Report: Music Signal Analysis Using Fourier Transform

1. Introduction

Music signals are complex, consisting of multiple frequencies that form a harmonic structure. Analysing these signals allows us to understand their composition, modify them for creative purposes, or enhance audio quality. The Fourier Transform (FT) is a mathematical tool that decomposes a signal into its constituent frequencies, providing insights into its spectral characteristics.

This report focuses on implementing Fourier Transform concepts in Python for practical exploration. Key demonstrations include frequency extraction, filtering, reconstruction, and visualisation using spectrograms.

2. Fourier Transform: Theoretical Overview

The Fourier Transform converts a time-domain signal into its frequency-domain representation. For a continuous signal $x(t)$:

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt$$

where:
$x(t)$: Time-domain signal
$X(f)$: Frequency-domain representation of the signal
j : imaginary unit
f : Frequency

For discrete signals, the Discrete Fourier Transform (DFT) is used:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\frac{j2\pi kn}{N}}$$

where:
N : Number of samples
k : Frequency bin index
$x[n]$: Discrete time-domain signal
$X[k]$: Frequency-domain representation

The Inverse Fourier Transform (IFT) reconstructs the time-domain signal from its frequency-domain representation.

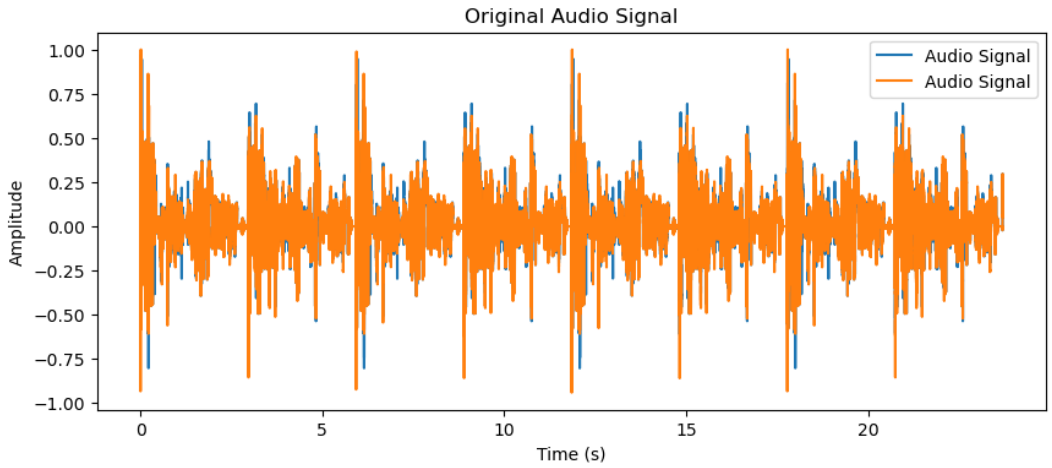
$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{\frac{j2\pi kn}{N}}$$

3. Practical Implementation in Python

Python's **numpy** and **scipy** libraries provide efficient tools for Fourier Transform computations. The following sections focus on implementing key operations: extracting frequencies, filtering, reconstruction, and spectrogram visualisation.

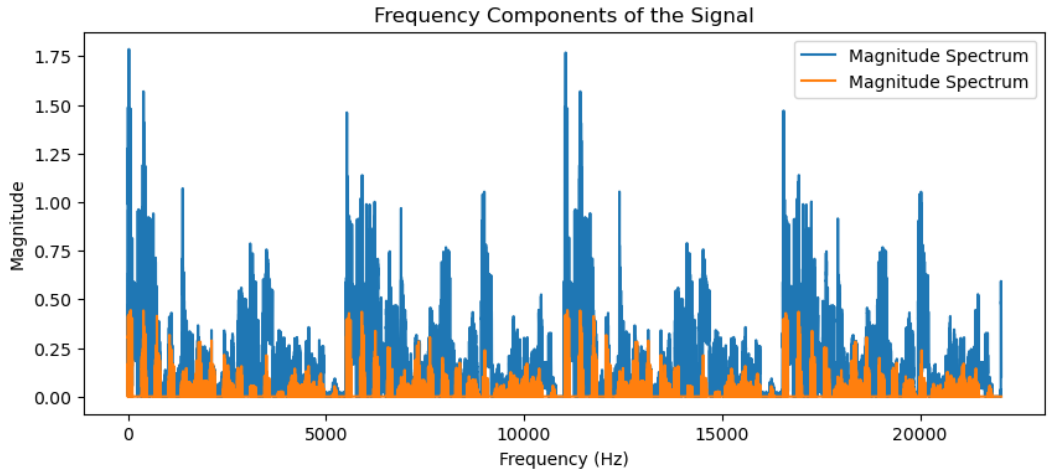
i. Loading and Visualising the Audio Signal

- **Audio Signal**: The audio file "**IndustrialDrumLoop.wav**" contains rhythmic drum beats, providing a clear representation of both low and high-frequency components.



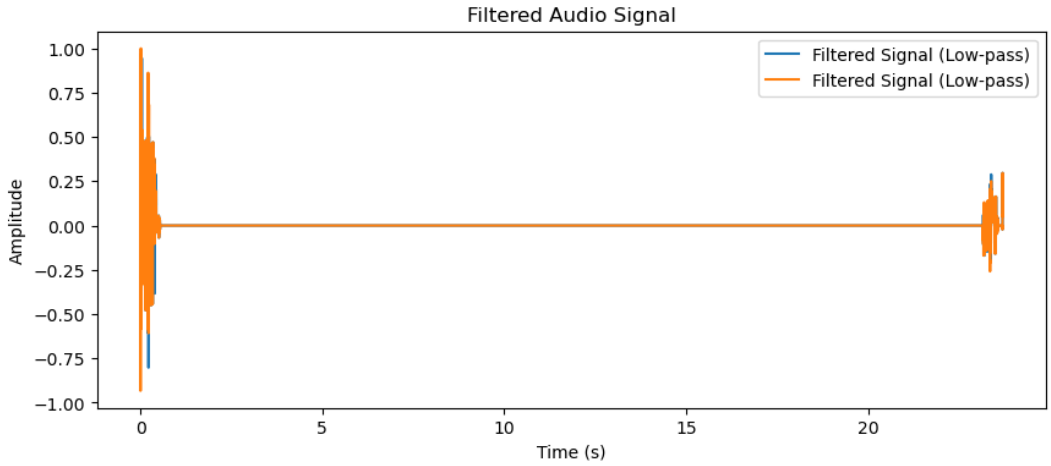
- **Observation**: The Industrial Drum Loop shows a repeating rhythmic pattern. The amplitude variations over time indicate periodic bursts of energy, characteristic of drum beats.

ii. Applying the Fourier Transform



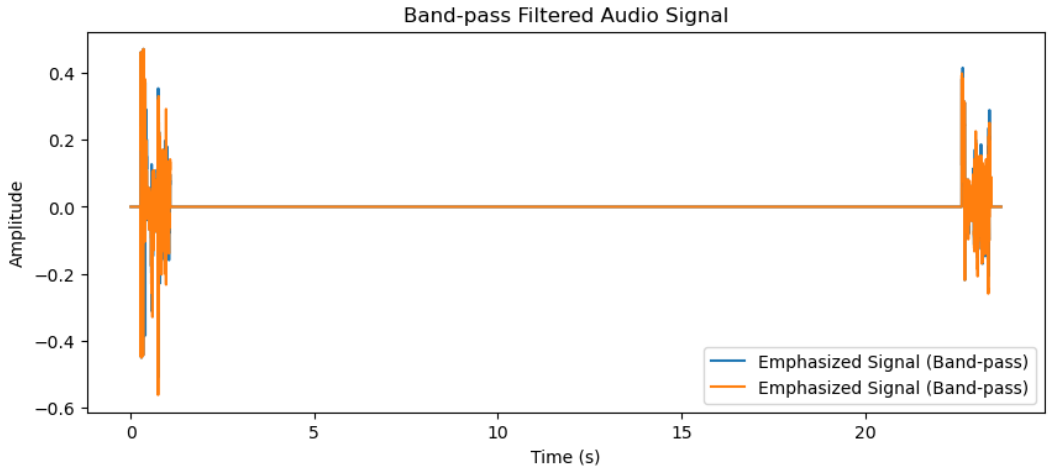
- **Observation**: The magnitude spectrum reveals multiple prominent peaks at various frequencies, indicating the presence of strong rhythmic components. The lower-frequency peaks represent the bass of fundamental drum hits, while the higher-frequency peaks correspond to sharper transients and total elements. The periodicity in the spectrum suggests a structured and repetitive nature of the drum loop.

iii. Frequency Filtering: Removing High Frequencies



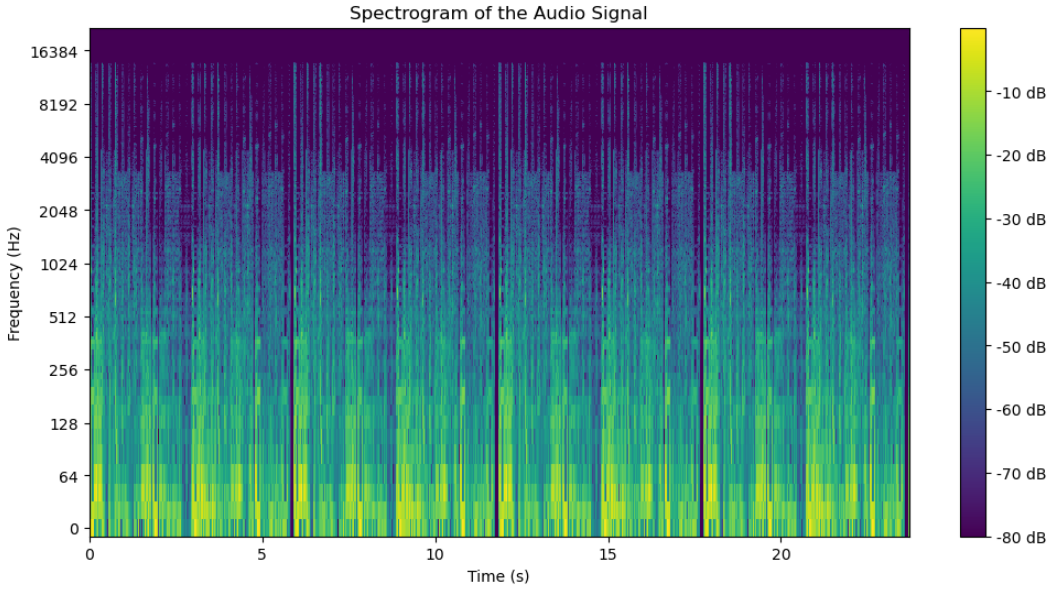
- **Observation**: After applying a low-pass filter with a cutoff frequency of 1000 Hz, the filtered signal retains only low-frequency components. This significantly reduces higher-frequency transients, resulting in a smoother waveform. Notably, most of the energy in the signal now appears at the start and end, corresponding to strong low-frequency components, while the middle of the signal has negligible amplitude.

iv. Band-Pass Filtering for Frequency Emphasis



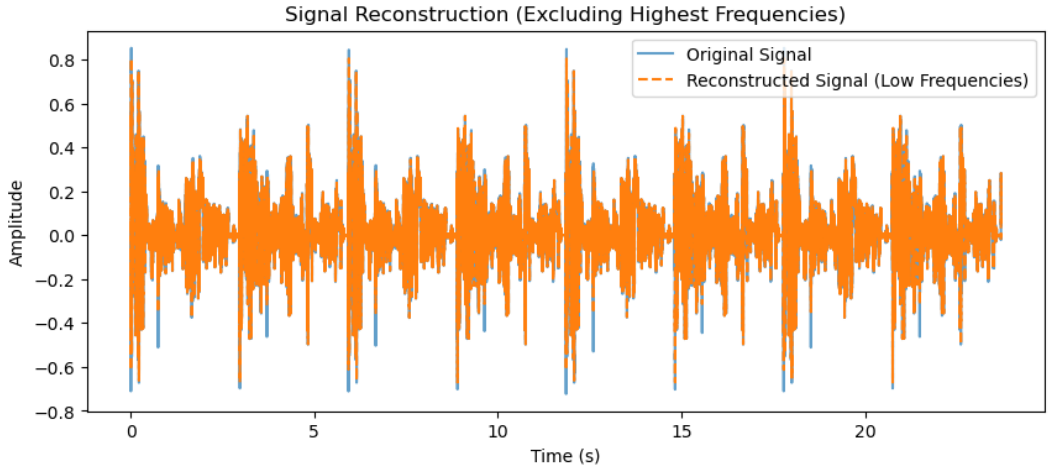
- **Observation**: After applying a band-pass filter to isolate frequencies between 500 Hz and 2000 Hz, the filtered signal retains the mid-range components of the audio. These correspond to the drum hits that lie within this frequency band, while both the low bass and high transients are suppressed. The waveform exhibits periodic bursts at the start and end of the signal, showing where the mid -frequency components are dominant.

v. Visualising Spectrogram



- **Observation**: The spectrogram of the Industrial Drum Loop reveals energy bursts in specific frequency bands corresponding to the drum hits. These bands appear as vertical streaks, highlighting the rhythmic structure.

vi. Signal Reconstruction



- **Visual Changes**: By excluding the highest frequencies, the reconstructed signal closely matches the original signal. However, sharp transients are slightly smoothed, indicating the absence of higher-frequency components.
- **Reconstruction Error:** The mean absolute reconstruction error (**0.0012**) remains minimal, showing that low frequencies capture the majority of the signal's energy.
- **Practical Application:**
  - This approach is useful for denoising signals where high-frequency noise dominates.
  - It is also beneficial in applications where only the primary low-frequency content is relevant, for e.g., speech signals or slow-varying musical notes.

4. Conclusion

The Fourier Transform effectively decomposes, filters, and reconstructs audio signals.

- Frequency analysis highlights the signal's structure, isolating critical components.
- Low-pass and band-pass filters demonstrate practical use cases, such as isolating bass or percussive sounds.
- Reconstruction confirms that lower frequencies carry most of the signal's energy, enabling compression while preserving quality.

This analysis can be applied in audio processing, including compression, noise removal, and feature extraction for machine learning tasks.

5. Limitations

a. **Fast Fourier Transform (FFT) Resolution**: The frequency resolution depends on the window size, and a balance between time and frequency resolution is necessary.
b. **Loss of Detail**: Filtering may result in the loss of high-frequency transients, impacting signal sharpness.
c. **Signal Assumptions**: The analysis assumes stationary signals, whereas real-world music signals often exhibit non-stationary behaviour.
d. **Reconstruction Quality**: Removing frequency bands reduces reconstruction accuracy, although low frequencies retain the core structure.

6. Bibliography

- Python libraries: NumPy, Matplotlib, SciPy, Librosa
- Textbook: John K. Hunter, Bruno Nachtergaele. *Applied Analysis*. Chapter 7: Fourier series.