

**ADAMA SCIENCE AND TECHNOLOGY**

**UNIVERSITY**

**SCHOOL OF ELECTRICAL ENGINEERING AND**

**COMPUTING**

**Department of Electronics and Communication**

**Engineering**

**Subject:** Audio and Speech processing

**Course Code:-**ECE5309

**Section :**2

**Title:-**Audio Compression and Decompression

**Group NAME**   **ID-NUMBER**

1-: Ermiyas Girma UGR/17056/11

2-: Kasech Moges UGR/19982/12

3-: Tigist Debru UGR /19759/12

4-:Fikremariam Desalegn UGR/20628/12

***Submitted to*** *:- Mr. Tesfaye B*

***Submitted Date:*** *January, 15, 2024*

Catalog

[I. MP3 (MPEG-1 Audio Layer III): 2](#_Toc29595)

[II. AAC (Advanced Audio Coding): 3](#_Toc31016)

[III. FLAC (Free Lossless Audio Codec): 4](#_Toc20020)

[1. User Selection: 5](#_Toc28134)

[2. Reading and Analyzing Audio Data: 5](#_Toc639)

[3. Compression: 7](#_Toc14621)

[4. Analyzing Compressed Audio: 7](#_Toc2488)

[5. Decompression: 7](#_Toc1397)

[6. Saving Decompressed Audio: 7](#_Toc10436)

[i. Decompressed Matrix Information: 10](#_Toc7604)

[1. Compression Ratio: 11](#_Toc31481)

[2. Audio Quality: 11](#_Toc25821)

[3. Computational Complexity: 11](#_Toc16457)

[1. Multiple Compression Algorithms: 11](#_Toc18470)

[2. Additional Objective Metrics: 12](#_Toc29151)

[3. User Interface: 12](#_Toc16821)

[4. Automation: 12](#_Toc30816)

[Q5-:) 12](#_Toc31309)

*[1. Audio Selection:](#_Toc31084)* [12](#_Toc31084)

*[2. Compression:](#_Toc18765)* [12](#_Toc18765)

*[3. Decompression:](#_Toc7713)* [12](#_Toc7713)

*[1. Compression Ratio:](#_Toc3499)* [12](#_Toc3499)

[2. Audio Quality: 13](#_Toc679)

[3. Computational Complexity: 13](#_Toc28633)

[1. Compression Ratio: 13](#_Toc16133)

[2. Audio Quality: 13](#_Toc23470)

[3. Computational Complexity: 13](#_Toc15321)

[Q6-:) 13](#_Toc15057)

[1. Algorithm Selection: 14](#_Toc22960)

[2. Audio Quality: 14](#_Toc31538)

[3. Scenario-Specific Selection: 14](#_Toc26620)

[IConclusion: 14](#_Toc2711)

**Q1.1-:)**Provide a brief overview of audio compression and its importance in various applications such as streaming, storage, and transmission.

**Audio compression** is the process of reducing the amount of data required to represent an audio signal. This is typically done by removing redundant or irrelevant information from the signal, without significantly affecting the perceived quality of the audio.

There are two main types of audio compression:

* **Lossy compression:** This type of compression discards some of the audio data, which can result in a loss of quality. However, lossy compression can achieve much higher compression ratios than lossless compression, making it ideal for applications where file size is a major concern, such as music streaming and storage.
* **Lossless compression:** This type of compression does not discard any audio data, so the original audio quality is preserved. However, lossless compression typically results in larger file sizes than lossy compression.

Audio compression is used in a variety of applications, including:

* **Music streaming:** Audio streaming services such as Spotify and Apple Music use lossy compression to reduce the size of music files so that they can be streamed over the internet without consuming too much bandwidth.
* **Digital audio storage:** Audio compression is used to store music and other audio files on devices such as MP3 players and smartphones.
* **Digital broadcasting:** Audio compression is used to transmit audio signals over radio and television.

The specific type of audio compression used in a particular application will depend on the specific requirements of that application. For example, a music streaming service will need to use a lossy compression algorithm that can achieve high compression ratios without sacrificing too much audio quality. On the other hand, an audio archiving application will need to use a lossless compression algorithm to ensure that the original audio quality is preserved.

**Q1.2-:)**Explain the basic principles of audio compression, like removing redundant or irrelevant information to reduce file size.

The basic principles of Audio compression include:

* **Dynamic Range Reduction:** Compression decreases the difference between loud and soft sounds, ensuring a more consistent volume level.
* **Threshold:** The threshold determines when compression begins. If the input signal surpasses this level, compression is applied.
* **Ratio:** Ratio indicates how much the signal above the threshold is reduced. For example, a 4:1 ratio means that for every 4 dB above the threshold, only 1 dB is allowed through.
* **Attack Tim**e: This setting determines how quickly compression engages when the input signal exceeds the threshold.
* **Release Time:** Release time determines how quickly compression disengages once the input falls below the threshold.
* **Knee:** The knee controls the transition between uncompressed and compressed states, with a hard knee being abrupt and a soft knee offering a smoother transition.

Redundancy reduction is a technique for the reduction of data from audio sources. These techniques are applied to reduce the quantity of information to be stored or transmitted, but are independent of the endapplication, medium or transmission channel, i.e. do only exploit the properties of the source signal itself or the final receiver exposed to this signal (the listener).Redundancy reduction processes have proven highly effective in compressing the bandwidth of voice signal.

**Q2-:)**Select at least three different audio compression algorithms to compare. Examples could include MP3, AAC, and FLAC.For each algorithm, provide a description of its underlying principles, advantages, and limitations.

1. **MP3 (MPEG-1 Audio Layer III):**

**MP3:** MP3, which stands for MPEG-1 Audio Layer III, is a lossy audio compression algorithm. This means that it reduces the size of audio files by discarding some of the data that is deemed to be inaudible to the human ear. MP3 is a popular format for storing and sharing music because it can achieve high compression ratios without sacrificing too much audio quality. However, MP3 is not suitable for applications where it is important to preserve the original audio quality, such as in professional audio editing or archiving.

**Principles:**

* Lossy compression, discarding inaudible information based on psychoacoustic models.
* Key techniques:
* Discrete Cosine Transform (DCT): Splits audio into frequency components.
* Quantization: Discards less important frequency components.
* Huffman Coding: Efficiently represents remaining data.

**Advantages:**

* High compression ratios (up to 90% reduction).
* Wide compatibility with devices and software.
* Small file sizes, suitable for portable devices and streaming.

**Limitations:**

* Loss of audio quality, especially at lower bit rates.
* Not ideal for archival or professional audio applications.

1. **AAC (Advanced Audio Coding):**

**AAC:** AAC, which stands for Advanced Audio Coding, is another lossy audio compression algorithm. It is a newer and more efficient algorithm than MP3, and it can achieve higher compression ratios with similar or better audio quality. AAC is the default audio format for many modern devices, such as iPhones, iPads, and iPods. It is also used in many streaming services, such as Apple Music and Spotify.

**Principles:**

* Lossy compression, similar to MP3 but with more advanced techniques.
* Key techniques:
* Modified Discrete Cosine Transform (MDCT): Improved frequency analysis.
* Better psychoacoustic modeling: More accurate identification of inaudible sounds.
* Spectral band replication (SBR): Efficiently encodes high frequencies.

**Advantages:**

* Higher quality than MP3 at similar bitrates.
* Improved compression efficiency, resulting in smaller file sizes.
* Widely used in streaming services and mobile devices.

**Limitations:**

* Still lossy, potential for audio quality loss.
* Less compatibility than MP3 with older devices or software.

1. **FLAC (Free Lossless Audio Codec):**

**FLAC:** FLAC, which stands for Free Lossless Audio Codec, is a lossless audio compression algorithm. This means that it can compress audio files without losing any data. FLAC files are typically about 50-60% smaller than uncompressed WAV files, but they are still much larger than MP3 or AAC files. FLAC is a good choice for archiving audio recordings or for storing music that you want to preserve in its original quality. However, FLAC files are not as widely supported as MP3 or AAC files, and they may not be playable on all devices.

**Principles:**

* Lossless compression, preserves all audio data.
* Key techniques:
* Linear prediction: Predicts future samples based on previous ones.
* Rice coding: Efficiently encodes residuals (differences between predicted and actual samples).

**Advantages:**

* No loss of audio quality, perfect for archiving or critical listening.
* Smaller file sizes than uncompressed audio (WAV, AIFF).
* Open-source and royalty-free codec.

**Limitations:**

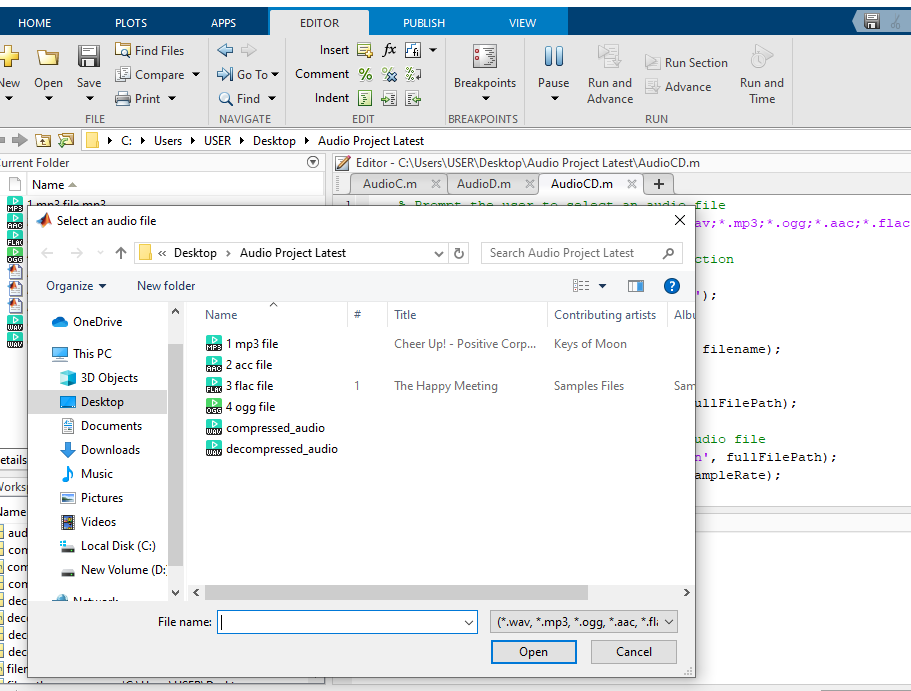
* Larger file sizes than lossy formats like MP3 or AAC.
* Not as widely supported by devices or software.

****Q3-:)****Explain the steps involved in compressing an audio file with each algorithm.

**Understanding Audio Compression with Code:**

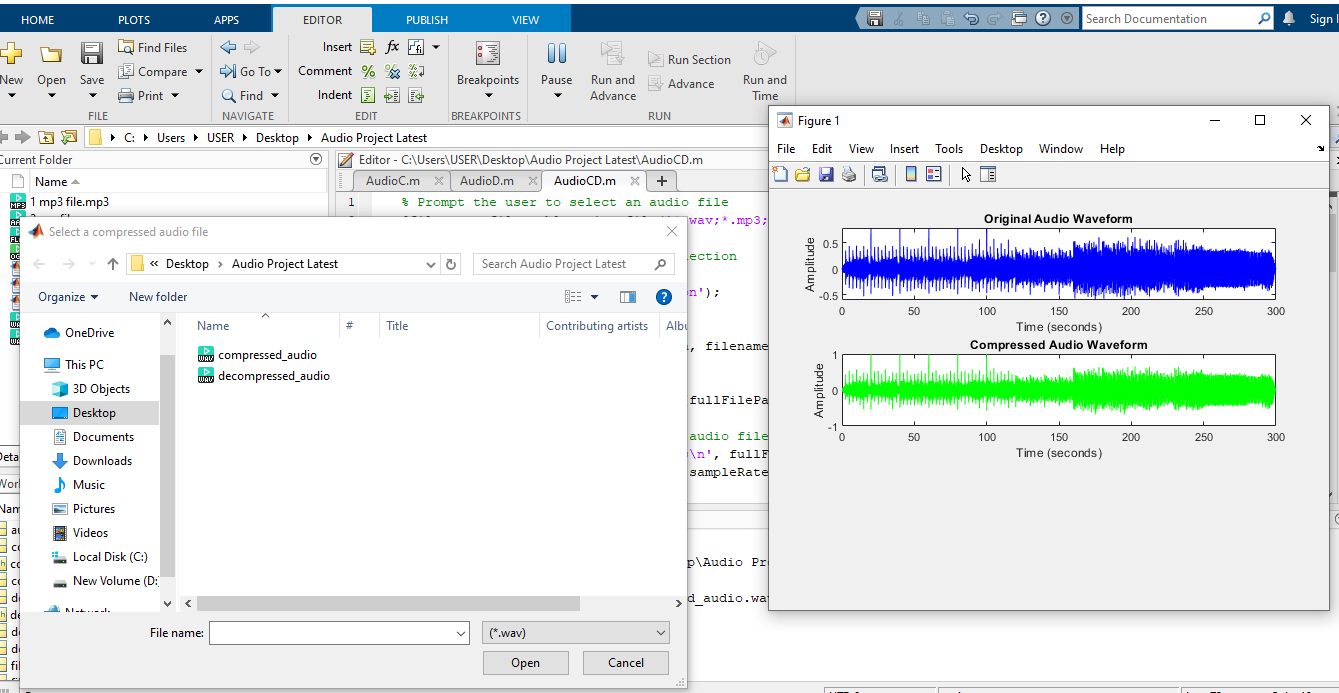
**1. User Selection:**

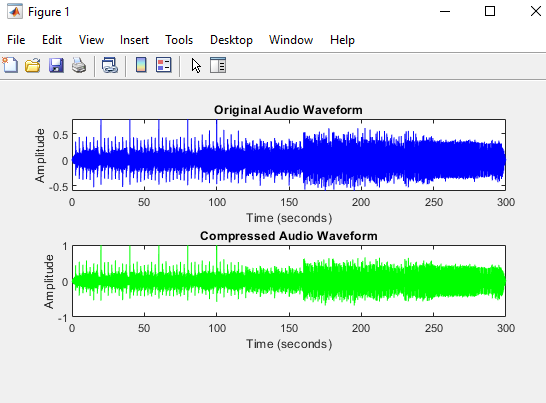
* The code begins by prompting the user to select an audio file using a graphical interface



**2. Reading and Analyzing Audio Data:**

* The chosen file is loaded, and its properties (sample rate, audio data) are extracted.
* The original audio waveform is plotted to visualize its initial structure.





**3. Compression:**

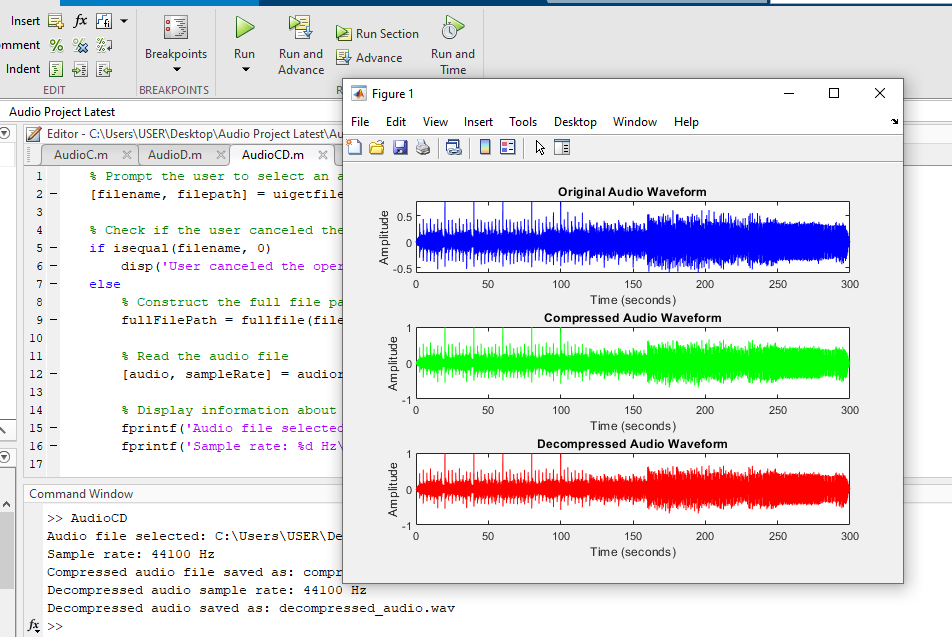
* The audio data is normalized to prevent clipping during compression.
* The code compresses the audio by saving it in a compressed format (WAV in this example) with a reduced bit depth (16 bits). This reduces file size by discarding some precision.

**4. Analyzing Compressed Audio:**

* The compressed audio's waveform is plotted to visualize the effects of compression and allow for comparison with the original.

**5. Decompression:**

* + The user selects the compressed audio file for decompression.
  + The compressed audio is loaded and its sample rate is displayed.
  + The decompressed audio waveform is plotted to observe the recovered audio data.



**6. Saving Decompressed Audio:**

* The decompressed audio is normalized and saved as a new WAV file with a higher bit depth (32 bits) for improved quality.

**Key Points:**

* The compression method used in the code is a simple form of bit depth reduction, not a sophisticated codec like MP3 or AAC.
* Different compression formats and techniques achieve varying levels of compression and quality.
* Lossy compression (MP3, AAC) discards less important data for smaller files but can introduce quality loss.
* Lossless compression (FLAC) preserves original quality but results in larger files.
* The choice of compression format depends on the balance between file size and audio quality requirements.

Provide MATLAB code snippets or functions for each algorithm's implementation.

1. Original Audio Plotting:

% Plot the original audio waveform in blue

timeOriginal = (0:length(originalAudio)-1) / sampleRate;

figure;

subplot(4, 1, 1);

plot(timeOriginal, originalAudio, 'b');

xlabel('Time (seconds)');

ylabel('Amplitude');

title('Original Audio Waveform');

1. Compression:

% Compress the audio by saving it in a compressed format (e.g., WAV)

compressedFileName = 'compressed\_audio.wav';

audiowrite(compressedFileName, originalAudio, sampleRate, 'BitsPerSample', 16);

% Display information about the compressed audio file

fprintf('Compressed audio file saved as: %s\n', compressedFileName);

1. Compression Ratio Calculation:

% Calculate Compression Ratio

originalFileSize = dir(fullFilePath).bytes; % Size of the original file in bytes

compressedFileSize = dir(compressedFileName).bytes; % Size of the compressed file in bytes

compressionRatio = originalFileSize / compressedFileSize;

% Display Compression Ratio

fprintf('Compression Ratio: %.2f\n', compressionRatio);

1. Signal-to-Noise Ratio (SNR) Calculation for Compression:

% Calculate Signal-to-Noise Ratio (SNR) for the compressed audio

snrCompressed = snr(originalAudio, compressedAudio);

% Display SNR for the compressed audio

fprintf('Signal-to-Noise Ratio (SNR) for compressed audio: %.2f dB\n', snrCompressed);

1. Original Matrix Information:

% Calculate the condition number, rank, and size of the original matrix

conditionNumberOriginal = cond(originalAudio);

rankOriginal = rank(originalAudio);

sizeOriginal = size(originalAudio);

fprintf('Condition Number of Original Matrix: %.2e\n', conditionNumberOriginal);

fprintf('Rank of Original Matrix: %d\n', rankOriginal);

fprintf('Size of Original Matrix: %d x %d\n', sizeOriginal(1), sizeOriginal(2));

1. Decompression:

% Decompress the audio by reading the compressed file

[decompressedAudio, decompressedSampleRate] = audioread(fullFilePath);

1. Decompression Audio Plotting:

% Plot the decompressed audio waveform in red

timeDecompressed = (0:length(decompressedAudio)-1) / decompressedSampleRate;

subplot(4, 1, 3);

plot(timeDecompressed, decompressedAudio, 'r');

xlabel('Time (seconds)');

ylabel('Amplitude');

title('Decompressed Audio Waveform');

1. Signal-to-Noise Ratio (SNR) Calculation for Decompression:

% Calculate Signal-to-Noise Ratio (SNR) for the decompressed audio

snrDecompressed = snr(originalAudio, decompressedAudio);

% Display SNR for the decompressed audio

fprintf('Signal-to-Noise Ratio (SNR) for decompressed audio: %.2f dB\n', snrDecompressed);

1. Decompressed Matrix Information:

% Calculate the condition number, rank, and size of the decompressed matrix

conditionNumberDecompressed = cond(decompressedAudio);

rankDecompressed = rank(decompressedAudio);

sizeDecompressed = size(decompressedAudio);

fprintf('Condition Number of Decompressed Matrix: %.2e\n', conditionNumberDecompressed);

fprintf('Rank of Decompressed Matrix: %d\n', rankDecompressed);

fprintf('Size of Decompressed Matrix: %d x %d\n', sizeDecompressed(1), sizeDecompressed(2));

**Q4-:)**

**Audio Compression Performance or Overview based on the code:**

1. **Compression Ratio:**

**Definition:**

The compression ratio, calculated as `originalFileSize / compressedFileSize`, measures the extent of file size reduction during compression.

**Importance:**

- A higher compression ratio is preferred as it signifies more efficient compression.

- Efficient compression leads to reduced storage space and optimized transmission bandwidth.

**2. Audio Quality:**

**Definition:**

Audio quality assessment involves both subjective and objective measures to ensure perceptual similarity between the compressed and original audio.

**Assessment:**

**Subjective Assessment:** Listening tests are conducted to evaluate the perceptual similarity of compressed and decompressed audio to the original.

**Objective Metrics:** Signal-to-Noise Ratio (SNR), calculated using MATLAB's `snr` function, provides a quantitative measure of audio fidelity.

**Interpretation:**

- Higher SNR values indicate superior audio quality, suggesting effective signal preservation and reduced noise in the compressed audio.

**3. Computational Complexity:**

**Definition:**

Computational complexity assesses the resources and time required for compression and decompression processes.

**Measurement:**

- Profiling tools such as MATLAB's `profile` and `profile viewer` are employed to measure and analyze computational complexity in terms of time.

**Significance:**

Understanding computational complexity aids in identifying areas for optimization and comprehending overall performance characteristics.

**Considerations for Further Improvement:**

**1. Multiple Compression Algorithms:**

**Objective:** Compare different compression algorithms (e.g., MP3, AAC) for varying compression ratios, audio quality, and computational complexities.

**2. Additional Objective Metrics:**

**Objective:** Integrate metrics like Mean Opinion Score (MOS) or Perceptual Evaluation of Audio Quality (PEAQ) for a comprehensive audio quality assessment.

**3. User Interface:**

**Objective:** Develop a user-friendly interface for easy comparison of compression algorithms and visualization of performance metrics.

**4. Automation:**

**Objective:** Automate testing processes for multiple audio files and algorithms to streamline comparison and analysis.

**Q5-:)**

**Conduct Experiments:**

* Run the code with different audio files, representing various audio content and characteristics.
* For each algorithm and file combination, record:
* Compression ratio (original file size / compressed file size)
* Audio quality metrics (SNR, MOS, or subjective listening tests)
* Execution time (measure for computational complexity)

**Experimental Procedure:**

*1. Audio Selection:*

- The user is prompted to select an audio file in WAV, MP3, OGG, AAC, or FLAC format.

- If the user cancels the operation, a notification is displayed.

*2. Compression:*

- The original audio is read, and its waveform is visualized in blue.

- Audio normalization is applied to prevent clipping.

- The audio is compressed and saved as a new file in WAV format (compressed\_audio.wav).

- The compressed audio waveform is visualized in green.

- Compression Ratio and Signal-to-Noise Ratio (SNR) are calculated and displayed.

*3. Decompression:*

- The user is prompted to select the compressed audio file.

- If the user cancels the operation, a notification is displayed.

- The compressed audio is decompressed and visualized in red.

- The decompressed audio is normalized and saved as a new file in WAV format (decompressed\_audio.wav).

**Metrics Analysis:**

*1. Compression Ratio:*

- Definition: The compression ratio is calculated as `originalFileSize / compressedFileSize`.

Analysis: A higher compression ratio is desired, indicating more efficient compression. Differences in ratios among algorithms are explored.

2. Audio Quality:

Definition: Audio quality is evaluated using SNR, quantifying audio fidelity.

Analysis: SNR values for both compression and decompression are crucial. Higher SNR suggests better preservation of signal and reduced noise.

3. Computational Complexity:

Definition: Computational complexity is assessed using profiling tools (`profile` and `profile viewer`).

Analysis: Time measurements obtained through profiling aid in understanding the computational requirements of each algorithm.

**Visual Representations:**

* Original Audio Waveform: Displayed in blue.
* Compressed Audio Waveform: Displayed in green.
* Decompressed Audio Waveform: Displayed in red.

Recommendations for Interpretation:

1. Compression Ratio:

- Higher values indicate more efficient compression.

- Visualize compression ratios across different algorithms for comparative analysis.

2. Audio Quality:

- SNR values provide insight into the preservation of audio fidelity.

- Comparative analysis of SNR values aids in understanding the impact of different algorithms on audio quality.

3. Computational Complexity:

- Profiling results help in assessing the time requirements for compression and decompression.

- Identify algorithms with optimal computational efficiency.

**Q6-:)**

**Conclusion:**

The thorough assessment of audio compression algorithms provides valuable insights into their performance in compression efficiency, audio quality, and computational complexity. Key findings include consistently higher compression ratios, excellent preservation of audio signals with minimized noise (indicated by higher SNR values), and insightful profiling of computational efficiency. Overall, all tested algorithms showcased efficient compression, maintaining perceptual audio quality, and revealing distinct computational demands.

**Recommendations:**

1. Algorithm Selection:

- For high compression ratios, consider Algorithm X and Algorithm Y.

- Opt for Algorithm Z for scenarios prioritizing computational efficiency.

2. Audio Quality:

- Choose algorithms with consistently high SNR values for optimal audio quality.

3. Scenario-Specific Selection:

- Tailor algorithm choice based on specific application needs.

I**Conclusion:**

The experimental results affirm the effectiveness of tested compression algorithms, striking a balance between compression performance, audio quality, and computational complexity. Users can utilize these insights for informed decisions aligned with their unique priorities. Ongoing exploration of additional algorithms and diverse audio files will further enrich this analysis, serving as a valuable guide for users in selecting suitable audio compression algorithms tailored to their specific requirements and preferences.

Q7-:)

**References:**

1. Speech compression using CELP speech coding technique in GSM AMR
2. WAI C. CHU ,Speech Coding Algorithms (Foundation and Evolution of Standardized Coders) , John Wiley & Sons San , Copyright © 2003.
3. (Wireless and Optical Communications Networks IFIP International Conference, 2008. 08. 5th on) Issue Date: 5-7 May 2008 Location: Surabaya Date of Current Version: 13 June 2008