Digital signal processing MATLAB exam - TRAN Gia Quoc Bao, ASI

Table of Contents

Default commands

```
clear all;
close all;
clc;
```

I. Context and variables to be analyzed

```
% Here we need to analyze the frequencies inside a signal to determine
% them (part II) and then to design suitable filters to extract those
% frequencies (part III). Finally there will be some conclusions on
 what we
% have done so far, in part IV.
```

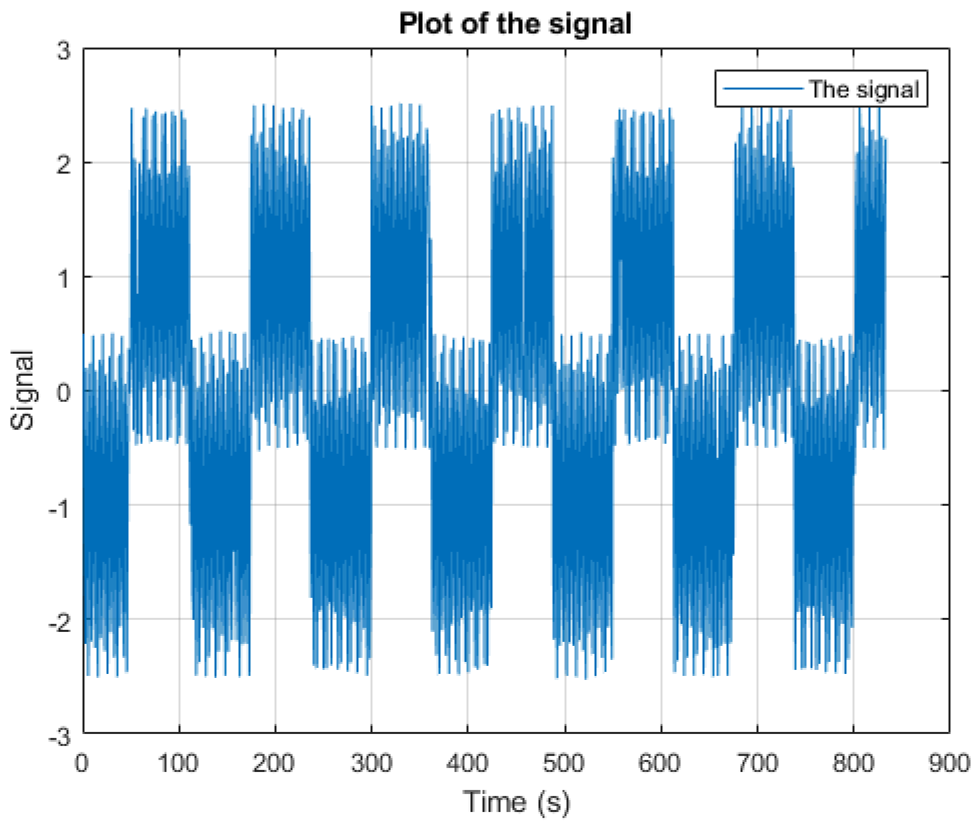
II. Spectral analysis

```
% The signal we want to analyze is called x, with a sampling frequency
 of fe.
% In this part we would like to spot the special, interesting
% frequencies in order to know what kinds of filters need to be made.

load('sujet2a.mat');
```

Plotting the signal

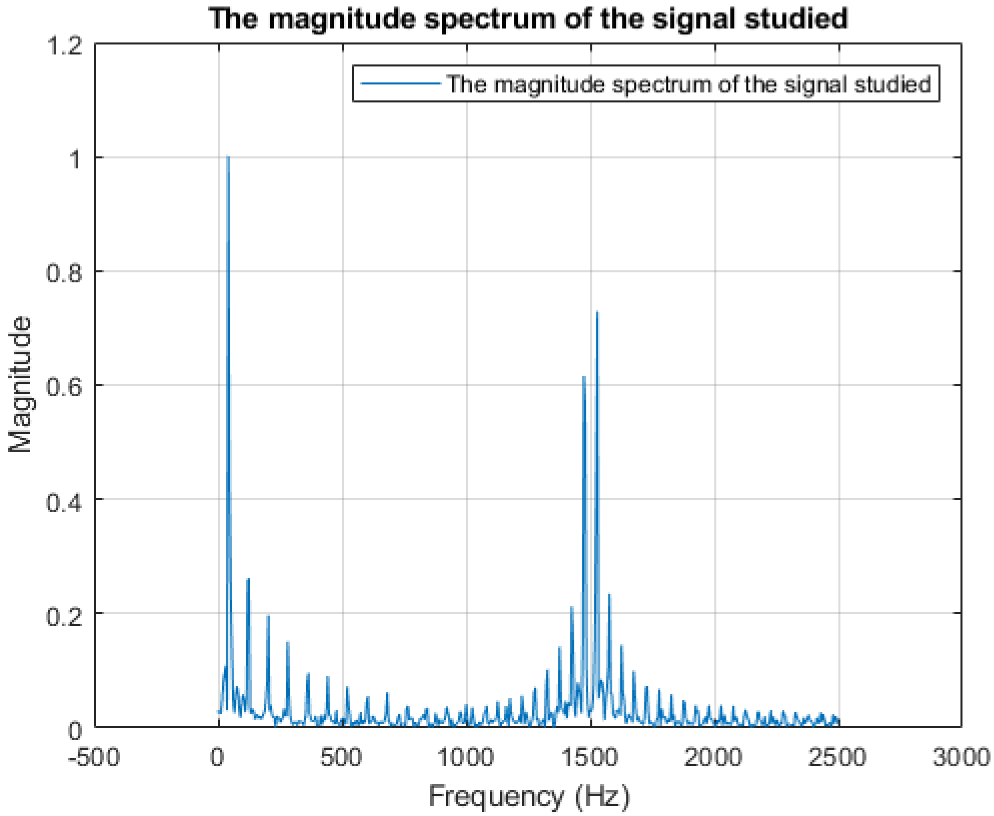
```
figure(1);
plot(x);
grid on;
```

```
xlabel('Time (s)');
ylabel('Signal');
legend('The signal');
title('Plot of the signal');

% This signal contains some frequencies that we need to analyzed,
 using fft
% and zero-padding.
```



Applying spectral analysis on the power consumption signal

```
figure(2);
[magnitude_signalSecond frequency_signalSecond] = Spectrum('signal
 studied', x, fe);

% Here in the analysis using the file "Spectrum" with contains zero-
padding
% and fft, I observed 1 peak at 41.5869 Hz and another one at
 1528.9307 Hz.
% So I will need a low-pass filter and a high-pass filter to separate
 them.
```



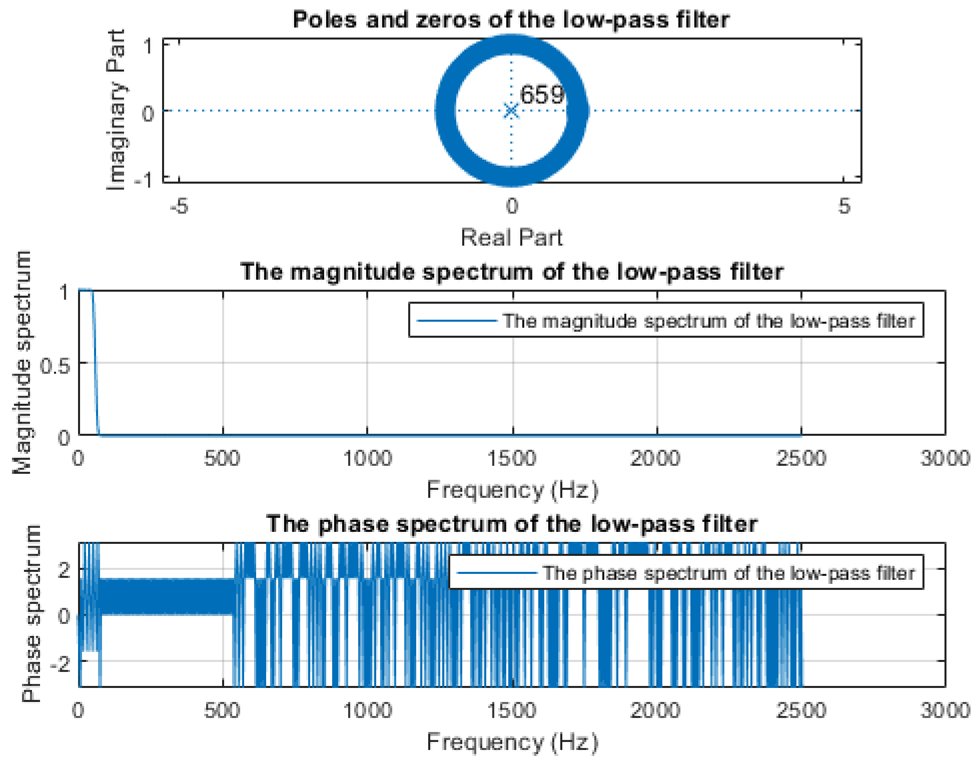
III. Digital FIR filters

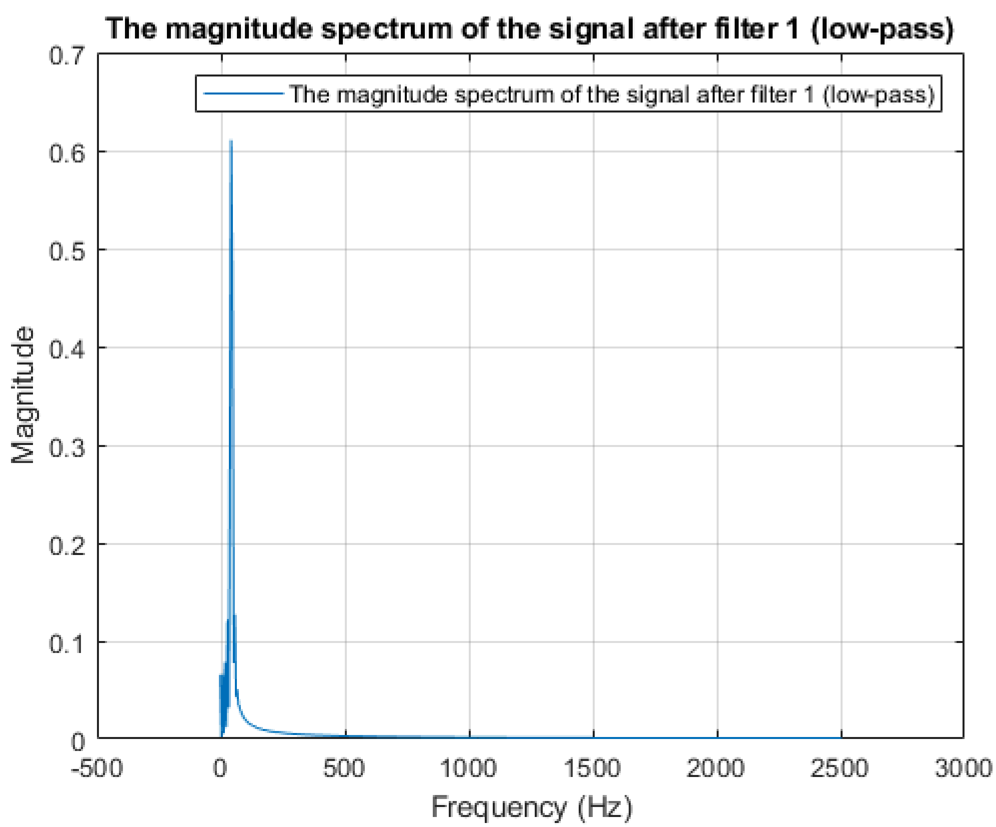
```
% In this part, we are trying to separate the components corresponding
% to each of the frequencies we determined.
% So we will first use a low-pass FIR filter for the f = 41.5869
% Hz, then a high-pass for the one with 1528.9307 Hz.
```

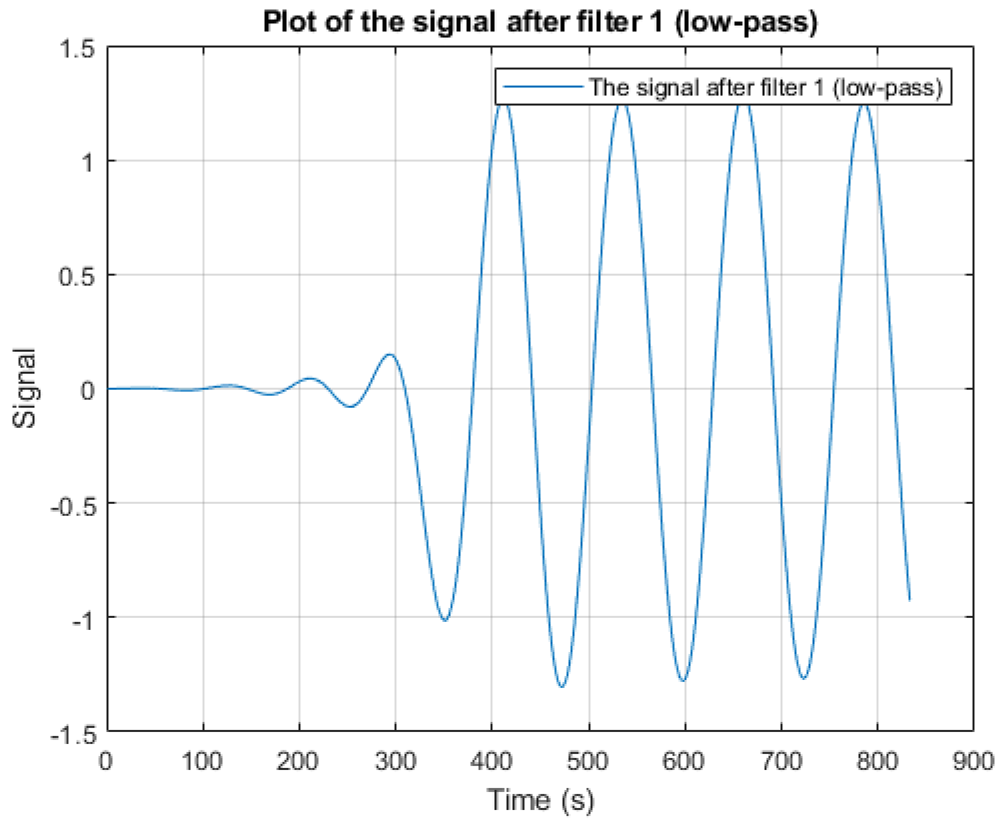
3.1 Low-pass filter

```
% The frequency we want is 41.5869 Hz. We will use a cut frequency
% at 60 as we know that this will include the one we need.

fc1 = 2*60/fe;

% Making the 1st window (Hamming)
deltaWindow1 = 0.005;
sizeWindow1 = round(3.3/deltaWindow1 - 1);

% I chose the delta to be 0.05 to greatly reduce to other unwanted
% frequencies while avoiding to much calculation for the computer.

% Making the window is optional. Normally with MATLAB we just need to
 use fir1 and
% then filter.
```

```
% Making the 1st filter
filter1 = fir1(sizeWindow1, fc1, 'low');

% Analysis of the 1st filter
figure(3);
[magnitude1 phase1 frequency1] = FilterVisu('low-pass filter',
 filter1, 1, sizeWindow1, fe);

% Looking at the magnitude spectrum we observe clearly that this is a
% low-pass filter. The gain for low frequencies is high and almost
 zero for
% the high frequencies. The cut is clear at the frequency we want:
 41.5869

% Applying the 1st filter on the signal
signalFiltered1 = filter(filter1, 1, x);

% Analysis of the obtained signal
figure(4);
[magnitude_signalFiltered1 frequency_signalFiltered1] =
 Spectrum('signal after filter 1 (low-pass)', signalFiltered1, fe);

% Looking at this we see that the high frequencies are filtered away
 and
% the small ones are much clearer. This shows the effectiveness of the
% low-pass filter we implemented.

figure(5);
plot(signalFiltered1);
grid on;
xlabel('Time (s)');
ylabel('Signal');
legend('The signal after filter 1 (low-pass)');
title('Plot of the signal after filter 1 (low-pass)');

% We observe the first pattern thanks to the filter we implemented.
 Later
% we will compare the results.
```


Poles and zeros of the low-pass filter
Imaginary Part
Real Part
659
The magnitude spectrum of the low-pass filter
Magnitude spectrum
Frequency (Hz)
The magnitude spectrum of the low-pass filter
The phase spectrum of the low-pass filter
Phase spectrum
Frequency (Hz)
The phase spectrum of the low-pass filter


The magnitude spectrum of the signal after filter 1 (low-pass)
The magnitude spectrum of the signal after filter 1 (low-pass)
Magnitude
Frequency (Hz)

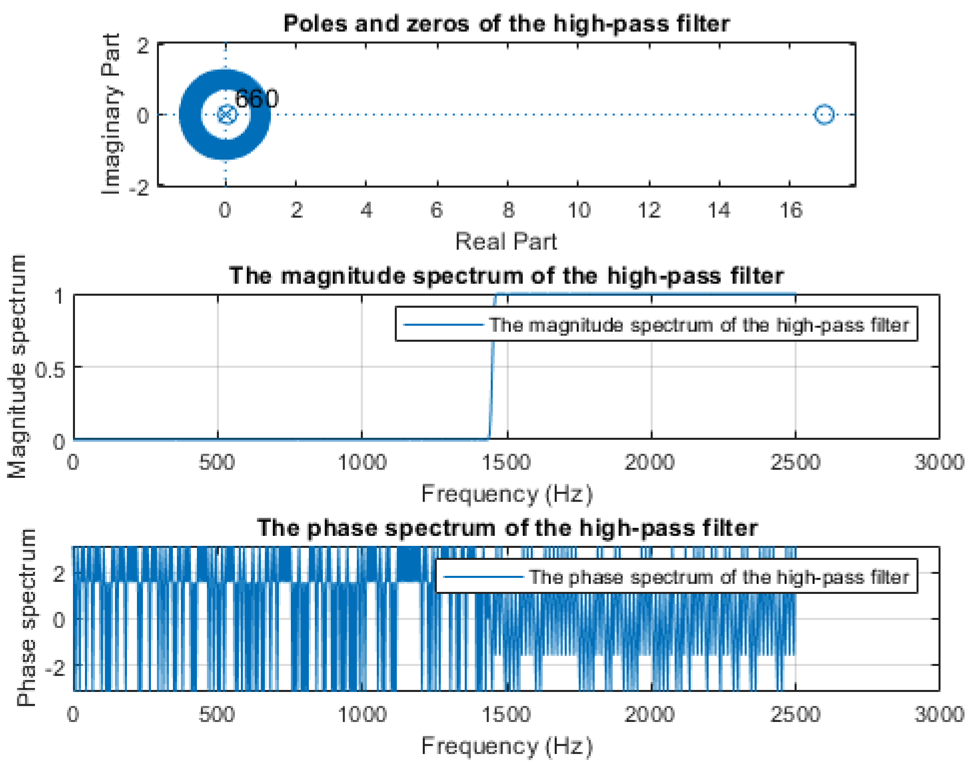


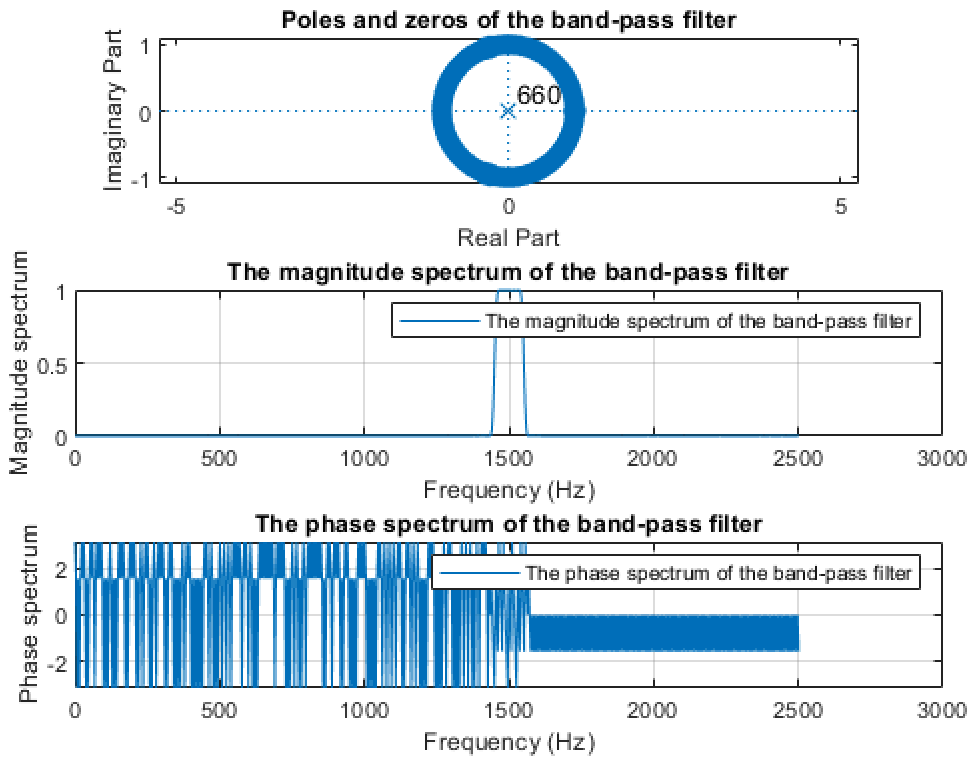
3.2 High-pass filter (and comparison with band-pass filter)

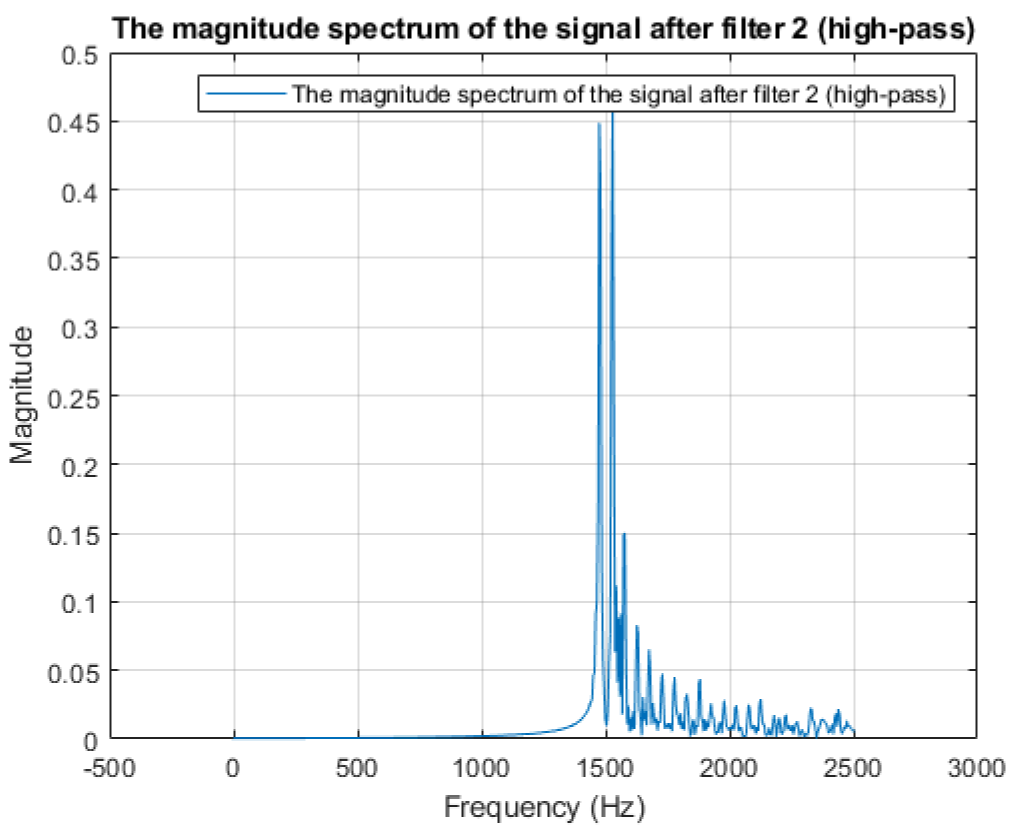
```
% The frequency we want is 1528.9307 Hz. We will use cut
% frequencies at 1450 Hz as we know that this will include the one we
 need.

fc2 = 2*1450/fe;

% Making the 2nd window (Hamming)
deltaWindow2 = 0.005;
sizeWindow2 = round(3.3/deltaWindow2 - 1) + 1;

% I chose the delta to be 0.05 to greatly reduce to other unwanted
% frequencies while avoiding to much calculation for the computer. Odd
% order symmetric FIR filters must have a gain of zero at the Nyquist
% frequency. So the order is being increased by one.

% Making the window is optional. Normally with MATLAB we just need to
 use fir1 and
% then filter.

% Making the 2nd filter
filter2 = fir1(sizeWindow2, fc2, 'high');
```

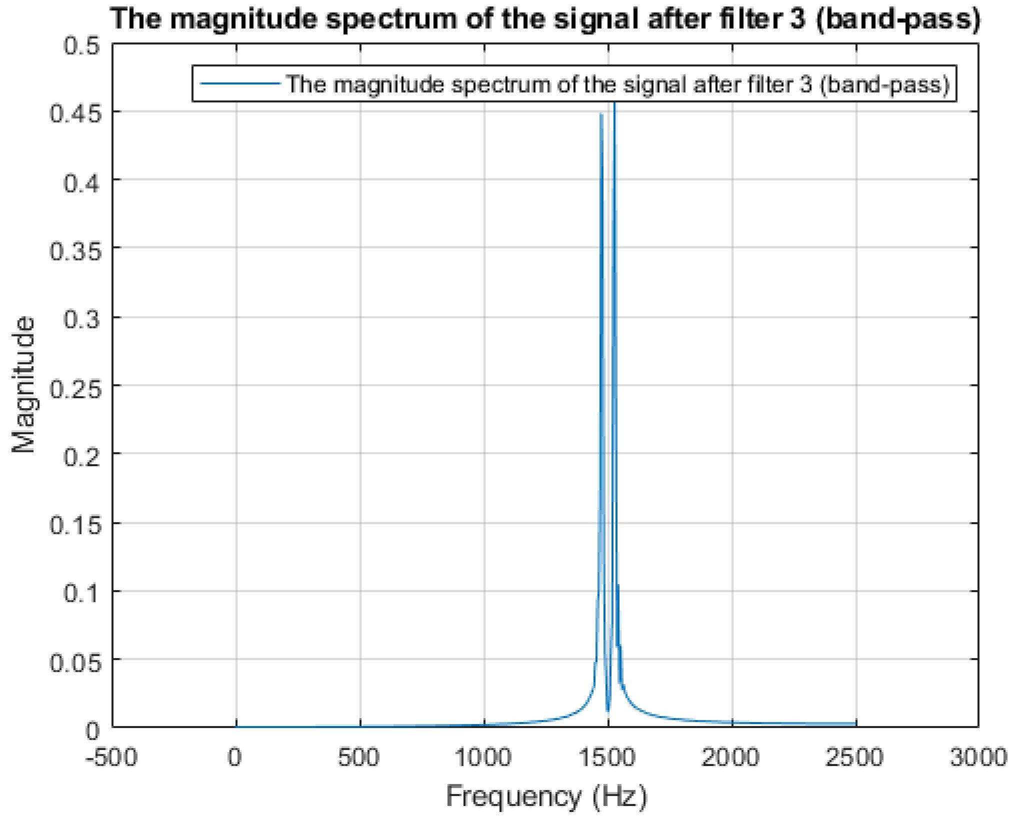
```
% I want to use a band-pass filter to compare with the high-pass one
filter3 = fir1(sizeWindow2, [fc2 fc2*1.07]);

% Analysis of the 2nd filter
figure(6);
[magnitude2 phase2 frequency2] = FilterVisu('high-pass filter',
 filter2, 1, sizeWindow2, fe);

% Looking at the magnitude spectrum we observe clearly that this is a
% high-pass filter. The gain for high frequencies is high and almost
 zero for
% the low frequencies. The cut is 1461 Hz which is good.

% Analysis of the 3rd filter
figure(7);
[magnitude3 phase3 frequency3] = FilterVisu('band-pass filter',
 filter3, 1, sizeWindow2, fe);

% Looking at the magnitude spectrum we observe clearly that this is a
% band-pass filter. The gain is only reserved for the frequencies in
 the
% band we choose. We cut at 1461 and 1540.

% Applying the 2nd filter and 3rd filter on the signal
signalFiltered2 = filter(filter2, 1, x);
signalFiltered3 = filter(filter3, 1, x);

% Analysis of the obtained signals
figure(8);
[magnitude_signalFiltered2 frequency_signalFiltered2] =
 Spectrum('signal after filter 2 (high-pass)', signalFiltered2, fe);

% By looking at this we see that the low frequencies have been greatly
% reduced by the filter. The high-frequency component is now quite
 clear to see.

figure(9);
[magnitude_signalFiltered3 frequency_signalFiltered3] =
 Spectrum('signal after filter 3 (band-pass)', signalFiltered3, fe);

% By looking at this we see that the frequencies outside the wanted
 band
% have been greatly reduced by the filter. The high-frequency
 component is
% now very clear to see, clearer than the high-pass filter.

figure(10);
plot(signalFiltered2);
grid on;
xlabel('Time (s)');
ylabel('Signal');
legend('The signal after filter 2 (high-pass)');
title('Plot of the signal after filter 2 (high-pass)');
```

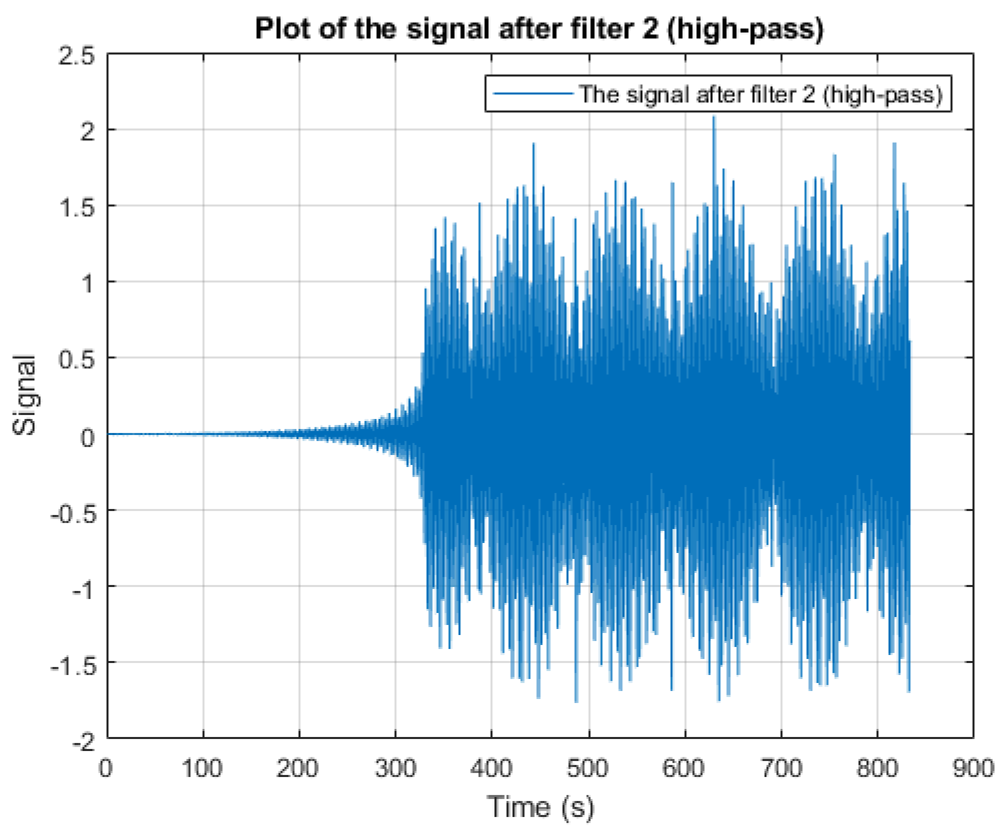
```
figure(11);
plot(signalFiltered3);
grid on;
xlabel('Time (s)');
ylabel('Signal');
legend('The signal after filter 3 (band-pass)');
title('Plot of the signal after filter 3 (band-pass)');

% We were able to extract the frequency. We see that the results
% obtained from the 2nd and 3rd filters are not very different as the
% component with the special frequency is extracted anyway. But of
 course
% the 3rd one gives clearer results as the very high frequencies are
% filtered away as well.
```

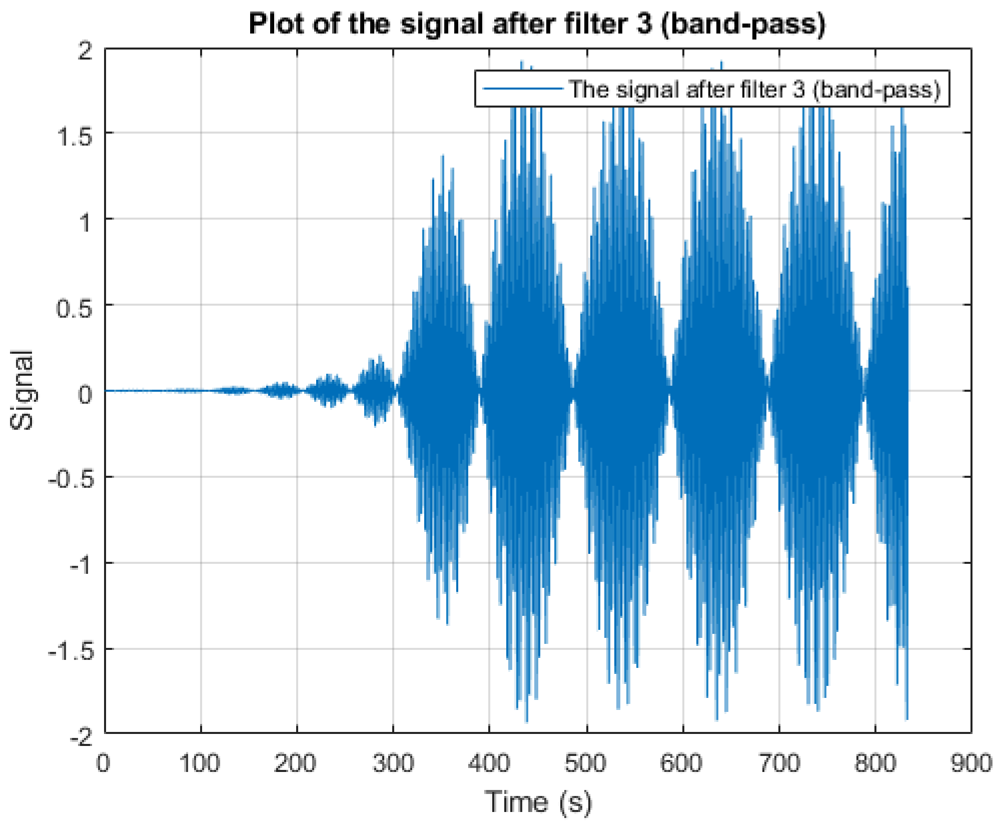



Poles and zeros of the band-pass filter
Imaginary Part
Real Part
660
The magnitude spectrum of the band-pass filter
Magnitude spectrum
Frequency (Hz)
The phase spectrum of the band-pass filter
Phase spectrum
Frequency (Hz)


The magnitude spectrum of the signal after filter 2 (high-pass)
Magnitude
Frequency (Hz)


The magnitude spectrum of the signal after filter 3 (band-pass)
The magnitude spectrum of the signal after filter 3 (band-pass)
Magnitude
Frequency (Hz)
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
-500
0
500
1000
1500
2000
2500
3000

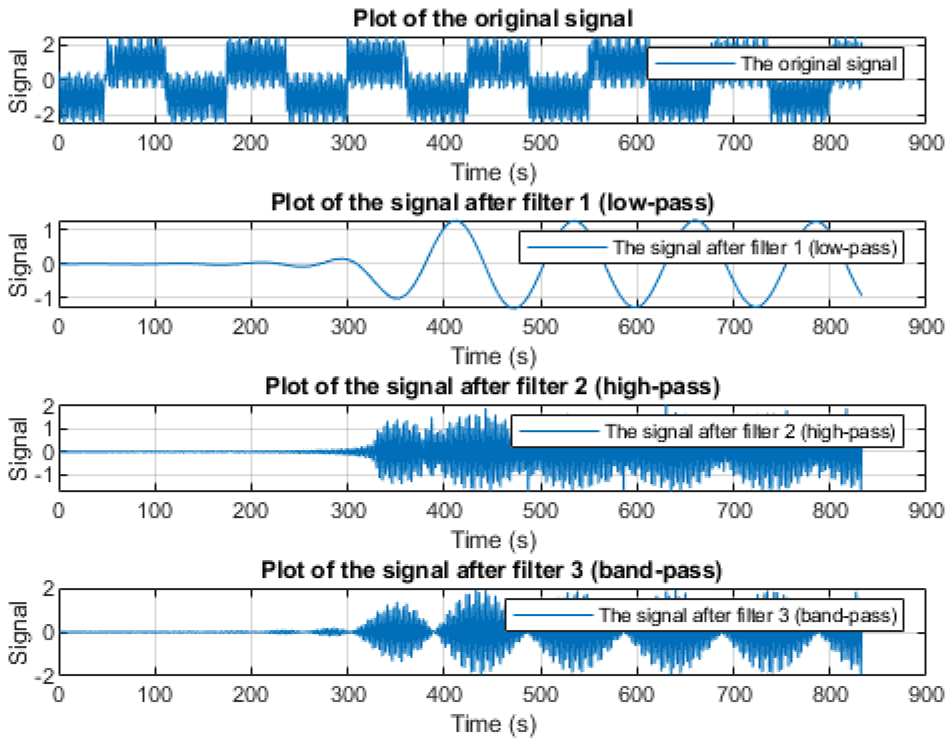

Plot of the signal after filter 2 (high-pass)
The signal after filter 2 (high-pass)
Signal
Time (s)
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
100
200
300
400
500
600
700
800
900



IV. Comparison and conclusion

```
figure(12);
subplot(411);
plot(x);
grid on;
xlabel('Time (s)');
ylabel('Signal');
legend('The original signal');
title('Plot of the original signal');
subplot(412);
plot(signalFiltered1);
grid on;
xlabel('Time (s)');
ylabel('Signal');
legend('The signal after filter 1 (low-pass)');
title('Plot of the signal after filter 1 (low-pass)');
subplot(413);
plot(signalFiltered2);
grid on;
xlabel('Time (s)');
ylabel('Signal');
legend('The signal after filter 2 (high-pass)');
title('Plot of the signal after filter 2 (high-pass)');
subplot(414);
```

```
plot(signalFiltered3);
grid on;
xlabel('Time (s)');
ylabel('Signal');
legend('The signal after filter 3 (band-pass)');
title('Plot of the signal after filter 3 (band-pass)');

% In comparison we see that the signals after being filtered are very
% different from the original one and each has its own frequency. Also
% the filtered signals are delayed by half the order of the filter so
 we could
% only plot them from there. We see that for the 3rd filter the signal
 is
% clearer than the 2nd one, thanks to the very high frequencies being
% filtered away.

% What we have done so far is first to analyze the signal in order to
% detect the special frequencies thanks to the file "Spectrum". Then,
 with
% the information obtained, we designed different suitable filters,
 studied
% the filters with the file "FilterVisu" (plotting poles and zeros,
% frequency response), and applied them on the signal to process it.
% This way we can see how each of the components varies.
% Through this we learned how to study a signal effectively using this
% 2-step procedure and reviewed the tools we have had in the course.

% Some practical lessons from the exercise: we need to normalise the
 fft,
% which we need zero padding to improve the visual aspect.
```



Published with MATLAB® R2019a