Mechatronic Systems Identification

Lab 3 - Matched Filter

Khaldoun Fayad - 409597

Witold Surdej - 407100

30.03.2024

Description

The aim of this laboratory is for us to learn the basics of signal filtration and its different utilization methods.

Task 1

This task aims to explore the cross-correlation of signals in Matlab and how to best discover the implications of different setups of correlated signals.

Task 1.1

In this task, a digital signal was created and then modified by adding a chirp signal, which changes in frequency over time, to a specific segment of it. This chirp signal was tailored to smoothly transition across its duration by applying a Hanning window, a technique used to minimize abrupt changes and spectral leakage, making the signal more coherent for analysis. Following this, the modified signal was analyzed through two primary methods: correlation and convolution.
The correlation process involved comparing the modified signal to the original chirp to identify where and how closely the chirp appears within the larger signal. This technique is crucial for detecting specific patterns or signals within a more complex or noisy environment. Conversely, convolution was explored as a means of blending two signals, highlighting its role in filtering and system analysis, which differs fundamentally from correlation's focus on similarity and pattern detection.

```
%% Task 1
fs = 2000; % sampling frequency
N = 2000; % number of samples
s = 2000;
y = zeros(s,1);
tp = (0:N-1)/fs; % time range
f1 = 10; %frequency one
f2 = 70; %frequency two
yp = chirp(tp,f1,tp(end),f2);
% Creating the Hann windowed
signal
y2 = yp.*hann(N)';
figure()
plot(tp,y2)
xlabel('time (s)')
ylabel('amplitude (-)')
```

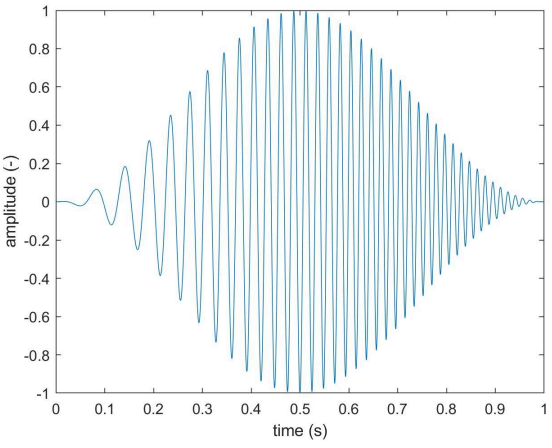

fig 1: shows the original signal

```
%Ex. 1.1
N = 200; % number of samples
s = 2000;
tp2 = (0:N-1)/fs; % time range
f1 = 10; %frequency one
f2 = 300; %frequency two
y = zeros(s,1);
yp = chirp(tp2,f1,tp2(end),f2);
% Creating the Hann windowed signal
y2 = yp.*hann(N)';
y(500:500+length(y2)-1) = y2;
[yc,lag] = xcorr(y,y2);
tc = lag/fs;
figure()
plot(tp,y)
hold on
plot(tc,yc)
xlabel('time (s)')
ylabel('amplitude (-)')
hold off
legend('Original Signal','Signal on the 500th sample')
```

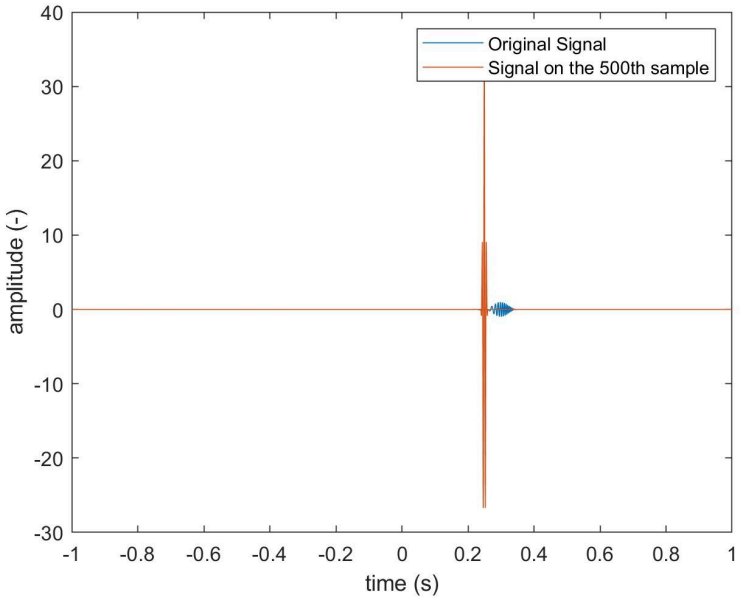

fig 2: shows the original signal compared to the newly created signal on the 500th sample

```
%Normalizing
n = normalize(y);
n2 = normalize(yc);
figure()
plot(tp,n)
hold on
plot(tc,n2)
xlabel('time (s)')
ylabel('amplitude (-)')
hold off
legend('Normalized Original Signal','Normalized Signal on the 500th sample')
```

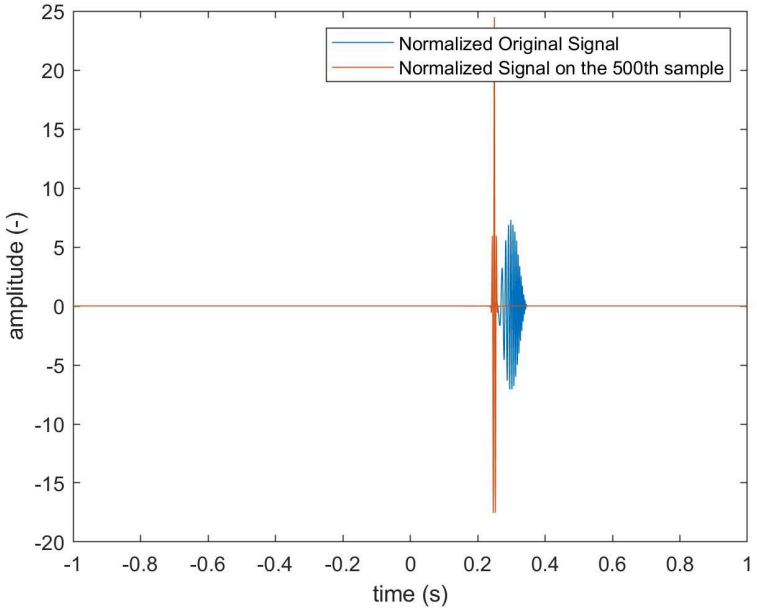

fig3: shows the original signal compared to the newly created signal on the 500th sample both normalized

```
%Trying with convolution
N = 200; % number of samples
s = 200;
tp2 = (0:N-1)/fs; % time range
f1 = 10; %frequency one
f2 = 300; %frequency two
s = 2000;
y = zeros(s,1);
yp = chirp(tp2,f1,tp2(end),f2);
% Creating the Hann windowed signal
y2 = yp.*hann(N)';
y(500:500+length(y2)-1) = y2;
h = conv(y,y2);
t_conv = ((0:length(h)-1)/fs);
n3 = normalize(h);
figure()
plot(tp,n)
hold on
plot(t_conv,n3)
xlabel('time (s)')
ylabel('amplitude (-)')
hold off
legend('Normalized Original signal','Convoluted normalized signal')
```

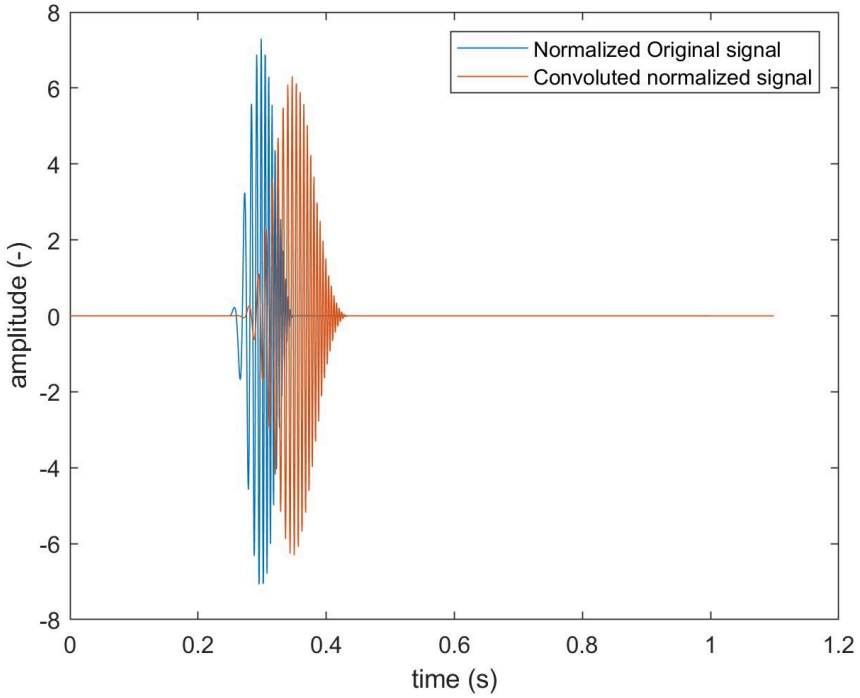

fig4: shows the original signal compared to the newly created signal on the 500th sample both normalized

Conclusions: By comparing the results of correlation and convolution operations on the same set of signals, the exercise highlights the fundamental differences between these two techniques. While correlation measures the similarity between two signals as one is slid over the other, convolution mathematically combines two signals. This distinction is crucial for understanding their respective applications: correlation is often used for signal detection and analysis, whereas convolution is key to signal filtering and system analysis.

The application of the cross-correlation function *xcorr()* to detect the presence and position of the yp signal within y showcases a common use case in signal processing for identifying the time delay (lag) between two related signals. This is especially relevant in areas such as radar signal processing, communications, and audio analysis, where determining the time difference of arrival or matching patterns within signals is necessary.

Task 1.2

The task involved calculating the correlation of signals generated with varying parameters and examining the effects of these variations on the correlation results, particularly focusing on the relationship between the time base length of the correlation outcome and the frequency band of the correlated signal.

```
%Ex. 1.2
fs = 2000; % sampling frequency
%f2=100
N = 200; % number of samples
s = 2000;
tp2 = (0:N-1)/fs; % time range
f1 = 30; %frequency one
f2 = 100; %frequency two
y = zeros(s,1);
yp = chirp(tp2,f1,tp2(end),f2);
% Creating the Hann windowed signal
y2 = yp.*hann(N)';
y(500:500+length(y2)-1) = y2;
[yc,lag] = xcorr(y,y2);
tc = lag/fs;
%Normalizing
n = normalize(y);
n2 = normalize(yc);
figure()
plot(tp,n)
hold on
plot(tc,n2)
xlabel('time (s)')
ylabel('amplitude (-)')
hold off
legend('Normalized Original Signal','Normalized Signal on the 500th
sample')
```

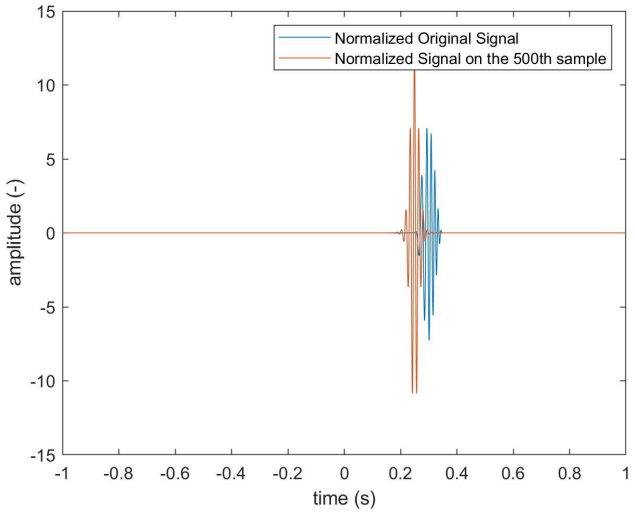

fig 5: shows the plot of the signal with f2 = 100 Hz

```
%f2=200
```

```
N = 200; % number of samples
s = 2000;
tp2 = (0:N-1)/fs; % time range
f1 = 30; %frequency one
f2 = 200; %frequency two
y = zeros(s,1);
yp = chirp(tp2,f1,tp2(end),f2);
% Creating the Hann windowed signal
y2 = yp.*hann(N)';
y(500:500+length(y2)-1) = y2;
[yc,lag] = xcorr(y,y2);
tc = lag/fs;
%Normalizing
n = normalize(y);
n2 = normalize(yc);
figure()
plot(tp,n)
hold on
plot(tc,n2)
xlabel('time (s)')
ylabel('amplitude (-)')
hold off
legend('Normalized Original Signal','Normalized Signal on the 500th sample')
```

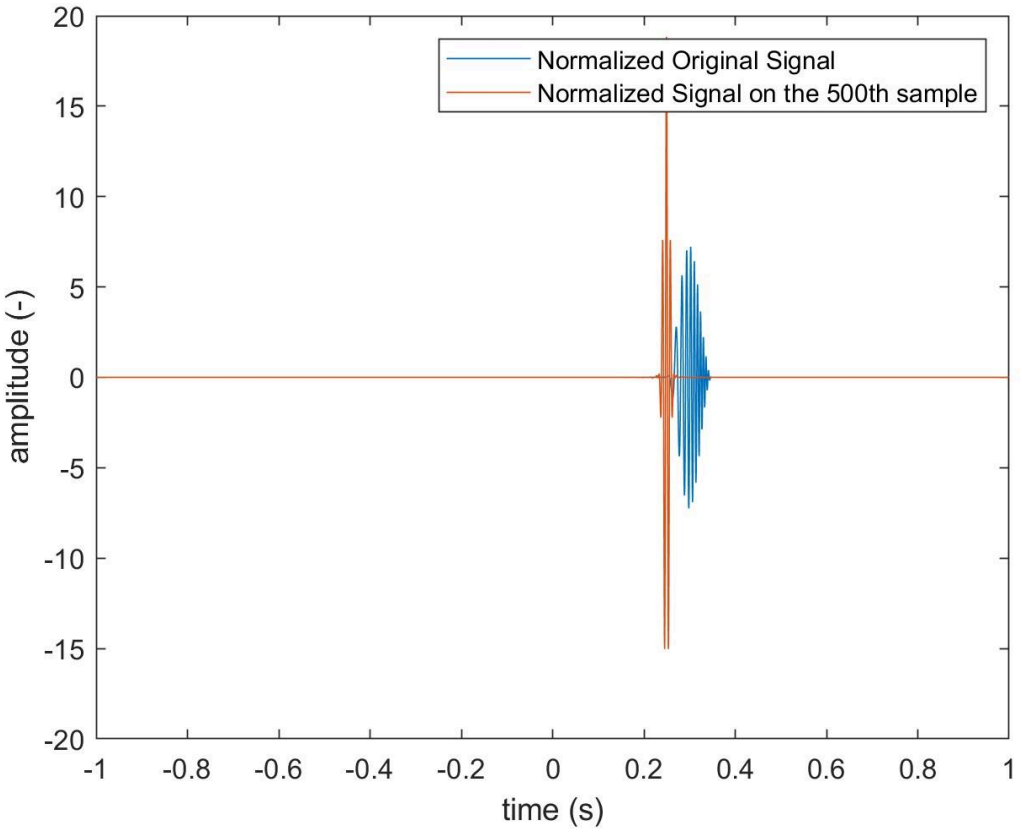

fig 6: shows the plot of the signal with f2 = 200 Hz

```
%f2=500
```

```
N = 200; % number of samples
s = 2000;
tp2 = (0:N-1)/fs; % time range
f1 = 30; %frequency one
f2 = 500; %frequency two
y = zeros(s,1);
yp = chirp(tp2,f1,tp2(end),f2);
% Creating the Hann windowed signal
y2 = yp.*hann(N)';
y(500:500+length(y2)-1) = y2;
[yc,lag] = xcorr(y,y2);
tc = lag/fs;
%Normalizing
n = normalize(y);
n2 = normalize(yc);
figure()
plot(tp,n)
hold on
plot(tc,n2)
xlabel('time (s)')
ylabel('amplitude (-)')
hold off
legend('Normalized Original Signal','Normalized Signal on the 500th sample')
```

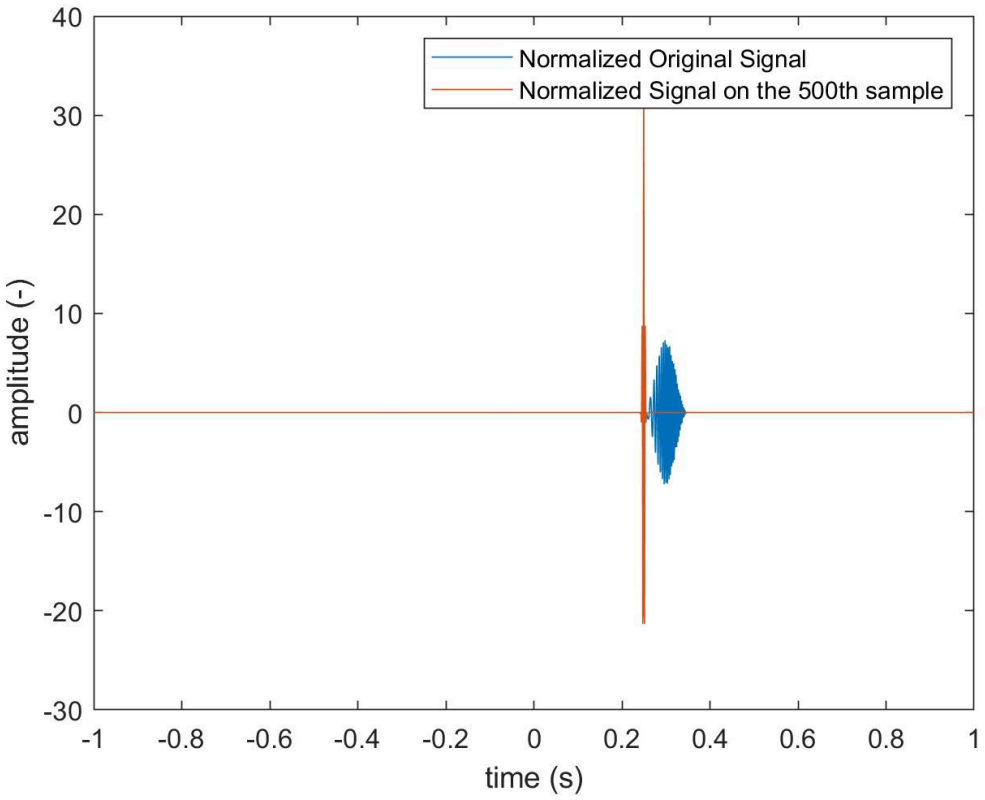

fig 7: shows the plot of the signal with f2 = 500 Hz

```
%f1=500 and f2=500
```

```
N = 200; % number of samples
s = 2000;
tp2 = (0:N-1)/fs; % time range
f1 = 500; %frequency one
f2 = 500; %frequency two
y = zeros(s,1);
yp = chirp(tp2,f1,tp2(end),f2);
% Creating the Hann windowed signal
y2 = yp.*hann(N)';
y(500:500+length(y2)-1) = y2;
[yc,lag] = xcorr(y,y2);
tc = lag/fs;
%Normalizing
n = normalize(y);
n2 = normalize(yc);
figure()
plot(tp,n)
hold on
plot(tc,n2)
xlabel('time (s)')
ylabel('amplitude (-)')
hold off
legend('Normalized Original Signal','Normalized Signal on the 500th sample')
```

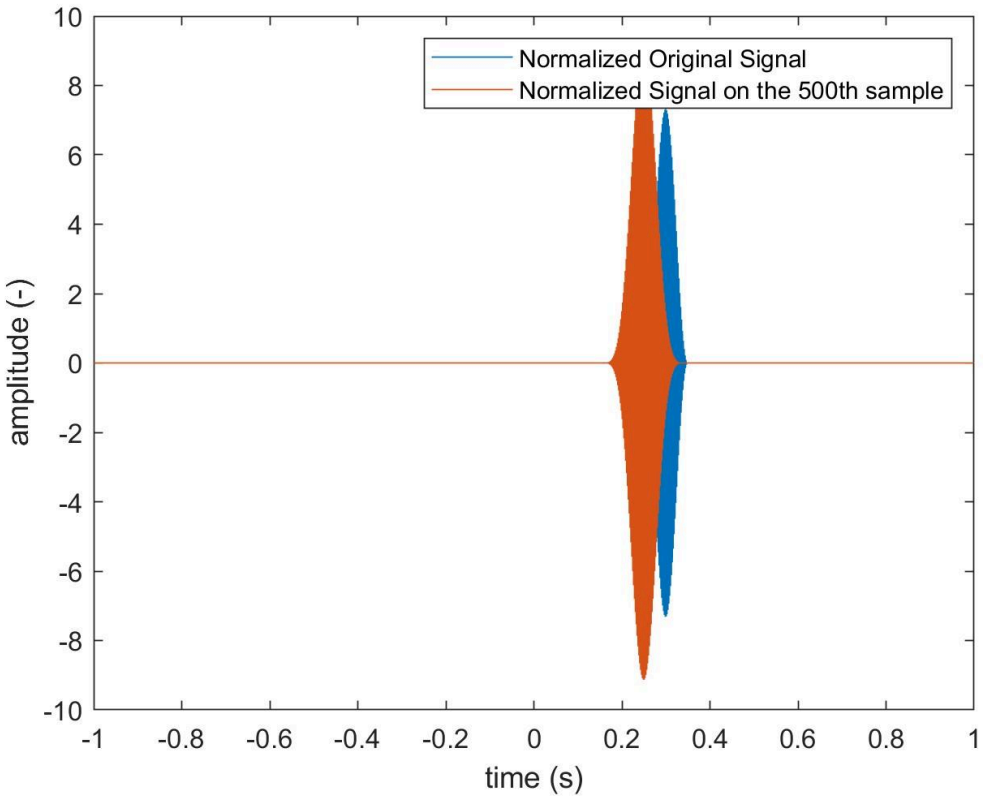

fig 8: shows the plot of the signal with f1 = 500 Hz and f2 = 500 Hz

Conclusions: The analysis revealed that the width of the correlation peak varies with the frequency range of the chirp signal. Specifically, a wider frequency range (difference between f1 and f2) tends to result in a narrower correlation peak. This is because a chirp signal with a wider frequency range changes its frequency more rapidly over its duration, leading to a more compact autocorrelation function. Conversely, a chirp signal with a smaller frequency range (including the case where f1 equals f2) tends to have a broader correlation peak, as the signal changes less over time. The exercise also underscores how the frequency content of a signal can affect its detectability and the preciseness of its localization in time. High-frequency signals or signals that sweep through a wide range of frequencies can be pinpointed more precisely in the time domain, as evidenced by the sharper correlation peaks. This principle is pivotal in applications like radar and communications, where identifying the exact timing of a signal's arrival or its duration is crucial.

Task 1.3

The instructions for the task describe a procedure for embedding a chirp signal (yp) into a longer signal (y) at two different instances and amplitudes, followed by a correlation analysis between the composite signal and the original chirp signal. Drawing conclusions from such an experiment, we can highlight several key insights relevant to signal processing, matched filtering, and the practical applications of correlation.

```
%Ex. 1.3
fs = 2000; % sampling frequency
N = 1000; % number of samples
s = 2000;
tp3 = (0:N-1)/fs; % time range
f1 = 50; %frequency one
f2 = 500; %frequency two
y = zeros(s,1);
yp = chirp(tp3,f1,tp3(end),f2);
% Creating the Hann windowed signal
y2 = yp.*hann(N)';
y(500:500+length(y2)-1) = y2;
[yc,lag] = xcorr(y,y2);
tc = lag/fs;
%Normalizing
n = normalize(y);
n2 = normalize(yc);
% Generating a new yp for the second addition
y_half = zeros(s,1);
yp_half = 0.5 * chirp(tp3,f1,tp3(end),f2); % Half the amplitude
y2_half = yp_half.*hann(N)';
% Second addition at 750th sample with amplitude 0.5
y_half(750:750+length(y2_half)-1) = y2_half;
[yc2,lag2] = xcorr(y_half,y2_half);
tc2 = lag/fs;
%Normalizing
n_half = normalize(y_half);
```

```
n2_half = normalize(yc2);
figure()
plot(tp,n)
hold on
plot(tc,n2)
plot(tp,n_half)
plot(tc2,n2_half)
xlabel('time (s)')
ylabel('amplitude (-)')
xlim([0 1])
hold off
legend('Normalized Original Signal','Normalized Signal on the 500th sample','Normalized Halfed Signal','Normalized Halfed Signal on the 750th sample')
```

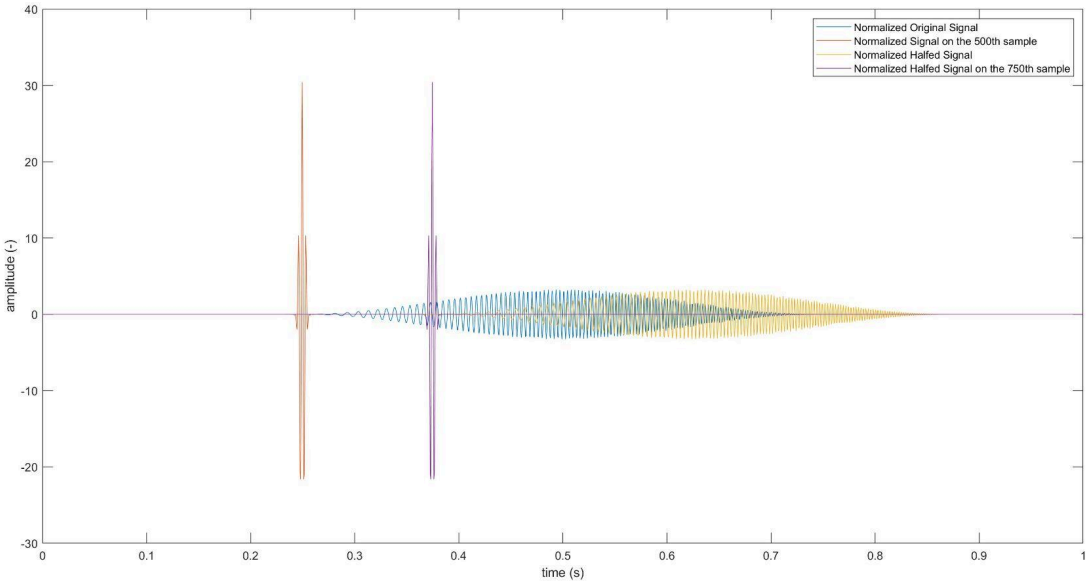

fig 9: shows the plots of the signals when different pulses were introduced.

Conclusions: The experiment shows that the amplitude of the signal being correlated (in this case, the amplitude of yp when it is added to y) directly influences the amplitude of the peaks in the correlation output. A higher amplitude of the embedded signal results in a higher peak, which can be crucial for applications that require estimating the strength or power of the signal of interest. The locations of the pulses in the correlation output closely relate to the locations where the yp signals were embedded in y. This relationship highlights how correlation can map the presence of a known signal within a larger signal back to its original time domain, an essential aspect of matched filtering used for signal detection and estimation.

The findings emphasize the practical implications of embedding and detecting signals of varying amplitudes and demonstrate the utility of correlation in signal processing applications. Understanding how signal characteristics like amplitude and frequency affect correlation outcomes is vital for designing effective detection and analysis algorithms

Task 1.4

The task involves adding white noise of varying amplitudes to a signal containing embedded chirp signals and then analyzing the effects of this noise on the detection of the chirp signals through correlation. From such an experiment, several insightful conclusions can be drawn, emphasizing the interaction between signal processing techniques and the impact of noise on signal detection.

```
%Ex. 1.4
fs = 2000; % Sampling frequency
amp_values = [0.5, 1, 2]; % Different amplitudes for noise
% Original signal creation as before
N = 1000; % Number of samples for yp
y = zeros(s,1); % Reinitialize y for clarity in demonstration
yp = chirp((0:N-1)/fs,f1,(N-1)/fs,f2); % Generating yp
y2 = yp.*hann(N)'; % Applying Hann window
y(500:500+N-1) = y2; % Inserting first signal
y(750:750+N-1) = y2 * 0.5; % Inserting second signal at half amplitude
for amp = amp_values
   % Add white noise to the y signal
   szum = amp * randn(length(y),1); % Generating noise
   y_noisy = y + szum; % Adding noise to the signal
   % Correlate the noisy y signal with the original yp signal
   [yc, lag] = xcorr(y_noisy, yp);
   tc = lag/fs;
   % Plotting
   figure;
   subplot(2,1,1);
   plot((0:length(y_noisy)-1)/fs, y_noisy);
   title(['Noisy Signal with amp = ', num2str(amp)]);
   xlabel('Time (s)');
   ylabel('Amplitude');
   subplot(2,1,2);
   plot(tc, normalize(yc, 'norm'));
   title(['Correlation Output with amp = ', num2str(amp)]);
   xlabel('Time (s)');
   ylabel('Normalized Amplitude');
end
```

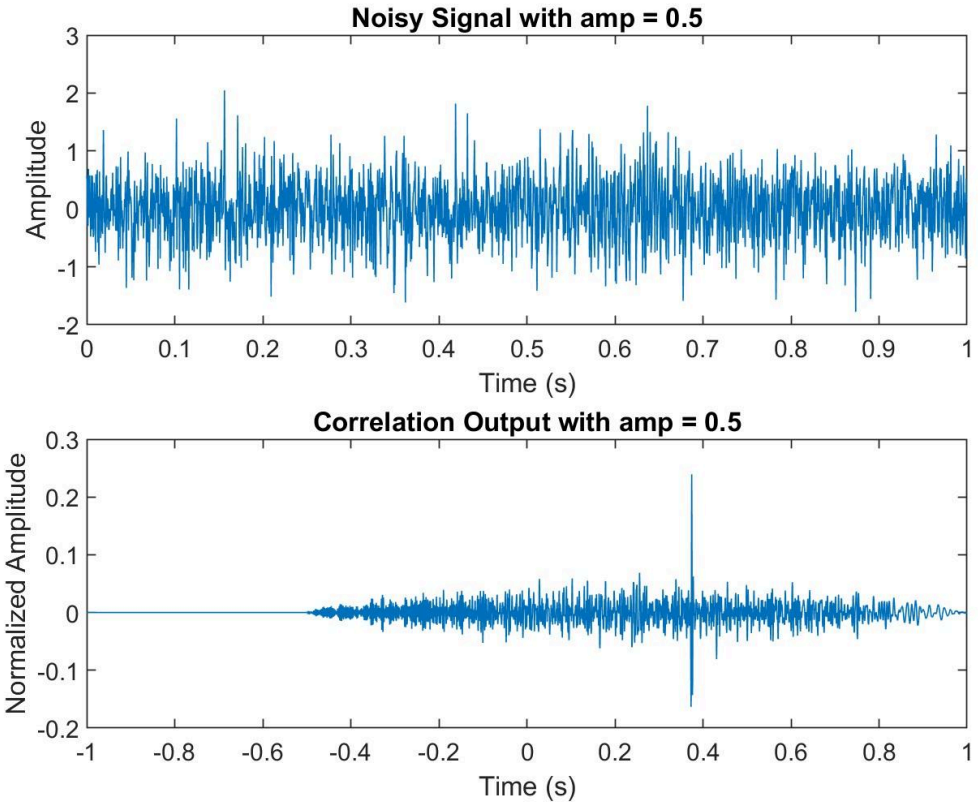

fig 10: shows the plots of the signals with 0.5 amplitude noise introduced.

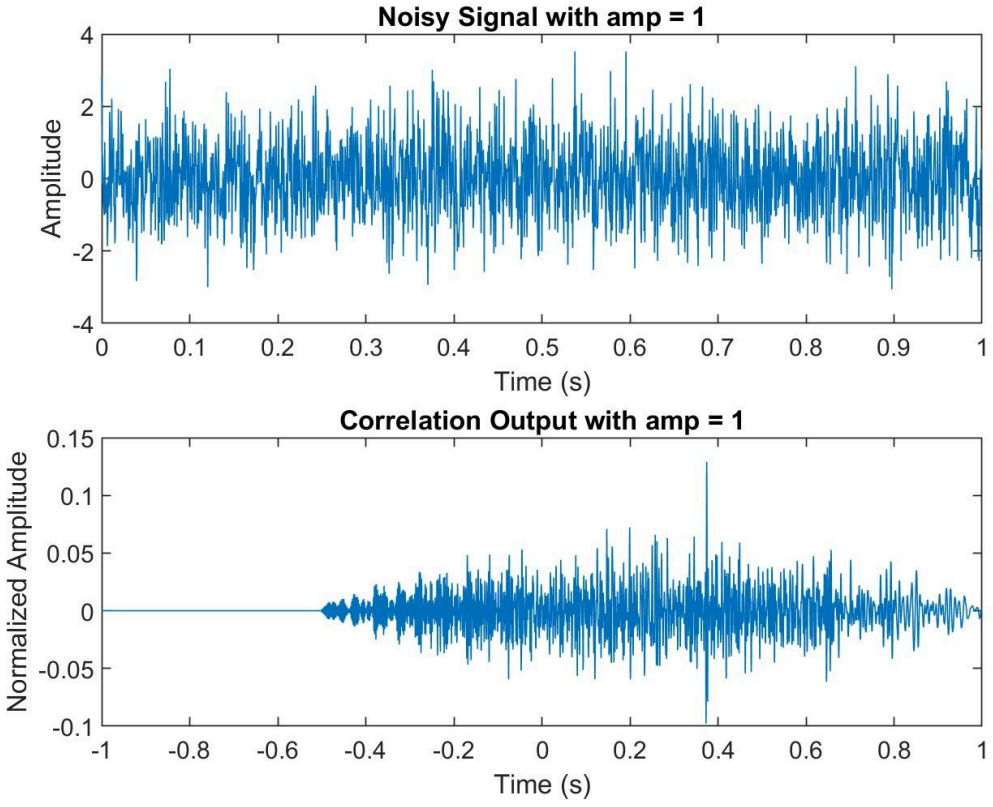

fig 11: shows the plots of the signals with 1 amplitude noise introduced.

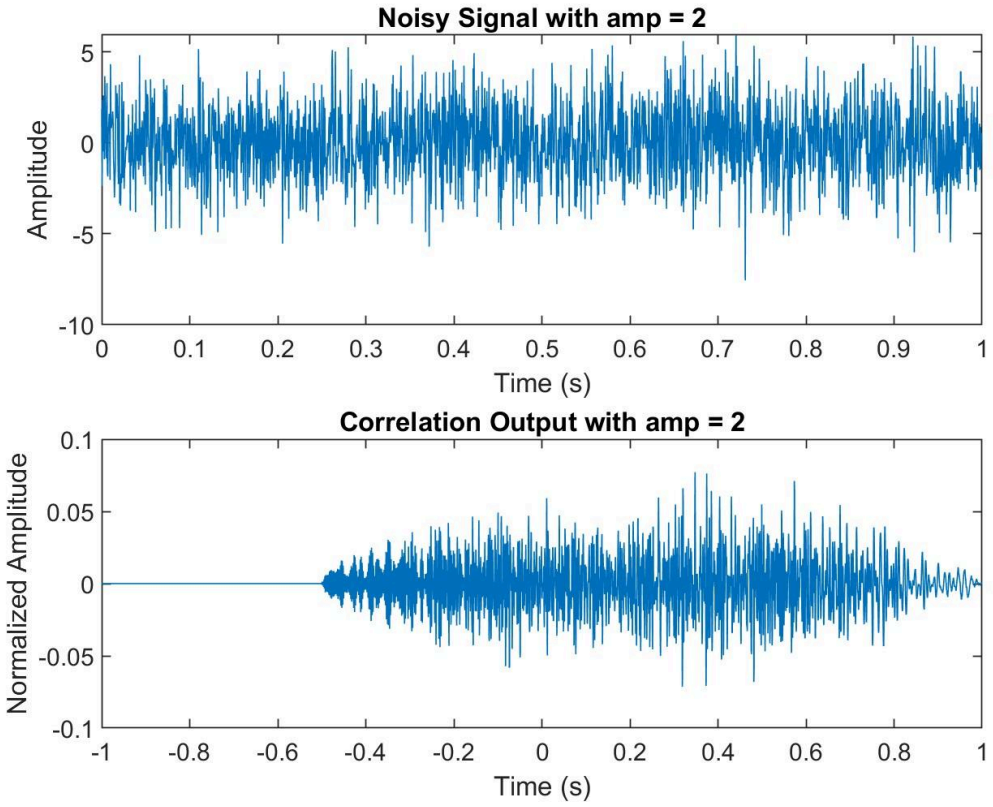

fig 12: shows the plots of the signals with 2 amplitude noise introduced.

Conclusions: The results show that with lower noise levels (amp = 0.5), the correlation peaks corresponding to the embedded chirp signals remain relatively distinct, allowing for the identification of the signal's presence and timing. However, as the noise amplitude increases to 1 and then to 2, these peaks become less pronounced and may eventually be obscured by the noise, highlighting the inverse relationship between noise amplitude and the detectability of signal features through correlation. The task underscores the importance of the signal-to-noise ratio (SNR) in signal processing applications. The visibility of the correlation peaks against the background noise effectively represents the SNR, with higher noise levels leading to lower SNR values. This experiment serves as a practical demonstration of how SNR affects the ability to extract meaningful information from a signal, a critical consideration in communications, radar systems, and audio processing. The findings have practical implications for the design and implementation of noise management strategies in signal processing systems. The ability to detect and analyze signals under various noise conditions is essential for optimizing system performance, necessitating techniques such as noise reduction, filtering, and the use of more robust correlation methods or matched filtering to improve detection capabilities.

Task 2

This task aims at showing us the practical applications of matched filtering on a real life signal to help us understand more how to apply it in the real world.

Task 2.1

In task 2.1 we want to explore the use of matched filtering in detection of the characteristic pulses.
Matched filtering is a fundamental technique used to extract known waveforms from signals corrupted by noise. This is achieved by performing cross-correlation between the signal and the waveform of interest. For instance, in seismic exploration, cross-correlating the vibroseis sweep (a specific waveform) with recorded seismic data aids in detecting subsurface structures. Essentially, matched filtering serves as a method for identifying a predefined signal or waveform within a noisy environment or the other way around (as in our case: detecting a known pulse within an audio file).

```
%% Task 2
%Ex. 2.1
%Load audio file
[data, fs] = audioread('audio.wav');
%Load standard pulse
load('Pulse.mat');
%Perform matched filtering on each channel
filtered_signals = zeros(length(data), size(data, 2));
for i = 1:size(data, 2)
   filtered_signals(:, i) = conv(data(:, i), flipud(yp), 'same');
end
%Plot original and filtered signals
figure;
subplot(4, 1, 1);
plot(data(:, 1)); %original signal for channel 1
title('Original Signal (Channel 1)');
xlabel('Time')
ylabel('Amplitude')
subplot(4, 1, 3);
plot(data(:, 2)); %original signal for channel 2
title('Original Signal (Channel 2)');
xlabel('Time')
ylabel('Amplitude')
subplot(4, 1, 2);
plot(filtered_signals(:, 1)); %filtered signal for channel 1
title('Filtered Signal (Channel 1)');
xlabel('Time')
ylabel('Amplitude')
subplot(4, 1, 4);
plot(filtered_signals(:, 2)); %filtered signal for channel 2
title('Filtered Signal (Channel 2)');
xlabel('Time')
ylabel('Amplitude')
```

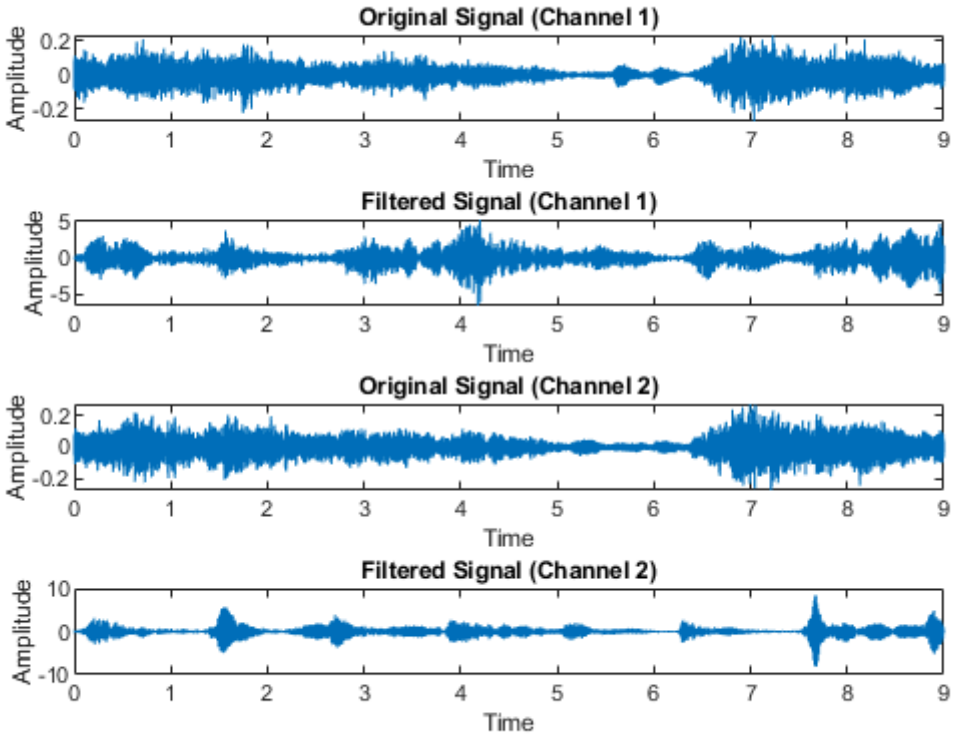

fig 13: Correlation of the pulse with the signal using matched filtering

Conclusions: The results show that the known pulse is present in both channels during the whole duration of the signal. From the correlation plots presented above we can see that the pulses are particularly noticeable in the 1st channel at 3, 4 and 8.5s and in the 2nd channel at 1.5, 7.6, 8.8s. Additionally the 2nd channel contains less pulses but when they are present they show higher correlation.
We were unable to clearly hear the pulse while listening to the audiofile.

Task 2.2

In task 2.2 we have attempted to solve the same problem as in the previous task this time using frequency domain. To this end the input signal and the pulse function were transformed using fft, multiplied using complex conjugate, and returned to the time domain for subsequent analysis.

```
%Ex.2.2
% Load audio file
[data, fs] = audioread('audio.wav');
%Load standard pulse
load('Pulse.mat');
%Compute Fourier transform of the audio signal
data_fft = fft(data);
%Apply zero-padding to the Fourier transform of the pulse
N = length(data);
yp_fft = fft(yp, N);
```

```
%Perform complex conjugate multiplication
filtered_signals_fft = data_fft .* conj(yp_fft.');
%Compute inverse Fourier transform
filtered_signals_time = ifft(filtered_signals_fft);
%Normalize filtered signals
filtered_signals_norm = normalize(real(filtered_signals_time));
%Define time vector in seconds
t = (0:length(data)-1) / fs;
%Plot original and filtered signals
figure;
subplot(4, 1, 1);
plot(t, data(:, 1)); %original signal for channel 1
title('Original Signal (Channel 1)');
xlabel('Time (s)')
ylabel('Amplitude')
xlim([0,9]);
subplot(4, 1, 3);
plot(t, data(:, 2)); %original signal for channel 2
title('Original Signal (Channel 2)');
xlabel('Time (s)')
ylabel('Amplitude')
xlim([0,9]);
subplot(4, 1, 2);
plot(t, filtered_signals_norm(:, 1)); %filtered signal for channel 1
title('Filtered Signal (Channel 1)');
xlabel('Time (s)')
ylabel('Amplitude')
xlim([0,9]);
```

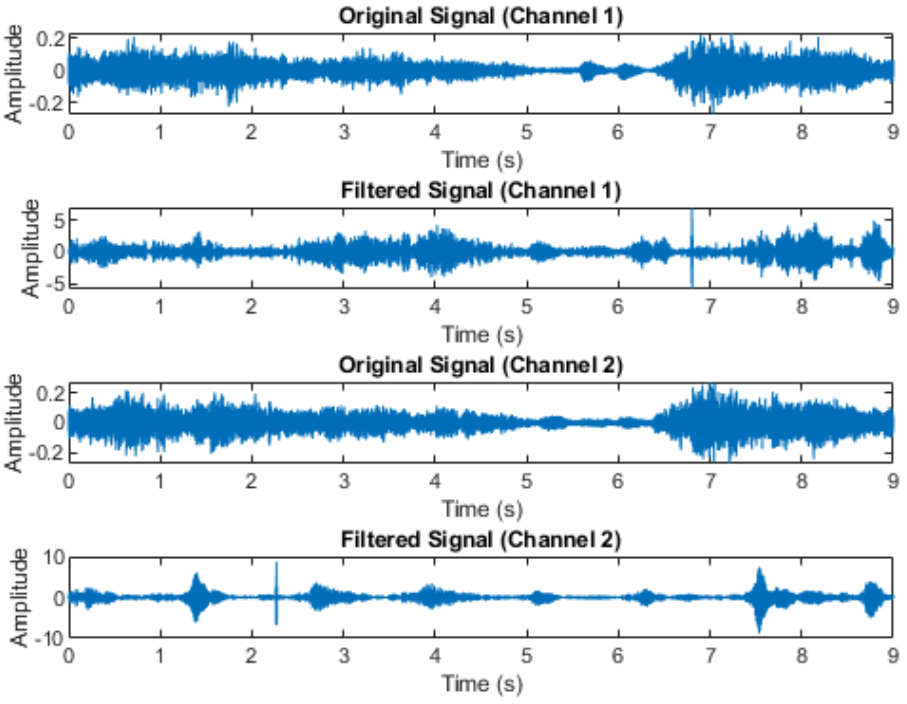

fig 14: Correlation of the pulse with the signal using frequency domain

Conclusions: The results show that the obtained filtered waveforms present the peaks at almost the same places as the results from the previous task with 2 exceptions in channel 1 at ~6.7s and in channel 2 at ~2.3s which present peaks significantly more pronounced than in the previous task.
Overall, such a result was somewhat expected since the method utilized in the previous task is also well suited for noise detection. Both methods have their own unique set of advantages, and can be interchanged based on the need of the particular task.