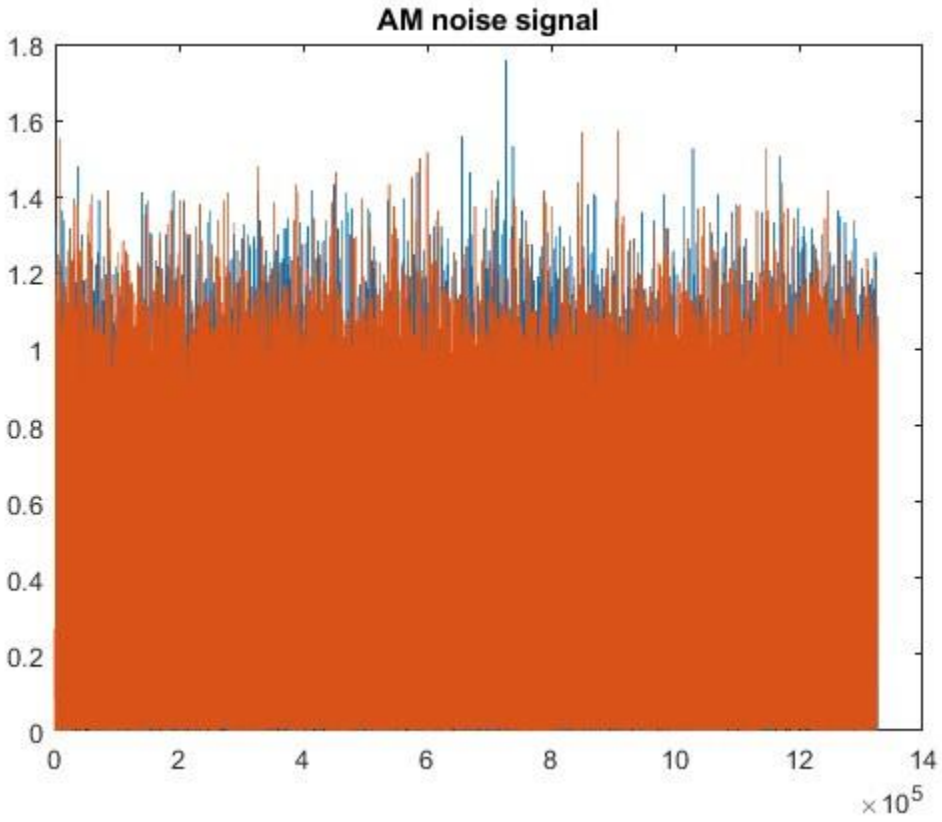
Create object for playing audio

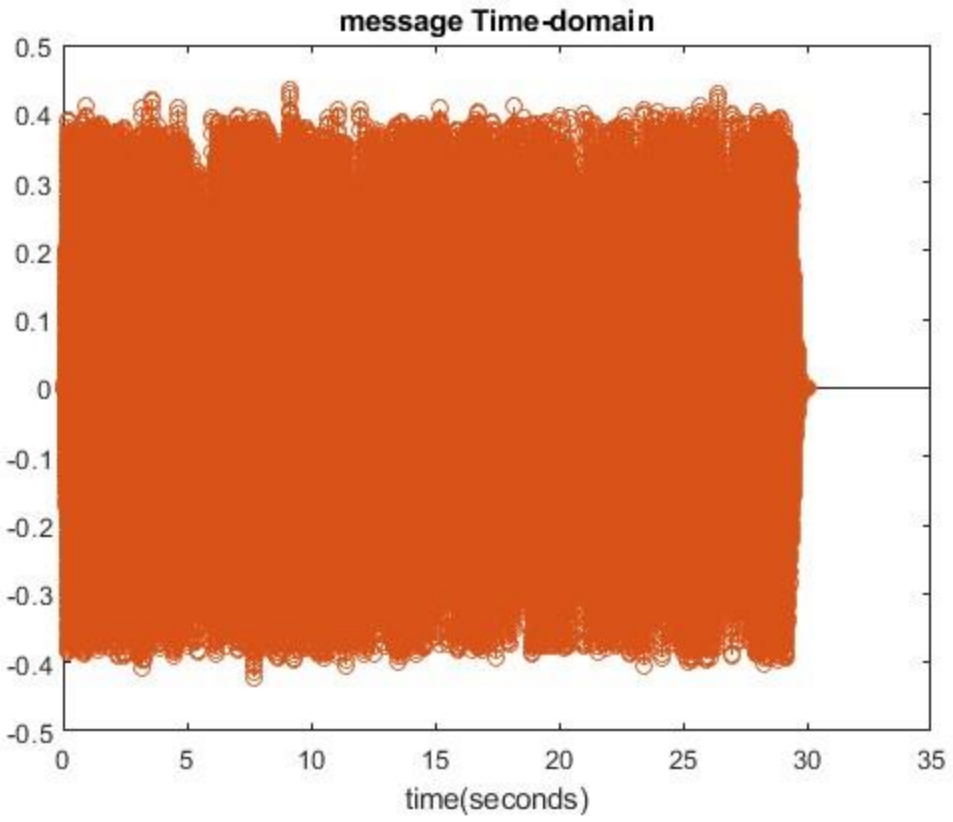
```
pl = audioplayer(m,Fs);  % original signal
```

```
%pl.play;
```

Plot audio

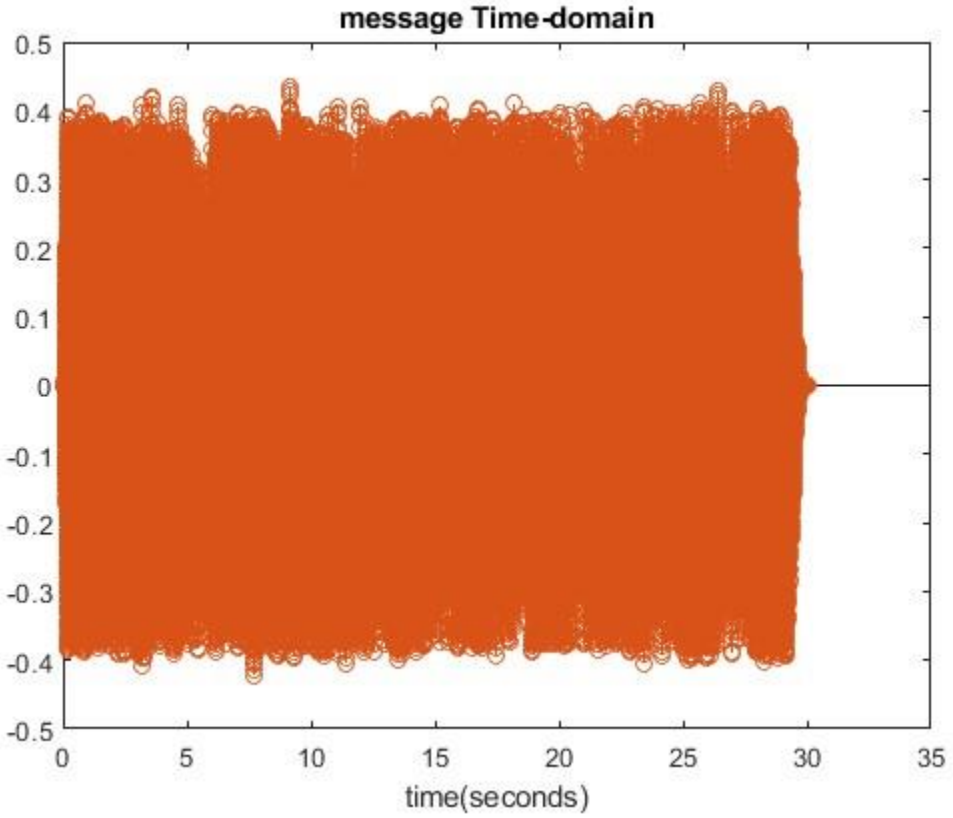
```
N = size(m, 1); figure;  stem(t, m); title('message Time-
domain');  xlabel('time(seconds)');
```

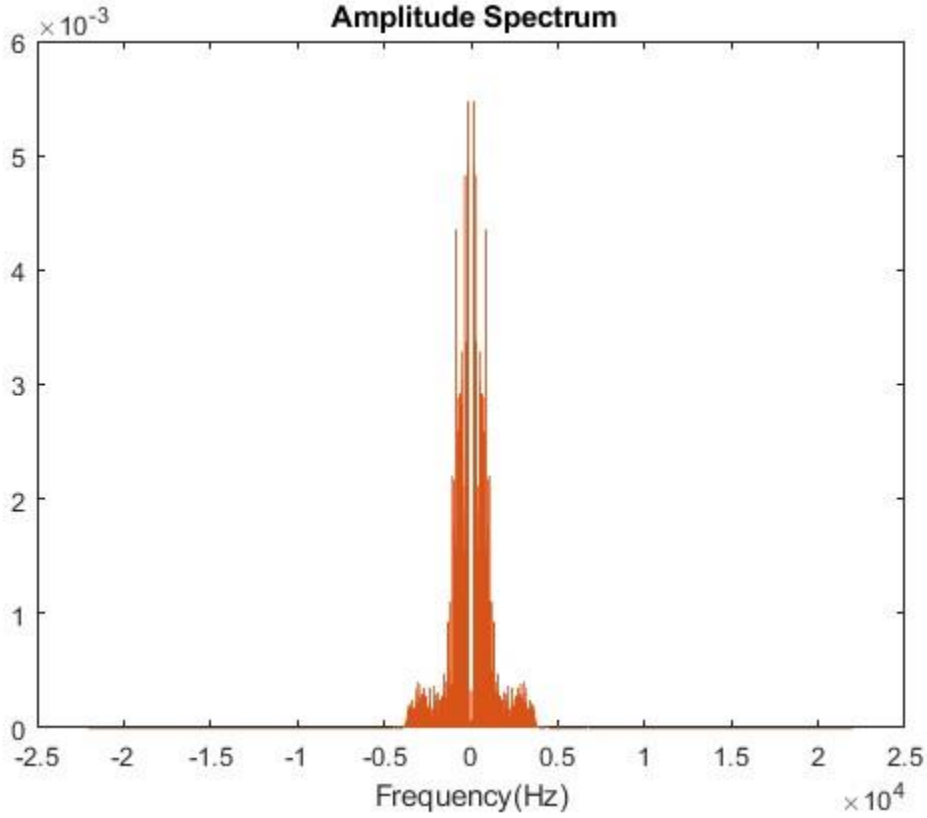




Plot the spectrum

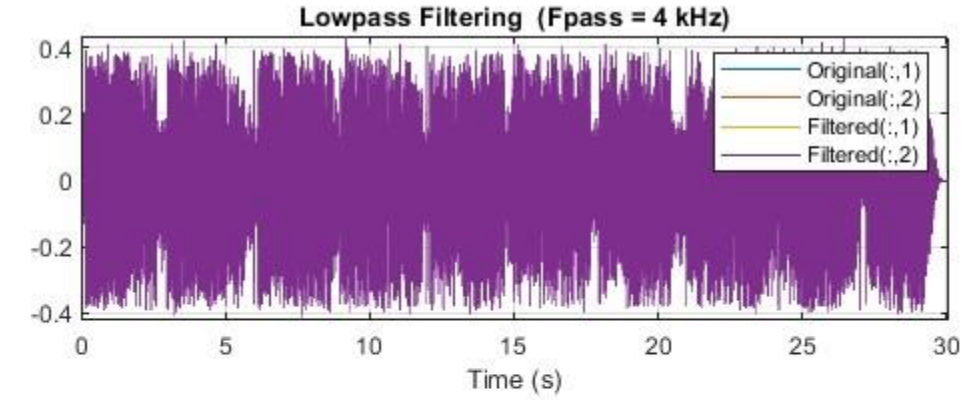
```
dm = Fs / N; w = (-(N/2):(N/2)-1)*dm; y = fft(m) / N;
% For normalizing y2 = fftshift(y); figure;  plot(w,
abs(y2));      title('      Amplitude      Spectrum');
xlabel('Frequency(Hz)');
```

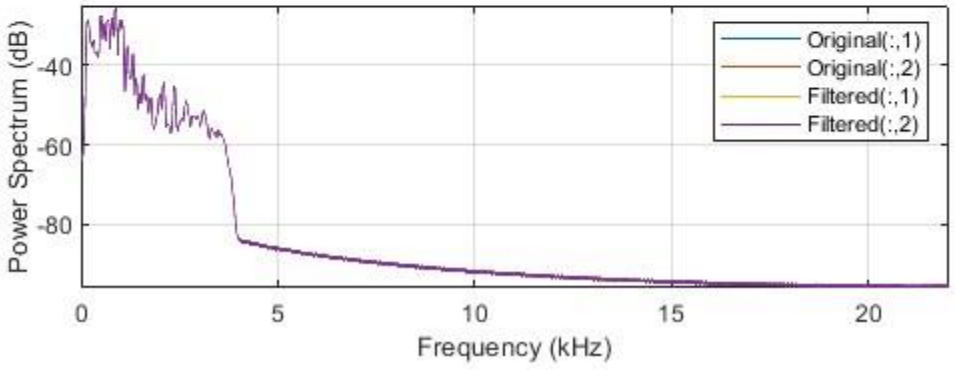

message Time-domain
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
5
10
15
20
25
30
35
time(seconds)


Amplitude Spectrum
×10-3
6
5
4
3
2
1
0
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2
2.5
Frequency(Hz)
×10^4

filtering at 4K Hz (Low pass filter)

```
lowpass(m,4000,Fs); filter_m =
lowpass(m,4000,Fs);
```





the average error between the original signal and the band limited signal.

```
err = immse(m,filter_m);
disp("the error =") disp
(err)
```

```
the error =
5.2813e-09
```

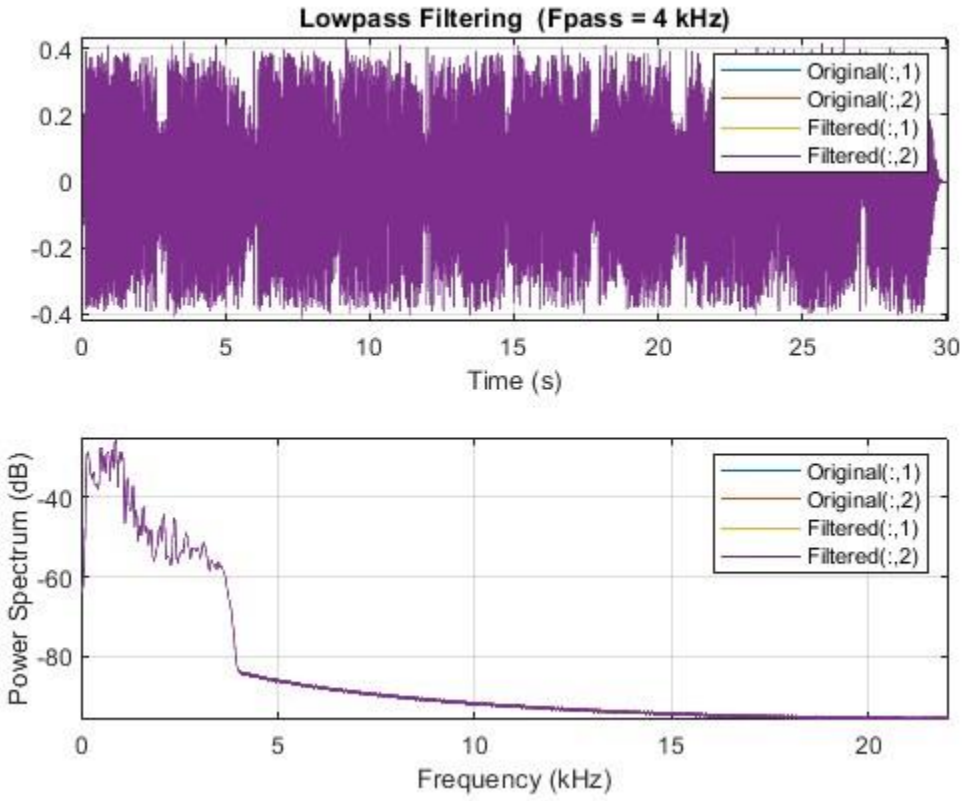
FM modulation of the recorded signal

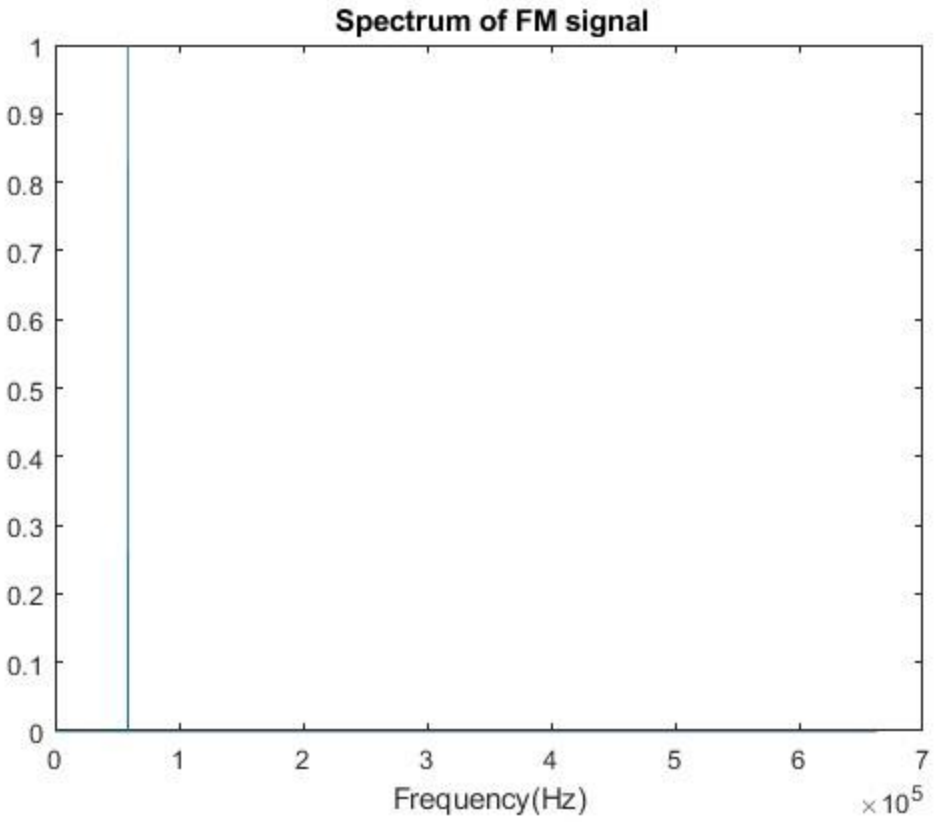
```
FM_modulate_signal = fmmod(m,Fs,1000000,2);
```

Spectrum of modulated signal

```
spectrumFM = fft(FM_modulate_signal);
```

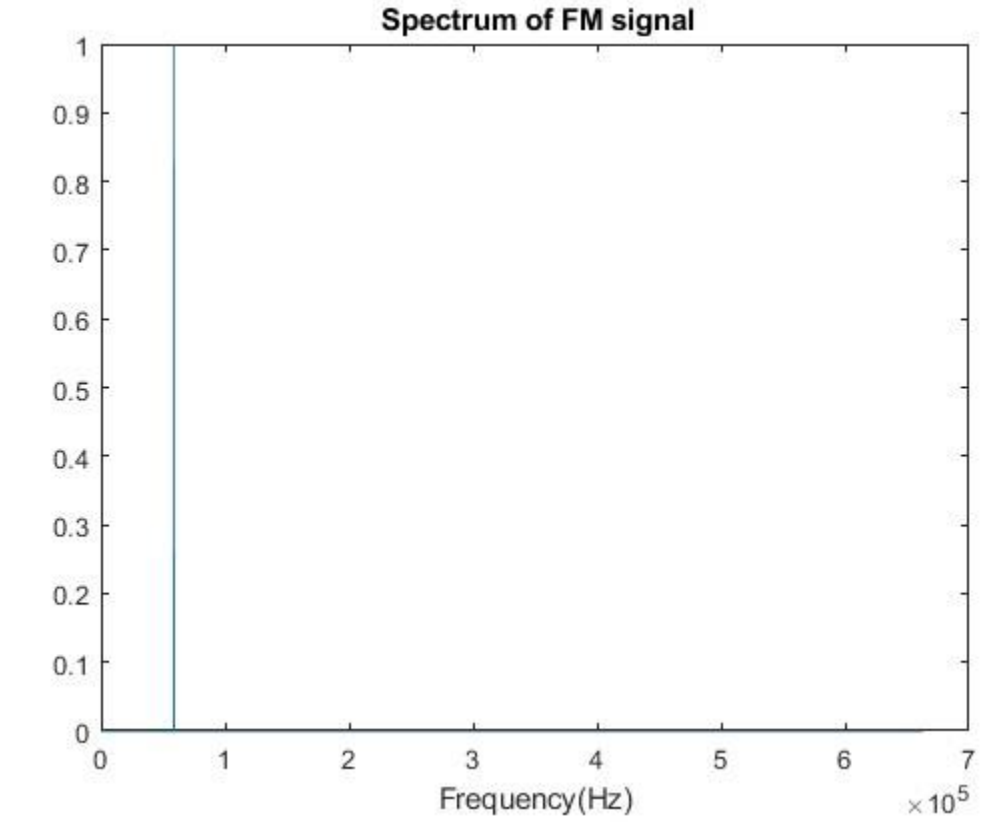
```
lengthOfSignal = length(m); normalizedSpectrumFm=
abs(spectrumFM/lengthOfSignal); frequencySpectrumFm=
normalizedSpectrumFm(1:lengthOfSignal/2+1);
frequencySpectrumFm(2:end-1) = 2*frequencySpectrumFm(2:end-1);
figure; plot(frequencySpectrumFm); title(' Spectrum of FM
signal'); xlabel('Frequency(Hz)');
```

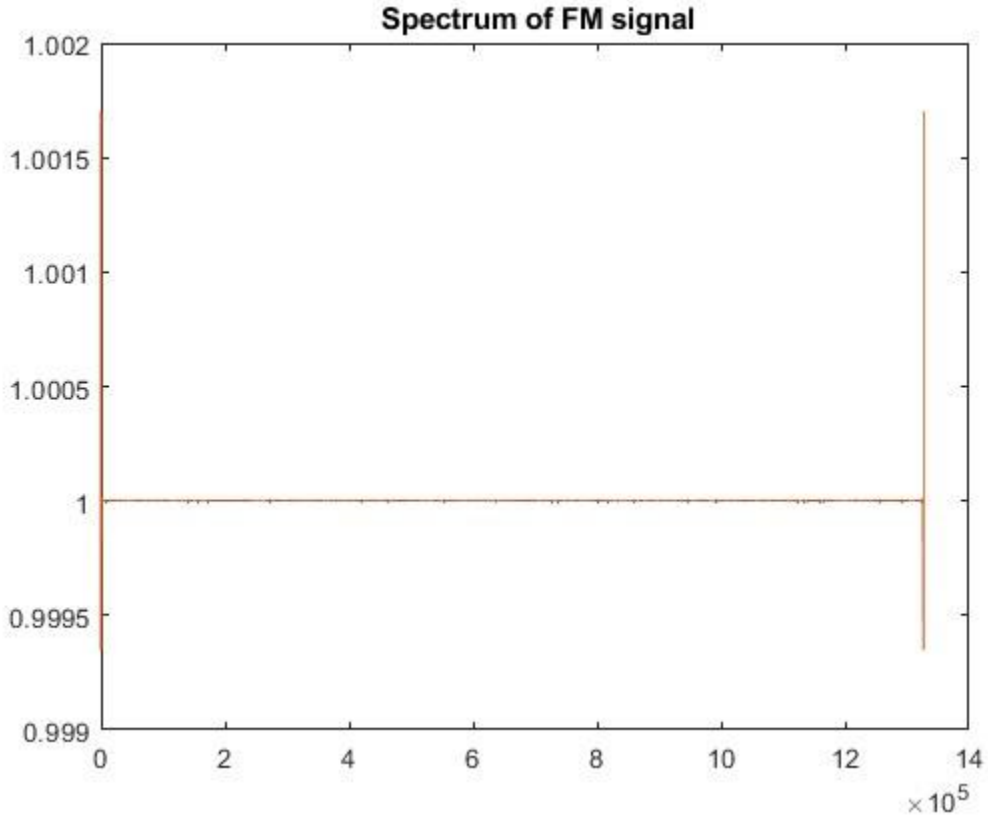




Envelope Detection based on Hilbert Transform and then FFT

```
envelope = abs (hilbert (FM_modulate_signal));
figure;  plot(envelope); title(' Spectrum of FM
signal');
```


Spectrum of FM signal
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
1
2
3
4
5
6
7
Frequency(Hz)
×10^5


Spectrum of FM signal
1.002
1.0015
1.001
1.0005
1
0.9995
0.999
0
2
4
6
8
10
12
14
×10^5

with noise

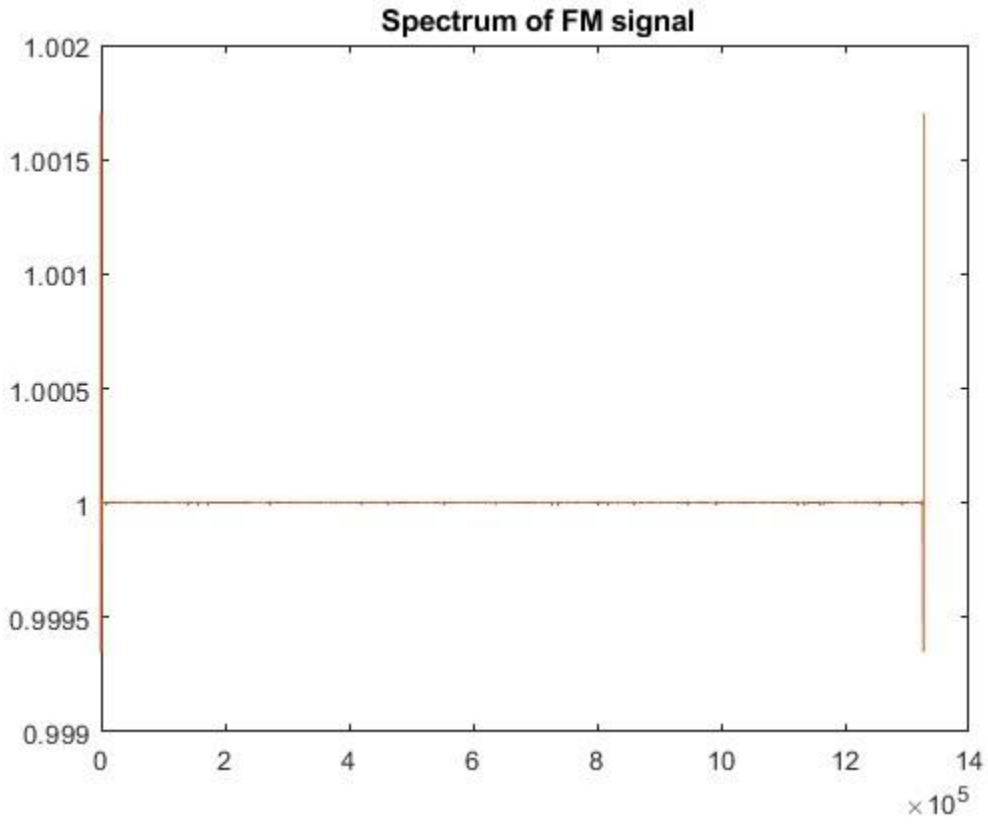
```
Noise_signal = awgn(m,10);
```

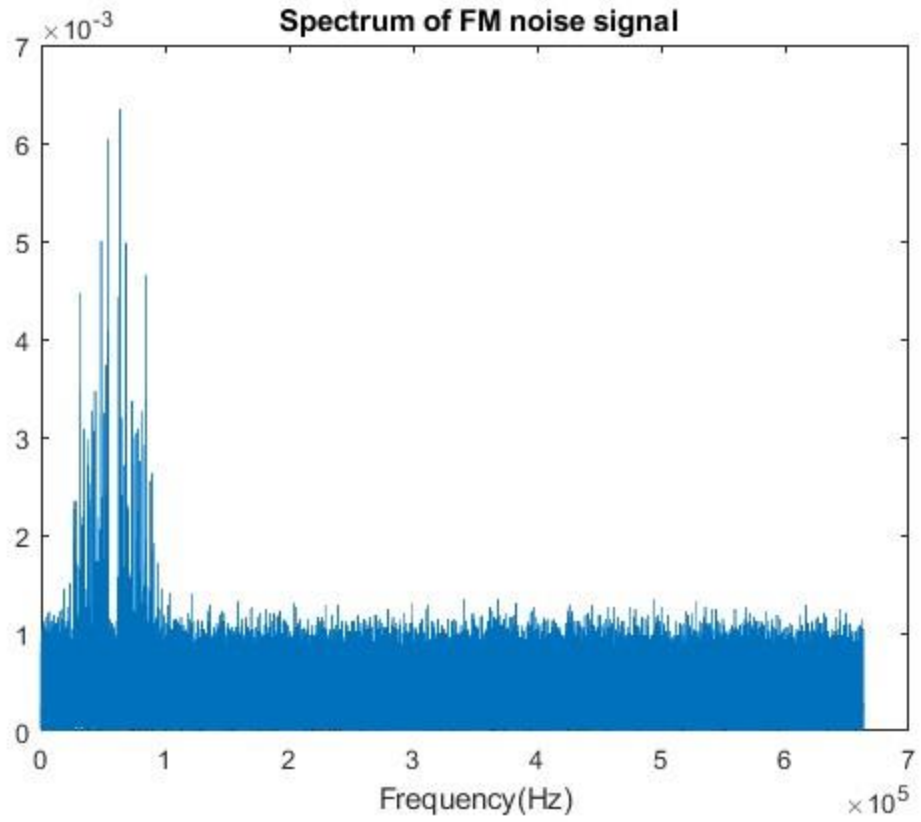
AM modulation of the recorded signal + Noise

```
FM_modulate_noise_signal = fmmod(Noise_signal,Fs,1000000,2);
```

Spectrum of modulated signal

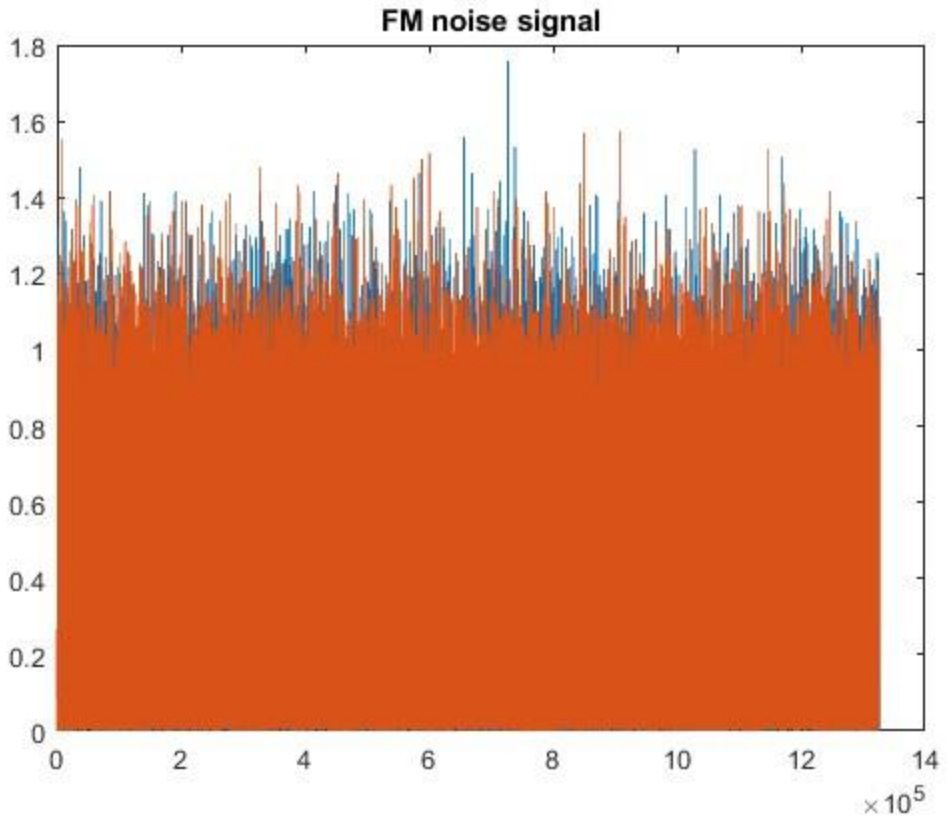
```
spectrumFM = fft(AM_modulate_noise_signal); lengthOfSignal =
length(m); normalizedSpectrumFm= abs(spectrumFM/lengthOfSignal);
frequencySpectrumFm= normalizedSpectrumFm(1:lengthOfSignal/2+1);
frequencySpectrumFm(2:end-1) = 2*frequencySpectrumFm(2:end-1);
figure; plot(frequencySpectrumFm); title(' Spectrum of FM noise
signal'); xlabel('Frequency(Hz)');
```





Envelope Detection based on Hilbert Transform and then FFT for noise signal

```
envelope = abs (hilbert (AM_modulate_noise_signal));
figure;  plot(envelope); title('FM noise signal');
```



Published with MATLAB® R2021a