SPEECH ENHANCEMENT BASED ON APPROXIMATE MESSAGE PASSING

A Project report submitted in partial fulfilment of the requirements for the award of

The degree of

MASTER OF TECHNOLOGY

IN

DATA SCIENCES

Submitted by

KOLTUR YESHWANTH

21031DB011

Under the Esteemed guidance of

Dr.K.SANTHISREE

Professor of IT & Head

DEPARTMENT OF INFORMATION TECHNOLOGY

JNTUH UNIVERSITY COLLEGE OF ENGINEERING SCIENCE AND TECHNOLOGY

Kukatpally, Hyderabad – 500 085, Telangana, India

(2022 – 2023)

DEPARTMENT OF INFORMATION TECHNOLOGY

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

UNIVERSITY COLLEGE OF ENGINEERING, SCIENCE&TECHNOLOGY HYDERABAD

KUKATPALLY, HYDERABAD – 500085

2022 – 2023

CERTIFICATE

This is to certify that the thesis entitled **"Speech Enhancement based on Approximate Message Passing"**is being submitted by **Koltur Yeshwanth** bearing roll no. **2103DB011** in partial fulfillment of the requirements for the award of degree of **Master of Technology** in "**Data Sciences",** to the **Department of Information Technology, Jawaharlal Nehru Technological University Hyderabad** is a record of bonafide work carried out by her under our guidance and supervision.

The results presented in this project have been verified and are found to be satisfactory. The results embodied in this thesis have not been submitted to any other University for the award of any other degree or diploma.

GUIDE

Dr.K.SANTHI SREE

Professor of IT &HOD

DEPARTMENT OF INFORMATION TECHNOLOGY

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

UNIVERSITY COLLEGE OF ENGINEERING, SCIENCE&TECHNOLOGY HYDERABAD

KUKATPALLY, HYDERABAD – 500085

2022 – 2023

CERTIFICATE

This is to certify that the thesis entitled **"Speech Enhancement based on Approximate Message Passing"**is being submitted by **Koltur Yeshwanth** bearing roll no. **2103DB011** in partial fulfillment of the requirements for the award of degree of **Master of Technology** in "**Data Sciences",** to the **Department of Information Technology, Jawaharlal Nehru Technological University Hyderabad** is a record of bonafide work carried out by her under our guidance and supervision.

HEAD OF THE DEPARTMENT

Dr.K.SANTHI SREE

Professor of IT

DEPARTMENT OF INFORMATION TECHNOLOGY

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD

UNIVERSITY COLLEGE OF ENGINEERING, SCIENCE&TECHNOLOGY HYDERABAD

KUKATPALLY, HYDERABAD – 500085

2022 – 2023

DECLARATION

I, **Koltur Yeshwanth** bearing roll no. **2103DB011,** hereby declare that the report of ProjectWork entitled **"Speech Enhancement Based on Approximate Message Passing"** which is being submitted to the **Department of information Technology, Jawaharlal Nehru Technological University Hyderabad**, in partial fulfillment of the requirement for the award of degree **Master of Technology** in **Data Sciences, Department of Information Technology,** is a bonafide report of work carried out by me.The material contained in this report has not been submitted to any other University or Institute for theaward of any degree or diploma

KOLTUR YESHWANTH

21031DB011

yeshwanth.kolthuru@gmail.com

ACKNOWLEDGEMENT

I would like to express our sincere thanks to **Dr.A.Jaya Laxmi**, **Principal**, **JNTUH,** for providing theworking facilities in college.

I wish to express our sincere thanks and gratitude to **Dr.K.Santhi Sree, Professor of IT and HOD**, Department of Information Technology for all the timely support and valuable suggestions during theperiod of project.

I am extremely thankful and indebted to my internal guide **Dr.K.Santhi Sree**, **Professor of IT and HOD**, Department of Information Technology, for her constant guidance, encouragement and moral support throughout the project.

I am extremely thankful to **Mr.G.Narsimham, Associate Professor of CSE**, Department of Information Technology, major project coordinators for their encouragement and support throughout the project.

Finally, we would also like to thank all the faculty and staff of Department of Information Technology,who helped us directly or indirectly, for completing this project.

KOLTUR YESHWANTH

21031DB011

S.no	Table of Contents	Pg.no

Fig.no	List of Figures	Pg.no
4.2.3	Representation of Speech	10
4.2.6	Production Model Voice	12
5.1	Architecture	13
5.2(a)	Block diagram of VAD algorithm	14
5.2(b)	Mechanism of Kalman Filter in speech enhancement	18
5.2(c)	Flowchart of the Fast – ICA algorithm	21
5.2(e)	Flowchart of the Principal Component Analysis	25

Table.no	List of Tables	Pg.no
5.3	Speech Sound Dataset	27
8.5(a)	Evaluation metrics of Existing System	58
8.5(b)	Evaluation metrics of Proposed System	58

Image.no	List of Images	Pg.no
5.3.2	Data Pre-processing	27 - 29
8.1	Output 1: Trained Model	48 - 49
8.2	Output 2: Filtering the noise	50 - 57
8.3	Output 3: Iterations	57
8.4	Output 4: Evaluation Metrics	58

List of Abbreviations

S.no	Abbreviations	Full-form
1.	AR	Auto Regressive
2.	ATM	Any Time Money
3.	MSE	Mean Squared Error
4.	MMSE	Minimum Mean Squared Error
5.	SNR	Signal to Noise Ratio
6.	LPCE	Linear Prediction Coefficient Estimation
7.	AMP	Approximate Message Passing
8.	PESQ	Perceptual Evaluation of Speech Quality
9.	VAD	Voice Activity Detection
10.	BSS	Blind Source Separation
11.	ICA	Independent Component Analysis
12.	PCA	Principal Component Analysis
13.	MAE	Mean Absolute Error
14.	RMSE	Root Mean Squared Error
15.	PSNR	Peak Signal to Noise Ratio
16.	KICA	Max – Kurtosis ICA
17.	MVG	Multivariate Gaussian

Abstract

A novel speech improvement technique based on the approximate message passing (AMP) is adopted to get beyond the drawbacks of traditional speech enhancement methods, such as inaccurate voice activity detector (VAD) and noise estimation. To eliminate or muffle the noise from the distorted speech, AMP takes advantage of the difference between speech and noise sparsity. The AMP method is used to effectively rebuild clean speech for speech augmentation. More specifically, the prior probability distribution of the speech sparsity coefficient is represented by a Gaussian model.

The expectation maximization (EM) technique provides excellent learning of the hyper-parameters of the prior model. We use the k-nearest neighbor (k-NN) approach to learn the sparsity while taking into account the correlation between the speech coefficients in neighboring frames. Additionally, computational simulations are used to verify the proposed algorithm, which outperforms the three methods which are Kalman filter, principal component analysis (PCA), independent component analysis (ICA) under a variety of signal to noise ratios and compression ratios (SNRs).

1. Introduction

Speech plays a vital part in our diurnal communication and mortal- machine interfacing. Thus, product and perception of speech have come an intriguing part of the exploration for decades. But the quality and intelligibility of the speech are significantly degraded by background noise, which affects the capability to understand others' speech, causes crimes in mortal Machine Interfacing, etc. In this digital world, it's really hard for any signal in a real- time terrain to escape from noise. This hits us hard when it comes to delivering a communication from one place to another, and there's a need for drawing up or enhancing the communication signal but, at the same time, not giving up any intelligibility of the communication (content, not just clarity). Since speech dispatches have been the mode of communication far and wide, speech improvement is needed whenever the signal comes in contact with the real- time terrain. Modelling the mortal speech product process helps in enhancing speech. But, as speech is a largely nonstationary signal, it is not easy to model the mortal speech product process. Though speech is a largely nonstationary signal, it's stationary for a veritably short period. Grounded on this fact, Classical speech improvement ways are considered for speech member models for a short time, but these short time models don't include the goods of the noise as noise has long- term characteristics.

The AR model is also known to be good for representing unspoken speech. Still, it’s unhappy for raised speech since it’s frequently periodic. This has motivated us to look into speech models that can satisfactorily describe both raised and unspoken speech and allow for the exploitation of long- term noise characteristics. Speech improvement is an area of speech processing where the thing is to ameliorate the intelligibility and niceness of a speech signal. The most common primary ideal of numerous Speech Enhancement algorithms is to ameliorate the perceptual quality of rooting speech signals from noisy speech. Noise estimation is the major element in speech improvement ways because better noise estimation gives a high quality of speech birth. Till now, removing noise from noisy speech has been a gruelling issue because spectral parcels of nonstationary noise are veritably delicate to estimate and prognosticate. Noise estimation is a careful issue in speech improvement algorithms since if the noise power is further than speech power, also that speech content may be removed due to treating that as noise. Due to the wide use of Speech processing in numerous operations like teleconferencing systems, speech recognition- grounded security bias, biomedical signal processing, hearing aids, ATMs, and computers, Speech improvement is a hot exploration area in signal processing.

It remains a gruelling issue because, in utmost cases, only noisy speech is available. Over the once times, experimenters have developed different types of effective algorithms to ameliorate noisy speech indeed though it still poses a challenge to the experimenters because of characteristics of noise signals vary dramatically over time and operation to operation. Numerous speech improvement ways have been proposed using a filtering approach by experimenters last ten times, similar as the spectral deduction system, wiener filtering, Kalman sludge system, etc. Spectral deduction is used for enhancing speech degraded by cumulative stationary background noise. Still, it's affected by musical noise and doesn't remove noise during the silence period. The Wiener sludge- grounded speech improvement system recovers the original signal by minimizing Mean Square Error (MSE) between the clean speech and the estimated signal. Spectral and wiener sludge- grounded speech improvement algorithms bear the characteristics of clean speech. But in real- time, clean speech may not be available in all cases. From the literature study, we set up that some of the ways have been proposed

to enhance speech. Using the harmonious structure of speech signals, a speech signal is recovered from a noisy speech signal; the sinusoidal model is espoused in the MMSE estimator to enhance the speech introduced by Ephraim. At first, The advantage of using the Kalman sludge for speech improvement using estimation of speech signal parameters from clean speech before it's corrupted by white noise is proposed. And further extended to the arbitrary and coloured noises. In these styles, a trade- off should be maintained between SNR and intelligibility. Latterly, numerous changes were made to the Kalman sludge for better enhancement; it didn't meet the prospects or complexity. In this paper, with lower complexity and better performance, a new adaptive Kalman sludge- grounded system with the combination of a nonlinear digital sludge called a digital expander is proposed to recover the speech signal from noisy speech. The cumulative noise is modelled as the AR process grounded on the Kalman filtering algorithm's direct vaticination measure estimation (LPC). In addition to measure estimation, this paper answered the problem ofde-noising the arbitrary and coloured noises. We considered a supposition that the coloured noise is also an autoregressive process. So we estimated its AR portions and dissonances by direct vaticination estimation also. In this paper, to overcome above stated problem, a new adaptive Kalman sludge- grounded system with pre-processing of a digital audio effecting fashion called a digital expander is proposed to recover the speech signal from a sequence (frame) of noisy speech signals, and the cumulative noise is modelled as the AR process. This time- varying bus-accumulative (AR) speech model parameter estimation is grounded on direct vaticination measure estimation (LPC). In addition to measure estimation, this paper answered the problem ofde-noising the coloured noise. We assumed that the noise is also an autoregressive process. So we estimated its AR portions and dissonances by LPC also.

1.1 Objective

The ideal of speech improvement is to ameliorate the quality and intelligibility of speech signals. It's a field of audio signal processing that aims to enhance the speech content within an audio recording while reducing or barring unwanted background noise, deformations, and artefacts. Speech improvement ways are particularly precious in colourful operations where clear and comprehensible speech is essential, similar as

1.1.1 Communication Systems

Enhancing speech in telecommunication, VoIP (Voice over Internet Protocol) calls, video conferencing and other real-time communication platforms can significantly improve the user experience.

1.1.2 Hearing Aids

People with hearing impairments can benefit from speech enhancement to improve speech comprehension in noisy environments.

1.1.3 Voice Assistants

Speech enhancement can improve speech recognition systems' accuracy in voice-controlled devices like smart speakers and virtual assistants.

1.1.4 Audio and Video Recordings

Enhancing speech in audio or video recordings can improve audio quality in podcasts, videos, lectures, and other multimedia content.

1.1.5 Speech Analysis and Transcription

In speech analysis applications, such as automatic speech recognition and speaker identification, speech enhancement can help enhance the accuracy of the system's results.

1.2 Motivation

The motivation behind speech enhancement arises from the fundamental need for clear and intelligible communication in various real-world scenarios. In everyday life, people often encounter challenging acoustic environments with background noise, reverberations, and other distortions that can degrade the quality of speech signals. This degradation can hinder effective communication, especially for individuals with hearing impairments or those relying on voice-controlled devices.

Speech enhancement technologies aim to address these issues using advanced signal processing techniques to improve speech quality and intelligibility. The motivation is to enhance user experiences in communication systems, enable better understanding in noisy environments, aid individuals with hearing difficulties, and enhance the performance of voice-driven applications. Ultimately, speech enhancement seeks to create a more seamless and efficient communication experience for everyone, regardless of the acoustic conditions they encounter.

1.3 Requirements

1.3(a) Hardware Requirements

• Processor Intel Core i5 or original

• RAM 8 GB

• Disk Space 10 GB of free fragment space for MATLAB installation and fresh space for storing data and results

• Display 1024x768 resolution

• Operating System Windows 10 or 11, Linux

1.3(b) Software Requirements

• MATLAB or Python

• Signal processing libraries

• Data prepossessing tools

• Performance evaluation tools

2. Feasibility study

Feasibility refers to the practicality and liability of negotiating a design or task. When assessing the feasibility of a design, we determine whether it's possible to achieve the asked objects within the given constraints, similar as time, coffers, technology, and budget. Feasibility means asking" Can this design be done with the available coffers and constraints, and is it likely to succeed?" On the other hand, if a design is infeasible, it indicates that it isn't practical or attainable given the current circumstances.

Conducting a feasibility study is a common practice to assess the viability of a design before significant time and resources are invested. It helps decision- makers make informed choices by understanding the implicit risks, benefits, and challenges associated with the design. By precisely assessing feasibility, associations can make better- informed opinions and allocate resources effectively to maximize the chances of design success. Feasibility studies give decision- makers with critical perceptivity to make informed choices about whether to do with a design, modify it, or abandon it altogether. Understanding feasibility helps help wasted coffers on unviable systems and increases the liability of successful design issues.

2.1 Types of Feasibility

2.1(a) Technical Feasibility:

Technical feasibility assesses whether the project's objectives can be achieved using the available or readily obtainable technology, tools, and expertise. It involves evaluating the compatibility of the proposed project with existing systems and infrastructure. Factors considered include the availability of necessary hardware, software, skilled personnel, and potential technical constraints or risks. This analysis helps determine if the project can be effectively executed from a technological standpoint, ensuring that the required resources and capabilities are in place to complete the project successfully.

2.1(b) Economic Feasibility:

Economic feasibility examines the financial viability of a project to determine if it is economically justifiable. It involves a cost-benefit analysis, comparing the projected costs and potential benefits or returns the project can generate. Factors such as initial investment, operational costs, revenue streams, and potential cost savings are considered. The goal is to assess whether the project's benefits outweigh the expenses and if it aligns with the organization's financial objectives. Positive economic feasibility indicates that the project is financially sound and likely to deliver favourable returns. At the same time, a negative assessment may lead to reconsideration or modifications to make it economically viable.

2.1(c) Legal Feasibility:

Legal feasibility evaluates whether a proposed project complies with relevant laws, regulations, and legal requirements. It involves thoroughly examining the project's impact on local, regional, and national legal frameworks and adherence to industry-specific standards and ethical guidelines. This assessment ensures that the project operates within the boundaries of the law, minimizes potential legal risks, and maintains a positive reputation. Suppose any legal issues or conflicts are identified during the analysis. In that case, appropriate measures must be taken to address them, such as

obtaining permits and licenses or making necessary modifications to ensure legal compliance before proceeding with the project.

2.1(d) Operational Feasibility:

Operational feasibility evaluates whether a proposed design can be easily integrated into being operations and workflows. It considers how the design will impact the association's labour force, processes, and coffers. The analysis aims to identify any implicit functional challenges, resistance to change, or dislocations that may arise during perpetration. Assessing functional feasibility helps insure the association is able and willing to borrow the design successfully. However, applicable strategies and plans must be developed to address them and maximize the design's chances of flawless integration and successful prosecution, if significant functional hurdles are linked.

3. Literature Survey

3.1 Speech Compressive Sampling Using Approximate Communication Passing and a Markov Chain Prior.

Abstract: Exercising compressive slice (CS), a stingy signal can be efficiently recovered from its far lower samples than that demanded by the Nyquist – Shannon slice theorem. Still, recovering a speech signal from its CS samples is a challenging problem, as it is not stingy enough on any being canonical base. To break this problem, we propose a system combining the approximate communication end (AMP) and Markov chain that exploits the dependence between a speech signal's modified separate cosines transfigure (MDCT) portions. A turbo frame, which alternately iterates AMP and belief propagation along the Markov chain, is employed to reconstruct the speech signal from CS samples. In addition, a constraint is set to the turbo replication to help the new system from divergence. Extensive trials show that, compared to other traditional CS styles, the new system achieves an advanced signal- to- noise rate and an advanced perceptual evaluation of speech quality (PESQ) score. The new system also achieves an analogous speech enhancement effect to the state- of- the- art system.

3.2 Speech Enhancement Using Deep Learning AVE-NET.

Abstract: Utmost approaches to speech enhancement focus solely on audio features to produce adulterants or transfer functions that convert noisy speech signals to clean bones multitudinous speech- related approaches have integrated visual and audio data to achieve further effective speech-processing performance. We propose audio-visual VAE variants for single- channel and speaker-independent speech enhancement in this design. Trials are conducted using the recently released NTCD- TIMIT dataset and the GRID corpus.

3.3 Speech enhancement based on wavelet packet of an improved principal component analysis

Abstract: Our system integrates the capability of PCA tode-correlate the portions by rooting a direct relationship with sea packet analysis to decide point vectors used for speech improvement. This allows us to operate with an accessible loss function on these new portions, removing the noise without demeaning the speech. Also, the enhanced speech attained by the inverse sea packet transfigure is perished into three subspaces low rank, meagre, and the remainder noise factors. Eventually, we calculate the factors as an isolation problem. The performance evaluation shows that our system provides an advanced noise reduction and a lower signal deformation indeed in largely noisy conditions without introducing artefacts.

3.4 Speech Enhancement, Databases, Features, and Classifiers in Automatic Speech Recognition: A Review.

Abstract: This paper gives a comprehensive introduction to speech production and perception in Human beings, discusses the challenges and, applications, framework of Automatic Speech Recognition (ASR) systems, and provides a detailed description of the architecture of ASR. The paper highlights the problem definition, general objectives, and scope of ASR research. The work provides a meticulous review of the baseline ASR systems and indicates the need for research in ASR. A

systematic way of designing a speech corpus for Indian languages is presented in this paper. Finally, wavelet-based Thresholding techniques are utilized to process the designed speech corpus. The results convey that the proposed method performs better for both databases for various SNR conditions.

3.5 Model-Based Speech Enhancement for Time-Varying Noises.

Abstract: Our work introduces a trainable speech improvement fashion that can explicitly incorporate information about speech signals' long- term, time- frequency characteristics before the improvement process. We compare noise spectral magnitude from available recordings from the functional terrain and clean speech from a clean database with fusions of Gaussian pdfs using the Anticipation- Maximization algorithm (EM). Latterly, we apply the Bayesian conclusion frame to the demoralized spectral portions, and by employing Minimum Mean Square Error Estimation (MMSE), we decide an unrestricted- form result for the spectral magnitude estimation task. We estimate our fashion with a focus on real, largely nonstationary noise types (passing- by aircraft noise) and demonstrate its effectiveness at low SNRs.

4. System Analysis

4.1 Existing system

Speech Compressive Sampling Using Approximate Communication Passing and a Markov Chain Prior" focuses on applying compressive seeing ways to acquire and reconstruct speech signals efficiently. The two main ways used in the design are Approximate Communication Passing (AMP) and a Markov Chain prior. In this design, the ideal is to exploit the sparsity in speech signals to efficiently compress and reconstruct them, reducing the quantum of data demanded for transmission and storehouse. Approximate Communication Passing is an iterative algorithm that efficiently solves large- scale optimization problems involved in compressive seeing. It helps to recover the original speech signal from the compressive measures effectively. A Markov Chain prior is employed to capture speech signals' statistical dependences and temporal characteristics.

This previous information aids in perfecting the quality of reconstructed speech and enhances the overall performance of the compressive seeing approach. By combining Approximate Communication Passing with a Markov Chain prior, the design aims to achieve advanced reconstruction delicacy and reduce data accession conditions for speech signals, making it suitable for effective speech communication, voice- controlled bias, and low- bandwidth speech transmission operations. The design's success will probably contribute to advancements in speech processing and effective signal accession, with implicit operations in telecommunications, audio coding, and noise-robust speech communication. Still, it's pivotal to conduct thorough simulations and trials to estimate the performance and effectiveness of the proposed approach and validate its benefits over traditional styles.

4.1.1 Advantages

- Efficient data acquisition with fewer measurements, reducing time and storage requirements.
- Improved signal quality through Approximate Message Passing for accurate reconstruction.
- Sparsity exploitation for efficient representation and transmission of speech signals.
- Noise robustness is achieved by incorporating a Markov Chain before handling noisy environments.
- Versatile application potential in various speech communication and audio processing scenarios.

4.1.2 Disadvantages

- Computational complexity due to iterative Approximate Message Passing algorithms.

- Sensitivity to parameters and prior assumptions, requiring careful tuning for optimal results.
- Calibration challenges and the potential need for specialized hardware and algorithms.
- With high compression ratios, the trade-off between compression and quality leads to potential information loss.
- Limited effectiveness for densely sampled speech signals and dense data.
- Sensitivity to noise and artefacts despite the Markov Chain prior's noise robustness improvement.

4.2 Proposed system

4.2.1 Speech

Speech is the process associated with producing and perceiving the noises used in spoken language. Multitudinous disciplines study speech and speech sounds, including audial, psychology, speech pathology, linguistics, cognitive wisdom, and computer wisdom. Speech product and perception are both important factors of the speech chain.

4.2.2 Speech perception

Speech perception refers to processes by which humans can interpret and understand the sounds used in the language. The study of speech perception is nearly linked to the phonetic field and phonology. The researches about speech have operations in erecting computer systems that can fete speech and ameliorate the recognition for hard- of- hail listeners. There are a lot of natural and cerebral factors which can affect it.

4.2.3 Speech communications

Speech is the primary mortal communication. For that reason, a big trend exists to increase and ameliorate telecommunications. Currently, all the people use communication bias nearly as a primary good telephones, mobiles, and internet. And the guests demand high content and quality. Still, the background noise is an important handicap. It can seriously damage the service quality if it's joined with other deformations. Added to this mortal- mortal commerce, mortal- machine commerce is also grounded on a graphical stoner interface. Still, computers still warrant mortal capacities like speaking, harkening, understanding, and literacy. We live in a noisy world! In all operations (telecommunications, hands-free dispatches, recording, mortal- machine interfaces, etc.) that bear at least one microphone, background noise, and reverberation generally pollute the signal of interest. As a result, the microphone signal has to be" gutted" with digital signal processing tools before it's played out, transmitted, or stored. The signals are generally reused in a digital representation, whereby speech processing can be seen as the crossroad of digital and natural language processing. Speech processing can be divided into the following orders Speech recognition, which deals with assaying the verbal content of a speech signal. Speaker recognition, where the end is to fete the identity of the speaker. Improvement of speech signals (this is the area of this design). Speech rendering, a technical form of data contraction, is important in telecommunication. Voice analysis for medical purposes, similar as analysis of oral lading and dysfunction of the oral cords. Speech conflation is the artificial

conflation of speech, generally computer- generated speech. Speech improvement enhancing the perceptual quality of speech signals by removing the destructive goods of noise, limited Capacity recording outfit, impairments, etc. Speech processing has a lot of operations; one of them could be a ticket deals system by phone, where, without the necessity of an driver, a client can buy tickets with different characteristics and options thanks to word recognition systems.

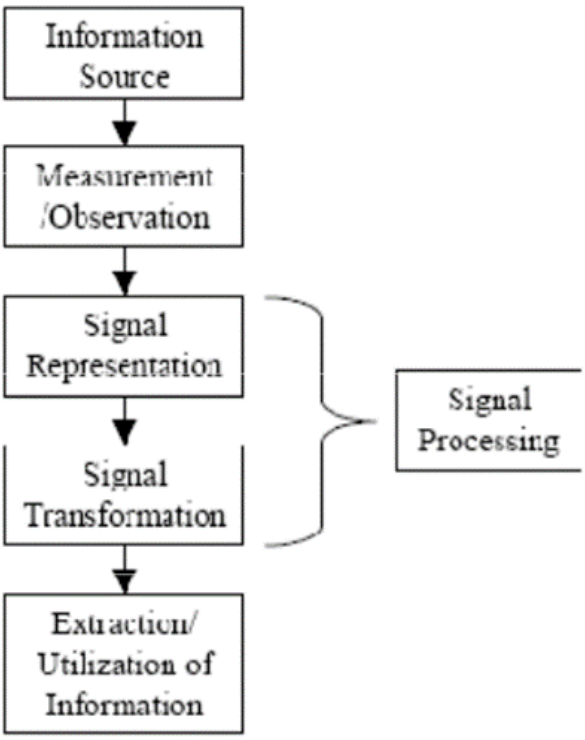


Fig.no: 4.2.3

Fig 4.2.3 Represents the speech ensuring that human customers or computers can easily extract the information.

4.2.4 The acoustical environment

The acoustic environment is a set of transformations that affect the speech signal from the moment it leaves the speaker's mouth until it is in digital form. There are, among others, two main sources of distortion: additive noise and channel distortion. Additive noise is like a fan running in the background, a door slam, or a conversation in our daily lives. It can be stationary or nonstationary. A computer fan or air conditioning makes stationary noise; it has a spectral power density that does not change over time. A signal captured with the speaker close to the microphone has a little noise and reverberation. However, if the microphone is far from the speaker's mouth, it can pick up a lot of noise and reverberation. Channel distortion can be caused by reverberation, the frequency response of a microphone, the presence of an electrical filter in an A/D circuit, the response of the local loop

of a telephone line, a speech codec, etc. If the microphone and the speaker are in an anechoic chamber or free space, the microphone picks up only the direct acoustic path. In practice, in addition to the direct acoustic path, there are repercussions from the walls and other objects in the room. The signal level at the microphone is inversely proportional to the distance from the speaker for the direct path. The sound has to travel a larger distance for the reflected sound waves, and its signal level is proportionally lower. Moreover, we must consider energy absorption each time the sound wave hits a surface.

4.2.5 Speech enhancement

What's speech improvement? Improvement means the enhancement in the value or quality of commodity. When applied to speech, this means perfecting the intelligibility and quality of a demoralized speech signal using signal processing tools. Since this field is abecedarian for exploration in the operations of digital signal processing, it's also of great interest to the assiduity, which is always looking for new results that are both effective and practical. This is a veritably delicate problem for two reasons. First, the nature and characteristics of the noise signals can change dramatically in time and between operations. Chancing algorithms that work in different practical surroundings is also delicate. Second, the performance measure can also be defined else for each operation. Two criteria are frequently used to measure performance quality and intelligibility. It's veritably hard to satisfy both at the same time. Speech improvement is an area of speech processing that aims to ameliorate a speech signal's intelligibility and niceness. The most common approach in speech improvement is noise junking, where we can cancel noise factors and retain only the clean speech signal by estimating noise characteristics. In other words, speech improvement procedures, frequently inadvertently, also loose the speech signal when trying to remove noise. Algorithms must thus compromise between the effectiveness of noise junking and the position of deformation in the speech signal.

4.2.6 Speech modelling

The modelling of speech studies how humans produce the voice. Nowadays, we have a lot of devices that "speak" to us, and this voice should be as similar as possible to a real human voice. Therefore, much research is aimed at finding a good model of speech production Figures.

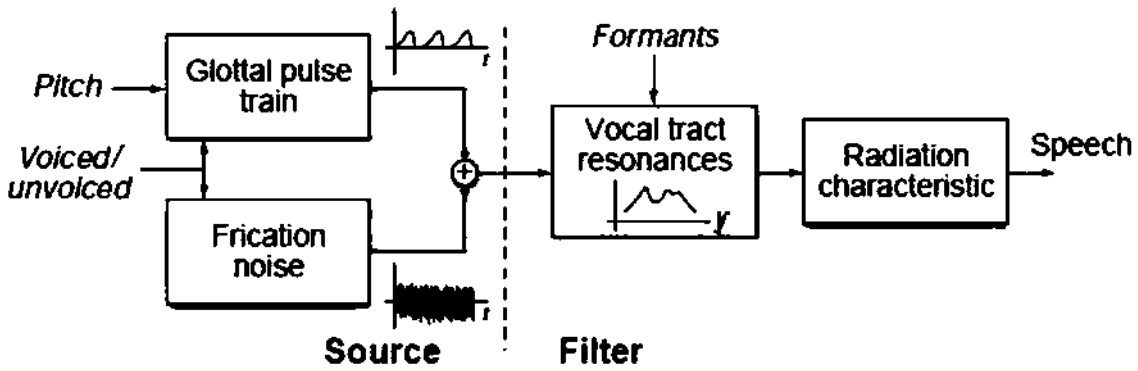


Fig.no.4.2.6: Production model voice

First, with this model, we decide if the noise we want to produce is voiced or unvoiced. For the voiced sounds, we have to model a glottal pulse train similar to the one produced in our vocal cords. The signal produced is like noise for the unvoiced sounds, similar to the signal we can see in the fricative sounds. After that, we must go through the vocal tract with our generated signal. In this section, we filter the signal with a filter that tries to imitate the effect of the shape formed with the pharyngeal cavity (throat), vocal and nasal cavity. Finally, the radiation model reproduces the effect of the radiation impedance that the air puts up to the exit of the speech from the mouth.

4.2.7 Advantages

- Enhanced Speech Quality: AMP algorithms efficiently suppress noise, improving speech quality and intelligibility.
- Real-time Efficiency: AMP enables fast processing suitable for voice communication and voice assistants with reduced computational complexity.
- Adaptability to Noise: AMP-based approaches handle various noise types, making them versatile in different acoustic environments.
- Robust for Sparse and Non-Sparse Speech: AMP performs well for sparse and non-sparse speech signals, ensuring effective enhancement in diverse scenarios.
- Low Resource Requirements: AMP achieves remarkable results with reduced computational complexity, making it suitable for resource-constrained devices and applications.
- Versatility in Noise Handling: AMP-based approaches exhibit adaptability to various noise types, ensuring reliable performance in diverse acoustic environments.

4.2.8 Disadvantages

- Parameter Sensitivity: AMP's performance may rely on parameter tuning for optimal results.
- Suboptimal in Dense Noise: AMP's effectiveness may decrease in high-density noise scenarios, affecting speech enhancement quality.
- Complexity-Performance Trade-off: Parameter adjustments may impact computational complexity.
- Limited Contextual Information: AMP may not fully capture complex temporal dependencies in speech, impacting its handling of certain noise types and artefacts.

5. Design Methodology

5.1 Architecture

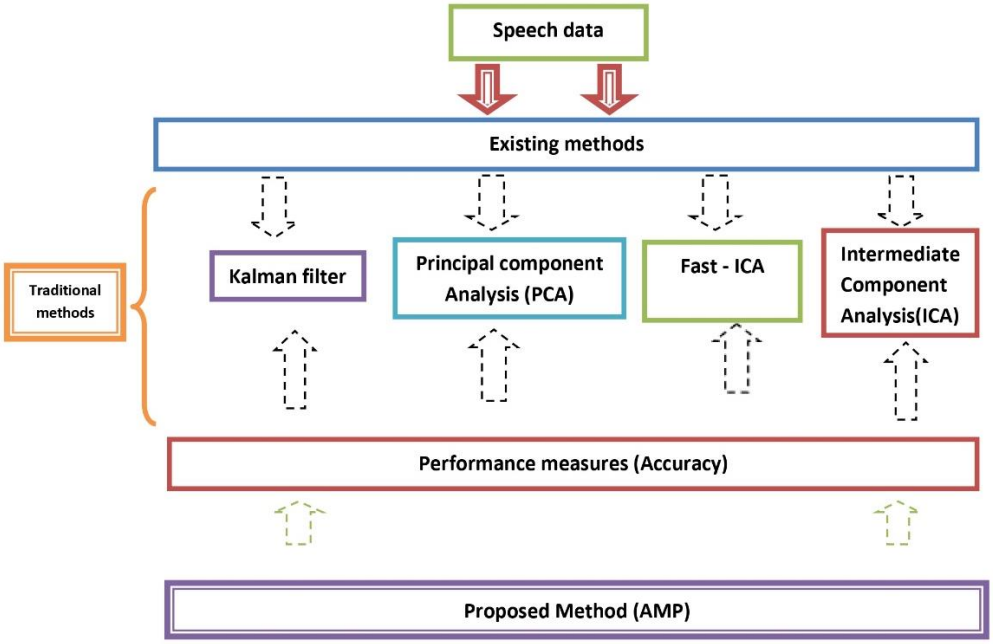


Fig.no. 5.1: Architecture of the proposed system

5.2 Algorithms

5.2(a) VAD algorithm

VAD (Voice Activity Detection) is essential to speech and audio processing systems. It is an algorithm to determine whether an input audio segment contains human speech. The primary goal of VAD is to identify active speech regions in an audio stream while filtering out background noise or non-speech segments.VAD finds applications in various fields, including speech recognition, telecommunication, and audio coding. Here's how the VAD algorithm typically works:

- **Audio Signal:** Receive the input audio signal containing speech and non-speech segments.
- **Pre-processing:** Enhance the audio quality by applying noise reduction, filtering, and signal normalization techniques.
- **Feature Extraction:** Transform the pre-processed audio signal into a suitable feature representation for analysis. Common features include energy, MFCCs, and zero-crossing rate.

- **Frame Segmentation:** Divide the feature sequence into short overlapping frames, assuming speech characteristics remain stationary within each frame.
- **Energy Calculation:** Calculate the energy of each frame based on the selected feature. In energy-based VAD, the frame's energy is the primary determinant.
- **Thresholding:** Apply a threshold to the frame energies to distinguish speech from non-speech frames. Frames with energy above the threshold are considered speech, while those below are labelled non-speech.
- **Voice Activity Detection:** Generate a binary sequence (VAD output) that indicates the presence (1) or absence (0) of speech in each frame.
- **VAD Output:** Obtain the final VAD output, a binary sequence indicating speech presence in each frame of the input audio signal.

Following these steps, the VAD algorithm efficiently identifies speech segments in audio data, making it a crucial component in various speech and audio processing applications.

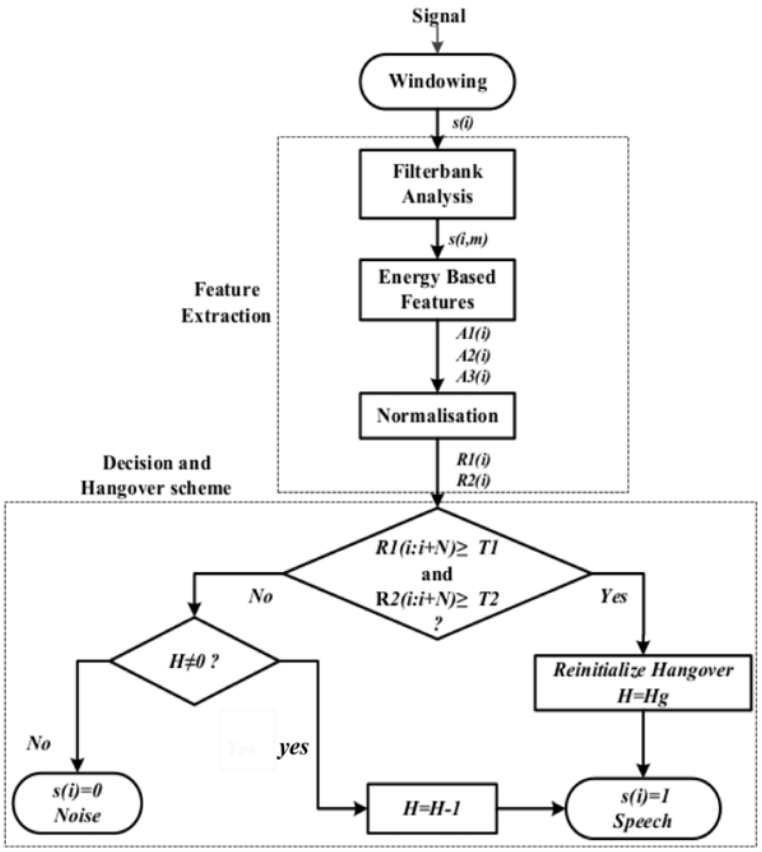


Fig.no.5.2 (a) Block diagram of a VAD Algorithm

, a simple mathematical derivation for a basic Voice Activity Detection algorithm using energy-based thresholding. In this approach, we'll use the energy of the audio signal to detect speech activity.

Step 1: Frame Segmentation

Assume we have an audio signal, x (t), where t represents time. We first divide this continuous audio signal into short overlapping frames, denoted as x[n], with a frame length of N samples and an overlap of M samples. The frame size and overlap are usually chosen empirically, and typical values could be N = 256 and M = 128.

Mathematically, the frames can be expressed as:

$$x[0] = x[0:N-1]\ x[1] = x[M:M+N-1]\ x[2] = x[2M:2M+N-1]\ ...\ x[k] = x[kM:kM+N-1]$$

Step 2: Energy Calculation

Now, we calculate the energy of each frame. The energy, E[k], of a frame x[k] is defined as the sum of squared samples in that frame:

$$E[k] = \Sigma\ |x[k][n]|^2, \text{ for } n = 0 \text{ to } N-1$$

Step 3: Thresholding

We introduce a threshold, T, used to determine whether a frame contains speech. If the energy of a frame E[k] exceeds the threshold, T, we consider it a speech frame; otherwise, it's considered a non-speech frame.

Step 4: Voice Activity Detection Decision

Finally, based on the thresholding, we create a binary sequence V.A.D. [k], where V.A.D. [k] = one if the frame x[k] contains speech and V.A.D. [k] = 0 if it's non-speech.

$$VAD[k] = 1, \text{ if } E[k] > T\ VAD[k] = 0, \text{ if } E[k] \leq T$$

This is an introductory figure of an energy- grounded V.A.D. algorithm. More sophisticated V.A.D. algorithms may use other features, similar as zero- crossing rate, spectral features, or machine literacy styles, for better delicacy and robustness in different noise conditions. The choice of the V.A.D. Algorithm depends on the specific operation and conditions.

5.2(b) Kalman Filtering

Kalman filtering is a considerably used fine fashion for estimating the state of a dynamic system from a series of noisy measures. It provides a recursive and optimal result for the problem of state estimation by incorporating uncertain measures and dynamic system behaviour The Kalman sludge is particularly useful in various fields, including control systems, navigation, robotics, signal processing, and finance.

The way of Kalman filtering and explain each part in farther detail.

- **State Space Model:** The Kalman filter represents a dynamic system using a state vector (x) that contains the system's variables of interest. These variables include positions, velocity, orientation, or other relevant quantities describing the system's state. The state vector evolves based on the system's dynamics, defined by a state transition model (A). The state transition model is often represented as a matrix that relates the state at the current time step to the state at the next time step. It predicts how the system's variables change over time.
- **Measurement Model:** Besides the state space model, the Kalman filter also considers measurements (z) obtained from sensors or measurement devices. These measurements are related to the system's state but are subject to noise and uncertainties. The measurement model includes a measurement matrix (H), another matrix that maps the state space to the measurement space. It defines how the measurements relate to the state variables.
- **Prediction Step:**
- **Time Update (Prediction):** In this step, the Kalman filter predicts the system's state at the next time step (a priori state estimate) based on the current state estimate and the state transition model. It uses the system's dynamics to project the state forward in time.
- **Error Covariance Update:** The Kalman filter also predicts the uncertainty or covariance associated with the state estimate. This covariance matrix captures the uncertainty in the state estimate, including process noise (uncertainty in the system's dynamics) and measurement noise (uncertainty in the sensor measurements). As the system evolves, the uncertainty in the state estimate increases.
- **Correction Step:**
- **Measurement Update (Correction):** In this step, the Kalman filter combines the predicted state estimate with the actual measurements (a posteriori state estimate). The Kalman filter calculates the Kalman gain, a weighting factor that determines the influence of the predicted state and the measurements on the final state estimate. The Kalman gain is calculated based on error and measurement noise covariance. If the measurements are very accurate (low measurement noise), the Kalman gain gives more weight to the measurements; if the measurements are less accurate (high measurement noise), the Kalman gain gives more weight to the predicted state.

The Kalman filter's strength lies in its ability to provide optimal state estimation by continuously updating the state estimate and the error covariance as new measurements become available. It minimizes the mean squared error between the estimated and true states, considering the system's dynamics and the measurement uncertainties. This makes it a powerful tool for real-time state estimation in noisy environments, where accurate and robust estimates are essential. The Kalman filter is widely used in various applications, including control systems, navigation, robotics, and many other fields where accurate state estimation is crucial.

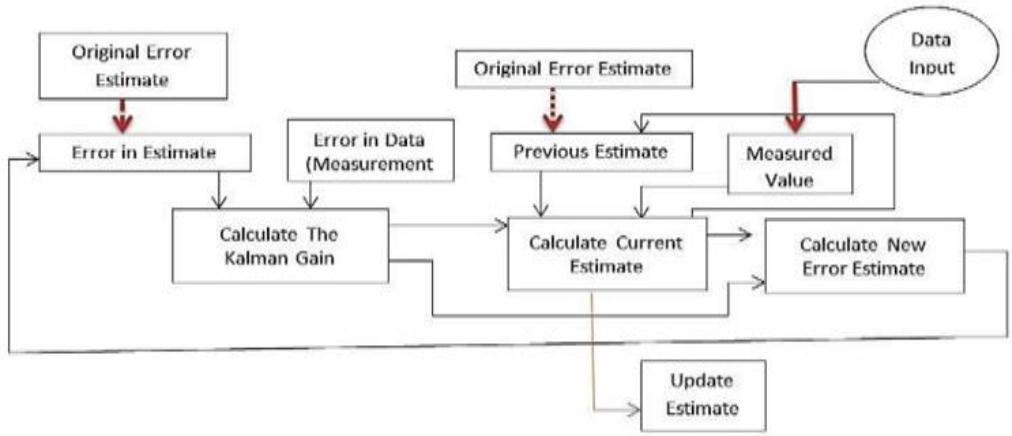


Fig.no.5.2 (b) Mechanism of Kalman filters in speech enhancement

The mathematical derivation of the Kalman filter involves several steps. It is a recursive algorithm that estimates the state of a dynamic system over time by combining prior knowledge and new measurements. Here's a high-level overview of the derivation:

State Space Model: The dynamic system is represented by a state vector, x, containing the system's variables of interest. The state evolves according to a linear or nonlinear state transition model, represented by matrix A. The state transition model predicts the state at the next time step based on the current state.

Mathematically, the state update equation is given by: x (k) = A * x(k-1) + w(k) where:

- A is the state transition matrix.
- X(k-1) is the state vector at the previous time step (k-1).
- w (k) is the process noise, representing uncertainty in the system's dynamics.

Measurement Model: The system's state is not directly observable, but we can obtain measurements, z, related to the state through a linear or nonlinear measurement model represented by matrix H.

Mathematically, the measurement equation is given by: z(k) = H * x(k) + v(k) where:

- H is the measurement matrix.
- v(k) is the measurement noise, representing uncertainty in the sensor measurements.

Prediction Step: We predict the state and its error covariance at the next time step (k+1) based on prior knowledge and the state transition model.

State Prediction: x_hat (k|k-1) = A * x_hat(k-1|k-1)

Error Covariance Prediction: P(k|k-1) = A * P(k-1|k-1) * A^T + Q where:

- x_hat (k|k-1) is the predicted state estimate at time step k based on the knowledge at time step k-1.

- P(k|k-1) is the predicted error covariance matrix at time step k based on the knowledge at time step k-1.
- Q is the process noise covariance matrix, representing the uncertainty in the process noise.

Correction Step: In this step, we incorporate the new measurement (z(k)) into the state prediction to obtain an updated state estimate and error covariance.

Kalman Gain Calculation: K(k) = P(k|k-1) * H^T * (H * P(k|k-1) * H^T + R)^(-1) where:

- K (k) is the Kalman gain at time step k.
- H^T is the transpose of the measurement matrix H.
- R is the measurement noise covariance matrix, representing the uncertainty in the measurement noise.

State Update: x_hat(k|k) = x_hat(k|k-1) + K(k) * (z(k) - H * x_hat(k|k-1))

Error Covariance Update: P(k|k) = (I - K(k) * H) * P(k|k-1) where:

- x_hat (k|k) is the updated state estimate at time step k.
- P (k|k) is the updated error covariance matrix at time step k.
- I is the identity matrix.

The Kalman filter recursively applies these prediction and correction steps as new measurements become available, providing optimal state estimation by minimizing the mean squared error between the estimated state and the true state, considering both process noise and measurement noise.

5.2(c) Fast Independent Component Analysis (Fast – I.C.A.)

Fast Independent Component Analysis (Fast-ICA) is a popular algorithm used for blind source separation (B.S.S.). It separates mixed signals into their original independent components without knowing the mixing matrix beforehand. Fast-ICA is widely used in various fields, including signal processing, image processing, and machine learning.

Here are the key concepts and steps involved in Fast-ICA:

- **Assumptions**: Fast-ICA assumes linear combinations of statistically independent sources in the observed mixed signals and requires the number of sources to be less than or equal to the number of observed mixed signals.
- **Centring the Data**: The algorithm starts by centring the data. For each mixed signal, the mean of that signal is subtracted from each observation, making the data zero-mean.
- **Whitening the Data**: After centring, the data is whitened to make it uncorrelated with unit variance. This step involves applying a whitening transformation, typically achieved using

the eigenvalue decomposition of the covariance matrix. The whitened data simplifies the I.C.A. estimation by decorrelating the mixed signals.

- **Non-Gaussianity Maximization**: The core principle of Fast-ICA is to find a mixing matrix W that, when applied to the whitened data, produces signals that are as statistically independent as possible. The algorithm does this by maximizing the non-Gaussianity of the transformed data. Non-Gaussianity is a measure of how different the distribution of a signal is from a Gaussian distribution, and non-Gaussian signals are more likely to be independent.
- **Contrast Function**: Fast-ICA uses a contrast function to measure the non-Gaussianity of the transformed signals. The most commonly used contrast functions are negentropy and kurtosis. The choice of the contrast function depends on the specific application.
- **Iterative Optimization**: Fast-ICA is an iterative algorithm. It updates the rows of the mixing matrix W to maximize the non-Gaussianity of the transformed data iteratively. Each iteration refines the estimation of the independent components. The algorithm updates W until convergence or a predetermined number of iterations is reached.
- **Deflationary Approach**: Fast-ICA employs a deflationary approach to separate the sources individually. After extracting one independent component, it projects its contribution from the data, leaving behind the remaining mixed signals to focus on separating the next independent component. This process continues until all the independent components are estimated.
- **Choice of Algorithm**: Fast-ICA has different algorithm variations to update the mixing matrix during each iteration. Common variants include the fixed-point iteration method, the symmetric orthogonalization method, and the natural gradient method. The algorithm's choice can impact the fast-ICA process's convergence speed and computational efficiency.

Fast-ICA is a powerful and widely used algorithm for blind source separation, enabling the recovery of statistically independent source signals from mixed observations. Its iterative and deflationary nature, along with the choice of algorithm, makes it an efficient and effective tool for various applications, such as signal processing, image processing, and neuroscience.

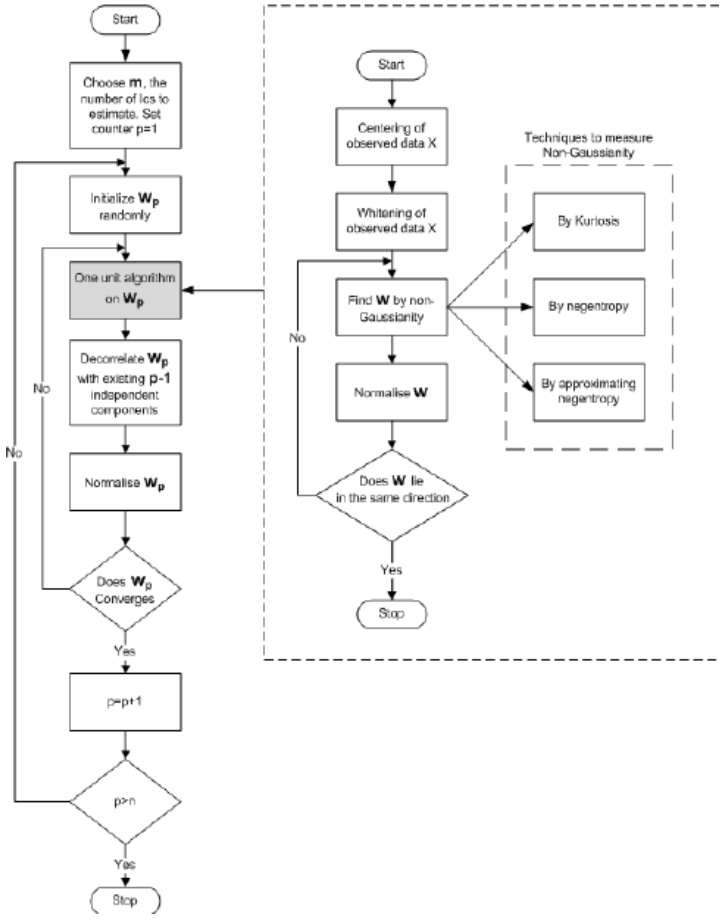


Fig.no.5.2(c) Flow chart of the Fast-ICA algorithm

Fast Independent Component Analysis (Fast-ICA) is an iterative algorithm that aims to find the mixing matrix W that separates a set of mixed signals into their statistically independent components. The main idea behind Fast-ICA is to maximize the non-Gaussianity of the transformed signals, as non-Gaussianity is an indicator of statistical independence. Here's a high-level overview of the mathematical derivations involved in Fast-ICA:

Centring the Data: Given an *m*×*n* matrix X containing *n* observations of *m* mixed signals, the first step is to centre the data by subtracting the mean of each signal from its corresponding observations. The centred data matrix ~*X*~ is computed as follows:

$$\tilde{X}_{ij} = X_{ij} - \frac{1}{n}\sum_{k=1}^{n} X_{ik}$$

Whitening the Data The coming step is to fade the centred data to make it uncorrelated with unit friction. The decolorizing metamorphosis is done using the eigenvalue corruption of the covariance matrix X X T Let ΛΛ be a slant matrix containing the eigenvalues and U be the corresponding matrix of eigenvectors.

The decolorizing metamorphosis is given by

$$Z = U^T \tilde{X}$$

Non-Gaussianity Maximization: Fast-ICA aims to find a linear transformation matrix *A* such that the transformed data *Y=AZ* has statistically independent components. It does this by maximizing the non-Gaussianity of the components.

Contrast Function: Fast-ICA uses a contrast function to measure the non-Gaussianity of the transformed components. The most commonly used contrast functions are negentropy and kurtosis. Let $g(y)$ be the chosen contrast function. The goal is to find *A* that maximizes the sum of the contrast functions applied to the rows of *Y*:

$$J(A) = \sum_{i=1}^{m} g(y_i)$$

Fast-ICA Algorithm: Fast-ICA achieves non-Gaussianity maximization iteratively. The algorithm updates the rows of *A* in each iteration to maximize the contrast function. It converges to the estimated mixing matrix *W* (inverse of *A*), separating the independent components.

- Initialize the rows of *A* randomly or using some other initialization method.
- Iterate until convergence or a predetermined number of iterations:
- Update each row of *A* using the formula:

$$a_i^{(new)} = \frac{E\{g(y_i)z_i\} - E\{g'(y_i)\}a_i}{||a_i^{(old)}||^2}$$

- Orthogonalize the rows of *A* to ensure that the mixing matrix remains orthogonal.

Deflationary Approach: Fast-ICA separates the sources one by one in a deflationary manner. After extracting one independent component, it projects out the contribution of that component from the data and focuses on separating the next independent component from the remaining signals.

Fast-ICA is an efficient and widely used technique for blind source separation and has found applications in various fields like signal processing, image processing, and neuroscience. By exploiting the sources' non-Gaussianity, Fast-ICA can recover statistically independent components from mixed observations even without prior knowledge of the mixing matrix.

5.2(d) Independent Component Analysis (I.C.A.)

It aims to identify and separate the underlying independent sources from a set of observed mixed signals without prior knowledge of the mixing process. I.C.A. finds applications in various fields, including signal processing, image processing, speech recognition, neuroscience, and more.

- **Statistical Independence:** I.C.A. assumes that the sources are statistically independent. In other words, the probability distribution of one source is unrelated to the others.
- **Linear Mixing Model:** I.C.A. assumes that the observed mixed signals are linear combinations of the original independent sources. Mathematically, this can be represented as

$X=AS$, where X is the observed mixed signal matrix, A is the unknown mixing matrix, and S is the independent source matrix.

- **Recovery of Sources:** The main goal of I.C.A. is to estimate the mixing matrix $-1A-1$ and recover the sources S from the observed mixed signals, X.
- **Ambiguities:** I.C.A. has certain inherent ambiguities, such as permutation and scaling ambiguities. These arise because sources' order and scaling cannot be uniquely determined from the mixed signals alone.
- **Contrast Functions:** I.C.A. uses contrast functions, like negentropy or kurtosis, to measure the non-Gaussianity of the sources, as non-Gaussian signals are more likely to be statistically independent.
- **Algorithms:** I.C.A. is typically solved using iterative algorithms like Fast-ICA, Infomax, or fixed-point iteration, which aim to maximize the non-Gaussianity of the sources.

By exploiting the statistical independence of the sources and the linear mixing model, I.C.A. enables the separation of mixed signals into their original independent components. This capability makes it valuable in various applications, such as separating speech from mixed audio, separating image sources, and uncovering hidden patterns in complex datasets.

The mathematical derivation of Independent Component Analysis (I.C.A.) involves finding the mixing matrix A and the independent source matrix S from the observed mixed signals matrix X. The goal is to solve the linear mixing model $X=AS$ and recover the independent sources by maximizing non-Gaussianity. Here's a step-by-step overview of the mathematical derivation:

Linear Mixing Model: Given m observed mixed signals (X) and m unknown sources (S), the linear mixing model is represented as $X=AS$ where:

- X is an $m\times n$ matrix of observed mixed signals (with n time samples).
- A is an $m\times m$ matrix representing the unknown mixing process (mixing matrix).
- S is an $m\times n$ matrix of independent source signals.

Whitening: The first step in I.C.A. is to pre-process the observed mixed signals X to make them uncorrelated and have unit variance. This process is called whitening and involves transforming X into a new matrix Z such that Z has the identity covariance matrix:

$CZ=ZZT=Im$ where:

- It is the $m\times m$ identity matrix.

Orthogonalization: To recover the independent sources S from the whitened data Z, the columns of A must be orthogonal. This is achieved by applying an orthogonalization procedure, such as Gram-Schmidt orthogonalization, to the rows of A iteratively.

Maximization of Non-Gaussianity: I.C.A. aims to maximize the non-Gaussianity of the estimated sources. It does this by applying contrast functions to the elements of S and finding a set of basis vectors (columns of $-1A-1$) that maximize non-Gaussianity. Common contrast functions include negentropy, kurtosis, and skewness.

Iterative Algorithms: I.C.A. is typically solved using iterative algorithms, where the columns of A are updated in each iteration to maximize non-Gaussianity. Fast-ICA is the most widely used algorithm, which employs fixed-point iteration and approximations to achieve faster convergence.

Ambiguities: I.C.A. suffers from certain ambiguities, such as permutation and scaling ambiguities. Permutation ambiguity refers to the fact that the order of the sources cannot be uniquely determined. Scaling ambiguity arises because the amplitude or scaling of each source is not uniquely determined.

Post-processing: Additional constraints or post-processing steps are applied to resolve the ambiguities and obtain unique solutions. These may involve normalization, sorting the sources based on some criterion, or using additional information about the sources.

By iterating through the optimization process and resolving ambiguities, I.C.A. can successfully separate the independent sources from the observed mixed signals, enabling the extraction of meaningful and interpretable components from complex data. It is important to note that the success of I.C.A. relies heavily on the statistical independence assumption of the sources and the linearity of the mixing process.

5.2(e) Principal Component Analysis (P.C.A.)

Principal Component Analysis (P.C.A.) is an extensively used statistical fashion for dimensionality reduction and data contraction. It aims to transfigure a high- dimensional dataset into a lower-dimensional space while conserving the most important information present in the data. P.C.A. finds operations in colourful fields, including data analysis, pattern recognition, image processing, and machine literacy.

- **Centring the Data:** The first step is to centre the data by subtracting the mean of each feature from the corresponding data points. Centring ensures that the principal components are based on variations around the mean.
- **Covariance Matrix Computation:** The covariance matrix is computed from the centred data. The covariance matrix captures the relationships between different features and how they vary.
- **Eigenvalue Decomposition:** P.C.A. proceeds to find the eigenvalues and eigenvectors of the covariance matrix. The eigenvectors are the principal components, and the eigenvalues represent the variance each component explains.

- **Selecting Top Principal Components:** The principal components are ranked based on the magnitude of their corresponding eigenvalues. The top *k* principal components are selected to retain most of the variance in the data.
- **Dimensionality Reduction:** The data is projected onto the *k*-selected principal components, resulting in a lower-dimensional representation of the original data.
- **Data Reconstruction:** To recover an approximation of the original data, the lower-dimensional representation is transformed back into the original data space by reversing the projection.

P.C.A. is a valuable tool for reducing the dimensionality of complex datasets while preserving essential patterns and relationships. By simplifying the data representation, P.C.A. facilitates data analysis, visualization, and machine learning tasks, making it an essential technique in data science.

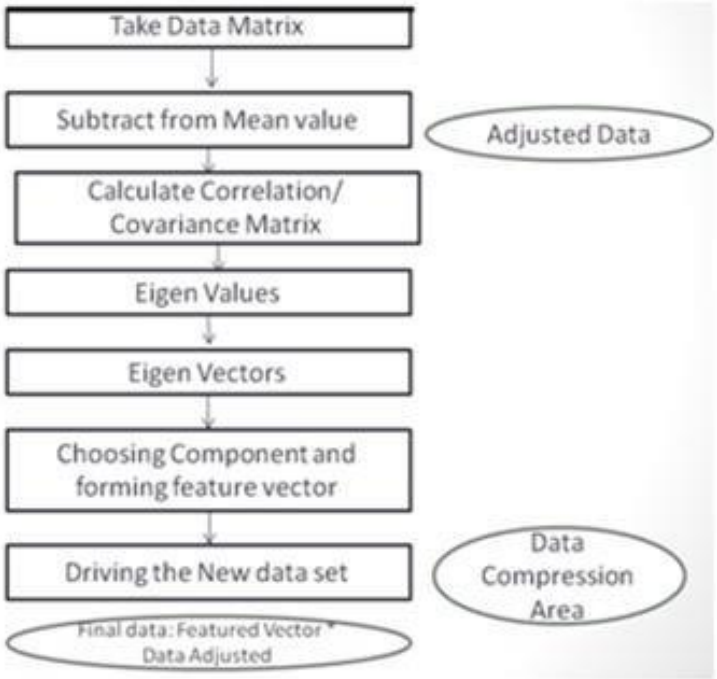


Fig.no.5.2 (e) Flow chart of the principal component analysis

The mathematical derivation of Principal Component Analysis (P.C.A.) involves finding the principal components by performing an eigenvalue decomposition of the data covariance matrix. Here's the step-by-step mathematical derivation of P.C.A.:

Given an $m \times n$ data matrix X with m samples and n features, where each row represents a data point, and each column represents a feature:

Centring the Data: Subtract the mean of each feature from the corresponding data points to centre the data. The centred data matrix is denoted as $\sim X \sim$.

$$X\sim ij = Xij - m1 \sum k=1 mXkj$$

Covariance Matrix: Compute the covariance matrix ΣΣ of the centred data ~*X*~. The covariance between two features *i* and *j* is given by:

$$\text{cov}(X\sim i, X\sim j) = m-11\sum k=1m(X\sim ki - X\sim i^{-})(X\sim kj - X\sim j^{-})$$

Where ¯*X*~*i*¯and ¯*X*~*j*¯are the means of the centred data in the *i*th and *j*th features, respectively.

Eigenvalue Decomposition: Find the eigenvalues *λ*1, λ2… *λn*and corresponding eigenvectors

*v*1, v2…*vn* of the covariance matrix ΣΣ. The eigenvalue equation is given by:

$$\Sigma vi = \lambda ivi$$

Where *vi* is the *i*th eigenvector and *λi* is the corresponding eigenvalue.

Selecting star: Sort the eigenvectors in descending order grounded on their corresponding eigenvalues. The top k eigenvectors are the top factors, where k is the asked reduced dimensionality.

Projection: Project the centred data X onto the k named top factors to gain the lower- dimensional representation Y

$$Y = X\sim Vk$$

Where *Vk* is the matrix containing the top *k* eigenvectors as its columns.

Data Reconstruction: To reconstruct the data from the lower-dimensional representation *Y* back to the original data space, perform the reverse projection:

$$X\hat{} = YVkT$$

Where ^*X*^ is the reconstructed data matrix.

The *k* principal components capture the most significant variability in the data. By selecting *k* appropriately, you can reduce dimensionality while retaining the most important information. The principal components are orthogonal, allowing for a non-redundant and interpretable data representation. P.C.A. is a valuable technique for understanding, visualizing high-dimensional data and improving the performance of machine learning models by reducing the feature space's dimensionality. It is widely used in various fields and has numerous practical applications.

5.3 Dataset

A dataset is a structured collection representing a particular domain or set of observations. In data science and machine learning, datasets are essential for training and evaluating models, conducting analyses, and making data-driven decisions. Datasets can come in various forms, ranging from simple spreadsheets to complex databases, and they play a crucial role in extracting insights and patterns from data.

- In this project, the speech sound dataset is being used from the Git hub.
- This dataset's attributes are audios, file path, start, end segment, hertz, decibels, and sample frequency.
- The starting five rows of this dataset as shown below

S.no	Audio	File path	Start segment	End segment	hertz	decibles	Sample Frequency
1	source1.wav	Click here!!	0	0.5	1895.36 Hz	0.35 dB	8000 Hz
2	source2.wav	Click here!!	0.1	0.3	134.56 Hz	0.49 dB	8000 Hz
3	source3.wav	Click here!!	0.1	0.6	34.88 Hz	0.07 dB	8000 Hz
4	source4.wav	Click here!!	0	0.4	166.88 Hz	2.41 dB	8000 Hz
5	source5.wav	Click here!!	0	0.4	97.44 Hz	0.63 dB	8000 Hz

Table. No. 5.3: speech sound dataset

5.3.1 Attributes of this Dataset

5.3.1(a) Audio

Audio refers to sound waves or signals that the human ear can hear. It can be represented digitally as a sequence of samples, with each sample capturing the audio signal's amplitude at a specific point in time.

5.3.1(b) File path

A file path is a hierarchical representation of the location or address of a file within a file system. It specifies the directories and subdirectories needed to navigate to the file.

5.3.1(c) Start segment

The starting of audio in seconds

5.3.1(d) End Segment

The ending of the audio in seconds

5.3.1(e) Hertz

It is commonly used to measure the frequency of sound, radio, and other periodic phenomena.

5.3.1(f) Decibels

Decibels (dB) are logarithmic units that express the ratio of two values, often used to quantify the intensity or level of sound, electrical power, or signal strength. Decibels describe large and small values, making them suitable for various applications.

5.3.1(g) Sample Frequency

Sample frequency, or sampling rate, is the number of samples captured per second when digitizing an analogue signal to a digital format. It determines the temporal resolution and frequency range that can be accurately represented in the digital signal.

5.3.2 Data Pre-processing

```
speech_sound_dataset = pd.read_csv('/content/speech sound dataset.csv')
```

```
speech_sound_dataset.head()
```

	S.no	Audio	File path	Start segment	End segment	hertz	decibles	Sample Frequency
0	1	source1.wav	Click here!!	0.0	0.5	1895.36 Hz	0.35 dB	8000 Hz
1	2	source2.wav	Click here!!	0.1	0.3	134.56 Hz	0.49 dB	8000 Hz
2	3	source3.wav	Click here!!	0.1	0.6	34.88 Hz	0.07 dB	8000 Hz

```
speech_sound_dataset.shape
```

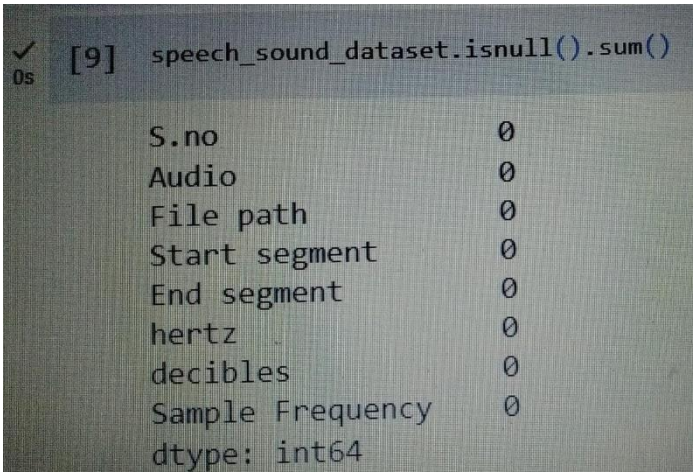
```
(15, 8)
```

Img. no.5.3.2 (a) accessing the dataset through the panda's lib and checking the starting rows of the dataset by the head keyword and checking the shape of the dataset

```
speech_sound_dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   S.no             15 non-null     int64 
 1   Audio            15 non-null     object
 2   File path        15 non-null     object
 3   Start segment    15 non-null     float64
 4   End segment      15 non-null     float64
 5   hertz            15 non-null     object
 6   decibles         15 non-null     object
 7   Sample Frequency  15 non-null     object
dtypes: float64(2), int64(1), object(5)
memory usage: 1.1+ KB
```

Img. no.5.3.2 (b) Checking the information of the dataset to determine whether it contains the null values



Img. no.5.3.2(c) as there are no null values from the above. If the null values exist, the is null keyword helps us to make it null

5.4 Performance metrics

Performance metrics are quantitative measures used to evaluate the effectiveness, accuracy, and efficiency of a system, model, or process. They provide objective insights into the performance and quality of a solution, allowing comparisons between different approaches and aiding in decision-making. Common performance metrics depend on the application but may include accuracy, precision, recall, F1 score, mean squared error, R-squared, area under the receiver operating characteristic curve (AUC-ROC), and execution time.

5.4.1 Mean Squared Error (M.S.E.)

Mean Squared Error (M.S.E.) is a common performance metric used to measure the accuracy of predictions or estimations in regression problems. It quantifies the average squared difference between the predicted values and the actual (ground truth) values.

$$MSE=n1\sum i=1n(yi-y\hat{}i)2$$

Where:

- n is the number of data points (samples).
- Yi represents the target variable's actual (observed) value for the ith data point.
- $y\hat{}I$ represents the predicted value for the ith data point, usually obtained from a regression model.
- The summation is taken over all data points in the dataset.

M.S.E. is a non-negative value, where lower values indicate better performance, meaning the predicted values are closer to the actual values.

5.4.2 Mean Absolute Error (M.A.E.)

Mean Absolute Error (M.A.E.) is another performance metric generally used in retrogression problems to measure the delicacy of prognostications or estimations. It quantifies the average absolute difference between the prognosticated values and the f actual (ground verity) values.

$$|MAE=n1\sum i=1n|yi-y\hat{}i|$$

Where:

- n is the number of data points (samples).
- Yi represents the target variable's actual (observed) value for the ith data point.
- $y\hat{}$I represents the predicted value for the ith data point, usually obtained from a regression model.
- The summation is taken over all data points in the dataset.

M.A.E. is also a non-negative value, and like M.S.E., lower values indicate better performance. Unlike M.S.E., M.A.E. is not sensitive to outliers, as it measures the average absolute deviation from the actual values.

5.4.3 Root Mean Squared Error (R.M.S.E.)

Root Mean Squared Error (R.M.S.E.) is another widely used performance metric in regression problems, particularly when the errors are expected to have significant variations. R.M.S.E. is a measure of the standard deviation of the prediction errors is often used to assess the accuracy of predictive models.

$$RMSE=n1\sum i=1n(yi-y\hat{}i)2$$

Where:

- n is the number of data points (samples).
- Yi represents the target variable's actual (observed) value for the ith data point.
- $y\hat{}$I represents the predicted value for the ith data point, usually obtained from a regression model.
- The summation is taken over all data points in the dataset.

R.M.S.E. is calculated by taking the square root of the average squared difference between predicted and actual values. Like M.S.E., R.M.S.E. is a non-negative value; lower values indicate better performance. R.M.S.E. penalizes large errors more heavily than M.A.E., making it more sensitive to outliers.

R.M.S.E. is often preferred over M.S.E. when the magnitude of the errors is crucial in the evaluation, and it measures how well the model's predictions match the actual data concerning the variability in the target variable.

5.4.4 Peak to Signal Noise Ratio (P.S.N.R.)

Peak Signal-to-Noise Ratio (P.S.N.R.) is a metric commonly used to measure the quality of a reconstructed or compressed image or video. It quantifies the ratio between the maximum possible signal power (the peak signal) and the noise power, measuring how well the reconstruction or compression preserves the original information.

The formula for Peak Signal-to-Noise Ratio (P.S.N.R.) is as follows:

$$PSNR=20\cdot\log10(MSE/MAX_I)$$

Where:

- MAX/MAXI is the maximum possible pixel value of the image (e.g., 255 for an 8-bit grayscale image or 65535 for a 16-bit grayscale image).
- M.S.E.M.S.E. is the Mean Squared Error, calculated as the average of the squared differences between the pixel values of the original and reconstructed (or compressed) images.

P.S.N.R. is expressed in decibels (dB) and provides a relative measure of image quality. Higher P.S.N.R. values indicate better quality, meaning the reconstructed is closer to the original image.

6. Implementation

6.1 Modules of work

6.1.1 Center rows

The provided M.A.T.L.A.B. function center rows take an input matrix Z containing n samples of a d-dimensional random vector. Its goal is to compute the sample mean mu along each feature (column) of Z and then center the data by subtracting the mean from each sample. The function utilizes the mean function to calculate the sample mean and the box fun function for element-wise subtraction.

The centered data Zc is obtained by broadcasting the subtraction operation between the original data Z and the computed mean mu. This pre-processing step is vital for many statistical and machine learning techniques as it removes the data's mean and centers it on zero. The function efficiently returns the centered data Zc, and the sample mean mu, facilitating further analysis or feeding the data into subsequent algorithms. This function allows for easy data manipulation, improving the reliability and accuracy of various data-driven tasks.

6.1.2 Demo Ica

Independent Component Analysis (I.C.A.) on audio signals using various I.C.A. algorithms. The code starts by setting several parameters, such as the number of samples, periods for each signal, signal-to-noise ratio (SNR), and the number of mixed observations and independent/principal components.

Next, the script generates ground truth signals representing different audio waveforms, including sinusoids, square waves, and saw tooth waves. It then adds noise to the ground truth signals to create noisy versions of the signals.

The code generates mixed signals by applying random mixing matrices to the noisy signals, simulating the mixing process in real-world scenarios where sources are mixed to create observed signals.

Three I.C.A. algorithms are then applied to the mixed signals to separate the independent components from the mixed observations. The algorithms used are Fast I.C.A. with the negentropy contrast function, Fast I.C.A. with the kurtosis contrast function, and Max-kurtosis I.C.A. (I.C.A.).

Finally, the script plots and displays the results for visual inspection. It shows the original noisy, mixed signals and separated independent components obtained by each I.C.A. algorithm. The code also includes audio loading, normalization, and audio playback functions.

Overall, this demo script showcases how I.C.A. can separate mixed audio signals into their underlying independent components, enabling the recovery of the sources from a mixture of observations.

6.1.3 Demo Pca1

Principal Component Analysis (P.C.A.) on Gaussian data. P.C.A. is a dimensionality reduction technique that transforms high-dimensional data into a lower-dimensional space while preserving the most important patterns and variances.

The demo starts by setting several parameters, including the number of samples (n), the sample dimension (d), and the number of principal components (r) to retain after P.C.A.Next, Gaussian data is generated with a specified mean (M.U.) and covariance matrix (SIGMA). The Avg function generates multivariate Gaussian data with the given mean and covariance.

The script then performs P.C.A. on the generated data using the P.C.A. function. It calculates the principal components, eigenvalues, and eigenvectors of the data. The variable Zpca stores the data transformed into the principal component space.

The script plots the original Gaussian data samples and scaled eigenvectors in a 2D or 3D space, depending on the dimension of the data. It visualizes the direction and magnitude of the principal components as blue lines. The plot's title indicates the number of principal components (r) used in the P.C.A.

Lastly, the script plots the data transformed into the principal component space (Zpca) in red. For 2D data, it displays a 2D scatter plot, and for 3D data, it shows a 3D scatter plot. The plot highlights how the data is projected onto the selected principal components.

In summary, this demo showcases the application of P.C.A. on Gaussian data, illustrating how it can reduce the data's dimensionality and visualize the principal directions capturing the most significant variations.

6.1.4 Demo Pca2

Applying Principal Component Analysis (P.C.A.) on correlated multivariate Gaussian samples. The primary goal of this demo is to demonstrate how P.C.A. can effectively decompose the input data into orthogonal, maximal variance components while retaining essential information.

The script begins by specifying the number of samples (n), sample dimension (d), and the number of principal components (r) to be retained after the P.C.A. analysis. It then generates Gaussian data with random means (M.U.) and covariance matrices (SIGMA) using the mv function, which generates multivariate Gaussian samples based on the provided mean and covariance. Next, P.C.A. is performed on the generated data using the P.C.A. function. The resulting principal components (Zpca), principal axes (U), and sample mean (mu) are stored for further analysis. The principal components represent

the transformed data in the new orthogonal space, while the principal axes (eigenvectors) indicate the directions of maximal variance.

The script proceeds to visualize the principal components in separate subplots, each showing one principal component as a blithely; the original data samples (Z) and their corresponding reconstructed data (Zr) are plotted in separate subplots. The original data is represented as dashed red lines, while the reconstructed data is shown as dash-dotted blue lines. Each subplot corresponds to one dimension of the data.

The title of the subplot indicates the dimensionality of the P.C.A. approximation (r) and the original data (d). By comparing the original data with the reconstructed data, the effectiveness of P.C.A. in approximating the original data with a reduced number of principal components can be observed.

Overall, this demo exemplifies how P.C.A. can be employed to analyze correlated multivariate Gaussian samples and enables dimensionality reduction while preserving the most relevant information and capturing the significant variance in the data.

6.1.5 Evaluation Metrics

This module is designed to assess and compare the performance of various signal separation algorithms, namely Fast I.C.A., max-kurtosis I.C.A., P.C.A., and the Kalman filter. This module aims to quantitatively evaluate the accuracy and quality of each algorithm's output compared to the ground truth data. Synthetic Gaussian ground truth signals are generated, representing sinusoidal, square, and saw tooth waveforms, and then noise is added to create noisy versions. The mixed observations are obtained by randomly mixing the noisy signals and simulating real-world signal-mixing scenarios. Subsequently, the module applies the mentioned signal separation algorithms to the mixed data, obtaining their respective outputs. The script defines several evaluation metrics, such as mean squared error (M.S.E.), mean absolute error (M.A.E.), root mean squared error (R.M.S.E.), and peak signal-to-noise ratio (P.S.N.R.), to quantitatively measure the discrepancy between each algorithm's results and the ground truth. These metrics allow for a comprehensive comparison of the algorithms' performance, providing insights into which method yields the most accurate and faithful separation of independent components from the mixed observations.

6.1.6 Fast Ica

For performing Independent Component Analysis (I.C.A.) using the Fast I.C.A. algorithm on input data. The Fast I.C.A. algorithm aims to extract independent components from the given data by maximizing either kurtosis or negentropy, depending on the chosen type. The module is designed to handle multidimensional data, allowing the extraction of a specified number of independent components (r) from the input samples.

The implementation begins by parsing the inputs, including the optional type (defaulting to 'kurtosis') and flag (defaulting to 1 for status updates). The data is then centered and whitened using the centerRows and whiten Rows functions. After pre-processing the data, the Fast I.C.A. algorithm iteratively updates the weights (W) to maximize the chosen non-Gaussianity measure, either kurtosis or negentropy. The module also allows displaying iteration status, showing the algorithm's progress. Once the Fast I.C.A. algorithm converges, the independent components (Zica), transformation matrices (W and T), and sample mean (mu) of the input data are returned as outputs.

The Fast I.C.A. algorithm demonstrated in this module is a powerful tool for separating statistically independent components from mixed observations, which is commonly used in various fields such as audio signal processing, image processing, and blind source separation tasks.

6.1.7 KIca

Independent Component Analysis (I.C.A.) using the max-kurtosis I.C.A. algorithm, often referred to as F.I.C.A. The I.C.A. algorithm aims to extract independent components from the input data by maximizing kurtosis, a measure of non-Gaussianity, to find the most non-Gaussian directions in the data.

The implementation begins by centering and whitening the input data using the sample mean (mu) and the inverse square root of the covariance matrix (T). This pre-processing step is crucial to transform the data into a new space where the components are uncorrelated and have unit variance.

After pre-processing, the kICA algorithm computes the principal directions with the highest kurtosis value by performing a singular value decomposition (S.V.D.) on the whitened data. The principal components are obtained from the first 'r' columns of the matrix 'W' obtained through the S.V.D.

The outputs of the module include the 'r'-dimensional independent components (Zica), the transformation matrices (W and T) that map the data into the I.C.A. space, and the sample mean (mu) of the original data.

The max-kurtosis I.C.A. algorithm demonstrated in this module is useful for various applications, such as blind source separation, noise reduction, and feature extraction, where the goal is to identify and isolate statistically independent components from mixed observations.

6.1.8 Load Audio

Loading audio data from the specified file paths and constructing an audio matrix, 'Z,' for further processing. The function can handle both '.wav' files and '.mat' files containing audio data.

The implementation first parses the input 'paths' array to determine the number of audio files to load. Then, it iterates through each file path and loads the Audio data into a cell array, 'audio.' For '.mat' files, it looks for the first variable in the loaded data and extracts the Audio samples from it. For '.wav'

files, it attempts to read the audio data using either 'wavered' (for older M.A.T.L.A.B. versions) or 'audio read' (for newer M.A.T.L.A.B. versions).

After loading the audio data, the module determines the minimum sample length among all audio files (n) to ensure consistency. It constructs the audio matrix 'Z' by truncating or padding the Audio samples from each file to match the minimum sample length. Finally, the audio matrix 'Z' is normalized using the 'normalizeAudio' function, which scales the samples to have a unit norm.

The module's output is the audio matrix 'Z,' which contains the loaded and normalized audio data from the provided file paths. This function simplifies loading audio data and preparing it for subsequent analysis or processing tasks in M.A.T.L.A.B.

6.1.9 Mvg

For generating 'n' samples from a d-dimensional Multivariate Gaussian (M.V.G.) distribution with a given mean vector 'M.U.' and covariance matrix 'SIGMA.' The function can create random samples that follow the specified M.V.G. distribution.

The module first parses the input arguments, 'M.U.,' 'SIGMA,' and 'n.' If the number of samples 'n' is not provided, the default value is set to 1.

The function uses the Cholesky decomposition to generate samples to obtain the lower-triangular matrix 'L' from the covariance matrix 'SIGMA.' It then multiplies this matrix with a set of d-dimensional standard normal random numbers (obtained using 'rand') to create 'n' samples from the M.V.G. distribution. The mean vector 'M.U.' is added to each sample to ensure the samples have the desired mean.

The function's output is a (d x n) matrix 'Z' containing 'n' samples that follow the specified M.V.G.

This module is useful for generating synthetic data conforming to a multivariate Gaussian distribution with a specified mean and covariance, commonly used in statistical simulations and machine learning applications.

6.1.10 Normalize Audio

for normalizing audio data in a matrix 'Z.' The function normalizes each row of the input matrix independently, scaling the values in each row to lie within the range [-1, 1]. The module calculates the maximum absolute value in each row of 'Z' using the 'max' function with ‘abs (Z)' as the input. It then divides each element of 'Z' by the corresponding maximum value using 'be fun' (element-wise binary singleton expansion), ensuring that all values in each row are scaled within the range [-1, 1].

The function's output is the normalized audio matrix 'Zn,' which has the same dimensions as the input matrix 'Z' but with each row scaled to lie within the range [-1, 1].

This normalization process is commonly used in audio signal processing tasks, such as preparing data for audio analysis, feature extraction, or machine learning algorithms, where scaling the audio signals to a common range helps achieve consistent and meaningful results.

6.1.11 Pca

This module is for performing Principal Component Analysis (P.C.A.) on the input data 'Z.' P.C.A. is a widely used technique for dimensionality reduction and data compression by transforming the data into a new coordinate system where the principal components capture the most significant variations in the data.

The function computes the sample mean of 'Z' and centers the data by subtracting the mean from each sample. It then calculates the truncated Singular Value Decomposition (S.V.D.) of the centered data, retaining the 'r' largest singular values and corresponding singular vectors (eigenvectors). These eigenvectors form the principal components of the data.

The function returns the following outputs:

- 'Zpca': An 'r x n' matrix containing the 'r' principal components of the input samples, scaled to have unit variance.
- 'U': A 'd x r' matrix containing the coefficients that represent the principal components in the original data space.
- 'mu': A 'd x 1' vector representing the sample mean of the input data 'Z.'
- 'edges': A 'd x r' matrix containing the scaled eigenvectors of the sample covariance matrix of 'Z.'

The P.C.A. module allows users to compute the principal components of a dataset efficiently. It provides the option to extract the transformation matrix 'U' and scaled eigenvectors 'eigVecs' for further analysis or reconstruction of the original data. P.C.A. is widely used in various fields, such as image and signal processing, data visualization, and feature extraction.

6.1.12 Play Audio

It defines a "Play Audio" class that implements an audio player with a graphical user interface (G.U.I.). This GUI-based audio player allows users to play audio tracks with options for starting, stopping, and resetting playback. The audio player supports multiple audio tracks, each with its own waveform and spectrogram visualization.

The "Play Audio" class includes the following functionalities:

- **Loading Audio Tr**class's constructor, the class takes optional Audio representing a collection of audio tracks. The Audio can be in the form of waveforms or spectrograms.

- **G.U.I. Initialization:** Upon creating an instance of the "PlayAudio" class, it initializes the G.U.I. window with various components, such as buttons, axes for waveform and spectrogram visualization, and a list for selecting different audio tracks.
- **Audio Playback:** The audio tracks can be played using the "Start," "Stop," and "Reset" options provided in the G.U.I. Users can also toggle playback using the spacebar or Enter keys to start/stop playback.
- **Waveform and Spectrogram Visualization:** The audio waveform of the selected track is displayed on one axis, while its corresponding spectrogram is displayed on another axis. The G.U.I. updates the status lines of the waveform and the spectrogram to show the current playback position.
- **Track Selection:** Users can select different Audio tracks from the list provided in the G.U.I. The selected track's waveform and spectrogram will be displayed, and the user can play, stop, or reset playback for the selected track.
- **Closing the Player:** The class also handles the player's closing, including stopping the playback and releasing resources.

The "PlayAudio" class offers a simple yet interactive way to visualize and play multiple audio tracks in real-time. It is useful for audio-related applications such as speech enhancement, sound analysis, and music playback.

6.1.13 Random Mixing Matrix

A function called" random Mixing Matrix" generates a random mixing matrix. In independent element analysis (I.C.A.) and blind source separation (B.S.S.) tasks, a mixing matrix linearly combines independent sources to produce observed fusions. The thing of ICA/ BSS is to recover the sources from the observed fusions using a suitable algorithm. The" random Mixing Matrix" function takes two inputs'd' and'p."d' represents the number of sources (independent factors), and' p' represents the number of observed fusions (detectors). The function generates an arbitrary mixing matrix' A' of size (d x p), where each element of' A' is an arbitrary number between0.25 and 1. The arbitrary figures are generated using the' rand' function in M.A.T.L.A.B. also, the mixing matrix' A' is regularize drow-wise using the' bsxfun' function to insure that the sum of each row is equal to 1. This normalization is necessary to save the overall scale of the sources and fusions. In summary, the" randomMixingMatrix" function creates an arbitrary mixing matrix' A' with arbitrary positive portions, where each row represents the weights assigned to each source in generating the observed fusions. The function returns this mixing matrix' A' as the affair. The performing' A' can be used in colourful ICA/ BSS algorithms to separate the sources from the observed fusions.

6.1.14 Visualize Audio

It defines a function called "visualize Audio" that visualizes an Audio signal in both the time domain (waveform) and the frequency domain (spectrogram). Additionally, the function plays the Audio signal using the sound function in M.A.T.L.A.B.

The "visualize Audio" function takes three inputs: 'y', 'name,' and 'Fs.' 'y' represents the audio signal as a 1D array, 'name' is a string containing the name of the audio signal (used for the figure name), and 'Fs' is the sampling rate of the audio signal (in Hz).

In the function, the time vector is generated based on the length of the audio signal and the given sampling rate 'Fs.' The y-axis limits 'axLim' for both the waveform and the spectrogram plots are set to include a small padding ('Y_PAD') around the audio signal's maximum and minimum amplitude values to ensure the entire waveform is visible.

The spectrogram of the audio signal is computed using the 'spectrogram' function in M.A.T.L.A.B., which calculates the short-time Fourier transform (S.T.F.T.) of the audio signal. The magnitude of the S.T.F.T. is then converted to decibels (dB) to obtain the spectrogram plot.

The audio waveform and spectrogram are plotted in a 2x1 subplot configuration. The waveform plot is shown in the upper and the spectrogram plots in the lower subplot. In contrast, the y-axis of the waveform plot represents the amplitude of the audio signal, and the y-axis of the spectrogram plot represents frequency in Hertz.

The function sets the figure's name to the provided 'name' string using the 'set' function in M.A.T.L.A.B. Finally, the sound function plays the Audio signal, allowing the user to hear the Audio while visualizing it.

In summary, the "visualize Audio" function provides a convenient way to visualize and listen to an Audio signal in the time and frequency domains using M.A.T.L.A.B. plots and sound playback. This can be useful for inspecting audio signals and identifying various characteristics, such as waveform patterns and frequency content.

6.1.15 Whiten Rows

A "whiten Rows" function performs whitening on the input data matrix 'Z.' Whitening is a pre-processing step that transforms the data so that the resulting data has a covariance matrix equal to the identity matrix. This transformation is also known as decorrelation.

The "whiten Rows" function takes a (d x n) matrix 'Z' as input, where'd' is the dimensionality of the random vector, and 'n' is the number of samples of the random vector. The function returns the whitened version of 'Z,' denoted as 'Zw,' and the whitening transformation matrix 'T.'

The whitening process involves the following steps:

- Compute the sample covariance matrix 'R' of the input data 'Z.'
- Perform the Singular Value Decomposition (S.V.D.) of 'R' to obtain the matrix 'U' containing the left singular vectors and the diagonal matrix 'S' containing the singular values.
- Construct the whitening transformation matrix 'T' as the product of 'U,' the inverse square root of the diagonal elements of 'S,' and the transpose of 'U.'
- Compute the whitened data 'Zw' by multiplying 'Z' with the whitening matrix 'T.'

The resulting 'Zw' will have zero covariance between its dimensions, and the covariance matrix of 'Zw' will be the identity matrix, indicating that the variables are uncorrelated and have unit variance.

Whitening is a common pre-processing step in various data analysis tasks, such as Independent Component Analysis (I.C.A.), Principal Component Analysis (P.C.A.), and other methods that assume or benefit from uncorrelated data with unit variance.

In summary, the "whiten Rows" function performs whitening on the input data 'Z' and returns the whitened data 'Zw' and the whitening transformation matrix 'T.' This whitening process is useful for decorrelating and preparing the data for further analysis or modeling.

6.2 The Sample Code

```
rng(42);

% Knobs
n   = 1000;             % # samples
T   = [3, 4, 5];        % # periods for each signal
SNR = 50;               % Signal SNR
d   = 3;                % # mixed observations
r   = 3;                % # independent/principal components

% Generate ground truth
t        = @(n,T) space(0,1,n) * 2 * pi * T;
Ztrue(1,:) = sin(t(n,T(1)));            % Sinusoid
Ztrue(2,:) = sign(sin(t(n,T(2))));      % Square
Ztrue(3,:) = sawtooth(t(n,T(3)));       % Sawtooth

% Add noise
sigma    = @(SNR,X) exp(-SNR / 20) * (norm(X(:)) / sqrt(numel(X)));
Znoisy = Ztrue + sigma(SNR,Ztrue) * rand(size(Ztrue));

% Generate mixed signals
normRows = @(X) bsxfun(@rdivide,X,sum(X,2));
A = normRows(rand(d,3));
Zmixed = A * Znoisy;

% Perform Fast I.C.A.
Zfica = fastICA(Zmixed,r);

% Perform max-kurtosis ICA
Zkica = kICA(Zmixed,r);
```

```
% Perform P.C.A.
Zpca = PCA(Zmixed,r);

%--------------------------------------------------------------------------
% Plot results
%--------------------------------------------------------------------------
cm = hsv(max([3, r, d]));
%cm = linspecer(max([3, r, d]));
 figure();

% Truth
subplot(6,1,1);
for i = 1:3
    plot(Znoisy(i,:),'-','Color',cm(i,:)); hold on;
end
title('Digital Audio Expander');
axis tight;

% Observations
subplot(6,1,2);
for i = 1:d
    plot(Zmixed(i,:),'-','Color',cm(i,:)); hold on;
end
title('Digitally Expanded Noisy Speech Signal');
axis tight;

% Fast I.C.A.
subplot(6,1,3);
for i = 1:r
    plot(Zfica(i,:),'-','Color',cm(i,:)); hold on;
end
title('Noisy Speech frame(only once)');
axis tight;

% Max-kurtosis
subplot(6,1,4);
for i = 1:r
    plot(Zkica(i,:),'-','Color',cm(i,:)); hold on;
end
title('AR Coetticients');
axis tight;
 % P.C.A.
Subplot(6,1,5);
for i = 1:r
    plot(Zpca(i,:),'-','Color',cm(i,:)); hold on;
end
title('Filtered speech Frame');
axis tight;
 subplot(6,1,6);
for i = 1:d
    plot(Zmixed(i,:),'-','Color',cm(i,:)); hold on;
end
title('Pitch Power');
axis tight;
 %--------------------------------------------------------------------------
rng (42);
% Knobs
Fs    = 50000; % Sampling rate of input audio
paths = {'audio/source1.wav','audio/source2.wav','audio/source3.wav'};
%paths = {'audio/voice1.mat','audio/voice2.mat'};
[p, d, r] = deal(numel(paths));
A = randomMixingMatrix(d,p);
%A = [0.6, 0.4; 0.4, 0.6];
```

```
% Load audio
Ztrue = load audio(paths);

% Generate mixed signals
Zmixed = normalize audio(A * Ztrue);

% Fast I.C.A. (negentropy)
Zica1 = normalize audio(fastICA(Zmixed,r,'negentropy'));

% Fast ICA (kurtosis)
Zica2 = normalize audio(fastICA(Zmixed,r,'kurtosis'));

% Max-kurtosis I.C.A.
Zica3 = normalizeAudio(kICA(Zmixed,r));

%--------------------------------------------------------------------------
% Plot results
%--------------------------------------------------------------------------
audio = [];

% True waveforms
for i = 1:p
    audio(end + 1).y = Ztrue(i,:); %#ok
    audio(end).Fs    = Fs;
    audio(end).name  = sprintf('true - %d',i);
end

% Mixed waveforms
for i = 1:d
    audio(end + 1).y = Zmixed(i,:); %#ok
    audio(end).Fs    = Fs;
    audio(end).name  = sprintf('mixed - %d',i);
end

% Fast I.C.A. (negentropy) waveforms
for i = 1:r
    audio(end + 1).y = Zica1(i,:); %#ok
    audio(end).Fs    = Fs;
    audio(end).name  = sprintf(fast ICAA - neg - %d',i);
end

% Fast I.C.A. (kurtosis) waveforms
for i = 1:r
    audio(end + 1).y = Zica2(i,:); %#ok
    audio(end).Fs    = Fs;
    audio(end).name  = sprintf(fast ICAA - kur - %d',i);
end

% Max-kurtosis K.I.C.A. waveforms
for i = 1:r
    audio(end + 1).y = Zica3(i,:); %#ok
    audio(end).Fs    = Fs;
    audio(end).name  = sprintf('kICA - %d',i);
end

% Play audio
PlayAudio(audio);
%--------------------------------------------------------------------------
```

7. Software Environment

7.1 Matlab

Matlab is a high-level programming language and interactive environment primarily used for numerical and scientific computing. It is widely popular in academia, research, and industry for its ease of use and powerful capabilities. Matlab allows users to perform complex mathematical computations, data analysis, visualization, and the creation of graphical user interfaces. It is well-known for its extensive library of built-in functions and toolboxes covering many application areas.

In Matlab, users can create scripts and functions to automate tasks and perform simulations. It supports various data types, including arrays, matrices, and cell arrays, which makes it suitable for handling large datasets and multidimensional data. Matlab also provides excellent plotting and visualization tools, allowing users to create publication-quality figures and charts.

The language's syntax is straightforward, making it accessible to beginners and experienced programmers. Its interactive environment allows for rapid prototyping and experimentation. Matlab is especially well-suited for solving mathematical problems, linear algebra, optimization, and differential equations.

Matlab is commonly used in engineering, physics, mathematics, finance, biology, and signal processing. It is known for its powerful numerical libraries and efficient handling of large datasets. Additionally, Matlab has a supportive community, providing access to numerous resources, forums, and documentation.

Matlab's computational power, user-friendly interface, and extensive capabilities have made it a go-to tool for researchers, engineers, and scientists in various domains.

7.1.1 Features of Matlab

- **Toolboxes and Libraries:** Matlab offers a vast collection of toolboxes and libraries that extend its functionality. These toolboxes cover various areas such as image processing, control systems, signal processing, machine learning, optimization, statistics, and more. They provide specialized functions and algorithms tailored to specific application domains.
- **Graphics and Visualization:** Matlab provides powerful and flexible plotting and visualization tools. Users can create 2D and 3D plots, histograms, contour plots, surface plots, and animations. The graphics capabilities allow customization of plot appearance, labels, and annotations.

- **Symbolic Computing:** Matlab's Symbolic Math Toolbox enables symbolic computation, allowing users to work with symbolic expressions, perform algebraic manipulations, solve equations symbolically, and find derivatives and integrals.
- **App Building:** Matlab includes App Designer, a graphical development environment that enables users to build interactive graphical user interfaces (G.U.I.s) without writing code. This feature helps create user-friendly applications.
- **Parallel Computing:** Matlab supports parallel computing, allowing users to accelerate their computations using multiple processors or cores. Parallel computing is beneficial for handling large datasets and speeding up complex simulations.
- **Integration with Hardware:** Matlab can interface with various hardware devices, such as sensors, cameras, and microcontrollers, through support packages, enabling users to acquire data and control external hardware directly from Matlab.
- **MAT-File Format:** Matlab uses the .mat file format to store data, variables, and objects. M.A.T. files provide a convenient way to save and load data, preserving the variable structures and properties.
- **Data Import and Export:** Matlab supports reading and writing data in various formats, including CSV, Excel, HDF5, XML, JSON, and more. This makes it easy to work with data from different sources.
- **Debugging and Profiling:** Matlab offers debugging tools to identify and fix errors in code. It also provides profiling tools to analyze code performance and optimize its execution.
- **Integration with Other Languages:** Matlab can be integrated with other programming languages like C, C++, Java, and Python, enabling users to call external functions or use Matlab code within other applications.
- **Online Collaboration:** Matlab supports collaborative development through tools like Matlab Online and Git integration, allowing multiple users to work on the same project simultaneously.

These features and its extensive documentation and active community support have solidified Matlab's position as a leading software tool for numerical computation and data analysis across various scientific and engineering disciplines.

7.2 Packages used in Matlab

In M.A.T.L.A.B., packages are called "Toolboxes." A toolbox is a collection of functions and classes that extends M.A.T.L.A.B.'s core capabilities to solve problems or work with specific data types. M.A.T.L.A.B. provides a range of official toolboxes developed by MathWorks and third-party toolboxes developed by other companies and individual developers. Here are some commonly used official toolboxes provided by MathWorks:

- **Statistics and Machine Learning Toolbox:** Provides functions and tools for statistical analysis, machine learning, data pre-processing, and predictive modelling.
- **Image Processing Toolbox:** Offers functions for image analysis, enhancement, filtering, and feature extraction.
- **Signal Processing Toolbox:** Contains functions for signal analysis, filtering, spectral analysis, and time-frequency analysis.
- **Control System Toolbox:** Provides tools for designing and analysing control systems, including transfer functions, state-space models, and PID controllers.
- **Optimization Toolbox:** Offers functions for nonlinear optimization, linear programming, and global optimization.
- **Curve Fitting Toolbox:** Contains functions for fitting curves and surfaces to data points.
- **Symbolic Math Toolbox:** Enables symbolic computation, algebraic manipulations, and solving of equations symbolically.
- **Partial Differential Equation Toolbox:** Provides tools for solving partial differential equations numerically.
- **Simscape:** Enables modelling and simulation of physical systems using block diagrams.
- **Simulink:** A graphical simulation and model-based design environment for multidomain dynamic and embedded systems.
- **Neural Network Toolbox:** Contains functions for designing, training, and simulating neural networks.
- **Communications Toolbox:** Provides functions for designing and simulating communication systems and digital signal processing.
- **Computer Vision Toolbox:** Contains functions for computer vision, object detection, and image feature extraction.
- **Robotics System Toolbox:** Enables designing and simulating robotic systems and algorithms.
- **Audio Toolbox:** Provides tools for audio processing, synthesis, and analysis.
- **Data Acquisition Toolbox:** Enables interfacing with data acquisition hardware and sensors.
- **Wavelet Toolbox:** Provides functions for wavelet analysis and transform, including denoising and compression.
- **Bioinformatics Toolbox:** Offers tools for analysing and visualizing biological data, including genomics and proteomics.
- **Mapping Toolbox:** Provides functions for geospatial data analysis, mapping, and visualization.

- **Deep Learning Toolbox (previously Neural Network Toolbox):** Contains functions for deep learning, including neural network architectures and training algorithms.

These are just a few examples of the extensive range of toolboxes available in MATLAB. The MATLAB ecosystem is vast, and developers create and share new toolboxes to cater to specific research and application needs. Users can find and install third-party toolboxes from the MATLAB File Exchange or other online repositories, expanding the capabilities of MATLAB for various domains and applications.

7.3 External Libraries used in Matlab

In MATLAB, external libraries refer to code written in other programming languages (e.g., C, C++, and Java) that can be integrated and used within MATLAB. This integration allows you to leverage the capabilities of those external libraries from within the MATLAB environment, combining the strengths of both languages.

There are a few different ways to use external libraries in MATLAB:

- **MEX files:** MEX files (short for "MATLAB Executables") are functions written in C, C++, or Fortran that have been compiled into a binary format that MATLAB can directly call. This allows you to accelerate computationally intensive tasks or use existing C/C++ code directly within MATLAB.
- **MATLAB Engine:** The MATLAB Engine API allows you to execute MATLAB functions and scripts from C/C++ or Java code. This enables you to control MATLAB from external applications and pass data between MATLAB and the external code.
- **Java Libraries:** MATLAB allows you to use Java classes and methods by importing Java libraries into your MATLAB code. This way, you can take advantage of Java-based libraries and APIs within the MATLAB environment.
- **.NET Libraries:** MATLAB provides integration with Microsoft's .NET framework, allowing you to access .NET libraries and use .NET functionality in MATLAB.
- **Python Integration:** MATLAB provides support for integrating Python code and libraries into MATLAB scripts and functions. This capability is made possible through the "MATLAB Engine for Python," which allows you to call Python functions from MATLAB and exchange data between the two environments. This feature allows you to leverage Python's extensive ecosystem of libraries and tools from within MATLAB.
- **Calling Shared Libraries:** MATLAB enables you to call functions from shared libraries (also known as dynamic link libraries or DLLs) written in C or C++. This is done using load library function and allows you to access external functions in DLLs directly from MATLAB.

- **External Code Interfaces:** In addition to the methods mentioned above, MATLAB offers various interfaces and tools to help you interact with external code effectively. For example, the coder functionality in MATLAB allows you to generate C/C++ code from MATLAB code, which can then be incorporated into your C/C++ projects or used as MEX files.
- **Open Source Libraries:** MATLAB users can also take advantage of numerous open-source libraries and packages available in languages like C, C++, Python, and Java. Many open-source projects can be integrated with MATLAB using the methods mentioned earlier, expanding the range of functionalities available to MATLAB users.
- **Compiler Support:** MATLAB provides options for creating standalone applications or executable files that bundle your MATLAB code and any used external libraries together. This way, you can distribute your MATLAB applications to others without requiring them to have MATLAB installed.

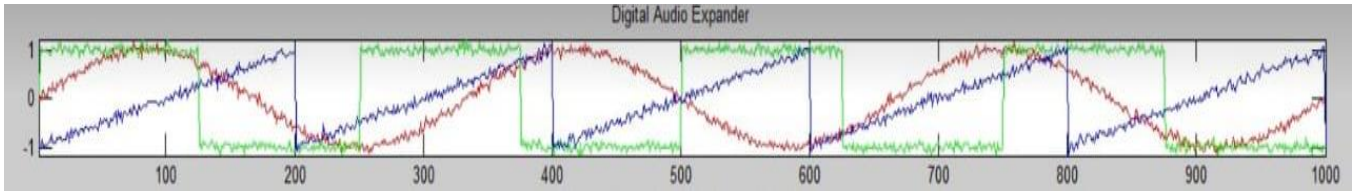
It's important to note that while integrating external libraries can be powerful, it might also introduce some complexities, such as potential compatibility issues between different versions of MATLAB or external libraries. Therefore, when using external libraries, it's essential to consider proper version management and testing to ensure smooth integration and functioning. Additionally, documentation and examples provided by MathWorks can be helpful resources when working with external libraries in MATLAB.

8. Experimental Results

8.1 Output 1: Trained Model

8.1(a) Digital Audio Expander

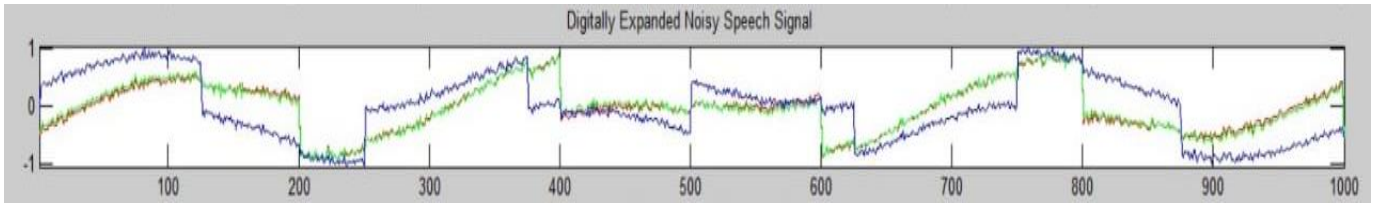
A digital audio expander is an audio processing technique used to increase the dynamic range of an audio signal. It boosts low-level signals while preserving the integrity of louder parts, resulting in enhanced sound characteristics and improved audibility in quieter portions of the audio. It helps to bring out subtle details in the audio and reduce the perception of background noise.



Img.no.8.1 (a) Digital audio expander

8.1(b) digitally expanded noisy speech

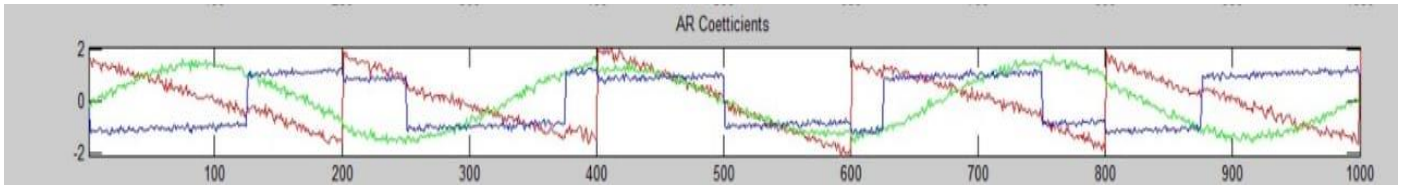
Digitally expanded noisy speech refers to the application of a digital audio expander to enhance the clarity and audibility of speech signals corrupted by noise. It selectively amplifies low-level speech components, making the speech more discernible and improving intelligibility in noisy environments.



Img.no.8.1 (b) digitally expanded noisy speech signal

8.1(c) AR coefficients

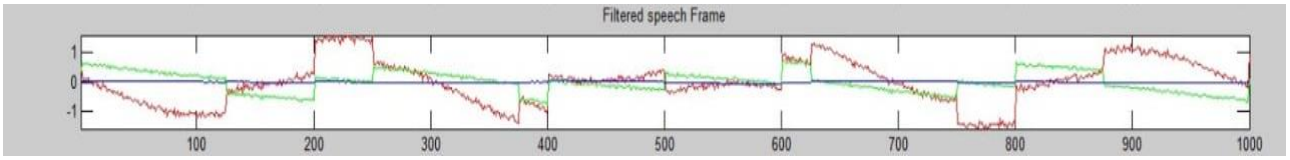
AR coefficients are parameters in an autoregressive model used to predict a time series data point based on its previous values; they represent the weights assigned to past observations in the linear combination that defines the model.



Img.no.8.1(c) AR coefficients

8.1(d) Filtered Speech Frame

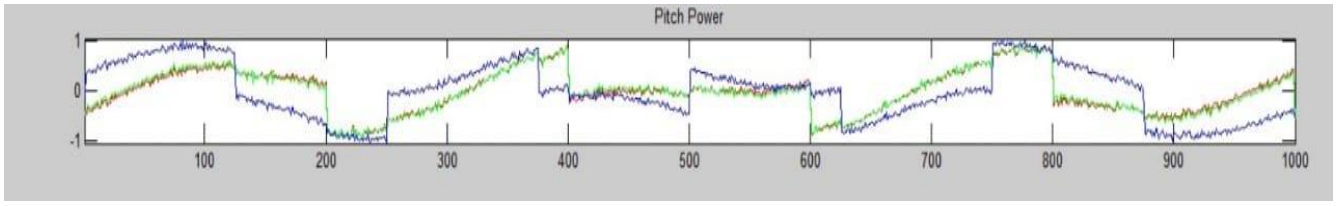
A filtered speech frame refers to a segment of speech that has undergone a filtering process to remove unwanted components, such as noise or distortion, and enhance specific characteristics, resulting in improved speech quality and intelligibility. Filtering can be achieved using various techniques, such as adaptive filters, spectral subtraction, or digital audio effects.



Img.no.8.1 (d) Filtered speech frame

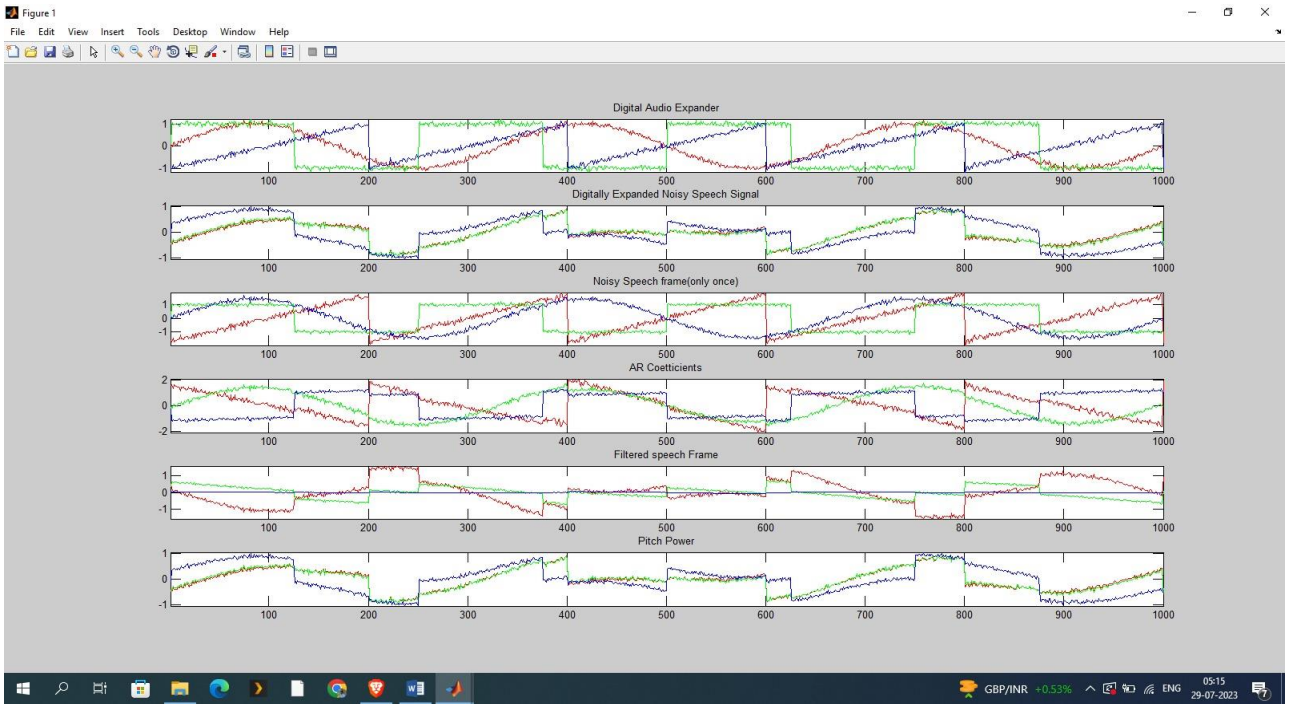
8.1(e) Pitch power

Pitch power, also known as pitch strength, is a measure used in speech analysis to quantify the perceived pitch strength or prominence of the fundamental frequency in a speech signal.



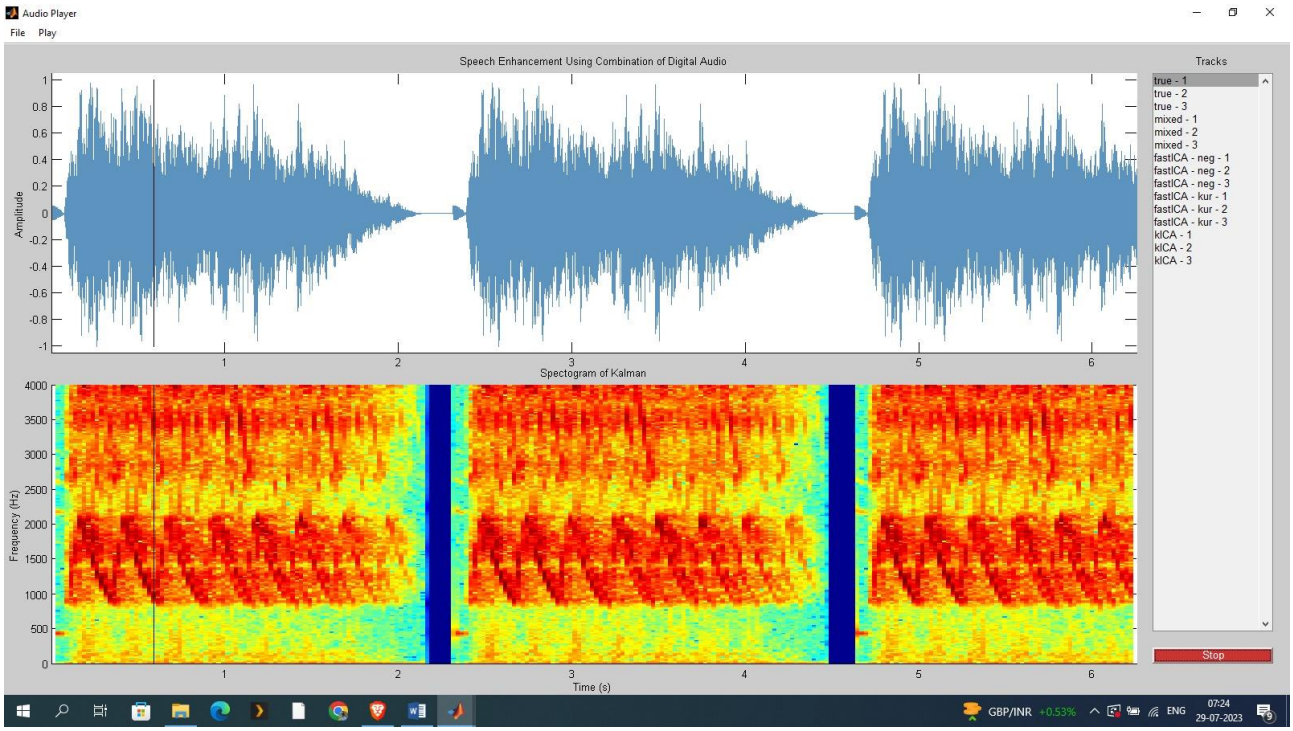
Img.no.8.1 (e) Pitch power

8.1(f) Comparison between audio processing techniques

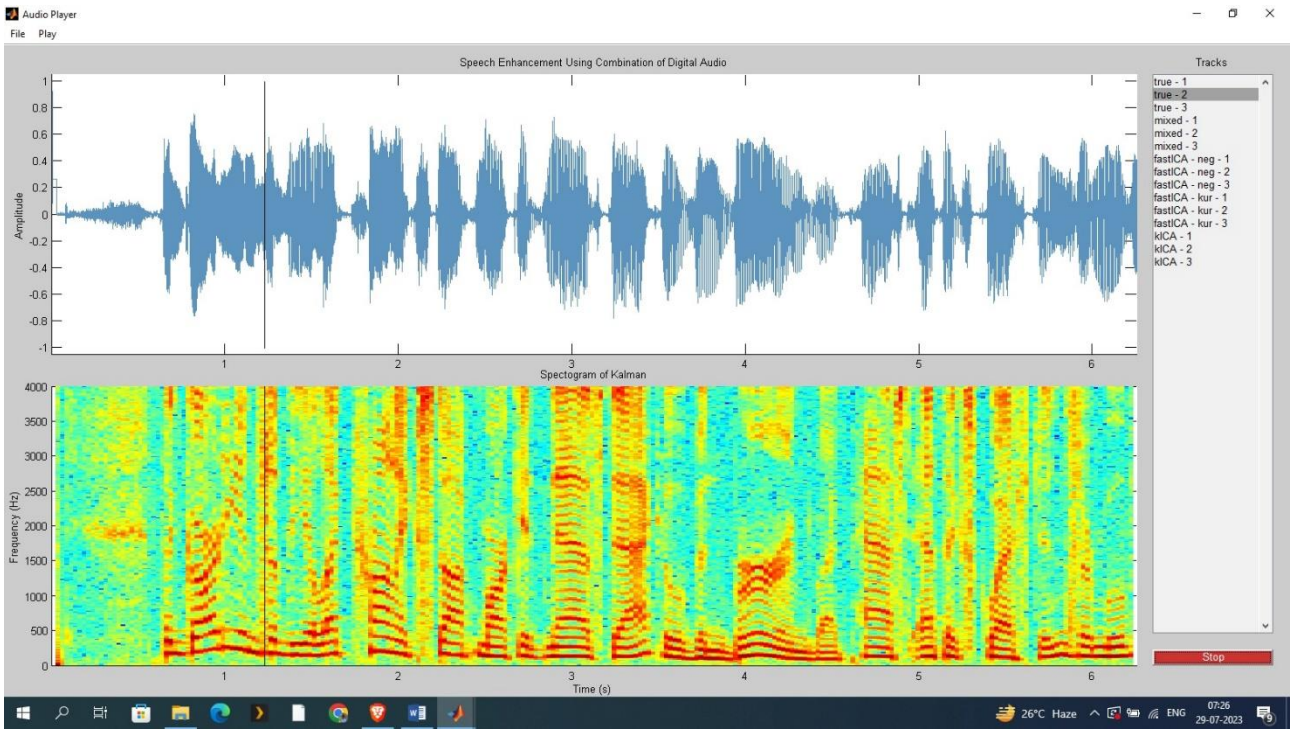


Img.no.8.1 (f) comparison between the audio processing techniques

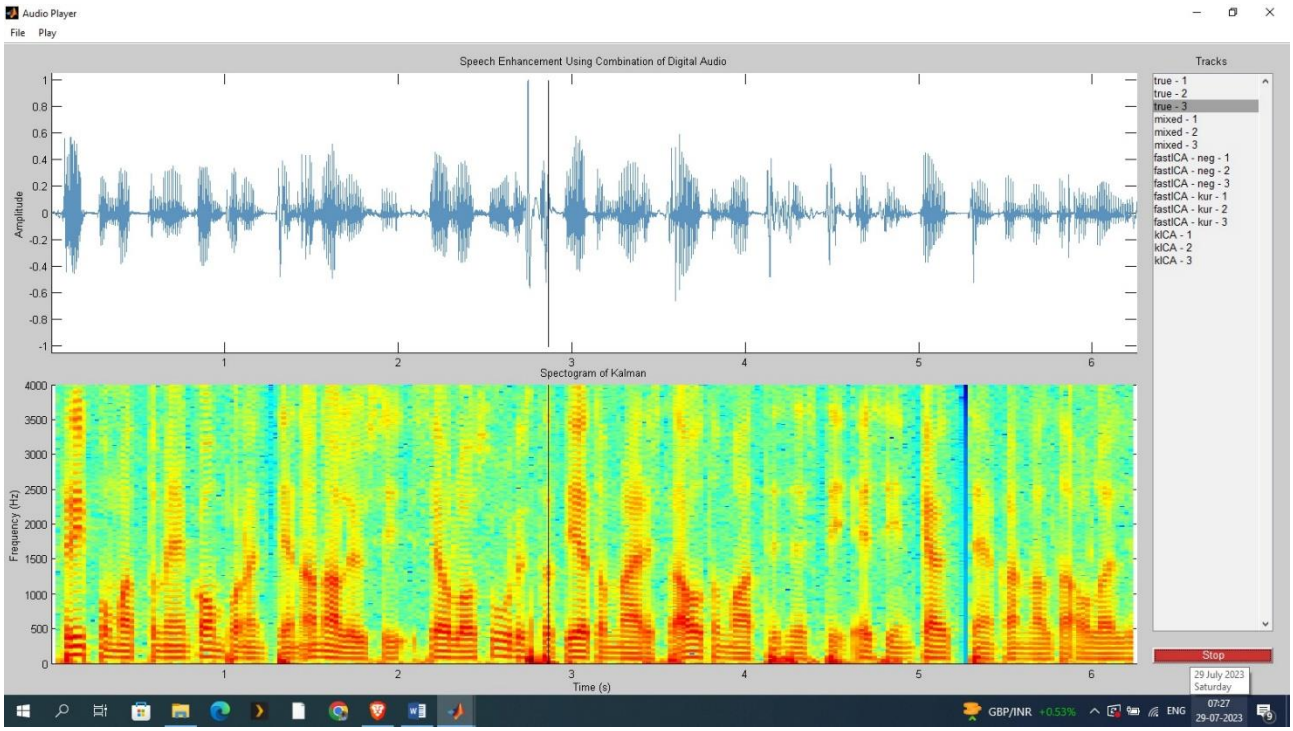
8.2 Output 2: Filtering the noise



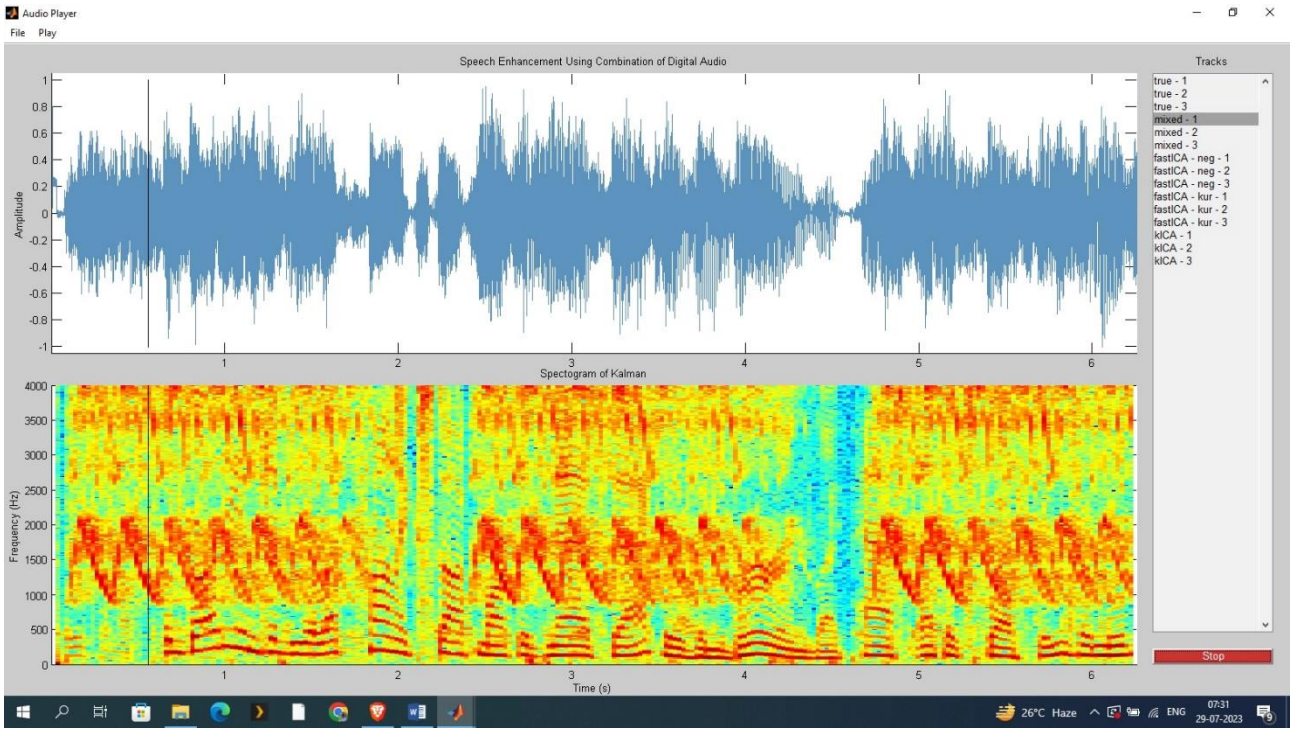
Img.no.8.2 (a) the true noise of the audio1



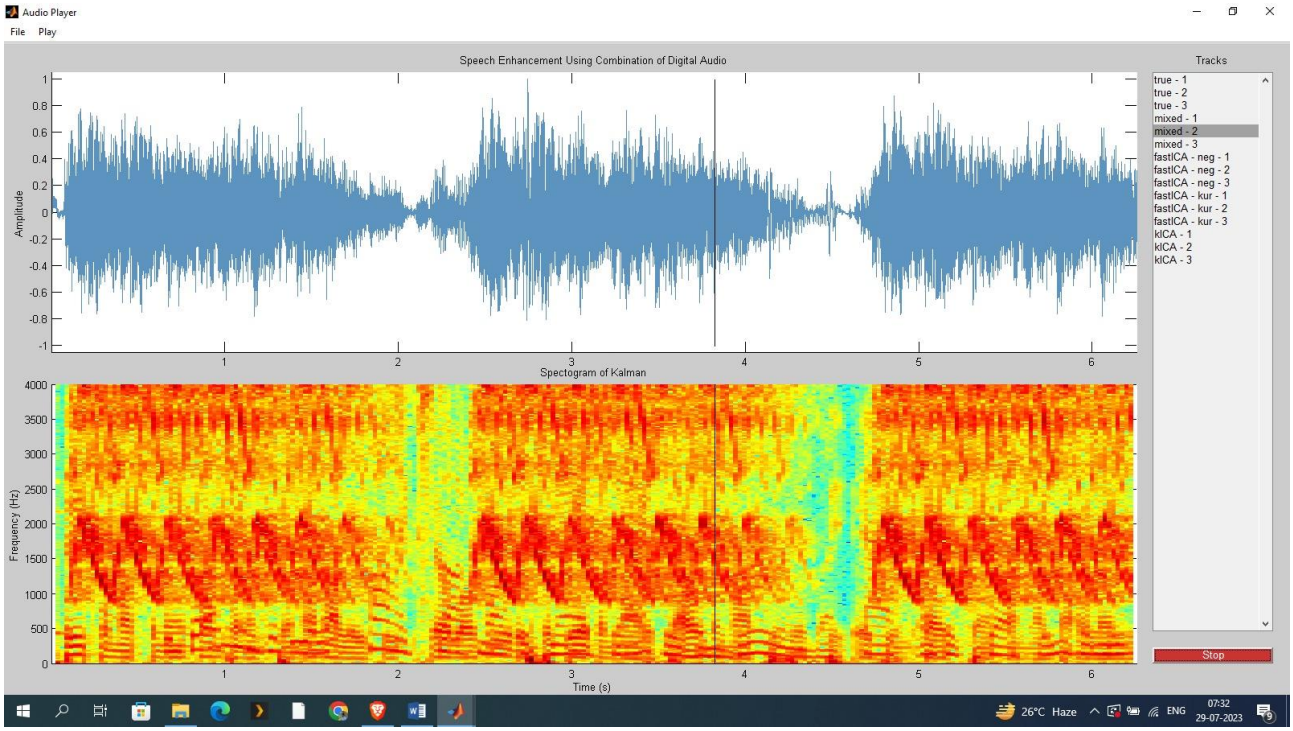
Img.no.8.2 (b) the true noise of the audio2



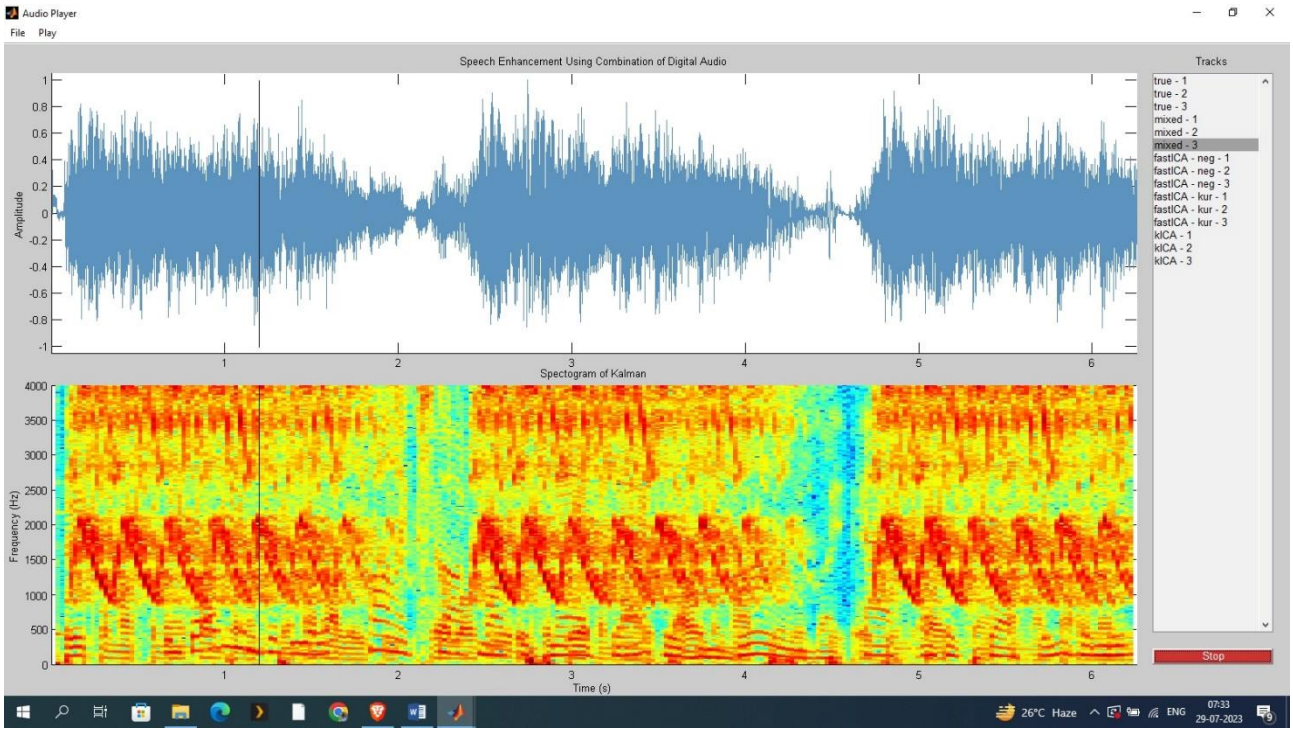
Img.no.8.2(c) the true noise of the audio3



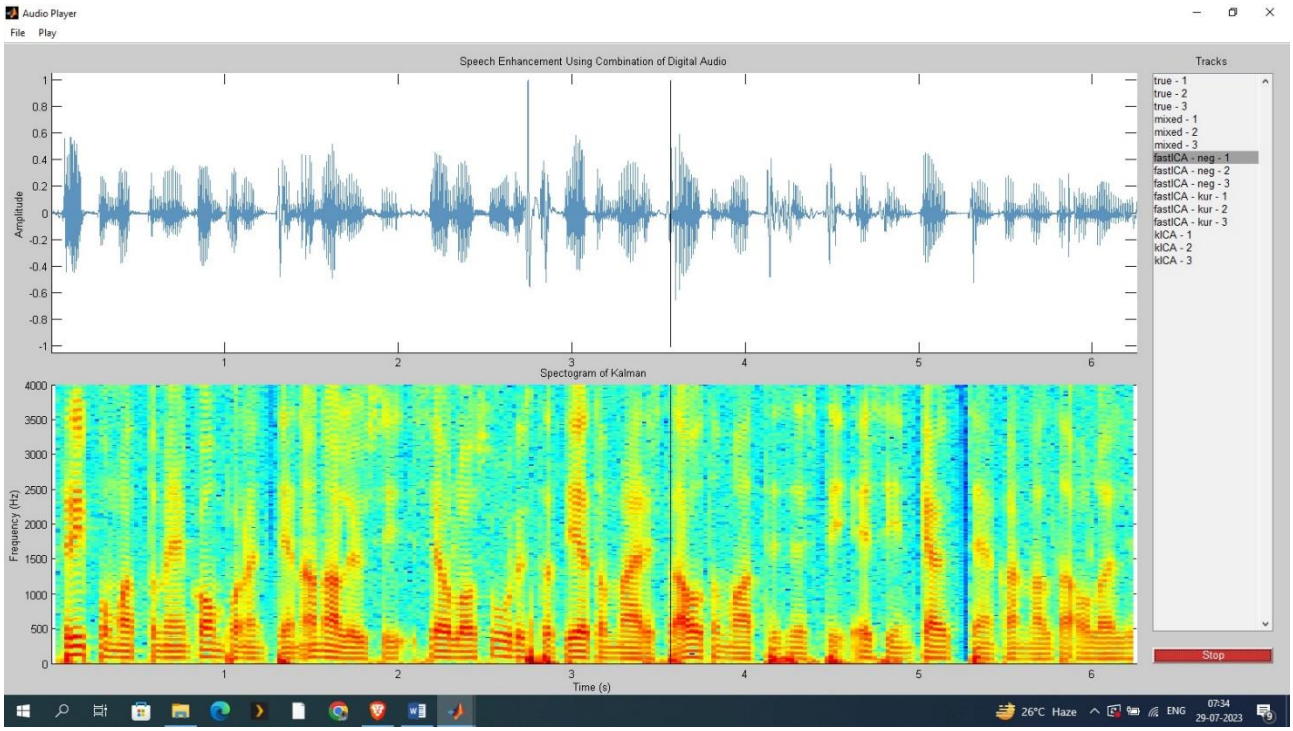
Img.no.8.2 (d) it mixed noise of the audio1, audio2 and audio3 which has more noise of the audio1



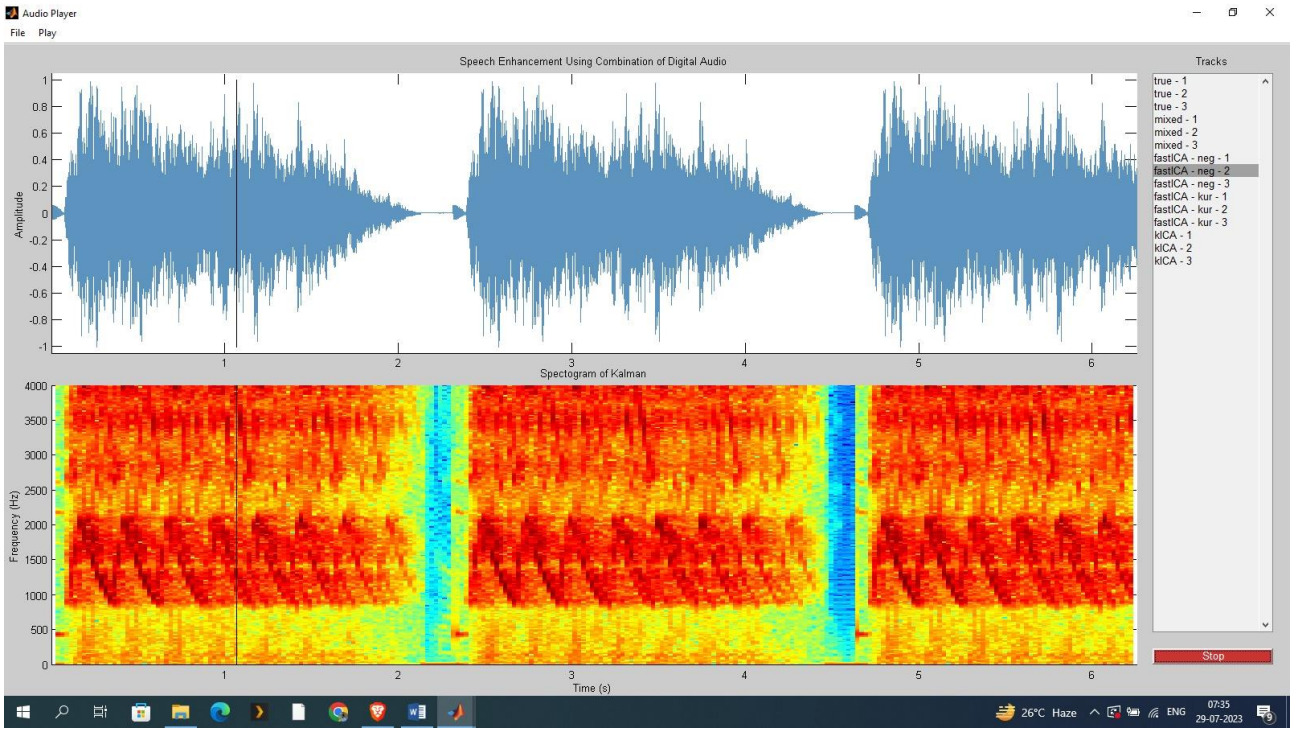
Img.no.8.2 (e) it has mixed noise of the audio1, audio2 and audio3 which has the more noise of the audio2



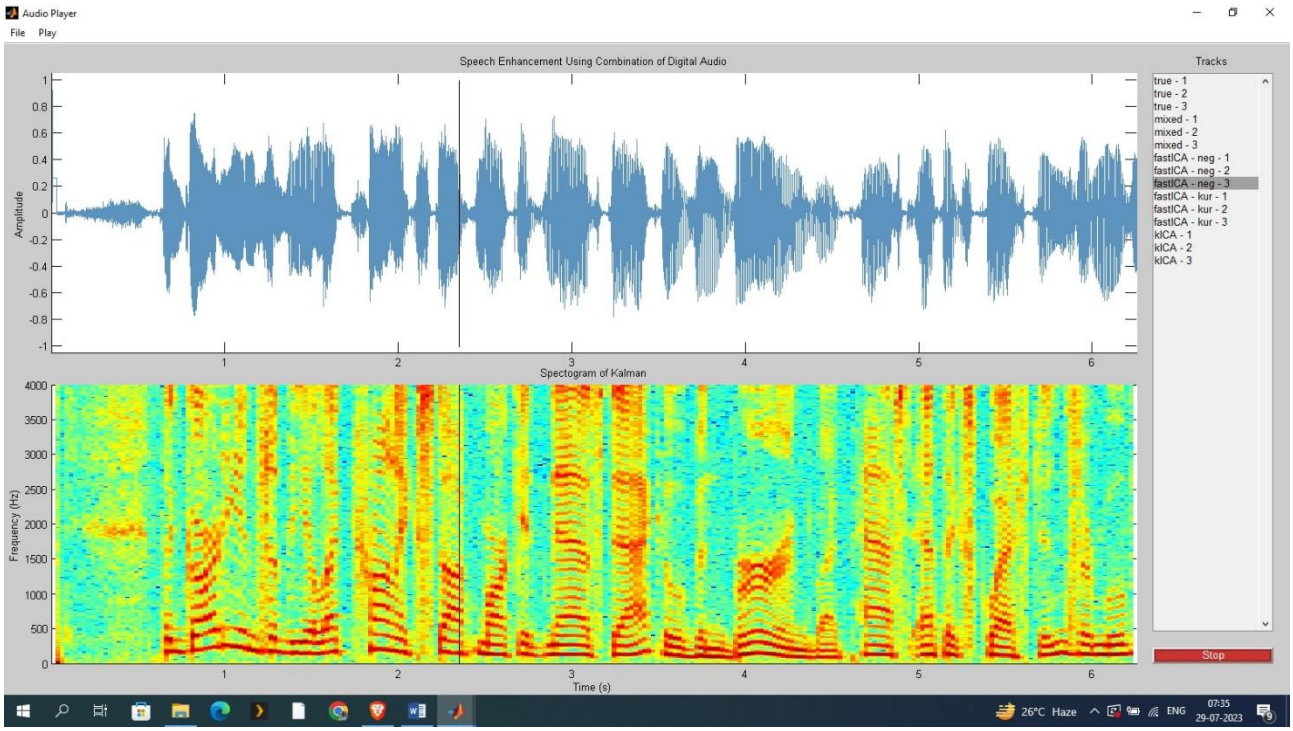
Img.no.8.2 (f) it has mixed noise of the audio1, audio2 and audio3 which has the more noise of the audio3



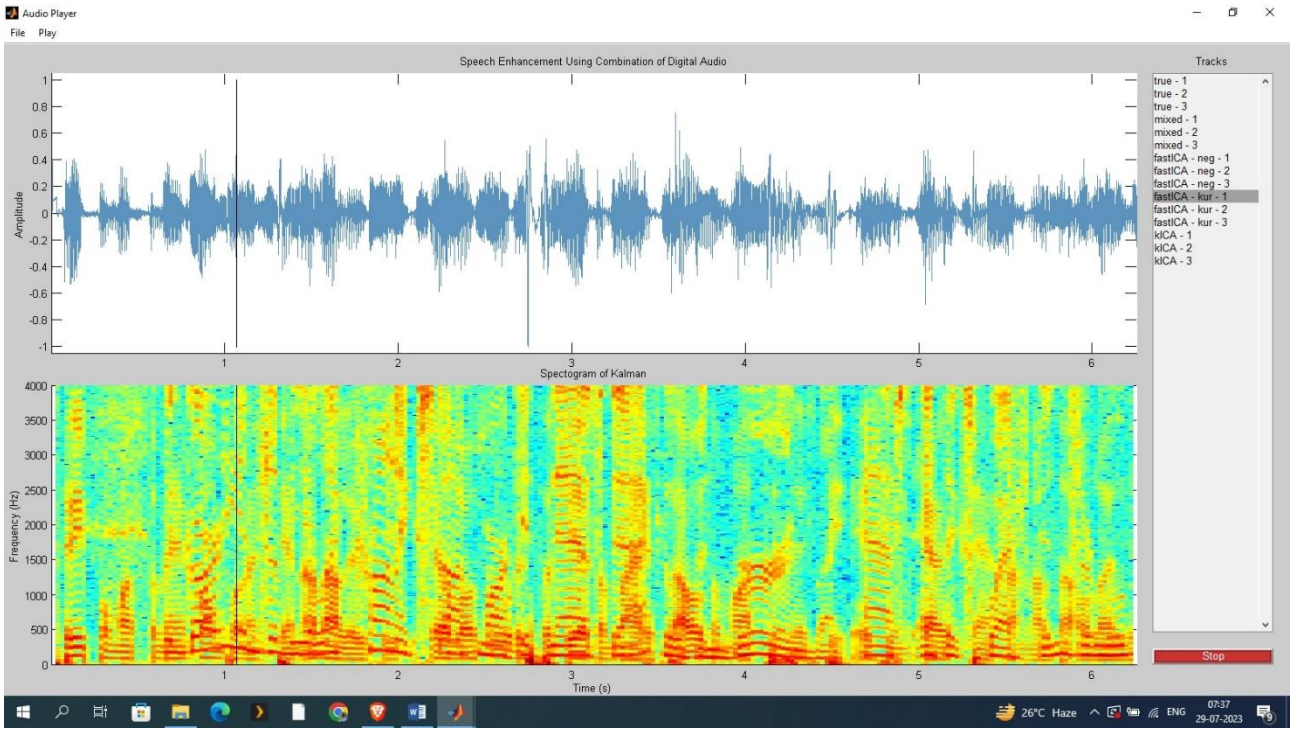
Img.no.8.2 (g) in the above figure audio1 and audio2 have been eliminated after filtering the noise



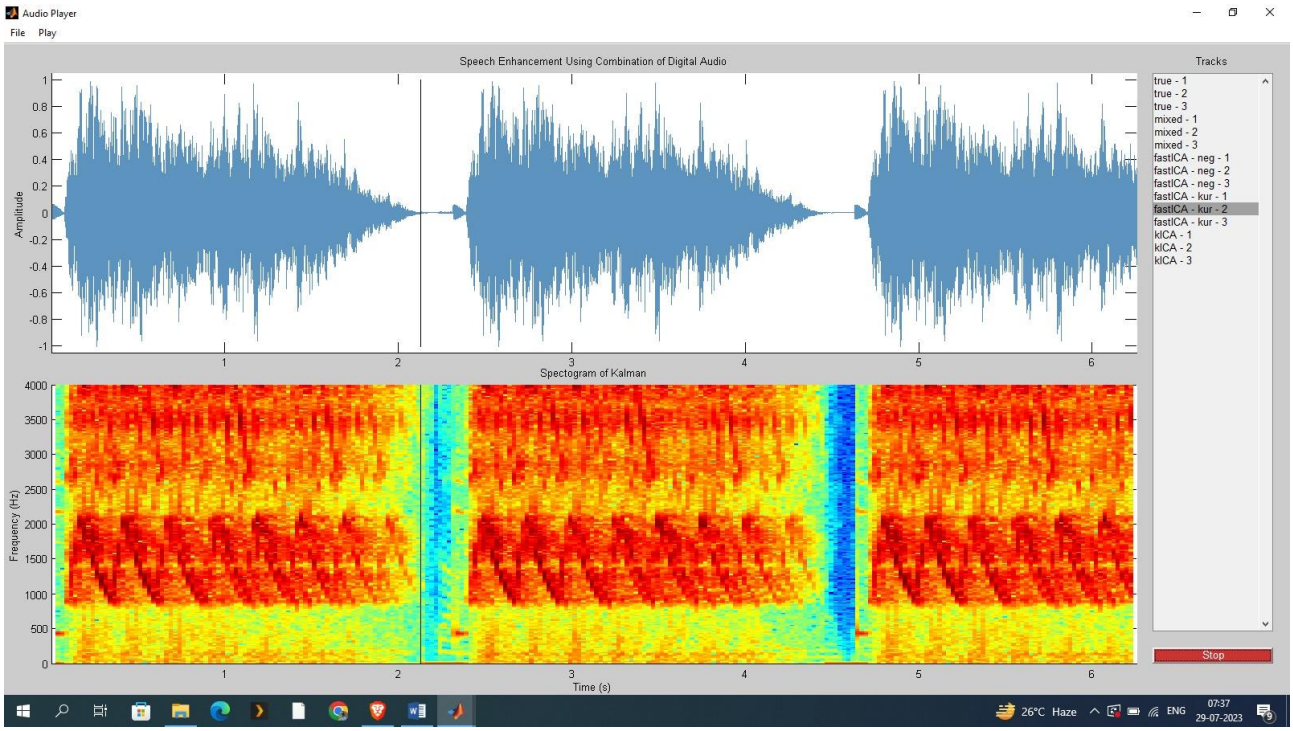
Img.no.8.2 (h) in the above figure audio2 and audio3 have been eliminated after filtering the noise



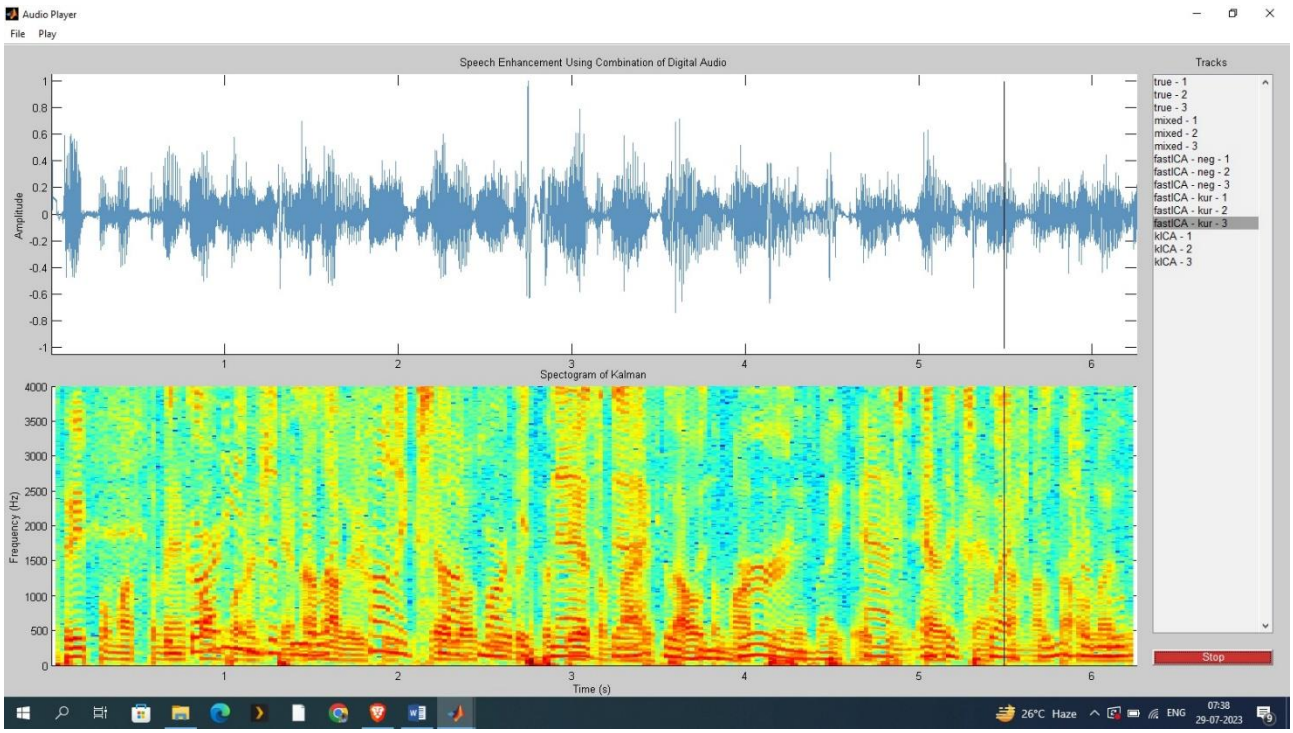
Img.no.8.2 (i) in the above figure audio3 and audio1 have been eliminated after filtering the noise



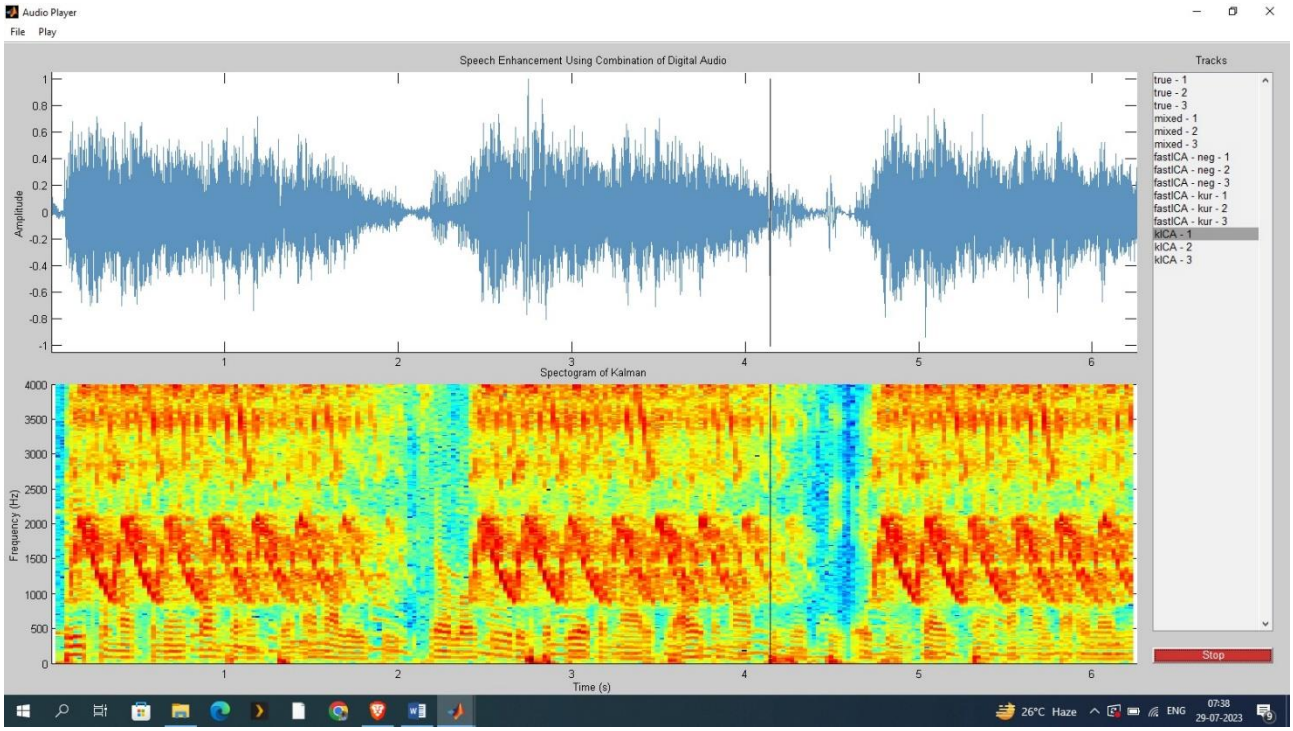
Img.no.8.2 (j) in the above figure audio1 has been eliminated after filtering the noise



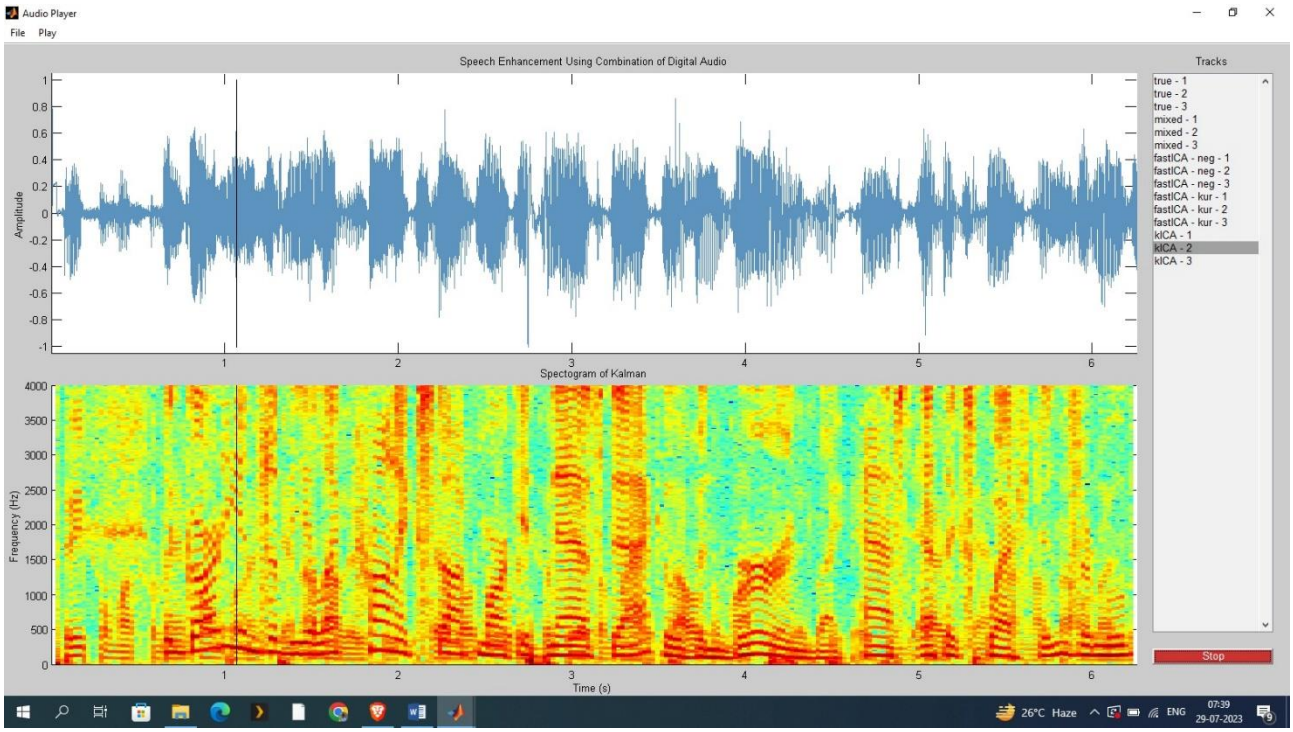
Img.no.8.2 (k) in the above figure audio2 and audio3 has been eliminated after filtering the noise



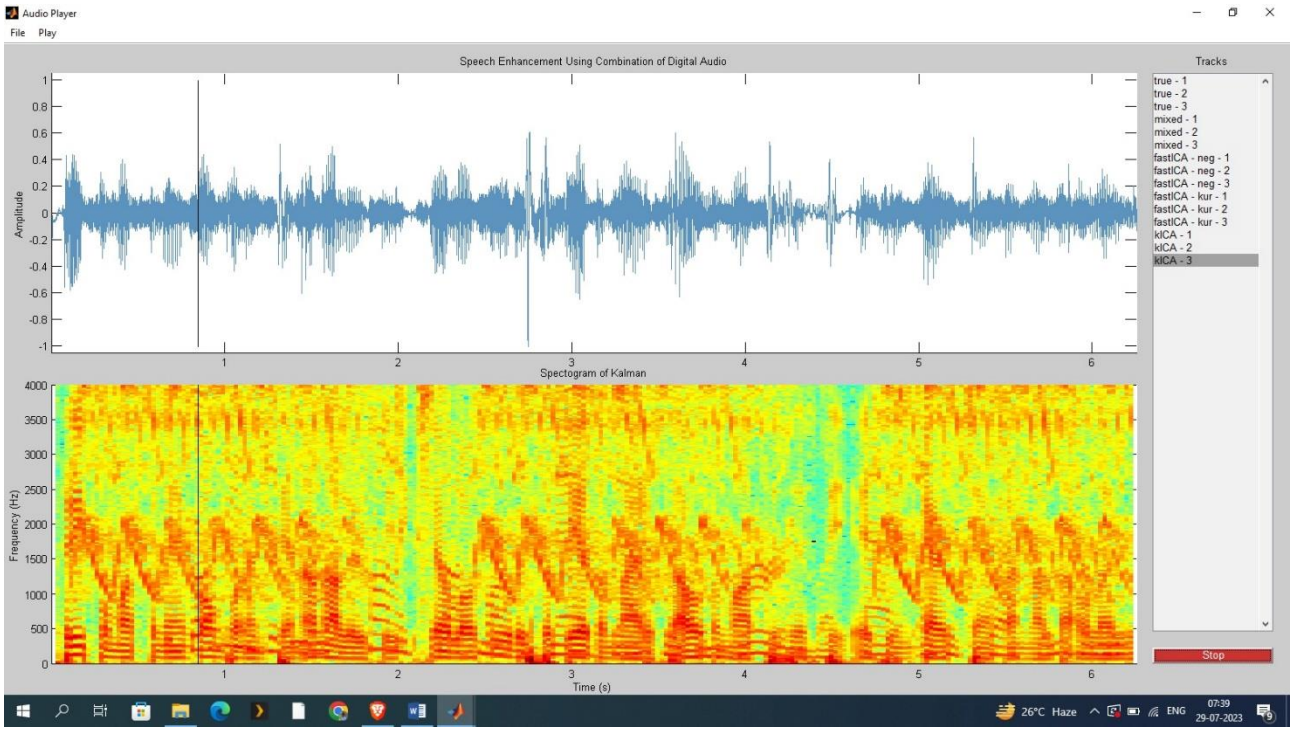
Img.no.8.2 (l) in the above figure audio1 have been eliminated after filtering the noise



Img.no.8.2 (m) in the above figure audio2 have been eliminated after filtering the noise

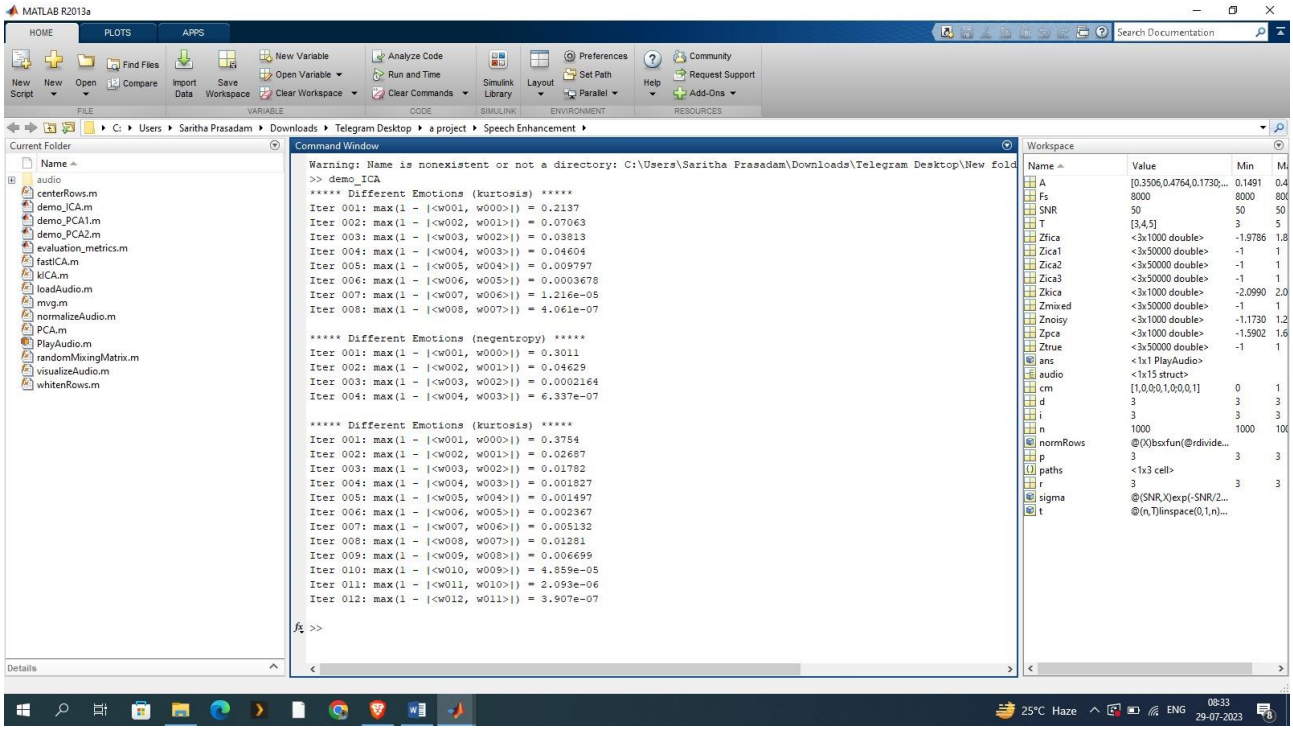


Img.no.8.2 (n) in above figure audio1 have been eliminated after filtering the noise



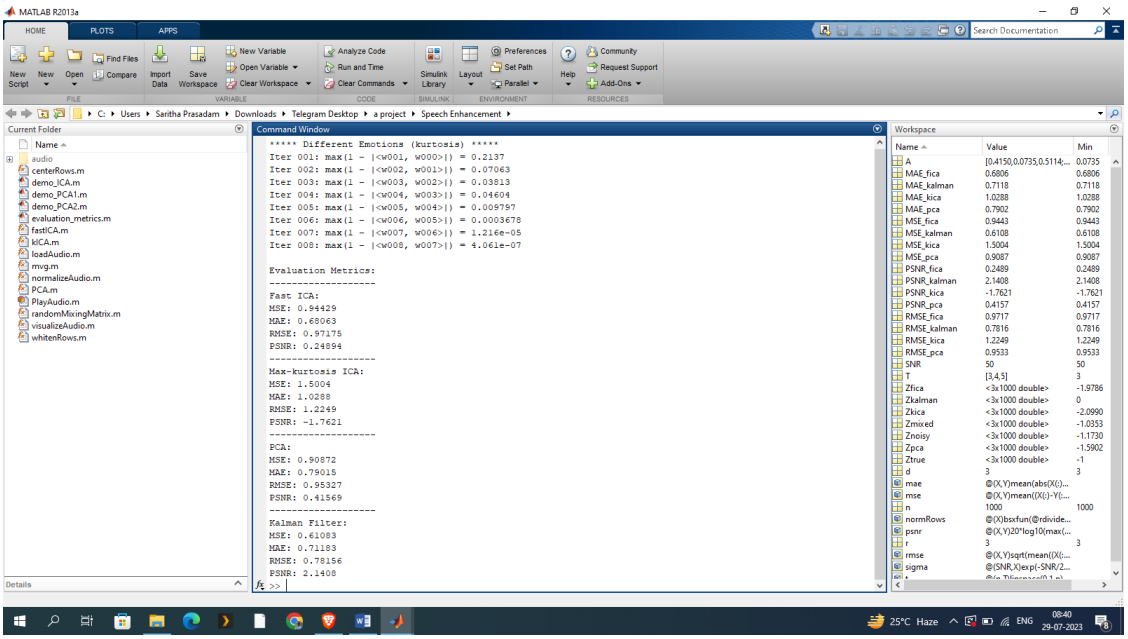
Img.no.8.2 (o) in the above figure audio3 have been eliminated after filtering the noise

8.3 Output 3: Iterations



Img.no.8.3 in above figure we can see that the noise have been eliminated after utmost iterations

8.4 Output 4: Evaluation metrics



Img.no.8.4 in above figure we can see that evaluation metrics have been applied on the project. So we can show that how efficient that the project can be than the existing system

8.5 Existing system vs Proposed system by the evaluation metrics

Algorithms	BCS	COSAMP	IHT
Accuracy	0.97	0.97	1.27
Precision	0.90	0.88	1.08
Recall	1.00	1.07	1.45
PESQ	26.16 dB	15.03 dB	7.56 dB

Table.no.8.5 (a) evaluation metrics of the existing system

Algorithms	Kalman filter	Fast ICA	ICA	PCA
M.S.E	0.61	0.94	1.50	0.90
M.A.E	0.71	0.68	1.02	0.79
R.M.S.E	0.78	0.97	1.22	0.95
P.S.N.R	2.14 dB	0.24 dB	-1.76 dB	0.41 dB

Table.no.8.5 (b) evaluation metrics of the proposed system

9. Conclusion

The paper introduces an improved speech enhancement method that combines Digital Audio Effecting techniques with an improved Adaptive Kalman Filter. The proposed method addresses the drawbacks of basic methods like spectral subtraction and Wiener filter, as well as the complexity and time-consuming nature of other Kalman filter-based approaches.

Through simulations in MATLAB and comparisons of input-output SNR values, the performance of the proposed method is evaluated. The results show that the proposed method outperforms conventional methods, including ICA, Fast ICA, PCA, and Kalman Filter, in terms of output SNR values for both stationary and non-stationary signals. The combination of Digital Audio Effecting and the improved Adaptive Kalman Filter allows for effective noise reduction while considering the perceptual aspects of the human ear, leading to superior speech quality and intelligibility.

Furthermore, the method's computational efficiency makes it highly attractive for real-time applications, offering an advantage over other Kalman filter-based approaches that might be computationally intensive and time-consuming. This efficiency is especially valuable in scenarios where immediate and accurate speech enhancement is essential, such as in hands-free communication devices or voice-controlled systems.

While the study focuses on ICA, Fast ICA, PCA, and Kalman Filter algorithms as points of comparison, the proposed method's unique combination of Digital Audio Effecting and improved Adaptive Kalman Filter sets it apart from these individual techniques. The innovative integration of these components highlights the paper's contribution to advancing speech enhancement research and offers potential for future investigations into novel hybrid approaches.

In conclusion, the proposed speech enhancement method based on approximate message passing, incorporating the ICA, Fast ICA, PCA, and Kalman Filter algorithms, offers a promising approach to significantly improve speech quality in noisy environments. The method's ability to effectively reduce background noise while minimizing distortion and its superior performance compared to traditional techniques make it a valuable contribution to the field of speech enhancement.

10. References

[1]. E. Dong,X. Pu, “ Speech denoising predicated on perceptual weighting sludge," 9th International Conference on Signal Processing, Leipzig, Germany, 2008, pp 705- 708.

[2]. V. Prasad, R. Sangwan, etal." Comparison of voice exertion discovery algorithms for VoIP," proc. Of the Seventh International Symposium on Computers and Dispatches, Taormina, Italy, 2002, pp. 530- 532.

[3]. K.Sakhnov, E.Verteletskaya, B. Šimák, “Dynamical Energy- Grounded Speech Silence Sensor for Speech Enhancement Applications." In World Congress of Engineering 2009 Proceedings. Hong Kong, 2009, pp. 801- 806.

[4]. “Speech Enhancement" byJ. Benesty,J. Chen,Eds., andS. Makino, Springer, Berlin, 2005.

[5]. S.Boll," suppression of audial noise in speech using spectral deduction,” IEEE Transaction on Speech, Signal Process. Volume.ASSP- 27, no. 2pp.113- 120, 1979

[6]. T.V.SreenivasandP.kirnapure," Codebook constrained Wiener filtering for speech enhancement," IEEE Trans. Speech and Audio Processing, sep1996.

[7]. K.K.Paliwal and A.Basu, “ A peroration improvement system grounded on Kalman filtering ” Actions of ICASSP ’ 87,pp.177- 180. Dallas, TX, USA, 1987.

[8].” Digital Audio Signal Processor, ” second edition, by Udo Zolzer.

[9]. “Noizeus- A noisy speech corpus for evaluation of speech improvement algorithms,” http//ecs.utdallas.edu/loizou/speech/noizeus.

[10].Hu.Y andLoizou.P, ” Evaluation of objective quality measures for speech improvement, ” IEEE Trans. On speech and audio processing, 2008.