Mechatronic Systems Identification

Lab 4 - Autocorrelation and cross-correlation using experimental data

Khaldoun Fayad - 409597

Witold Surdej - 407100

04.04.2024

Description

The aim of this laboratory is for us to learn how to practically implement cross-correlation and autocorrelation in a real scenario

Task 1

This task aims to explore the cross-correlation of signals in Matlab and how best to discover the implications of different setups of correlated signals, it was performed using a procedure where two microphones were placed at a certain distance from one another and we were to find the distance between them using correlation.

Task 1.1

From the outlined task, we can infer that the experiment was designed to demonstrate the application of cross-correlation in determining the spatial relationship between two audio signals captured by microphones placed at different distances from a sound source. This setup leverages the properties of sound propagation and the principle of cross-correlation to estimate the distance between two recording devices. Based on the task instructions, here are several conclusions that could form a comprehensive report:

```
%% Task 1
%Ex. 1.1
%Setup 1
[audio,fs] = audioread("twomicrophones1.wav");
N = length(audio);
tp = (0:N-1)/fs;
figure()
subplot(2,1,1)
plot(tp,audio(:,1))
xlabel('time (s)')
ylabel('amplitude (-)')
title('Audio from 1st microphone')
subplot(2,1,2)
plot(tp,audio(:,2))
xlabel('time (s)')
ylabel('amplitude (-)')
title('Audio from 2nd microphone')
%Correlation without compensation
[corr, lag] = xcorr(audio(:,1),audio(:,2));
t = lag/fs;
figure()
plot(t,corr)
xlabel('time (s)')
ylabel('amplitude (-)')
title('Correlated signal')
%Correlation with compensation
ycomp1 = audio(:,1) - mean(audio(:,1));
ycomp2 = audio(:,2) - mean(audio(:,2));
```

```
timedelay =

   -0.0025
```

```
[corr2, lag2] = xcorr(ycomp1,ycomp2);
t2 = lag2/fs;
figure()
plot(t2,corr2)
xlabel('time (s)')
ylabel('amplitude (-)')
title('Correlated and compensated signal')
%Finding the distance
v = 345;
[maxCorr,index] = max(abs(corr2));
timedelay = lag(index)/fs
distance = abs(timedelay * v)
```

fig 1: shows the results of the time delay and distance of the first setup

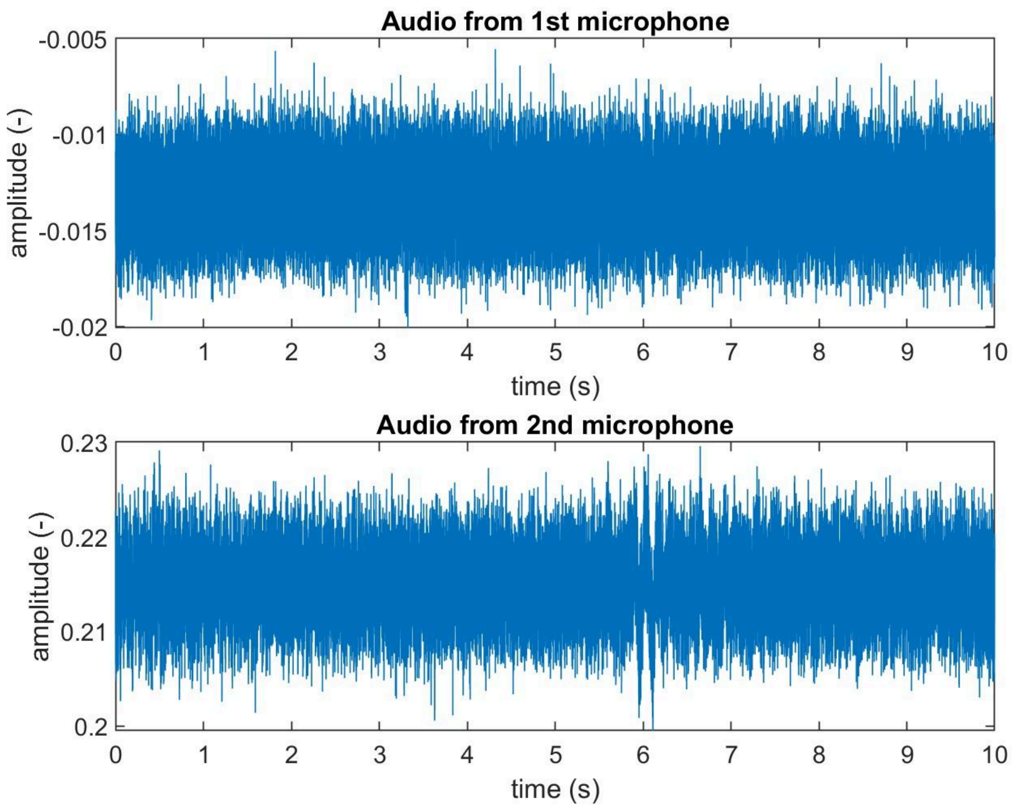


fig 2: shows the plot of the signal captured from both microphones

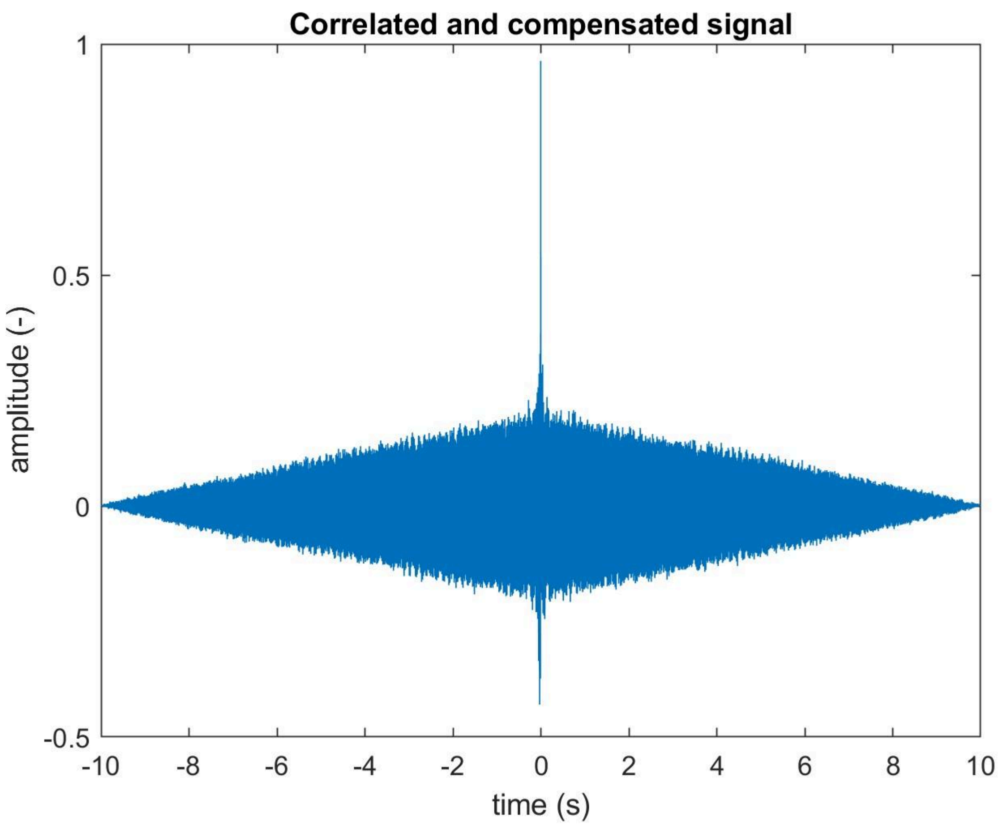
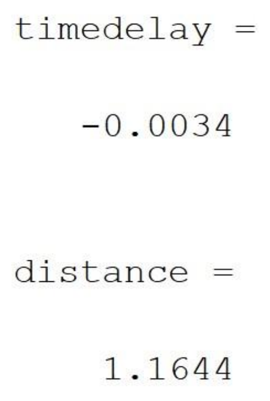


fig 3: shows the plot of the correlated and compensated signal

```
%Setup 2
[audio,fs] = audioread("twomicrophones2.wav");
N = length(audio);
tp = (0:N-1)/fs;
figure()
subplot(2,1,1)
plot(tp,audio(:,1))
xlabel('time (s)')
ylabel('amplitude (-)')
title('Audio from 1st microphone')
subplot(2,1,2)
plot(tp,audio(:,2))
xlabel('time (s)')
ylabel('amplitude (-)')
title('Audio from 2nd microphone')
%Correlation without compensation
[corr, lag] = xcorr(audio(:,1),audio(:,2));
t = lag/fs;
figure()
plot(t,corr)
xlabel('time (s)')
ylabel('amplitude (-)')
title('Correlated signal')
%Correlation with compensation
ycomp1 = audio(:,1) - mean(audio(:,1));
```

```
ycomp2 = audio(:,2) - mean(audio(:,2));
[corr2, lag2] = xcorr(ycomp1,ycomp2);
t2 = lag2/fs;
figure()
plot(t2,corr2)
xlabel('time (s)')
ylabel('amplitude (-)')
title('Correlated and compensated signal')
%Finding the distance
v = 345;
[maxCorr,index] = max(abs(corr2));
timedelay = lag(index)/fs
distance = abs(timedelay * v)
```



```
timedelay =

   -0.0034


distance =

    1.1644
```

fig 4: shows the results of the time delay and distance of the second setup

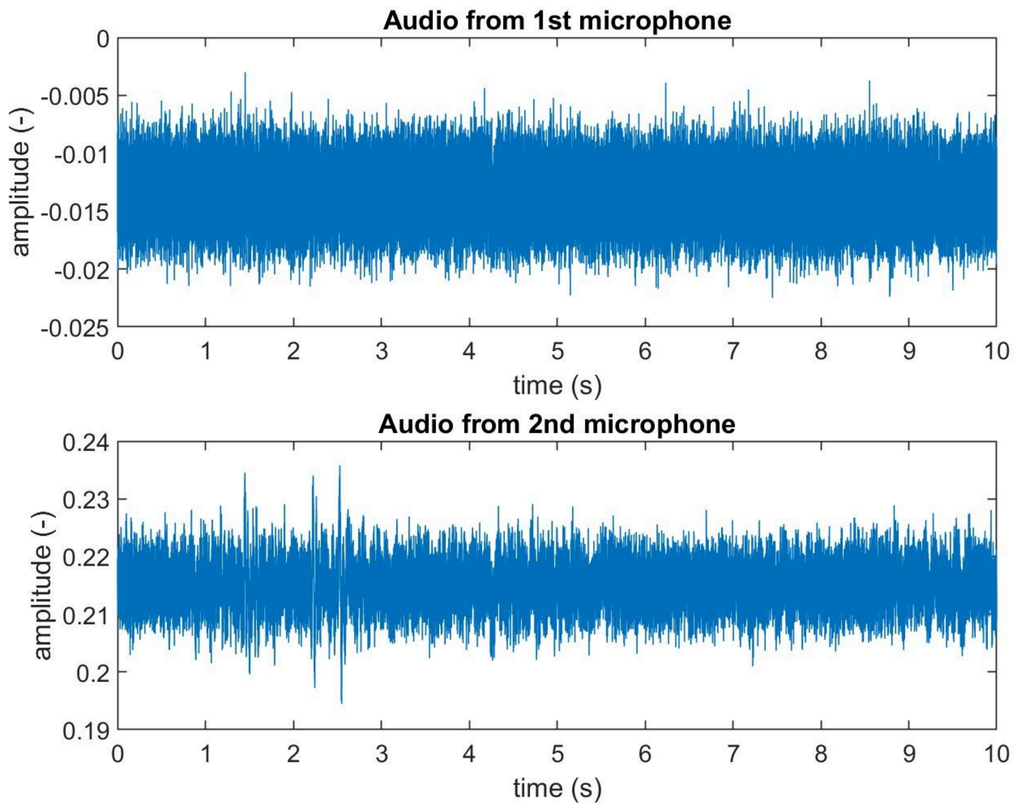


fig 5: shows the plot of the signal captured from both microphones in the second setup

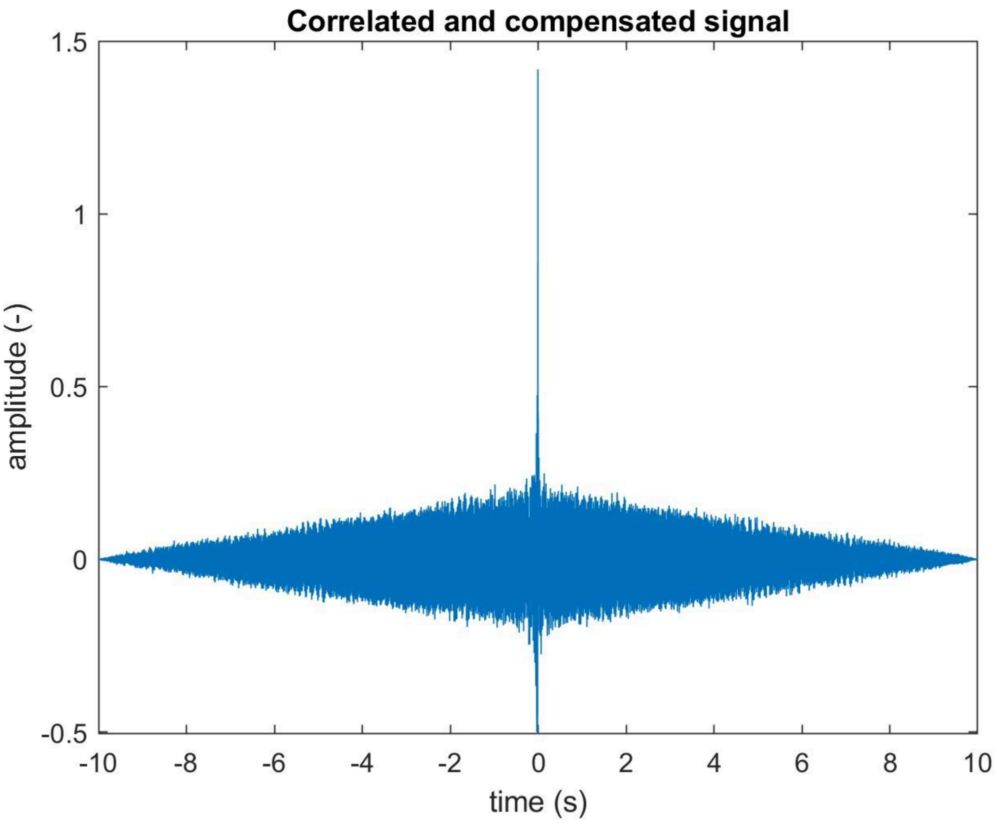


fig 6: shows the plot of the correlated and compensated signal of the second setup

Conclusions: The experiment underscores the utility of cross-correlation for analyzing and comparing audio signals from different sources or locations. By correlating the signals recorded by two microphones, it's possible to identify the time delay between them, which reflects the distance difference from the sound source. This demonstrates cross-correlation's value in signal processing applications, such as sound localization, spatial audio recording, and acoustic environment mapping.

The need to compensate for fixed components in the recordings by subtracting the average signal highlights the importance of preprocessing in signal analysis. Such steps improve the accuracy of subsequent analyses by reducing bias and enhancing the signal-to-noise ratio, making it easier to identify significant features like the point of maximum correlation.

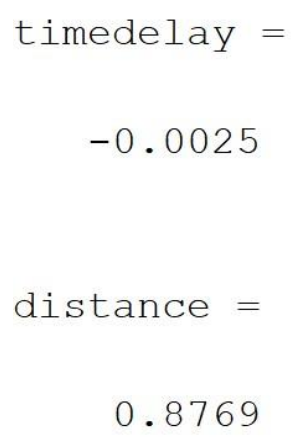
By using the known speed of sound to calculate the distance between the microphones based on the delay identified through cross-correlation, the experiment effectively applies theoretical principles of sound propagation to a practical scenario. This approach can be extended to various applications, including acoustic measurement, environmental sensing, and audio forensics.

Based on the constant of the speed we have which is *v = 345 m/s* we were able to deduce that the time delay in the first and second setups to be *25 ms* and *34 ms* consecutively. Due to these values, the calculated distance of the first setup was *0.8769 m,* and for the second setup, it was *1.1644 m.*

Task 1.2

This task demonstrates a sophisticated approach to estimating the distance between two microphones using signal processing techniques, specifically through the analysis of spectral transmittance and impulse response derived from the recording. From this experiment, several key conclusions can be drawn that encapsulate the principles of acoustics, signal processing, and their application in spatial analysis.

```
%Ex. 1.2
%Setup 1
[audio,fs] = audioread("twomicrophones1.wav");
N = length(audio);
tp = (0:N-1)/fs;
ycomp1 = audio(:,1) - mean(audio(:,1));
ycomp2 = audio(:,2) - mean(audio(:,2));
Hw = (fft(ycomp2))./(fft(ycomp1));
%Calculating time from fourier
ht = ifft(Hw);
figure()
plot(tp,ht)
xlabel('time (s)')
ylabel('amplitude (-)')
title('Plot of the inversed signal')
%Finding the distance
v = 345;
[maxCorr,index] = max(abs(ht));
timedelay = index/fs
distance = abs(timedelay * v)
```



```
timedelay =

   -0.0025


distance =

    0.8769
```

fig 1: shows the results of the time delay and distance of the first setup

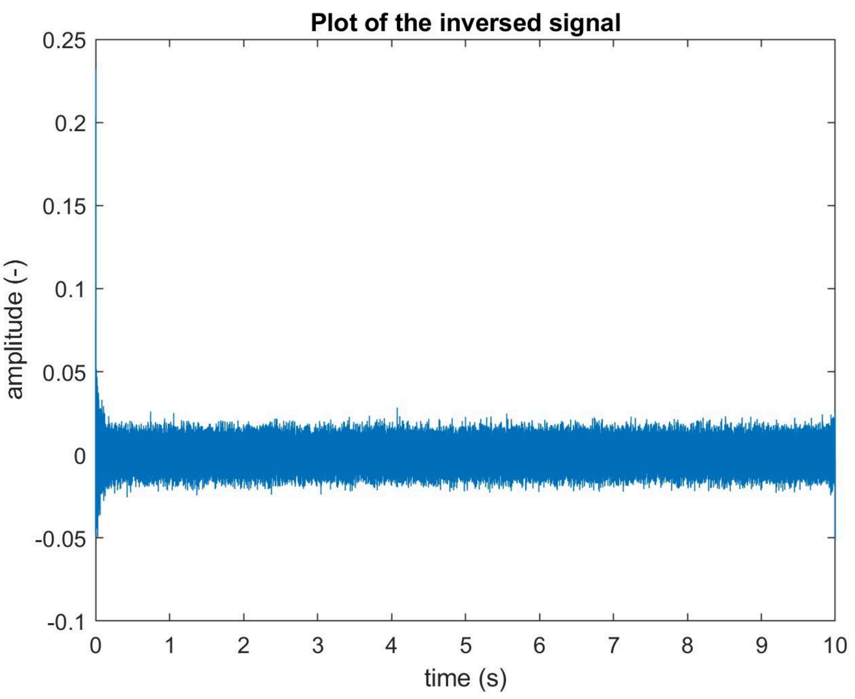


fig 7: shows the plot of the inversed signal after the inverse Fourier function was performed on it.

Conclusions: By calculating the spectral transmittance H(ω) between the Fourier transforms of the input and output signals, the experiment showcases how to assess the frequency response of a system—in this case, the acoustic path between a sound source and the microphones. This analysis provides insights into how different frequencies are attenuated or amplified by the physical environment, which has implications for understanding the acoustic properties of spaces and designing audio equipment.
The derivation of the impulse response h(t) through the inverse Fourier transform of H(ω) bridges the frequency-domain analysis back to the time domain, offering a direct way to examine how sound propagates through space over time. The impulse response characterizes the system's output in response to a unit impulse at the input, revealing the time it takes for sound to travel between the microphones.
This method shows that there is more than one way to try to calculate spatial distances between audio devices and both seem to provide very accurate and precise results.

Task 2

This task aims at determining the distance between two speakers emitting the same signal captured by a single microphone. To this end, we will discuss the approach to solving such a task using auto-correlation.

Task 2.1

The first task consists of trying to find the distance between the two speakers by calculating the autocorrelation of the signals emitted by each speaker. The secondary peak in the correlation plot should indicate the moment when the microphone received the signal from the further away speaker. We can then compare the distance with the highest peak (which is always centered at 0 in the autocorrelation plot) and determine the time lag between the receiving of the two signals.

To this end the signals were loaded into Matlab, compensation was performed to get rid of stationary elements and autocorrelation was performed using *xcorr()* function. Finally, the positions of the two main peaks were extracted and the distance between the speakers was computed. The code as well as the resultant figures are presented below.

```
%Loading the data
[audio_ref,fs] = audioread('twospeakers1_ref.wav');
[audio1,fs] = audioread('twospeakers1.wav');
[audio2,fs] = audioread('twospeakers2.wav'); %we are overwritting fs cuz
its the same for all 3
%compensating
ycomp_ref = audio_ref-mean(audio_ref);
ycomp_1 = audio1-mean(audio1);
ycomp_2 = audio2-mean(audio2);
%autocorrelating ycomp_1
[c,lags] = xcorr(ycomp_1,ycomp_1);
[c2,lags2] = xcorr(ycomp_ref,ycomp_ref);
[c3,lags3] = xcorr(ycomp_2,ycomp_2);
```

```
%time vector
t = lags/ fs;
%plotting
figure()
plot(t,c), hold on
plot(t,c2)
plot(t,c3)
xlabel('time [s]') %change to seconds
ylabel('amplitude (-)')
legend('signal1','reference', 'signal 2')
%extract peak values
[val1, loc1]=max(abs(c))
[val2, loc2]=max(abs(c(10*fs+1:10.0025*fs)))
%velocity
v=345;
%distance in signal 1
timedelay=loc1/fs-10.002548
distance_between=abs(timedelay*v)*100
%distance in signal 2
timedelay2=loc1/fs-10.0044375
distance_between2=abs(timedelay2*v)*100
```

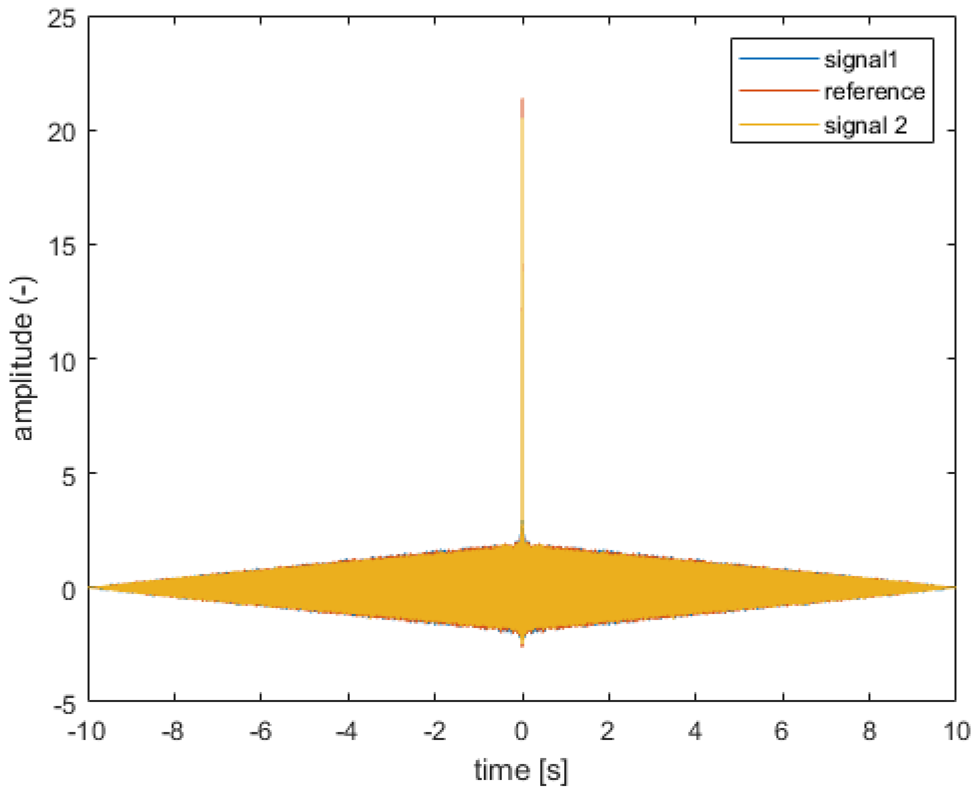


fig 7: autocorrelation plot of the three signals

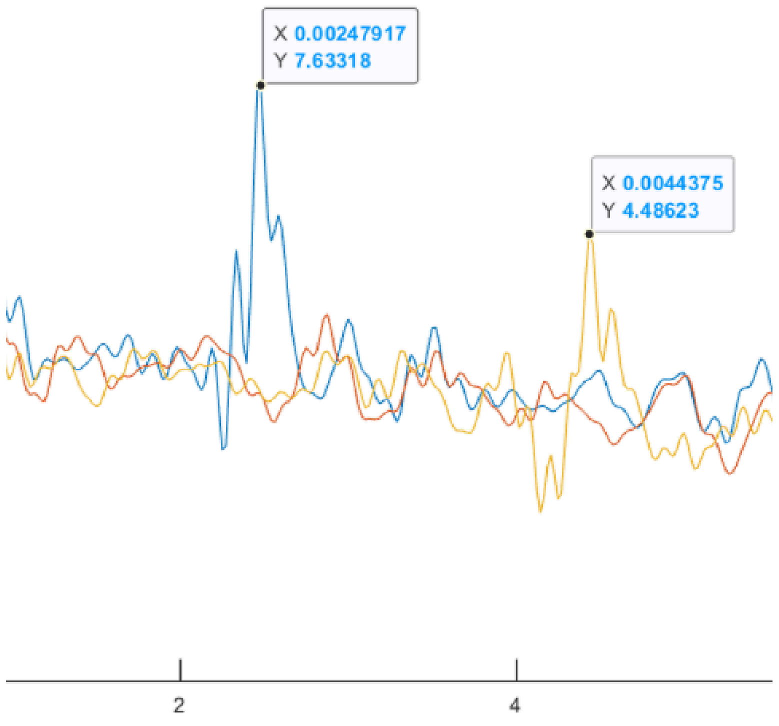
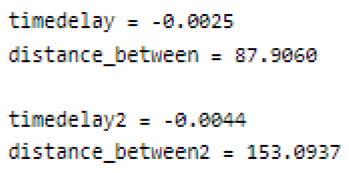


fig 8: peaks obtained by autocorrelation signals 1 and 2 plotted over reference signal



```
timedelay = -0.0025
distance_between = 87.9060

timedelay2 = -0.0044
distance_between2 = 153.0937
```

fig 9: A time delay and distance between the speakers in signal 1 and signal 2

Conclusions: The results indicate that the distance between the speakers in the 1st case was ~88cm and in the 2nd case was ~153cm, which showcases that the method of determining the distances between two sources of sound using autocorrelation is a valid and reliable method to approximate the real distance between the sources. The results are not fully reliable due to the presence of noise and other disturbances (such as slight variation in the speed of sound) which can disturb the results.

Task 2.2

In task 2.2 we are dealing with an identical problem as in the previous task, this time we will however filter the signal using two different filters and determine their impact on the accuracy of the results..

```
%Ex. 2.2
%loading the data
[audio_ref,fs] = audioread('twospeakers1_ref.wav');
```

```
[audio1,fs] = audioread('twospeakers1.wav');
[audio2,fs] = audioread('twospeakers2.wav'); %we are overwritting fs cuz
its the same for all 3
%compensating
ycomp_ref = audio_ref-mean(audio_ref);
ycomp_1 = audio1-mean(audio1);
ycomp_2 = audio2-mean(audio2);
%filtering the 1st signal with filter fstop = 1000
yf=filter(Hd1,squeeze(ycomp_1));
%filtering the 1st signal with filter fstop = 300
yf2=filter(Hd2,squeeze(ycomp_2));
%time vector for plotting
N=length(audio2);
tp = (0:N-1)/fs;
%plotting
figure()
plot(tp,yf)
%autocorrelating
[c,lags] = xcorr(yf,yf);
[c2,lags2] = xcorr(ycomp_ref,ycomp_ref);
[c3,lags3] = xcorr(yf2,yf2);
%time vector for cr
t = lags/ fs;
%plotting
figure()
plot(t,c), hold on
plot(t,c2)
plot(t,c3)
xlabel('time [s]') %change to seconds
ylabel('amplitude (-)')
legend('signal1','reference', 'signal2')
%finding the maximum
[val1, loc1]=max(abs(c))
[val2, loc2]=max(abs(c3))
%velocity
v=345;
%distance in signal 1
timedelay=loc1/fs-10.00260417
distance_between=abs(timedelay*v)*100
%distance in signal 2
timedelay2=loc2/fs-10.00447917
distance_between2=abs(timedelay2*v)*100
```

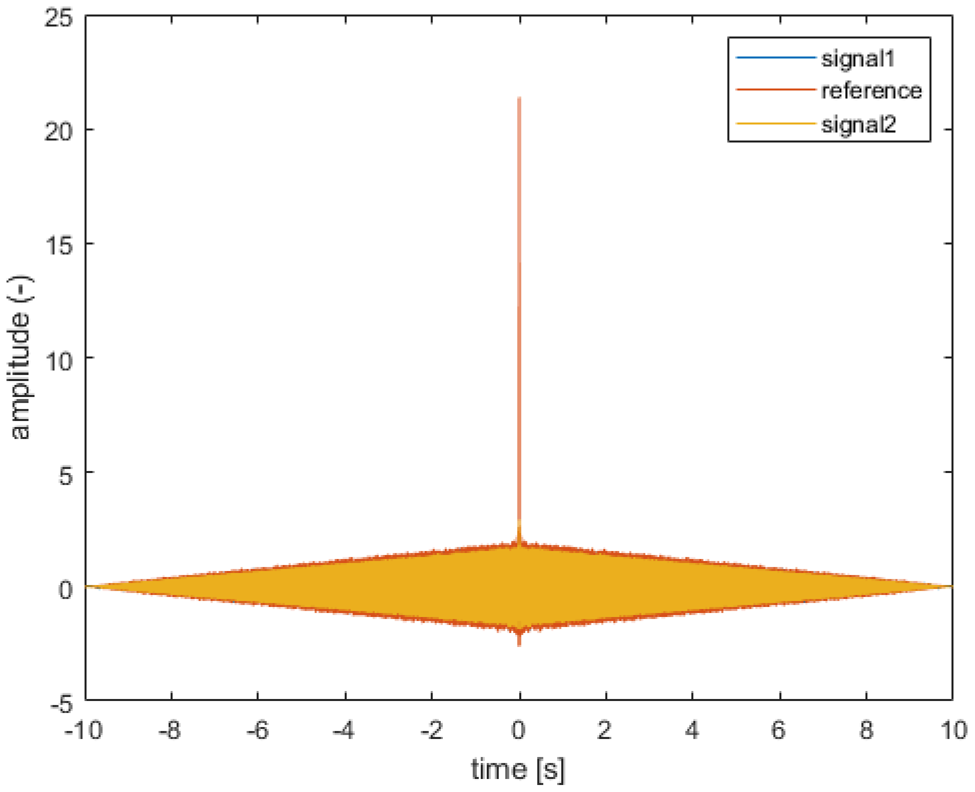
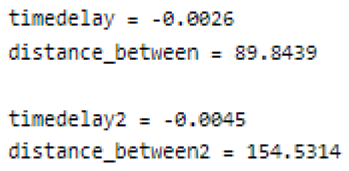


fig 10: Autocorrelation plot of the two filtered signals and the reference



```
timedelay = -0.0026
distance_between = 89.8439

timedelay2 = -0.0045
distance_between2 = 154.5314
```

fig 11: A time delay and distance between the speakers in signal 1 and signal 2

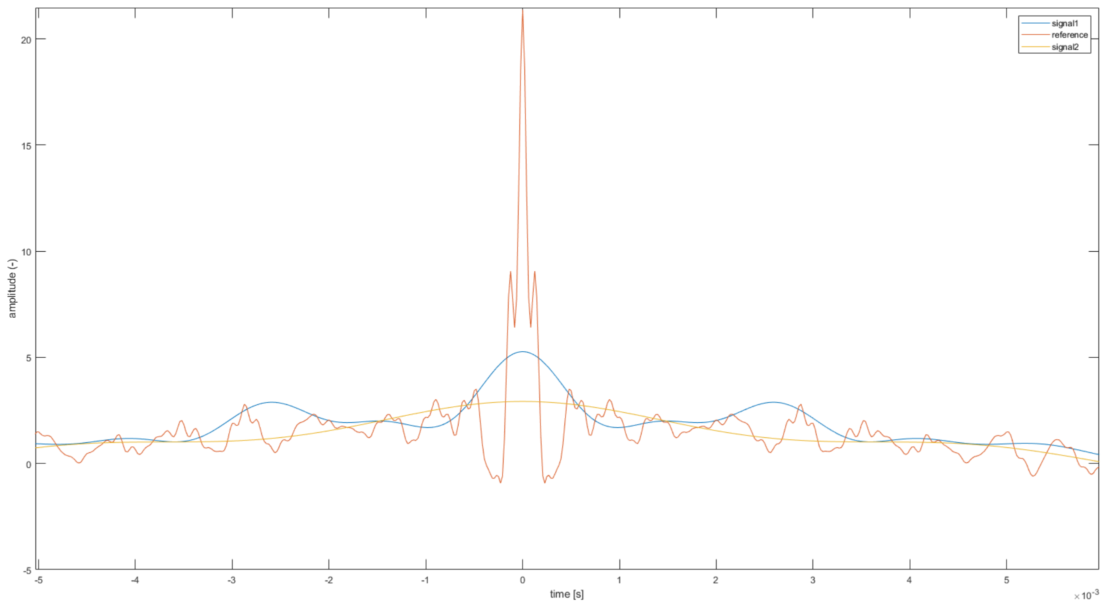


fig 12: Effects of the filtering on the peaks of the autocorrelation plot

Conclusions: The results show that the obtained distances slightly changed concerning the previous task. This change was expected since the filtering gets rid of some of the noise initially present in the signal. Additionally, figure 12 showcases that the filtering had a smoothing effect on the curves of the autocorrelation plots. Thanks to it the peaks are more clearly visible and it's easier to select the maximum.
Autocorrelation in spatial calculations is a tool for understanding the degree of similarity or pattern repetition across a spatial domain. By comparing a dataset with itself over various lags, it showcases the periodicity of features within the data. This technique is crucial in fields like geography, meteorology, and environmental science, where it helps in identifying spatial patterns, assessing the scale of variability, and making predictions about spatial phenomena.