EE200 Report: Audio Signal Restoration via Multi-Stage Cascade Filtering

Parv Gupta

June 30, 2025

1 Objective

The primary objective of this experiment is to restore a corrupted audio signal by removing multiple, distinct unwanted sounds. This is achieved by designing and implementing a cascade of digital filters tailored to specific frequency-domain problems identified in the source signal. The experiment aims to demonstrate the practical application of filter design, system analysis using Bode and Pole-Zero plots, and spectral analysis for verification.

2 System Overview and Tools Used

The entire analysis and restoration process was implemented using Python 3. The core functionalities were provided by the following open-source libraries:

- **Librosa**: For robust audio loading and time-frequency analysis.
- **NumPy**: For numerical operations and handling audio data as arrays.
- **SciPy (signal module)**: For filter design (`butter`), application (`filtfilt`), and spectral analysis (`welch`, `freqz`, `tf2zpk`).
- **Matplotlib**: For visualization of spectra, frequency responses, and pole-zero diagrams.
- **Soundfile**: For saving the final restored audio as a WAV file.
- **IPython.display**: For inline playback of audio in notebooks.

3 Theoretical Background

3.1 Digital Filters

A digital filter is a system that performs a mathematical operation on a discrete-time signal to reduce or enhance certain aspects of that signal. For LTI systems, the transfer function in the z-domain is:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 + \sum_{k=1}^{N} a_k z^{-k}} \tag{1}$$

3.2 Butterworth Filter

Butterworth filters are a type of IIR filter known for their maximally flat magnitude response in the passband, which avoids ripples and preserves tonal fidelity — essential for audio.

3.3 Filter Types Used

- **High-Pass Filter (HPF)**: Attenuates frequencies below the cutoff f_c while passing higher ones.
- **Band-Stop Filter (BSF)**: Suppresses frequencies between two specified cutoffs f_{low} and f_{high}.

3.4 System Analysis Tools

3.4.1 Bode Plot

The Bode plot describes the system's frequency response:

- **Magnitude (in dB)**:
$$20 \log_{10} |H(e^{j\omega})| \tag{2}$$
- **Phase**: Shows the phase shift introduced by the filter.

3.4.2 Pole-Zero Plot

This plot represents:

- **Poles ($\times$)**: System singularities, must lie inside the unit circle for stability.
- **Zeros (o)**: Frequencies where system response is zero.

3.5 Zero-Phase Filtering

The `filtfilt` function applies the filter forward and then backward, canceling phase distortion.

3.6 Spectral Analysis

- **Spectrogram**: Time-varying frequency content.
- **Power Spectral Density (PSD)**: Shows power vs. frequency using Welch's method.

4 Experimental Procedure

1. **Audio Loading**: The audio file `song_with_2piccolo.wav` was loaded.
2. **Filter Design**:
   - 5th-order HPF at 250 Hz.
   - Five 4th-order BSFs at 1100–1400 Hz, 1650–1850 Hz, 1420–1600 Hz, 1700–17200 Hz, and 4000–16000 Hz.
3. **Filter Validation**: Bode and Pole-Zero plots generated for each filter.
4. **Filter Application**: Filters applied using `filtfilt` to preserve phase.
5. **Normalization and Export**: Audio saved as `restored_music.wav`.
6. **Spectral Analysis**: Spectrograms, PSD, and FFT magnitudes plotted for both original and restored audio.
7. **Audio Playback**: Final output was audibly verified using Python.

5 Results and Observations

- **Bode Plots**: Confirmed correct frequency attenuation for each filter.
- **Pole-Zero Plots**: All filters were stable (poles inside the unit circle).
- **Spectrogram**: Clearly showed suppressed noise bands after filtering.
- **PSD**: Peaks at noisy frequencies were flattened in the restored signal.
- **Auditory Quality**: Audio was noticeably cleaner with reduced high-pitched and low-frequency interference.

6 Conclusion

The multi-stage cascade filter system successfully removed various types of interference in the audio. The experiment highlighted the effectiveness of combining theory (filter design and frequency analysis) with practice (Python implementation and audio processing). Bode plots and Pole-Zero plots were essential tools for verifying filter design, while PSD and spectrograms validated the final results.