Selective Noise Cancellation

EC208: Mini Project 1: S.Y. (EXTC) Sem: IV Academic Year 2023-24

Team Members:

Shawez Shaikh-2022200104

Gaurang Rajvaidya-2022200099

Shivam Singh-2022200110

Under Guidance of Prof. Payal Shah

And

Alumni Mentor Kartik Chakole

Department of Electronics & Telecommunication Engineering Sardar Patel Institute of Technology
Munshi Nagar, Andheri(W), Mumbai-400058 UNIVERSITY OF MUMBAI

2023-2024

Abstract

Have you ever woken up without understanding what it was, but knowing for sure that some sound isn't right? Sound identification is one of our instincts that keeps human beings safe. Sounds play a significant role in our life, Starting from recognizing a predator nearby to being inspired by music, to groups of human voices, to the cry of a bird. Inevitably, developing audio classifiers is a crucial task in our lives.

Ordinarily, it is essential to classify the sounds' source and is already widely used for various purposes. In music, there's a classifier for the genre of music. Recently similar systems began to be used to classify Human/ bird calls, historically done by Ornithologists. Their goal is to categorize human voice, considering it is challenging to discover voice calls from the fields or noisy surroundings.

Consider a scenario where a user is wearing ear-worn devices on a beach and desires to listen to the calming sounds of the ocean while blocking out any human speech nearby. Similarly, while walk- ing on a busy street, the user may wish to reduce all sounds except for emergency sirens; or while sleeping, they may want to listen to the alarm clock or baby sounds but not the noise from the street. In another scenario, **the user may be on a plane and desire to hear hu- man speech and announcements but not the sound of a crying baby**. Or while hiking, the user may want to listen to the birds chirping. **Work from home mother want to hear their only their baby noise while attending meet.**

Contents

Chapter 1

Introduction

Based on the provided information, it appears that your project revolves around implementing noise cancellation techniques using signal processing methods, specifically applied through Python programming. Here's how you can extrapolate on each point:

1. Project focus: The main focus of the project is to utilize signal processing methods for noise cancellation, indicating a dedication to improving audio quality through technological means.

2. Specific application: Python programming is chosen as the tool for implementing these noise cancellation techniques, highlighting its versatility and effectiveness in handling signal processing tasks.

3. Noise identification and mitigation: Python algorithms are employed to accurately identify and effectively mitigate unwanted noise components present in the audio signals, showcasing the power of computational approaches in addressing real-world problems.

4. Implementation platform: The choice of Jupyter Notebook as the implementation platform suggests a preference for an interactive and exploratory development environment, allowing for easy experimentation and documentation of the process.

5. Practical solution: The project aims to provide a practical solution for enhancing audio quality in various environments, indicating a focus on real-world applicability and usability rather than purely theoretical advancements.

8. Aim: The primary goal is to demonstrate the feasibility and effectiveness of signal processing techniques for noise reduction, indicating a desire to validate the practical benefits of the proposed approach.

9. Potential applications: By showcasing potential applications of such technology in improving audio systems' performance, the project aims to highlight the broader impact and relevance of its findings beyond the specific implementation context.

Overall, Our project appears to be a comprehensive exploration of using signal processing methods, implemented in Python, to address the challenge of noise cancellation in audio signals, with a focus on real-world applicability and potential future advancements.

Chapter 2 Literature

Review

Selective Noise Cancellation using MachineLearning

Revati Sule
Electronics and Telecommunication
NMIMS University
Mumbai, India
revatisule02@gmail.com

Akshat Kolekar
Electronics and Telecommunication
NMIMS University
Mumbai, India
akshatkolekar@gmail.com

Keval Patel
Electronics and Telecommunications
NMIMS University
Mumbai, India
kevalpatel.kjp@gmail.com

Avinash Tandle
Assistant Prof. EXTC Department
NMIMS University
Mumbai, India
avinash.tandle@nmims.edu

Algorithms for adaptive noise cancellation

- Least Mean Square (LMS) algorithm,
- Kwong Least Mean Square algorithm,
- Recursive Least Square Algorithms,
- NLMS,
- K-Nearest Neighbor (KNN)
- Logistic Regression (LR)

How training datasets are curated :

The image is a flowchart of the data collection process from the research paper on selective noise cancellation using machine learning. Here's the step-by-step process:

1. Download required audio file: The initial step involves obtaining the necessary audio files for the dataset.
2. Convert audio file to 16-bit sampled rate: After downloading, each audio file is converted to a 16-bit sampled rate to standardize the data format.
3. Segregation of audio files into clusters: The audio files are then grouped into clusters, possibly based on characteristics like noise type or environment.
4. Computing features of audio file manually: For each cluster, features of the audio files are computed manually, which likely involves analyzing and extracting relevant audio characteristics.
5. Entering feature values into dataset: The computed feature values are then entered into a dataset, creating a structured format for the machine learning models to process.

6. Assigning values of ‘1’ & ‘0’ to audio files manually: Finally, audio files are manually labeled with '1' or '0', which could indicate the presence or absence of certain audio features or noise types.

Further development Scope:

Real-Time Noise Cancellation: There is potential to further develop the model to enable real-time noise cancellation, allowing for immediate and dynamic adjustments to changing noise environments. This could involve the integration of advanced signal processing techniques and faster computational algorithms to achieve real-time adaptive noise cancellation.

Semantic Hearing: Programming Acoustic Scenes with Binaural Hearables

Bandhav Veluri*
Paul G. Allen School, University of Washington, Seattle, WA, USA
bandhav@cs.washington.edu

Malek Itani*
Paul G. Allen School, University of Washington, Seattle, WA, USA
malek@cs.washington.edu

Justin Chan
Paul G. Allen School, University of Washington, Seattle, WA, USA
jucha@cs.washington.edu

Takuya Yoshioka
Microsoft, One Microsoft Way, Redmond, WA, USA
tayoshio@microsoft.com

Shyamnath Gollakota
Paul G. Allen School, University of Washington, Seattle, WA, USA
gshyam@cs.washington.edu

"Semantic Hearing: Programming Acoustic Scenes with Binaural Hearables" encompasses the introduction of a novel capability for hearable devices known as "semantic hearing." This capability enables hearable devices to focus on or ignore specific sounds from real-world environments in real-time while preserving spatial cues.

Technical findings:

- Introduction of the first neural network capable of achieving binaural target sound extraction:
  - Authors presented a neural network capable of extracting binaural target sounds amidst interfering sounds and background noise.

- Design of a training methodology for generalization to real-world use:
  - Authors devised a training methodology to ensure the binaural network's generalization to real-world scenarios, including reverberations, multipath, and head-related transfer functions (HRTFs).
  - This methodology involved synthesizing training data from multiple datasets, including HRTF data and binaural room impulse responses from real rooms.
  - The aim was to enable the network to generalize to users and real-world environments not present in the training dataset.

Scope for Future development

1. Imbalance in training examples across classes:
 - Some classes, like "speech," have significantly more training examples (494) compared to others like "car horn" (60).
 - Suggested solution: Collect more examples across all classes to potentially improve system performance.

2. Inherent difficulty in separating certain classes:

- Classes like music and human speech share many characteristics, making separation challenging.
- Tasks such as separating speech from background music with vocals can be difficult.
- Similarly, separating music from classes like alarm clock sounds or bird chirping is challenging.

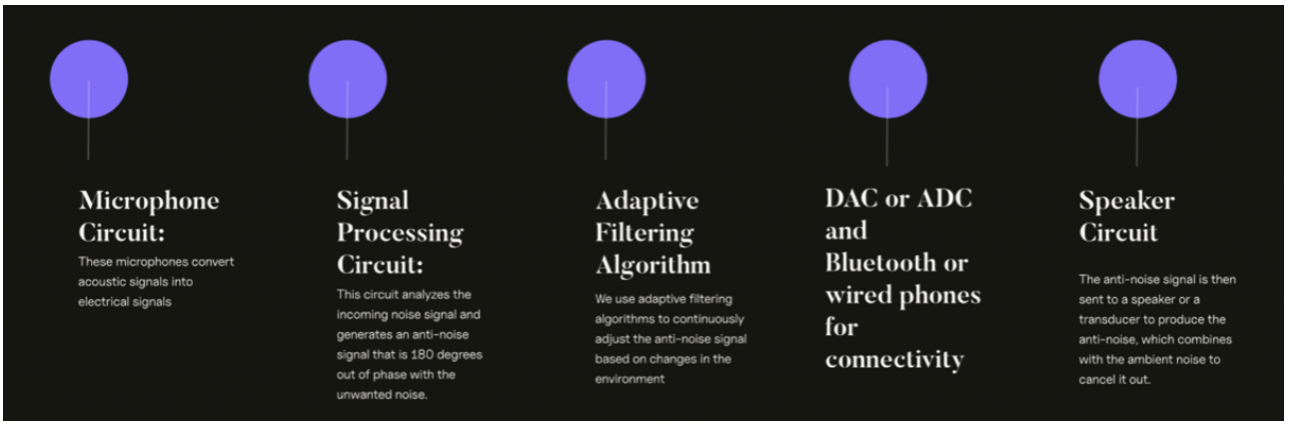
3. Lack of real-world data utilization in training methodology:
- Training methodology doesn't incorporate real-world data with the hardware used in evaluations.
- Suggested solution: Collect training data in real-world scenarios and with actual hardware to potentially enhance system performance.

4.Limitation in hardware form factor:
- The hardware used involves binaural earphones and a noise-canceling headset, which may not be convenient or practical.
- Suggested solution: Simplify the form factor by using a single device for recording and playback, such as commercial noise-canceling headsets with microphone access.

Chapter 3

Project Objectives

1. To findout, understand, implement current measures for Active Noise Cancellation.
2. To realise signal processing techniqus use for noise reduction techniques
    1. Understanding How stationary noise reduction technique works
    2. Smoothing and Masking filter
3. To create an algorithm allowing user to select noises to be muted\cancelled.
4. To ensure that the above function causes no minimum delay in relay of the sounds in real time conversation
5. To create a feature allowing users to store audio files not present in the available data set and selectively cancel or hear the audio.



Project Progress(till now):

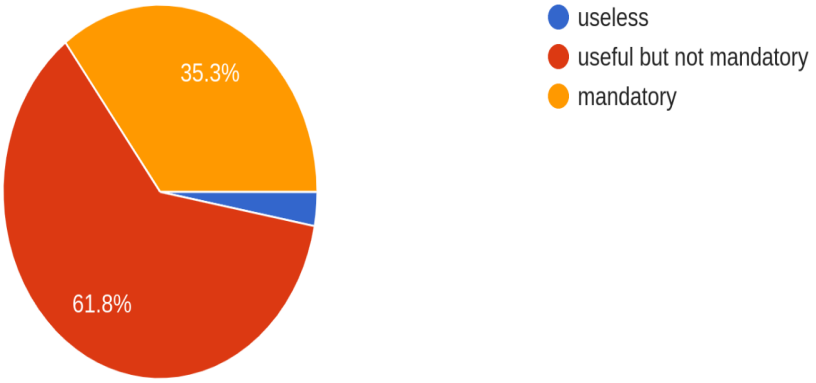
- [X] Interfacing microphone with esp32/ microcontroller
- [X] Signal processing/ noise reduction techniques on python script:
- [ ] Applying FFT on live recording taken from esp32/microcontroller
- [ ] Applying CNN algorithm/ Ml model for selectively reduce noise

- [ ] Increasing dataset for further testing

Chapter 4 Field Survey

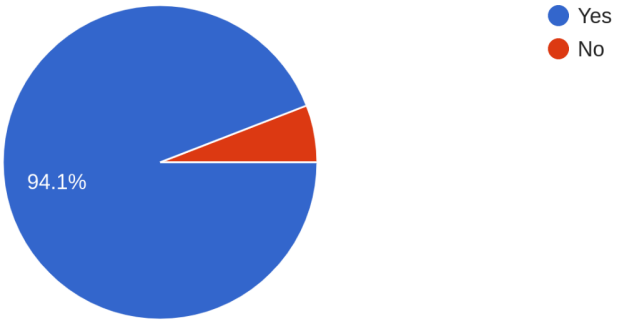
What are your thoughts on the role of the active noise cancelling feature of earphones?

34 responses



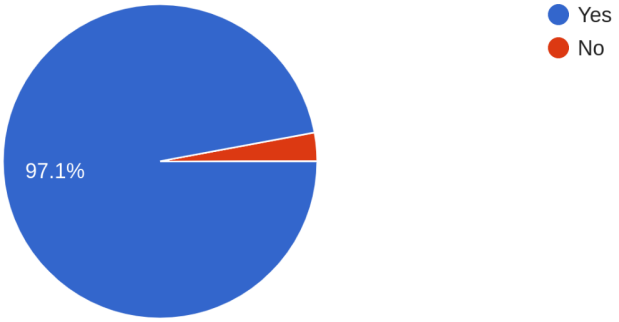
Would you like to have a feature in which you can control any noise/disturbance of your surroundings by simply cancelling or allowing it at your own will?

34 responses



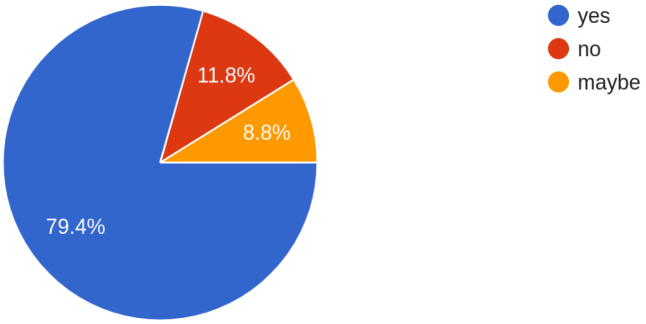
Would you like it more if active noise cancelling becomes selective noise cancelling by user's selection?

34 responses



Would you like a feature in your earphone in which you can turn off the disturbing sound of constructions but only allow the honks of any vehic...ing around while listening to your favourite music?

34 responses



Summary::

- Based on the pie chart, we can infer that the average person uses their Bluetooth device for about 4 hours a day.
- Apple products are the most popular brand with 52.9% of the users surveyed because of their sound quality and well built products. Sony comes in second at 14.7%.
- The most common purpose for using earphones is entertainment, at 44.1% followed by study, and traveling (combined) at 35.3%.
- Most people (47.1%) are satisfied with the active noise cancellation (ANC) feature of their earphones. That leaves 52.9% of people who are either unsatisfied (17.6%) or believe the ANC feature could be improved (35.3%).
- The common problem people face with Bluetooth devices is that the Active Noise Cancellation (ANC) feature is not very effective. Specifically, the text says that the ANC feature is “not up to the mark” and “not quite impressive”.

Here are some of the things users say about the ANC feature in the image:

- ❖ It is not effective in loud noises or crowded places.
- ❖ It does not allow some ambient sound in when desired.
- ❖ It needs to be more advanced and user-specific.

- Based on the survey, a very large majority of people (94.1%) would like to have a feature that allows them to control surrounding noises. This suggests that noise can be a significant disturbance for many people. The survey indicates that people wants to cut out disturbance like traffic noise, construction noise, or noise from other people.
- Most people (79.4%) would like a feature in their earphones that allows them to turn off disturbing sounds while listening to music. This suggests that a significant number of people find noise to be a distraction when using earphones.The survey also shows that 11.8% of people would prefer to have all sounds come through their earphones, and 8.8% of people are undecided.

Inference:

Solving ANC Problems:

- **Ineffectiveness in certain situations:** The survey likely revealed that users find current ANC technology ineffective in specific scenarios like loud environments or with certain types of noise (e.g., construction noise, honking vehicles). Selective noise cancelling could address this by allowing users to focus on cancelling specific unwanted sounds while letting in others.
- **Lack of user control:** The survey might have indicated that users desire more control over the ANC feature. Selective noise cancelling empowers users to personalize their noise cancellation experience, tailoring it to their specific needs and preferences.

Improving Audio Quality:

- **Unwanted noise masking desired sounds:** The survey might have shown that current ANC sometimes blocks out desired ambient sounds like important announcements or conversations. Selective noise cancelling allows users to enjoy their chosen audio content without sacrificing important environmental cues.
- **Preserving natural sound:** Survey participants might have expressed concerns about current ANC technology affecting the overall sound quality by creating an unnatural listening experience. Selective noise cancelling offers the potential to eliminate unwanted noise while preserving the natural soundscape.

The field survey suggests that a significant portion of users are dissatisfied with the current limitations of ANC technology. Selective noise cancellation presents a solution by offering a more personalized and user-controlled approach to noise management. This feature has the potential to significantly improve user satisfaction with ANC, audio quality, and overall user experience.

Chapter 5:Market Survey

Main inference:

High Demand for Selective Noise Cancelling:

- **Market Trend:** The overall noise cancelling market is experiencing significant growth, projected to reach $45.4 billion by 2031. This indicates a strong consumer desire for technology that reduces unwanted noise.
- **Survey Results**: The high percentage of users (94.1%) wanting control over surrounding noises suggests a clear demand for selective noise cancelling features.
- **Specific Needs:** The survey highlights the desire to block specific noises like construction sounds and honking vehicles. This indicates that current ANC technology may not be fully addressing user needs.

Selective Noise Cancelling as a Solution:

- **Customization**: Selective noise cancelling offers a more personalized approach, allowing users to control which sounds are blocked and which are allowed through. This caters to the varying preferences identified in the survey.
- **Improved User Experience:** By providing greater control over the listening environment, selective noise cancelling can enhance focus, improve audio quality, and reduce overall listening fatigue.
- **Market Opportunity:** The strong demand for noise control, combined with the limitations of current ANC technology, creates a significant opportunity for the development and implementation of selective noise cancelling features.

Therefore, based on the market survey and broader market trends, selective noise cancelling appears to be a highly desired feature that has the potential to further revolutionize the earphone and headphone market.

The main principle:

"Active Noise Cancellation (ANC) works by using microphones to pick up ambient sound and then producing sound waves that are the exact opposite (anti-phase) of the ambient noise. When these anti-noise waves combine with the ambient noise, they cancel each other out, effectively reducing the overall noise level."

PROS of ANC:

- Improved Audio Quality
- Enhanced Focus
- Protects Hearing

- Better Communication
- Travel Comfort
- Customizable Experience

CONS OF ANC:

- **Sound Quality Impact**
- **Price**
- **Compatibility Issues:** ANC may not work optimally in all environments or with all types of audio content. Certain frequencies or types of noise may not be effectively canceled, leading to a less-than-ideal experience in some situations.
- **Reduced Awareness**
- **Potential for Health Risks**
- **Device Size and Weight**
- ANC is more effective at reducing low-frequency sounds (e.g., airplane engine noise, traffic rumble) than high-frequency sounds (e.g., voices, birds chirping).
- ANC is most effective at reducing continuous, predictable sounds rather than sudden, unpredictable noises
- **Latency: ANC systems introduce a slight delay in processing audio signals due to the time required for noise analysis and cancellation.**
- **Power consumptions**

PRODUCTS:

1)Sony WF-1000XM4

FREQUENCY RESPONSE (BLUETOOTH® COMMUNICATION)
20–20,000 Hz (44.1 kHz sampling) / 20–40,000 Hz
(LDAC 96 kHz sampling, 990 Kbps)

- Bluetooth earbuds with the new Integrated Processor **V1(QN1e chip)**
- **Battery life - 24 hours long battery with Noise Cancelling.**
- **Frequency Response (Bluetooth Communication)-20Hz - 20,000Hz(44.1kHz sampling) / 20Hz - 40,000Hz(LDAC 96kHz sampling, 990kbps)**
- **BEST TWS of the year-2022**
- **IPX4 water resistance rating.**
- **5-minute quick charge gives you up to 60 minutes of play time.**
- **Microphone format: Built-In**
- **Microphone technology: Omnidirectional, Stereo**
- **Item Weight:41g**
- **Review:⅘**
- *The ANC is adaptive, meaning it adjusts according to the surrounding environment to optimize noise cancellation performance.*
- ***They utilize dual noise sensor microphones—one feed-forward and one feedback microphone—to detect and analyze ambient sounds both outside and inside the ear canal. This allows for more precise noise cancellation tailored to the wearer's environment.***
- *Sony's Adaptive Sound Control technology is employed Using **built-in sensors**, including an **accelerometer and gyroscope**, the earphones can detect whether the user is stationary, walking, running, or in transit. Based on this information, the ANC level is dynamically adjusted to optimize noise reduction while preserving important environmental sounds, such as announcements or approaching vehicles.*
- ***Customizable ANC Settings:** Users have the flexibility to customize the ANC settings according to their preferences using the companion smartphone app. This app allows users to fine-tune the level of noise cancellation, adjust the sensitivity of ambient sound passthrough (Ambient Sound Control), and even create personalized ANC profiles tailored to specific environments.*
- ***High-Quality Audio Performance:** sony's proprietary driver technology and support for high-resolution audio codecs such as LDAC.*
- *The Sony WF-1000XM4 is estimated to achieve reductions of up to **20-25 decibels** in certain frequencies, particularly in the lower frequency ranges where ANC is most effective*

2.Apple AirPods Pro:

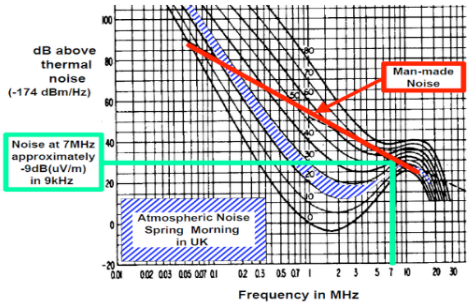
- ***Dual Microphones:*** *Each AirPod Pro earbud is equipped with two microphones, one facing outward and one facing inward.*
- ***Noise Cancellation Algorithm:*** *algorithm continuously monitors and produces inverse sound waves.*
- ***Seal Testing:*** *advanced vent system combined with the customizable silicone ear tips to create a tight seal in the ear canal.*
- ***Transparency Mode:*** *When users need to hear their surroundings, such as when crossing the street or having a conversation, they can switch to Transparency Mode. This mode uses the built-in microphones to allow ambient sounds to pass through the earbuds, providing clarity and awareness without needing to remove the AirPods Pro.*
- ***Adaptive EQ:*** *automatically adjusts the audio output based on the shape of the user's ear.*

3.Bose QuietComfort Earbuds:

- Bluetooth 5.1 connectivity.
- Compatible with Bose Music app for customization
- Adaptive ANC: continuously monitors and adjusts to the environment, adapting its noise-canceling levels to match the level of ambient noise.
- Customizable Levels:through the app
- StayHear Max Tips:maximum fitting
- Transparency Mode: This mode uses the earbuds' microphones to amplify ambient sounds.

Chapter 6 Theory

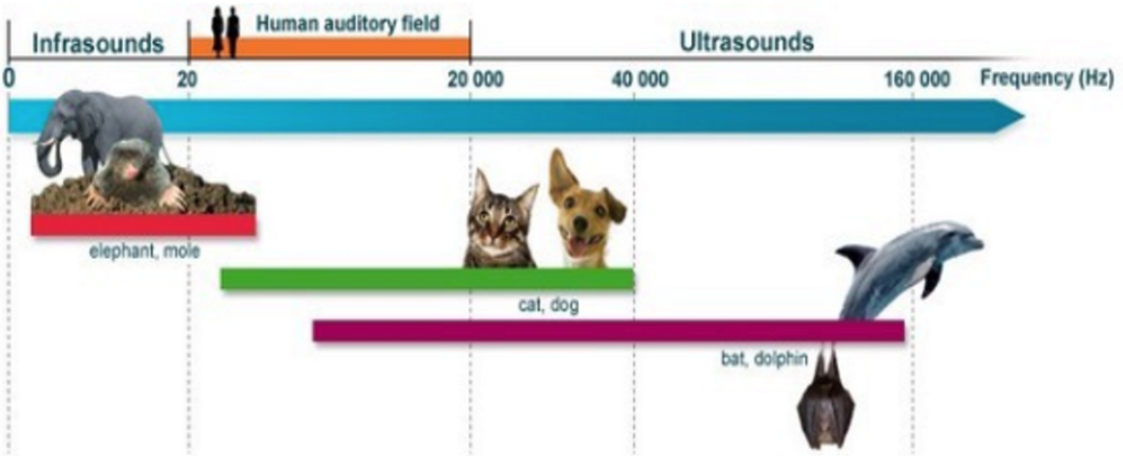
Noise



Here's a breakdown of the different noise sources shown on the graph:

- **Thermal Noise:** This is the lowest level of noise shown on the graph, and it is caused by the random motion of electrons in conductors. Thermal noise is always present, regardless of the temperature.
- **Atmospheric Noise:** This type of noise is caused by lightning and other electrical activity in the atmophere. It is strongest at lower frequencies and weakens as the frequency increases.
- **Man-Made Noise:** This type of noise is produced by human-made sources such as electrical power lines, radio stations, and vehicles. Man-made noise can vary depending on location and time of day.

The graph shows that man-made noise is the strongest type of noise at frequencies above about 10 MHz. Atmospheric noise is the strongest type of noise at lower frequencies. Thermal noise is the weakest type of noise at all frequencies.



-

A)Information extraction from the given Audio signal

- Audio waves are the vibration of air molecules whenever any sound happens and sound travels from originator to the receiver in the form of a wave.
- As such, this wave has 3 properties to it — Amplitude, Frequency and Time.
- → Amplitude represents the magnitude of the wave signal and it is usually measured in
  decibels (dB).
- → Time is the time scale, as we all know it.
- → Frequency represents how many complete cycle the wave takes in one second and it is
  measured in Hz.

B) What is DSP(Digital Signal Processing)

Digital Signal Processing (DSP) is a specialized branch of engineering and mathematics that deals with the manipulation, analysis, and interpretation of digital signals. It involves converting analog signals into digital form, performing various operations on them such as filtering, compression, and enhancement, and then converting them back into analog signals for interpretation or further processing

C) Fast Fourier Transform

Fast Fourier Transform (FFT): Efficient algorithm converting signals from time domain to frequency domain, enabling fast analysis of signal components.

1. **Frequency Domain Analysis**: FFT reveals frequency distribution of signal, aiding noise identification.
2. **Noise Identification**: Noise manifests as random fluctuations across frequencies in the frequency domain.
3. **Filtering**: Apply filters (like low-pass, high-pass) in the frequency domain to attenuate or remove noise components while preserving signal.

4. **Inverse Transform**: After filtering, perform inverse FFT to convert the signal back to time domain.
5. **Signal Reconstruction**: The filtered signal is ready for further analysis or processing, with noise reduced.

D) Short Term Fourier Transform

Short-Time Fourier Transform (STFT): Technique for analyzing signal variations over both time and frequency by breaking the signal into short segments.

1. **Time-Frequency Analysis**: STFT provides a time-varying representation of the frequency content of a signal, allowing analysis of signal changes over time.
2. **Windowing**: The signal is divided into short segments using a window function, typically overlapping, to capture time-varying frequency content.
3. **Frequency Content Visualization**: STFT generates a spectrogram, displaying the time evolution of frequency components, useful for analyzing changes in audio signals over time

4. **Applications in Audio Processing**:
    1. **Speech Processing:** STFT is used in speech recognition, enhancement, and synthesis.
    2. **Audio Effects:** STFT enables various audio effects like time stretching, pitch shifting, and reverberation.
    3. **Music Analysis:** STFT helps in music transcription, instrument identification, and genre classification.
    4. **Noise Reduction:** STFT aids in identifying and attenuating noise components in audio signals.
5. **Inverse Transform**: Inverse STFT reconstructs the signal in the time domain from the modified frequency domain representation, facilitating further processing or playback.

STFT is a fundamental tool in audio processing, offering insights into the time-varying frequency content of signals and enabling various audio manipulation and enhancement techniques.

E) Masking and Smoothing Filter

Masking :

- Masking refers to the phenomenon where the perception of one sound is affected by the presence of another sound, either in frequency or time.
- In frequency masking, louder sounds (maskers) make nearby quieter sounds (maskees) less audible.
- Smoothing filters can indirectly affect masking by reducing the amplitude fluctuations in a signal, potentially reducing the likelihood of one sound masking another.
- For example, if a smoothing filter is applied to reduce the amplitude variations of background noise, it may make it less likely to mask quieter sounds in the foreground.
- Additionally, smoothing filters can also be used as preprocessing steps in psychoacoustic models to improve the accuracy of masking predictions, which are then used in audio compression or noise reduction algorithms.

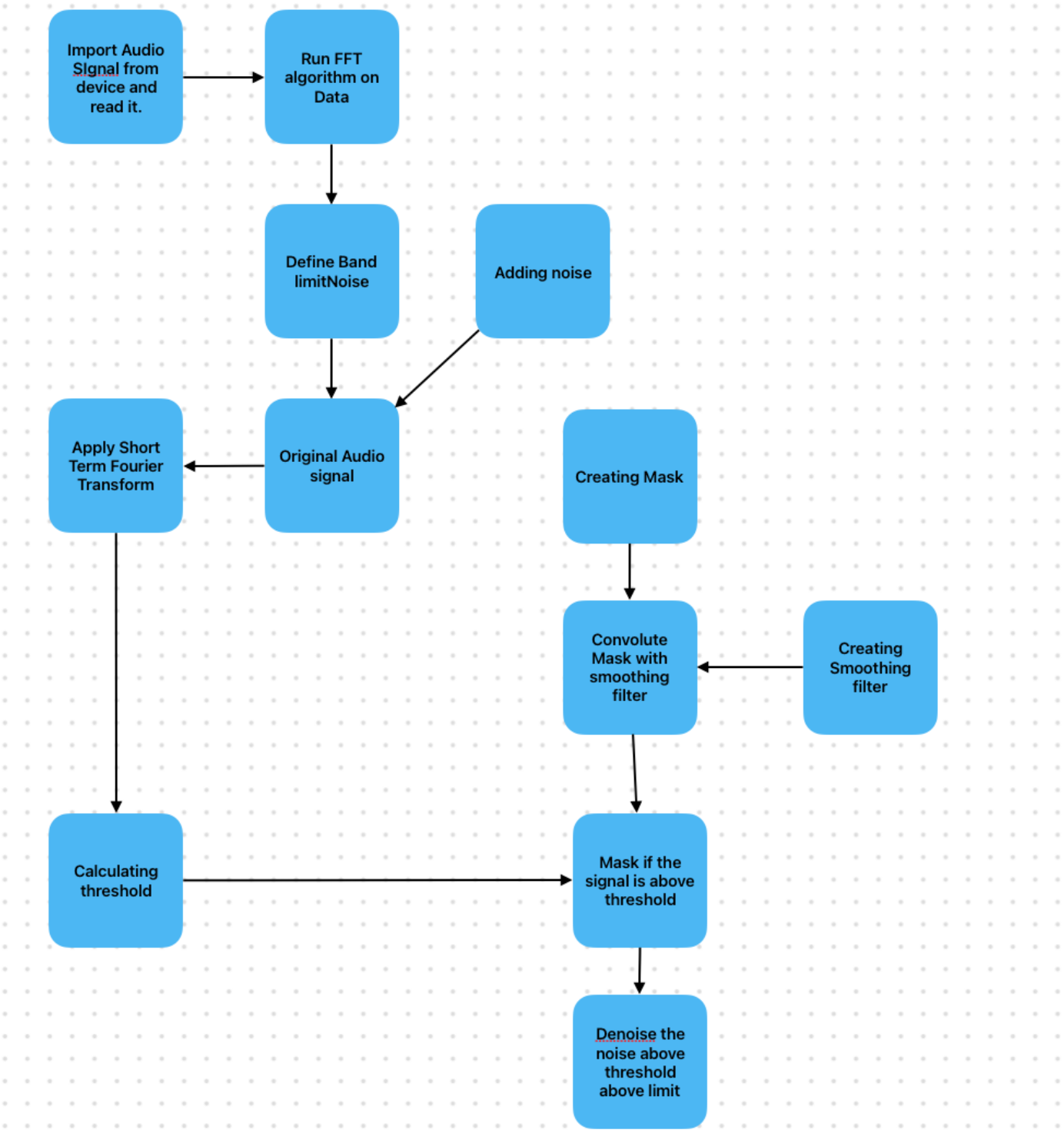
Smoothing Filters:

- Smoothing filters are used to reduce high-frequency noise or sharp transitions in a signal.
- They work by averaging neighboring data points, effectively blurring the signal and reducing its variability.
- Common types include moving average filters, Gaussian filters, and median filters.
- In audio processing, smoothing filters can be applied to reduce background noise or to create a more natural transition between adjacent audio samples.

While smoothing filters and masking serve different purposes in signal processing, they can interact in audio processing applications. Smoothing filters can indirectly affect masking by altering the amplitude characteristics of a signal, potentially influencing the perception of sound masking in both frequency and time domains.

Chapter 7

Flow Chart

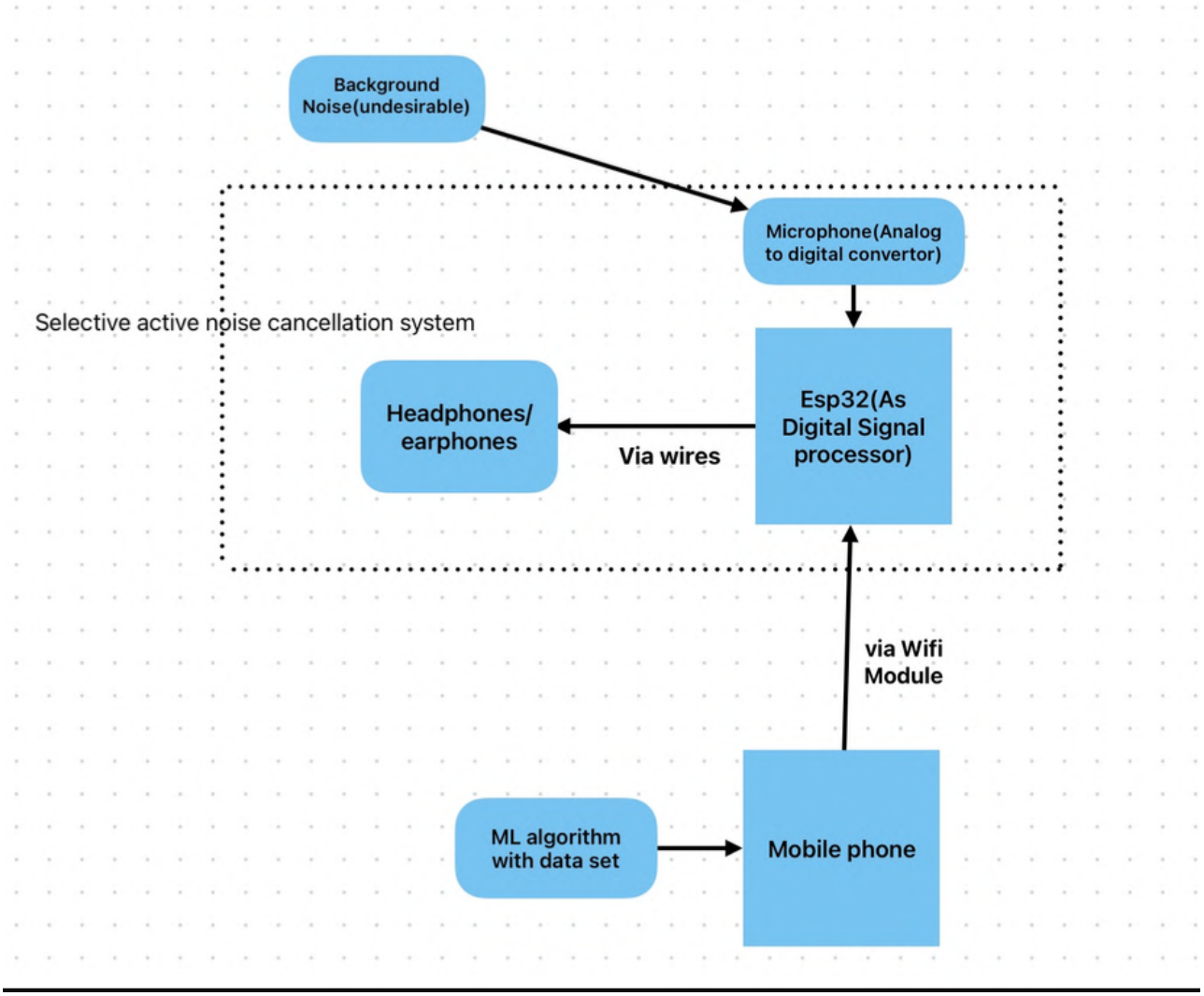


The algorithm used in the provided Python code snippet is primarily based on signal processing techniques for noise reduction in audio signals. Here's a breakdown of the key algorithms and techniques utilized:

1. **Fast Fourier Transform (FFT)**:
    - The FFT algorithm is used to analyze the frequency components of audio signals by transforming them from the time domain to the frequency domain. This allows for frequency-based operations such as filtering and spectral analysis.
2. **Band-Limited Noise Generation**:
    - The `band_limited_noise` function generates band-limited noise in the frequency domain using FFT. This noise generation technique is often used for simulating various types of noise in audio signals.
3. **Noise Reduction Techniques**:
    - The `noisereduce` library is used to apply noise reduction techniques to the audio signal. This includes stationary noise reduction using statistical analysis and adaptive filtering approaches.
4. **Masking and Filtering**:
    - The custom `removeNoise` function calculates noise statistics and applies a mask to attenuate noise components in the frequency domain. Filtering and smoothing techniques are used to refine the noise reduction process and improve signal quality.
5. **Real-Time Audio Visualization**:
    - Additional algorithms and techniques are employed for real-time audio visualization using `pyaudio` and `pyqtgraph`. These include real-time FFT analysis and spectrogram plotting for visualizing audio signals and noise reduction effects.

Overall, the algorithm combines various signal processing techniques, including FFT, noise generation, statistical analysis, and filtering, to achieve noise reduction in audio signals, both in offline and real-time scenarios.

Block Diagram



Plotting audio from device

Python

```
noise_len = 2 # seconds
noise = band_limited_noise(min_freq=4000, mx_freq = 12000, samples=len(data), samplerate=rate)*10
noise_clip = noise[:rate*noise_len]
audio_clip_band_limited = data+10*noise


fig, ax = plt.subplots(figsize=(20, 4))
ax.plot(data)

# Display the audio
ipd.display(ipd.Audio(data, rate=rate))
```

Plotting Noise

```
fig,ax=plt.subplots(figsize=(20,3))
ax.plot(noise)

IPython.display.Audio(data=noise,rate=rate)
```

Noise+OGAudio Plot

```
#noise added
noise_len = 2 # seconds
noise = band_limited_noise(min_freq=4000, max_freq = 12000, samples=len(data), samplerate=rate)*10
noise_clip = noise[:rate*noise_len]
audio_clip_band_limited = data+10*noise

fig.ax= plt.subplots(figsize=(20,3))
ax.plot(audio_clip_band_limited)

IPython.display.Audio(data=audio_clip_band_limited,rate=rate)
```

Plotting Filtered#1 Audio

Python

```
def band_limited_noise(min_freq, samples=1024,samplerate=1):
    freqs= np.abs(np.fft.fftfreq(samples, 1/samplerate))
    f= np.zeros(samples)
    f[np.logical_and(freqs >= min_freq, freqs <= max_freq)]=1
    return fftnoise(f)
```

Apply Stationary Noise technique

```
#Noise reduce for stationary noise
noise_reduced= nr.reduce_noise(y=data, sr=rate, prop_decrease=0, stationary=False)

fig.ax= plt.subplots(figsize=(15,4))
ax.plot(reduced_noise)

fig, axs = plt.subplots(nrows=2, figsize=(15,6))
axs[0].plot(data[3000:5000])
axs[0].plot(noise_reduced[3000:5000])
axs[0].plot(data)
axs[0].plot(noise_reduced)
```

Plotting Filtered Audio#2

```
#sec time filter
fig, ax = plt.subplots(figsize=(15,5))
ax.plot(audio_clip_band_limited)
ax.plot(reduced_noise)

IPython.display.Audio(data=reduced_noise, rate=rate)
```

Short Time Fourier Transform

```
import time
from datetime import timedelta as td
import librosa
import scipy.signal
import numpy as np
import matplotlib.pyplot as plt

def _stft(y, n_fft, hop_length, win_length):
    return librosa.stft(y=y, n_fft=n_fft, hop_length=hop_length, win_length=win_length)
```

Inverse STFT to recover the audio

```
def istft(y, hop_length, win_length):
    return librosa.istft(y, hop_length, win_length)
```

Amplitude to Decibel Conversion

```
def amp_to_db(x):
    return librosa.core.amplitude_to_db(x, ref=1.0, amin=1e-20, top_db=80.0)
```

Decibel to Amplitude Conversion

```
def db_to_amp(x):
    return librosa.core.db_to_amplitude(x, ref=1.0)
```

Plotting the spectrogram with a motive to view the computed signal as the input

Python

```
def plot_spectrogram(signal, title):
    fig, ax = plt.subplots(figsize=(20, 4))
    cax = ax.matshow(
        signal,
        origin="lower",
        aspect="auto",
        cmap=plt.cm.seismic,
        vmin=-1 * np.max(np.abs(signal)),
        vmax=np.max(np.abs(signal)),
    )
    fig.colorbar(cax)
    ax.set_title(title)
    plt.tight_layout()
    plt.show()
```

Calculate statistics over noise

```
#Calculate threshold
def plot_statistics_and_filter(mean_freq_noise, std_freq_noise, noise_thresh, smoothing_filter):
    fig, ax = plt.subplots(ncols=2, figsize=(20, 4))
    plt_std, = ax[0].plot(std_freq_noise, label='Std. power of noise')
    plt_std, = ax[0].plot(noise_thresh, label="Noise threshold (by frequency)")
    ax[0].set_title("Threshold for mask")
    ax[0].legend()
    cax = ax[1].matshow(smoothing_filter, origin="lower")
    fig.colorbar(cax)
    ax[1].set_title("Filter for smoothing mask")
    plt.show()
```

Spectral filtering to remove noise from audio recordings

```
def removeNoise(
    audio_clip,
    noise_clip,
    n_grad_freq=2,
    n_grad_time=4,
    n_fft=2048,
    win_length=2048,
    hop_length=512,
    n_std_thresh=1.5,
    prop_decrease=1.0,
    verbose=False,
    visual=False,
):
    if verbose:
        start = time.time()

    noise_stft = _stft(noise_clip, n_fft, hop_length, win_length)
    noise_stft_db = amp_to_db(np.abs(noise_stft))
```

```
    # Calculate statistics over noise
    mean_freq_noise = np.mean(noise_stft_db, axis=1)
    std_freq_noise = np.std(noise_stft_db, axis=1)
    noise_thresh = mean_freq_noise + std_freq_noise * n_std_thresh

    if verbose:
        print("STFT on noise:", td(seconds=time.time() - start))
        start = time.time()

    sig_stft = _stft(audio_clip, n_fft, hop_length, win_length)
    sig_stft_db = amp_to_db(np.abs(sig_stft))

    if verbose:
        print("STFT on signal:", td(seconds=time.time() - start))
        start = time.time()

    # Calculate value to mask db
    mask_gain_db = np.min(amp_to_db(np.abs(sig_stft)))
    print(noise_thresh, mask_gain_db)
```

BLANK……

Imports Library and Take audiofile of 16bits input from device

Python

```
import IPython
from scipy.io import wavfile
import scipy.signal
import scipy
import numpy as np
import matplotlib.pyplot as plt
import librosa


import matplotlib.pyplot as plt
import IPython.display as ipd
import soundfile as sf

import urllib.request

import wave
%matplotlib inline

#load data
wav_loc = r'/Users/shawez/Noise-Reduction/Audiofile.wav'
rate, data = wavfile.read(wav_loc,mmap=False)
data=data/32768
```

To generate white noise with a specific frequency spectrum using fftnoise()

```
def fftnoise(f):
    f = np.array(f, dtype="complex")
    Np = (len(f) - 1) // 2
    phases = np.random.rand(Np) * 2 * np.pi
    phases = np.cos(phases) + 1j * np.sin(phases)
    f[1 : Np + 1] *= phases
    f[-1 : -1 - Np : -1] = np.conj(f[1 : Np + 1])
    return np.fft.ifft(f).real
```

Creating band-limited noise() within a given frequency range in time domain

```
def band_limited_noise(min_freq, max_freq, samples=1024, samplerate=1):
    freqs = np.abs(np.fft.fftfreq(samples, 1 / samplerate))
    f = np.zeros(samples)
    f[np.logical_and(freqs >= min_freq, freqs <= max_freq)] = 1
    return fftnoise(f)
```

BLANK……

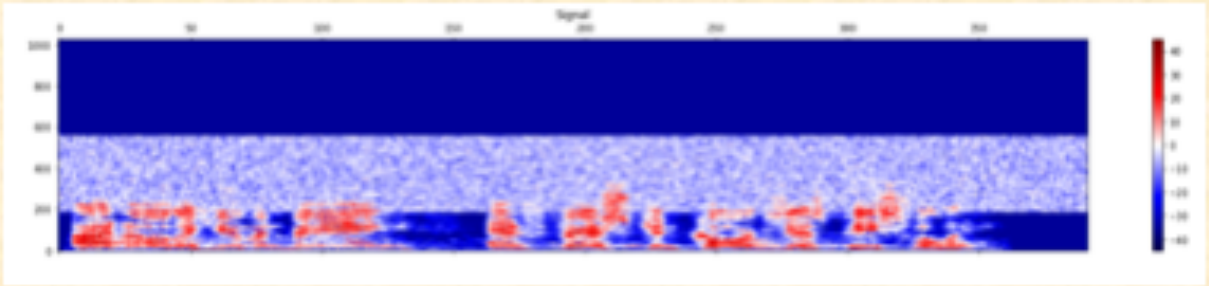
```
    # Creating a smoothing filter
    smoothing_filter = np.outer(
        np.concatenate(
            [
                np.linspace(0, 1, n_grad_freq + 1, endpoint=False),
                np.linspace(1, 0, n_grad_freq + 2),
            ]
        )[1:-1],
        np.concatenate(
            [
                np.linspace(0, 1, n_grad_freq + 1, endpoint=False),
                np.linspace(1, 0, n_grad_freq + 2),
            ]
        ),
    )
    smoothing_filter = smoothing_filter - smoothing_filter / np.sum(smoothing_filter)

    db_thresh = np.repeat(
        np.reshape(noise_thresh, [1, len(mean_freq_noise)]),
        np.shape(sig_stft_db)[1],
        axis=0,
    ).T

    if verbose:
        print("Masking:", td(seconds=time.time() - start))
        start = time.time()
```

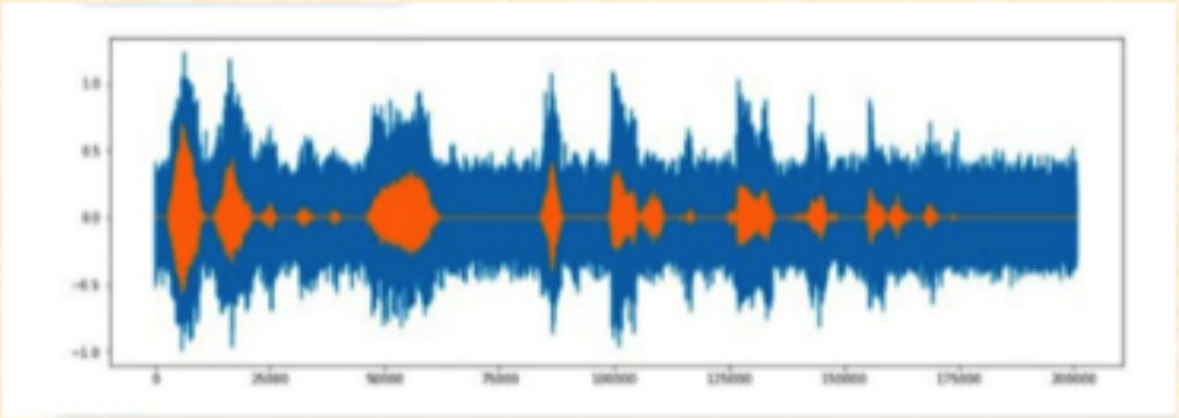
```
    # Convolve the mask with the smoothing filter
    sig_mask = scipy.signal.fftconvolve(sig_mask)
    print("Masking:", td(seconds=time.time() - start))
    start = time.time()

    # Convolve the mask with a smoothing filter
    sig_mask = scipy.signal.fftconvolve(sig_mask, smoothing_filter, mode="same")
    sig_mask = sig_mask * prop_decrease

    if verbose:
        print("Mask convolution:", td(seconds=time.time() - start))
        start = time.time()
```

```
    # Mask the signal
    sig_stft_db_masked = (
        sig_stft_db * (1 - sig_mask) + np.ones(np.shape(mask_gain_db)) * mask_gain_db * sig_mask
    )  # Mask real
    sig_imag_masked = np.imag(sig_stft) * (1 - sig_mask)
    sig_stft_amp = (
        db_to_amp(sig_stft_db_masked) * np.sign(sig_stft) + 1J * sig_imag_masked
    )

    if verbose:
        print("Mask application:", td(seconds=time.time() - start))
        start = time.time()

    # Recover the signal
    recovered_signal = istft(sig_stft_amp, hop_length, win_length)
    recovered_spec = amp_to_db(
        np.abs(_stft(recovered_signal, n_fft, hop_length, win_length))
    )

    if verbose:
        print("Signal recovery:", td(seconds=time.time() - start))

    if visual:
        plot_spectrogram(sig_stft_db, title="Signal")
        plot_statistics_and_filter(mean_freq_noise, std_freq_noise, noise_thresh, smoothing_filter)
        plot_spectrogram(sig_mask, title="Mask applied")
        plot_spectrogram(sig_stft_db_masked, title="Masked signal")
        plot_spectrogram(recovered_spec, title="Recovered spectrogram")

    return recovered_signal
```

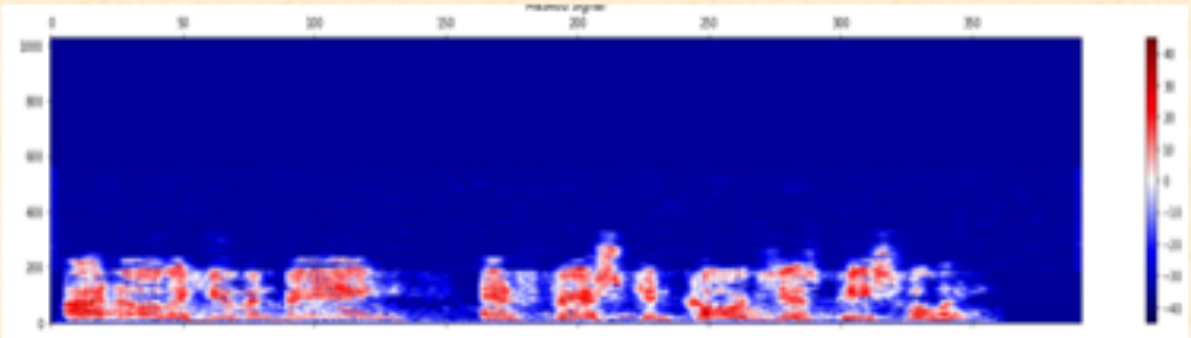
Chapter 9: Simulaton

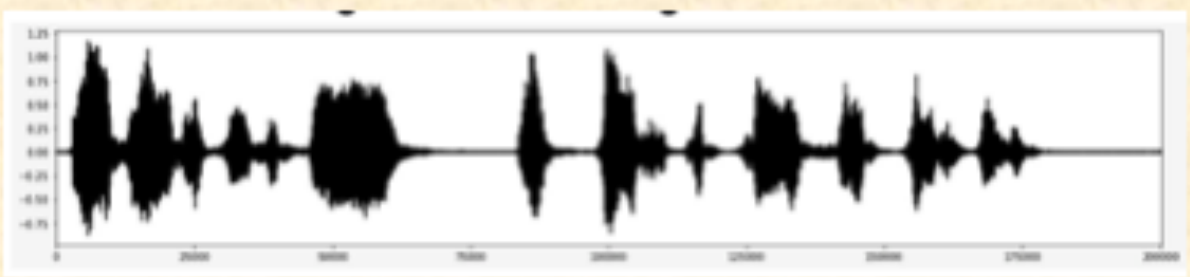
SIgnal with Noise





Signal recovered





Chapter 10

Conclusions

1. Conducted research: You began by reviewing multiple research papers to gather insights and establish a foundation for your project.

2. Utilized FFT: Fast Fourier Transform (FFT) is a powerful algorithm for analyzing the frequency content of signals. You applied FFT to both noise and data signals, likely to understand their frequency characteristics.

3. Set frequencies and wavelengths: By optimizing frequencies and wavelengths, you aimed to enhance signal processing efficiency and accuracy, possibly by selecting specific frequency ranges relevant to your analysis.

4. Added noise signal: Incorporating noise into the training data set can help improve the robustness of your model by exposing it to various conditions and enhancing its ability to generalize.

5. Applied STFT and ISTFT: Short-Time Fourier Transform (STFT) and Inverse Short-Time Fourier Transform (ISTFT) are essential for analyzing non-stationary signals over time. These techniques allow you to examine signal characteristics within short time intervals.

6. Converted amplitude into dB scale: Converting amplitude into decibel (dB) scale is a common practice in signal processing, as it provides a logarithmic representation that better aligns with human perception and facilitates analysis, especially in audio processing.

7. Implemented a threshold-based mask: By using a threshold-based mask, you aimed to eliminate unwanted frequencies or noise components from the signal, enhancing the signal-to-noise ratio and improving the quality of your analysis.

8. Convolved the mask with a smoothing filter: Convolution with a smoothing filter likely helped to refine the mask and smooth out any artifacts introduced during the thresholding process, leading to better signal recovery.

9. Completed all computations to reconstruct the audio signal: After applying various signal processing techniques and optimizations, you successfully reconstructed the audio signal, presumably achieving your desired outcome.

Bibliography

1. https://github.com/Paulattredies/Noise(Algorithm)
2. https://youtu.be/dZrShAGqT44?si=JwNnraKdiRrN-1y0(Fourier Transform using Python)
3. https://towardsdatascience.com/background-noise-removal-traditional-vs-ai-algorithms-9e7ec5776173(RNN bs Weiner filter)

1. https://www.kaggle.com/code/shawez/methods-for-sound-noise-reduction/edit(Bird classification)