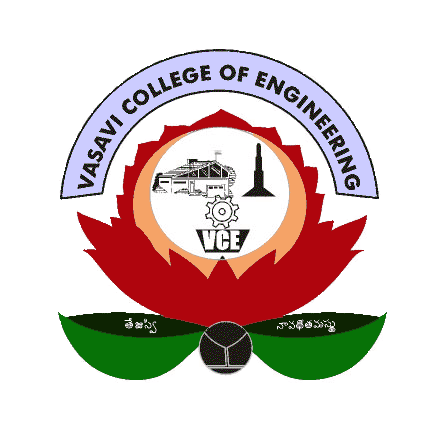
**VASAVI COLLEGE OF ENGINEERING**

SIGNALS AND SYSTEMS LABORATORY

**MULTIPLICATION OF AUDIO SIGNALS**

**IN TIME DOMAIN AND FREQUENCY DOMAIN**

ECE B 2/4 3rdsem

***Submitted By:***

1602-21-735-117

1602-21-735-119

1602-21-735-115

Contents: page no.

* Introduction 3
* Fast Fourier transform 4
* Audio signals 6
* Multiplication of signals 7
* Frequency domain 8
* Time domain 12
* MATLAB code 13
* MATLAB code explanation 16
* Future aspects 17
* Results 18
* References 20

# Introduction:

The following report contains different topic explanations about signals and audio signals

As we know one of our five senses is ability to hear, how do we hear? via out ear.

Did we question how does our ears work? They have an ear drum inside them, the ear drum collects sound waves and channels them into our ear canal. The sound waves are audio signals

Right now, we are trying to jump into the concept of audio signals

What are audio signals and how are they interpreted?

If they are a type of signals, can we do any operations on them and alter the signal? We have investigated this and tried to code our experience with the operation of multiplication of audio signals

Here in this project we will be working with two recorded voice signals on which the operations are performed in frequency as well as in time domain.

# Fast Fourier transform:

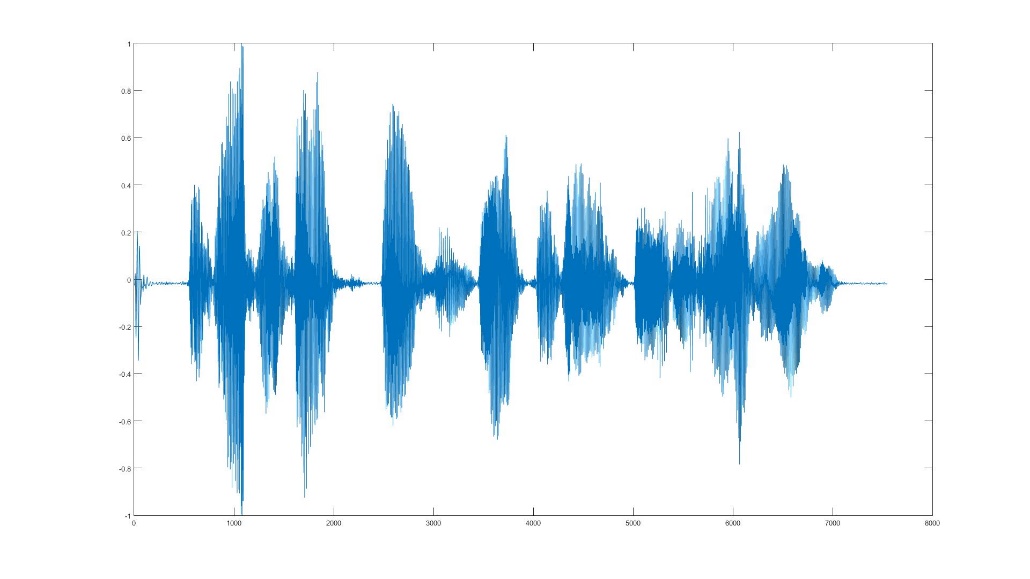
The Fast Fourier Transform (FFT) is a widely used algorithm in signal processing that allows for the efficient computation of the Discrete Fourier Transform (DFT) of a signal. The DFT is a mathematical operation that transforms a time-domain signal into a frequency-domain signal, which can then be analysed and manipulated. This can be useful for a variety of tasks such as filtering, spectral analysis, and modulation/demodulation of signals.

In signal processing, the FFT is used to analyse signals in the frequency domain to extract useful information. For example, it can be used to identify the frequencies present in a signal, which can be useful in tasks such as speech recognition, audio compression and noise reduction. It can also be used for filtering signals, for example, removing noise or unwanted frequency components. Additionally, it can be used to analyse and design control systems, as well as in image and video processing.

The FFT is also used in other fields such as telecommunications, audio processing, and mechanical vibrations analysis. As it allows to process large amount of data quickly, it is considered a standard tool in many signal processing applications. It is also widely used in scientific computing and engineering because it allows to solve problems that would be otherwise computationally infeasible.

Overall, the FFT is a powerful tool that allows for efficient and accurate frequency-domain analysis of signals, making it an essential tool in signal processing and many other fields.

# Audio Signals:

An audio signal is an electrical representation of sound. It is a time-varying voltage or current that corresponds to the air pressure changes of a sound wave. Audio signals are typically in the frequency range of 20 Hz to 20 kHz, which is the range of human hearing. They can be generated by various sources like a human voice, musical instruments, or even nature.

Engineers work with audio signals to design and improve audio equipment such as microphones, speakers, and audio processors such as equalizers, compressors, and noise reduction filters. They are often digitized for storage and processing in electronic devices such as computers and smartphones. With the help of audio signal processing techniques, engineers can enhance the overall audio quality and reduce noise in the signal.

# Multiplication of audio signals:

Multiplying two audio signals is a technique used in audio signal processing to combine the characteristics of two different signals into one. This process is known as signal mixing or signal multiplication, and it is commonly used in a variety of audio applications such as music production, broadcasting, and audio post-production.

When two audio signals are multiplied, the resulting signal inherits the frequency and amplitude characteristics of both input signals. This can result in a variety of interesting effects such as the creation of harmonics, the addition of reverberation, and the enhancement of certain frequency bands.

One of the most common uses of signal multiplication is in the creation of reverberation effects. By multiplying an audio signal with a pre-recorded impulse response of a room or other space, engineers can simulate the acoustic characteristics of the space and add a sense of depth and dimension to the sound.

Another use of signal multiplication is in the creation of harmonics. By multiplying an audio signal with a sine wave at a specific frequency, engineers can add harmonic content to the original signal, resulting in a richer and more complex sound. This technique is commonly used in music production and sound design.

Signal multiplication can also be used to enhance certain frequency bands in an audio signal. By multiplying an audio signal with a band-pass filter, engineers can selectively boost or cut certain frequency ranges to achieve a desired sound.

Overall, signal multiplication is a powerful technique in audio signal processing, and it can be used to achieve a wide range of effects and improve the overall sound quality of an audio signal. By understanding the principles of signal multiplication, engineers can create more dynamic and engaging audio experiences for their audiences.

# Frequency domain:

The frequency domain refers to the representation of a signal in terms of its frequency content. In contrast to the time domain, which represents a signal as a function of time, the frequency domain represents a signal as a function of frequency. This can be done using the Fourier Transform, which converts a signal from the time domain to the frequency domain.

The frequency domain representation of a signal can provide useful information about the signal, such as the frequency components that make up the signal, and their amplitudes. This information can be used for various signal processing tasks such as filtering, modulation, and compression.

For example, in audio signal processing, the frequency domain representation can be used to identify the frequency components of a sound, and remove unwanted noise or enhance specific frequencies to improve sound quality. In image processing, the frequency domain representation can be used for image compression and denoising. In telecommunications, the frequency domain representation can be used for modulation and demodulation of signals.

In the frequency domain, each frequency component is represented by a complex number, which encodes both the amplitude and phase of that frequency component in the signal. The magnitude of the complex number represents the amplitude, while the phase represents the timing of the frequency component relative to the beginning of the signal.

In summary, frequency domain representation of a signal is a way to represent a signal in terms of its frequency content. It gives a detailed understanding of the frequency components of the signal and their amplitudes, which can be used for various signal processing tasks such as filtering, modulation, and compression. The frequency domain representation is often represented by complex numbers which encodes amplitude and phase of the frequency component in the signal.

The following are few examples of representation of signals in frequency domain:

* Audio Signal Processing: In audio signal processing, the frequency domain representation of an audio signal can be used to identify the frequency components of a sound, and remove unwanted noise or enhance specific frequencies to improve sound quality. For example, a low-pass filter can be applied to remove high-frequency noise from an audio signal, or a high-pass filter can be applied to remove low-frequency noise.
* Image Processing: In image processing, the frequency domain representation of an image can be used for image compression and denoising. For example, the Discrete Cosine Transform (DCT) can be used to compress an image by removing the high-frequency components that are less visually significant. On the other hand, a Gaussian low-pass filter can be applied to remove noise from an image by removing high-frequency components.
* Telecommunications: In telecommunications, the frequency domain representation of a signal can be used for modulation and demodulation of signals. For example, in Amplitude Modulation (AM) and Frequency Modulation (FM) the message signal modulates the amplitude and frequency of the carrier signal, respectively.
* Medical Imaging: In medical imaging, the frequency domain representation of an image is used to extract features of the image that are important for diagnosis. For example, a Radon transform can be used to extract prominent features of an X-ray image, such as the location and shape of tumors.
* Spectral Analysis: In Spectral Analysis, the frequency domain representation of a signal is used to identify the frequency components of a signal and their amplitude. This can be used for various applications such as identifying the frequency of vibration in a machine or identifying the frequency of sound in a room.

# Time domain:

The time domain refers to the representation of a signal as a function of time. In contrast to the frequency domain, which represents a signal in terms of its frequency content, the time domain represents a signal in terms of its amplitude over time. A signal in the time domain can be represented by a waveform, which displays the amplitude of the signal at each point in time.

The time domain representation of a signal can provide useful information about the signal, such as the amplitude of the signal at specific points in time, and how the amplitude of the signal changes over time. This information can be used for various signal processing tasks such as signal filtering, modulation, and compression.

For example, in audio signal processing, the time domain representation can be used to identify the amplitude of a sound at specific points in time, and remove unwanted noise or enhance specific amplitudes to improve sound quality. In image processing, the time domain representation can be used for image compression and denoising. In telecommunications, the time domain representation can be used for modulation and demodulation of signals.

In the time domain, each point in a signal is represented by a real number, which encodes the amplitude of the signal at that point in time. The time domain representation of a signal can be analyzed using various time domain analysis techniques such as time-domain filtering, time-domain averaging, and time-domain correlation.

In summary, time domain representation of a signal is a way to represent a signal as a function of time. It gives a detailed understanding of the amplitude of the signal at specific points in time and how the amplitude of the signal changes over time. The time domain representation is often represented by real numbers which encodes the signal's amplitude at a specific point. It can be used in various fields such as audio signal processing, image processing, and telecommunications.

# **MATLAB code for multiplication of recorded audio signals**

close all;

clear;

clc;

% audio recording

Fs =4000;

Channels = 1;

bits = 16;

%recording first audio file

r1 = audiorecorder(Fs,bits, Channels);

duration = 5; disp('audio recording started x1');

recordblocking(r1,duration);

disp('audio recording stopped x1');

X1 = getaudiodata(r1);

% creating first audio file

filename = 'myvoice1.wav';

audiowrite(filename,X1,Fs);

pause(5);

%recording second audio file

r2 = audiorecorder(Fs,bits, Channels);

duration = 5; disp('audio recording started x2');

recordblocking(r2,duration);

disp('audio recording stopped x2');

X2 = getaudiodata(r2);

% creating second audio file

filename = 'myvoice2.wav';

audiowrite(filename,X2,Fs);

%listening to the first audio sound

disp('first audio file');

sound(X1,Fs,bits);

pause(10);

%listening to second audio sound

disp('second audio file');

sound(X2,Fs,bits);

% graph the audio x1

t = 0:1/Fs:(length(X1)-1)/Fs;

subplot(2,2,1); plot(t,X1,'LineWidth',1.5);

xlabel('time(sec)'); ylabel('Amplitude');

title('audio time domain - x1(t)')

n=length(X1);

Y1=fft(X1,n);F=0:(n-1)\*Fs/n;

F\_0=(-n/2:n/2-1).\*(Fs/n);

Y\_0=fftshift(Y1);

AY\_0=abs(Y\_0);

subplot(2,2,2);plot(F\_0,AY\_0,'LineWidth',1.5);

xlabel('Frequency(Hz)'); ylabel('Amplitude');

title('Audio frequency domain - X1(W)')

%graph the audio x2

t = 0:1/Fs:(length(X2)-1)/Fs;

subplot(2,2,3); plot(t,X2,'LineWidth',1.5);

xlabel('time(sec)'); ylabel('Amplitude');

title('audio time domain - x2(t)')

n=length(X2);

Y2=fft(X1,n);

F\_1=(-n/2:n/2-1).\*(Fs/n);

Y\_1=fftshift(Y2);

AY\_1=abs(Y\_1);

subplot(2,2,4);plot(F\_1,AY\_1,'LineWidth',1.5);

xlabel('Frequency(Hz)'); ylabel('Amplitude');

title('Audio frequency domain - X2(W)')

%Multiplying 2 audio signals in time domain

x3=X1.\*X2;

% creating multiplied audio file

filename = 'myvoice1\_2.wav';

audiowrite(filename,x3,Fs);

pause(10);

%listening to combinational sound

disp('multiplication audio file');

sound(x3,Fs,bits);

%graphing the Multiplied audio signal in time domain

t = 0:1/Fs:(length(x3)-1)/Fs;

figure;

subplot(2,1,1); plot(t,x3,'LineWidth',1.5);

xlabel('time(sec)'); ylabel('Amplitude');

title('audio signals multiplication time domain')

%Multiplying 2 audio signals in frequency domain

y3=Y1.\*Y2;

%%graphing the Multiplied audio signal in Frequency domain

if(length(X1)>length(X2))

n=length(X1);

else

n=length(X2);

end

F\_3=(-n/2:n/2-1).\*(Fs/n);

Y\_3=fftshift(y3);

AY\_3=abs(Y\_3);

subplot(2,1,2);plot(F\_3,AY\_3,'LineWidth',1.5);

xlabel('Frequency(Hz)'); ylabel('Amplitude');

title('audio signals multiplication frequency domain')

# ***How does this code work ?***

The code above is a MATLAB script that demonstrates the process of recording, saving, playing, and analyzing audio signals. The script uses the *audiorecorder* and *audiowrite* functions to record and save two audio signals, X1 and X2, with a duration of 5 seconds each. The script also uses the sound function to play the two signals back and the plot and *fft* functions to display the signals in both the time and frequency domain. The script also performs element-wise multiplication of the two signals in both the time and frequency domain, resulting in a third signal, x3.

The script begins by setting the audio recording parameters such as the sampling rate (Fs), number of channels (Channels), and bits (bits) to specific values. The *audiorecorder* function is then used to record the two audio signals, X1 and X2, with a duration of 5 seconds each. The *audiowrite* function is then used to save the two signals as .wav files, named 'myvoice1.wav' and 'myvoice2.wav' respectively.

The script then uses the *sound* function to play back the two signals, allowing the user to hear the recorded audio. The script also uses the plot and fft functions to display the signals in both the time and frequency domain. The time domain plots show the amplitude of the signals as a function of time, while the frequency domain plots show the amplitude of the signals as a function of frequency.

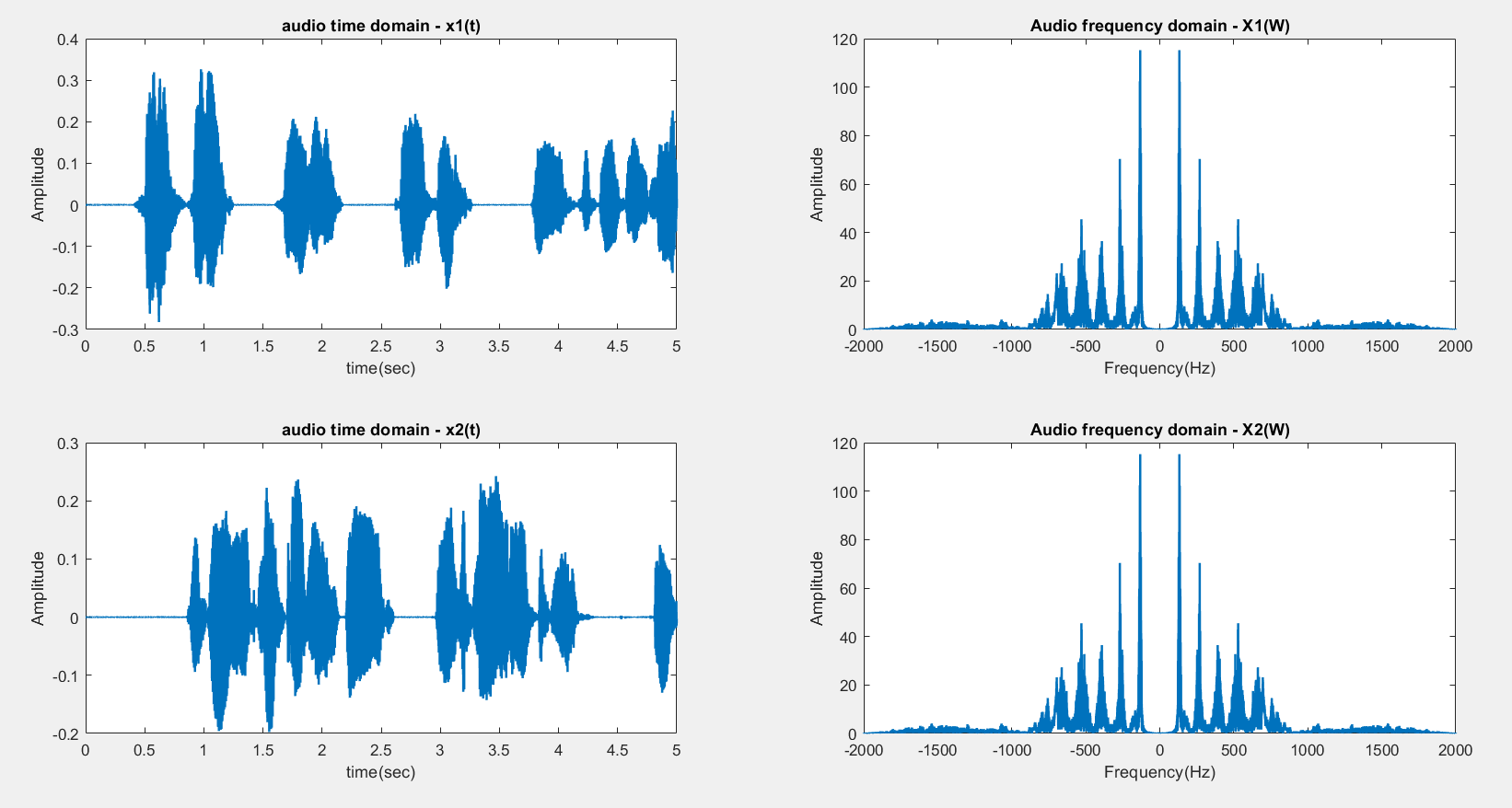
The script then performs element-wise multiplication of the two signals in both the time and frequency domain. In the time domain, the script multiplies the two signals element-wise, resulting in a third signal, x3. This signal is also saved as a .wav file, named 'myvoice1\_2.wav'. In the frequency domain, the script multiplies the two signals element-wise and plots the result in the frequency domain. This step is useful to see how the frequency content of the signals changes after multiplication.

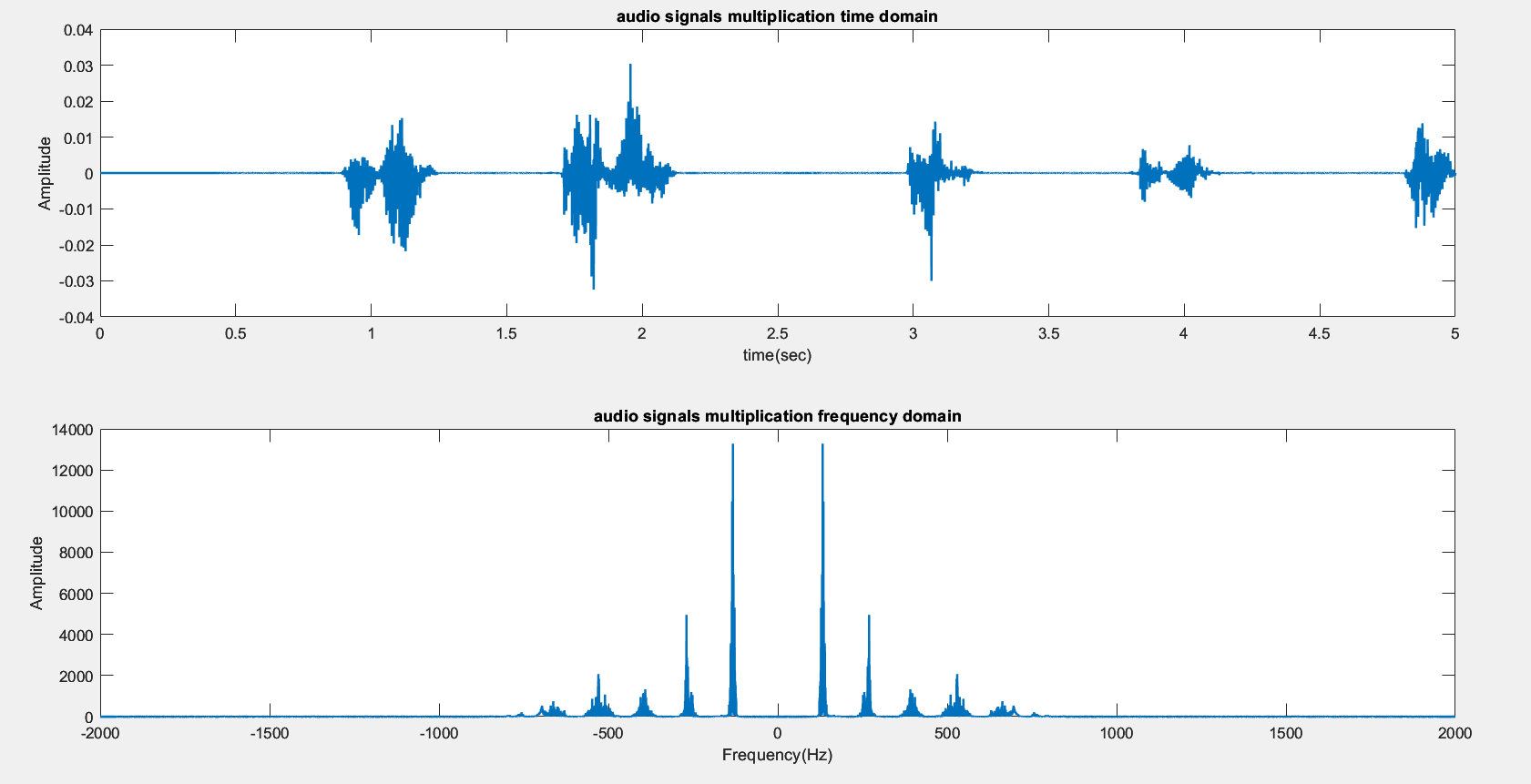
# **Future aspects:**

In the future, the multiplication of signals could be used in more advanced applications such as speech and audio recognition, image, and video compression, and even in telecommunications. With the increasing use of machine learning and artificial intelligence, the multiplication of signals could also be used in more complex applications such as natural language processing, computer vision, and self-driving cars. The possibilities are endless, and the field of signal processing is continually evolving, making it an exciting area for research and development.

In summary, the code above is a simple demonstration of the process of recording, saving, playing, and analyzing audio signals in MATLAB. It shows how multiplication of signals can be used in audio processing to combine multiple signals into one and how multiplication can be used in time and frequency domain. Additionally, it highlights the real-life applications of multiplication of signals in various fields and the future scope of this operation in advanced applications such as speech and audio recognition, image and video compression, and telecommunications

# **Result:**





The above graphs give information about time and frequency domain representation of 2 voice signals recorded through MATLAB. It also contains the time and frequency domain representation of audio signal which is obtained by the multiplication of 2 audio signals in time as well as frequency domain.

# ***References:***

MATLAB documentation:

<https://in.mathworks.com/help/matlab/ref/audiorecorder.html?s_tid=doc_ta>

<https://in.mathworks.com/help/matlab/ref/sound.html?s_tid=doc_ta>

<https://in.mathworks.com/help/matlab/ref/fft.html?s_tid=doc_ta>

Thank you