Audio Loudness and Perception Analysis Project

Welcome to our presentation on audio loudness and perception analysis. analysis. We'll explore how we used Python to analyze various audio characteristics and their impact on human perception.

Selin Sargın

Efe Şahin

Serhat Erdoğan

Ali Batuhan Zeytun

Fatma Nur Küçükkeskin

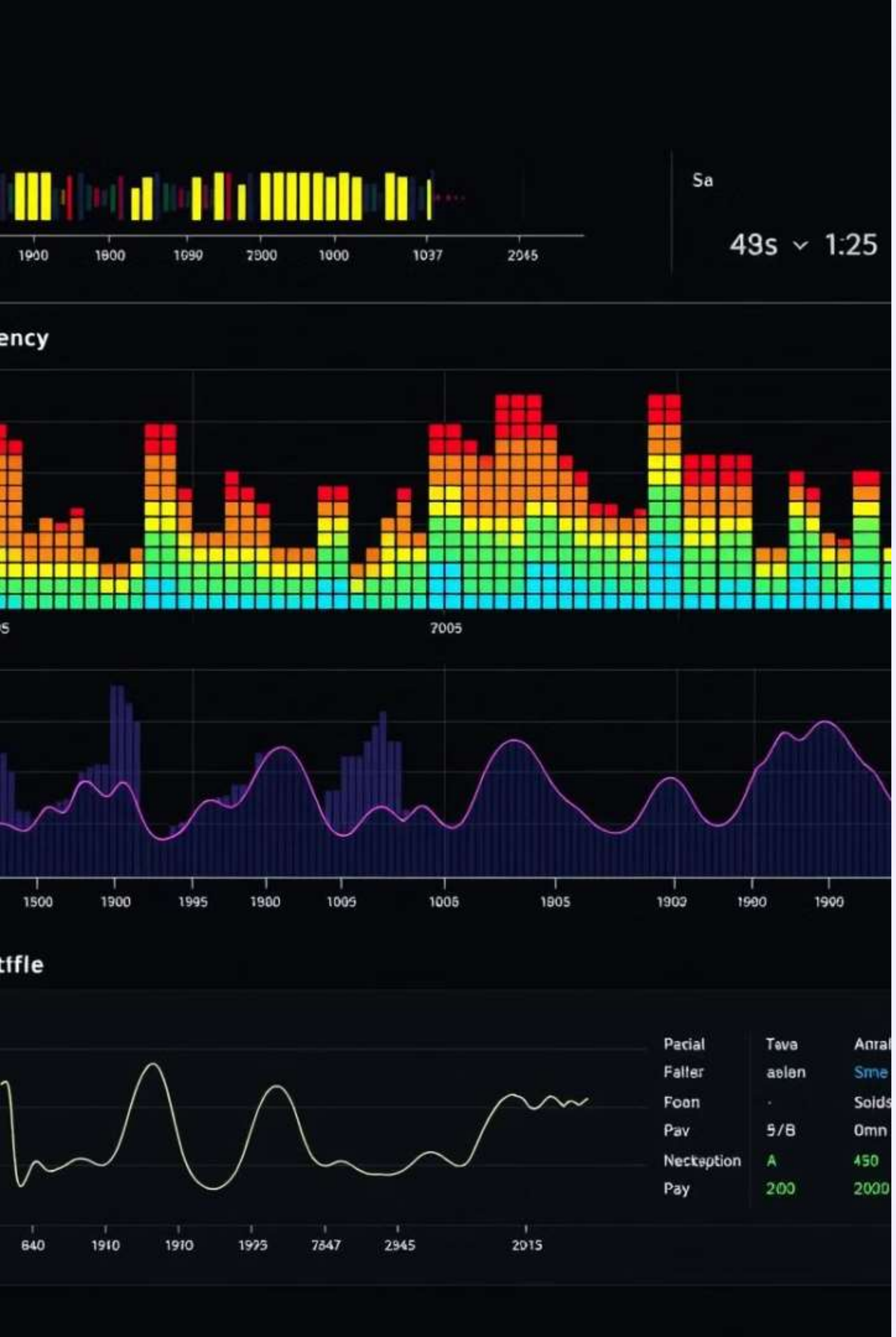
Project Overview

Objective

This project aims to analyze audio loudness and perception by evaluating RMS, dB values, and frequency components of selected audio files.

Methodology

Using Python, we analyze the audio data, visualize waveforms, frequency spectra, and perceptually adjusted A-Weighting metrics to gain insights into the relationships between audio characteristics and human perception.



Data Collection

We collected a diverse range of audio files, including music, speech, and environmental sounds, with varying loudness levels and frequency spectra.

Feature Extraction

Using Python libraries, we extracted relevant features from the audio files, such as RMS, dB values, and frequency spectra.

A-Weighting

We applied A-Weighting to the extracted features to adjust the human ear's frequency response, providing a more accurate representation of perceived loudness.

Load the audio file

```
audio_path = '1.wav'
y, sr = librosa.load(audio_path, sr=None)
print(f"Sampling Rate: {sr} Hz")
print(f"Frame Count: {len(y)}")
```

This code loads the audio file and calculates the sampling rate and frame count. The sampling rate determines the resolution of the audio

Visualization

```
plt.figure(figsize=(10, 4))
librosa.display.waveshow(y, sr=sr)
plt.title("Waveform")
plt.xlabel("Time (s)")
plt.ylabel("Amplitude")
plt.show()
```

This visualization shows the amplitude change of the sound over time.

RMS and dB Calculation

```
rms = np.sqrt(np.mean(y**2))
rms_db = 20 * np.log10(rms)
print(f"RMS Value: {rms:.5f}")
print(f"Sound Level (dB): {rms_db:.2f}")
```

RMS gives the energy level of the sound, and dB gives the logarithmic measurement of this level. This helps us understand how loud the sound is.

A-Weighting Part

```python
def a_weighting_filter(frequencies):
    """
    A-Weighting filter calculation.
    """
    f = frequencies
    Ra = (12194**2 * f**4) / ((f**2 + 20.6**2) * ((f**2 + 107.7**2
        )**0.5) * ((f**2 + 737.9**2)**0.5) * (f**2 + 12194**2))
    A = 20 * np.log10(Ra) - 20 * np.log10(12194**2 / ((20.6**2) *
        (12194**2)))
    return A
```

Weight (Ra) is calculated according to frequency values.

These weights are converted to decibel (dB) values on a logarithmic scale.

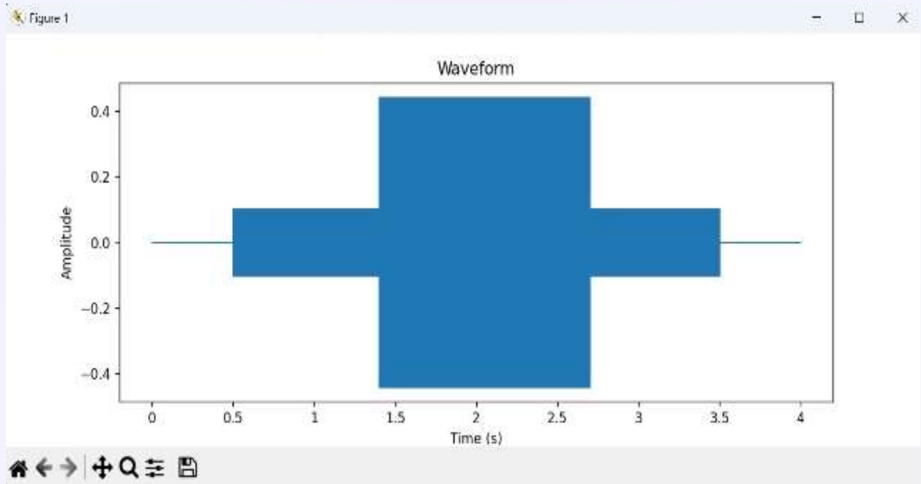
```python
a_weight = a_weighting_filter(fft_frequencies)

fft_magnitude_weighted = fft_magnitude + a_weight

plt.figure(figsize=(10, 4))
plt.plot(fft_frequencies, fft_magnitude_weighted)
plt.title("A-Weighted Frequency Spectrum")
plt.xlabel("Frequency (Hz)")
plt.ylabel("Weighted Magnitude (dB)")
plt.xlim(0, sr // 2)
plt.show()
```

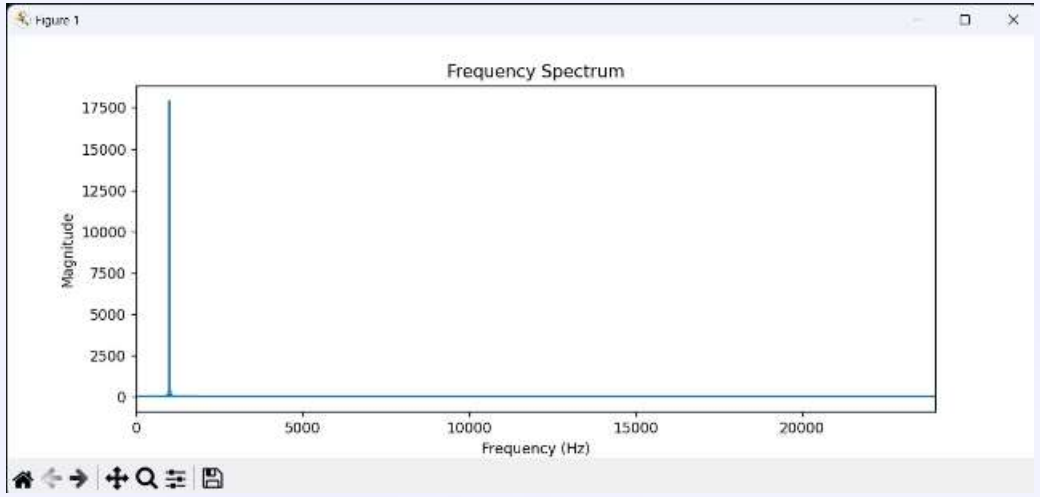
The frequency spectrum is weighted with the A-Weighting filter. The results are visualized in a way that suits human perception. The graph clearly shows the perceptual magnitudes and effects of the frequency components.

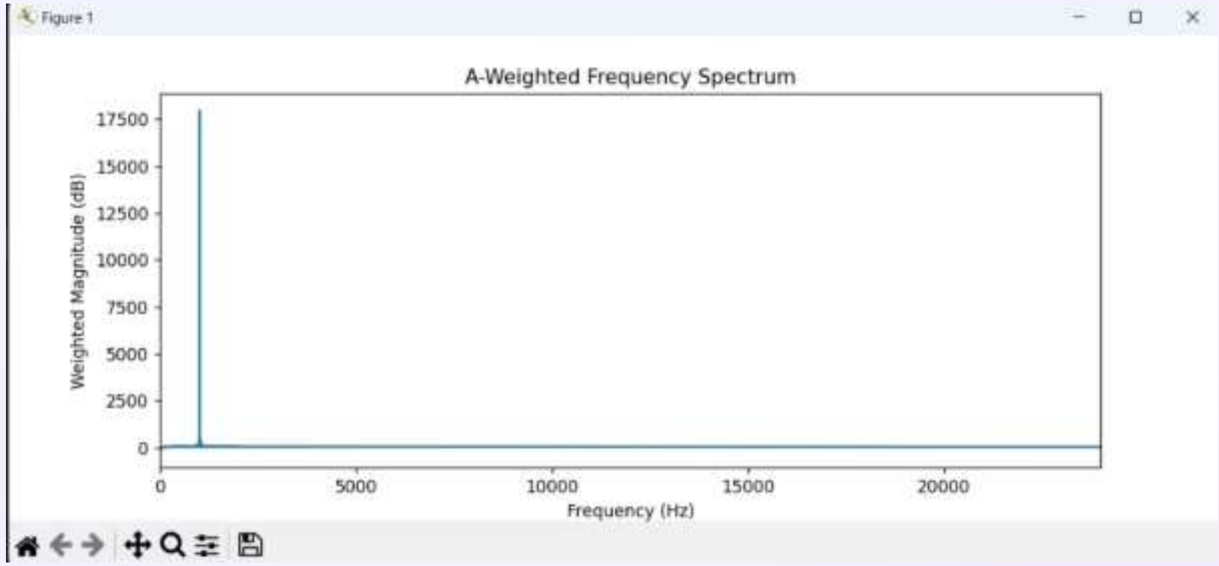
RelgateTest.wav



This graph shows the waveform of the RelgateTest.wav file. The horizontal axis represents time, while the vertical axis shows amplitude values. The waveform has a regular square wave shape, indicating that the signal has a periodic character.

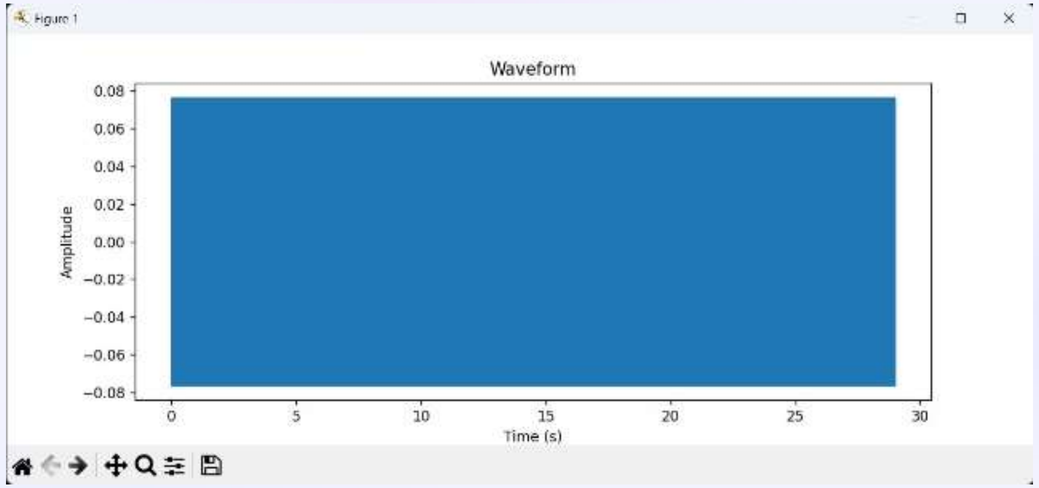
Here, the raw frequency spectrum of the file is displayed. A strong peak at low frequencies is visible, showing that the fundamental tone of the signal is very prominent.

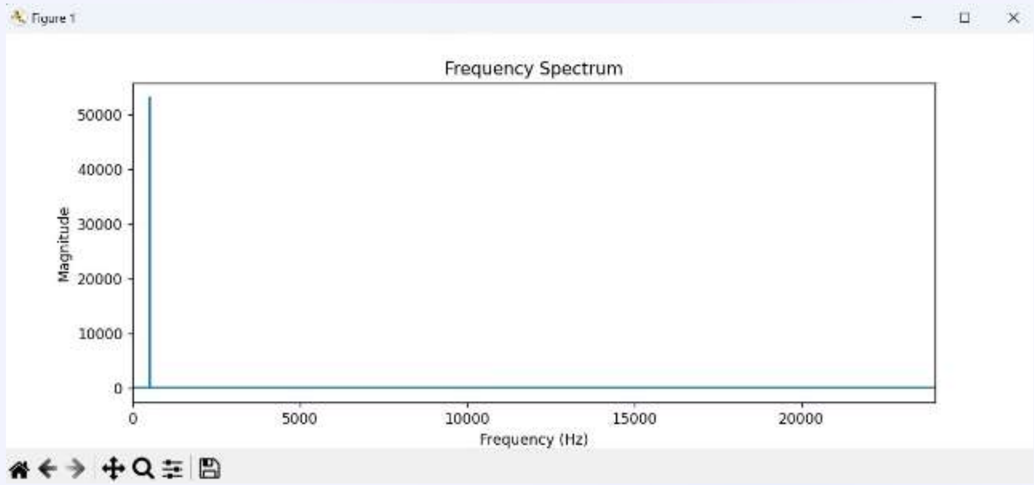


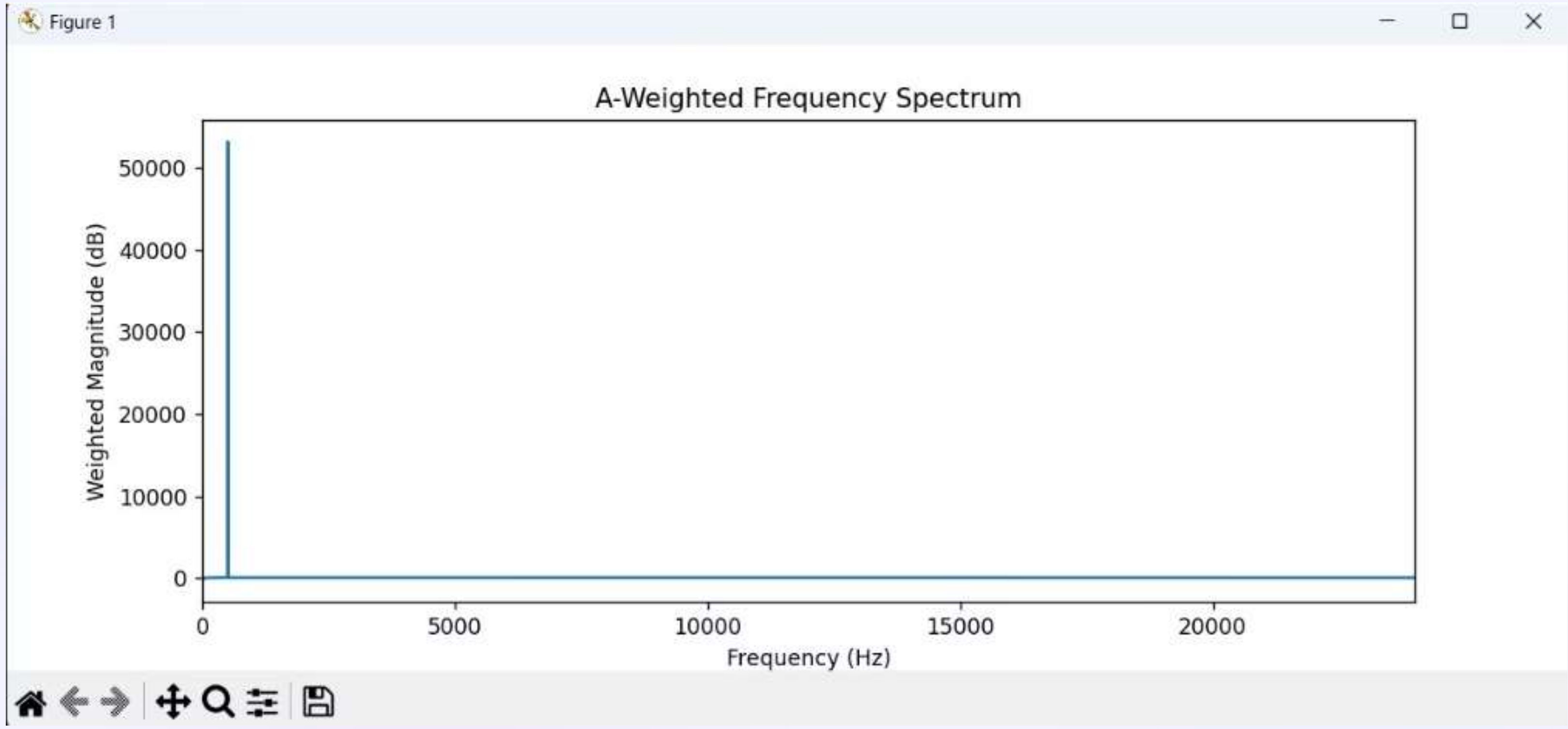


The final graph presents the A-weighted frequency spectrum. Adjusted to match the human ear's perception, this analysis highlights the dominance of low frequencies, while higher frequencies are less emphasized.

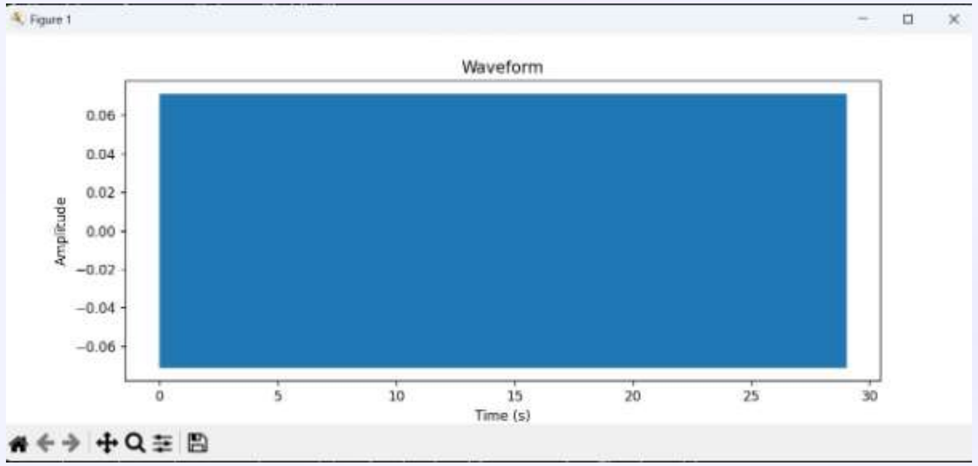
500Hz.2ch.wav

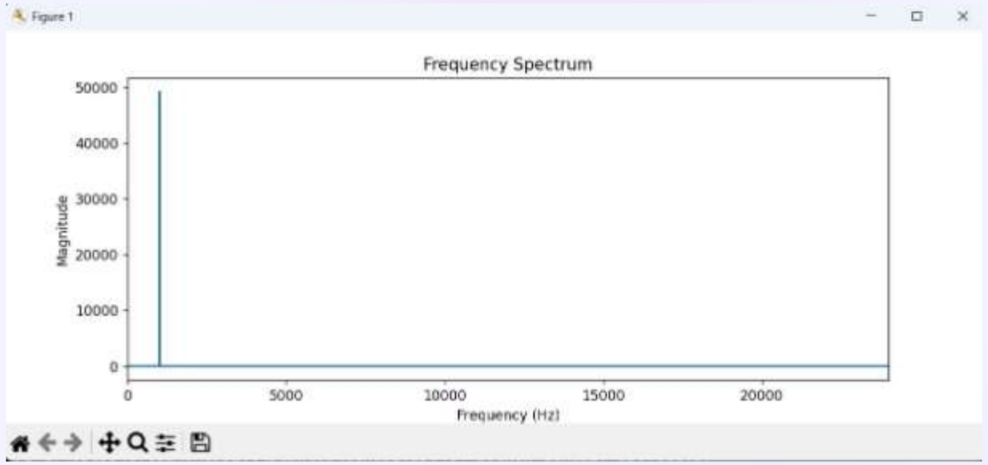


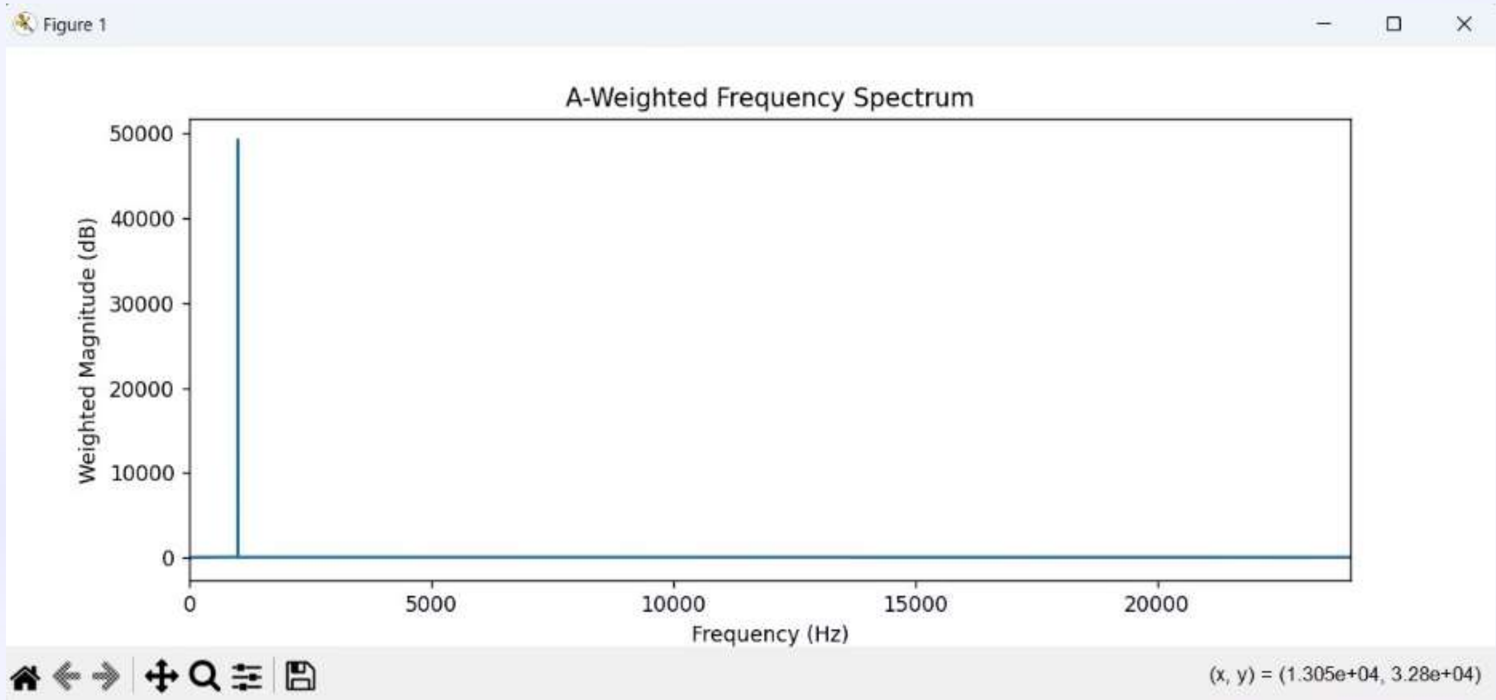




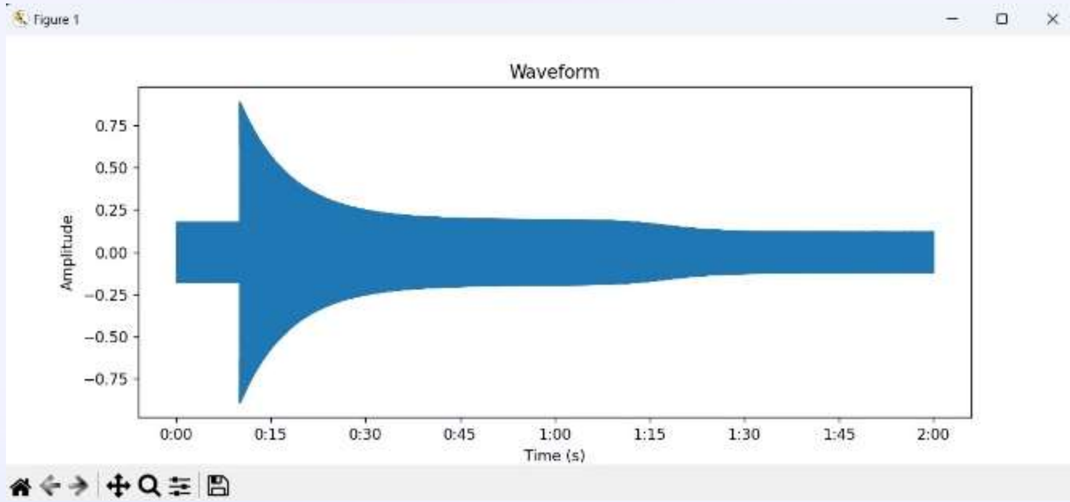
1000Hz.2ch.wav

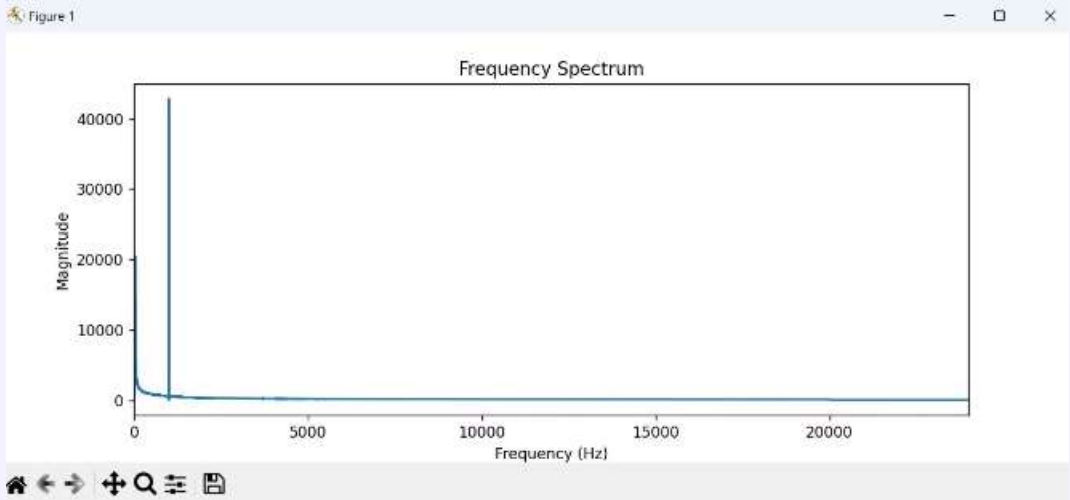


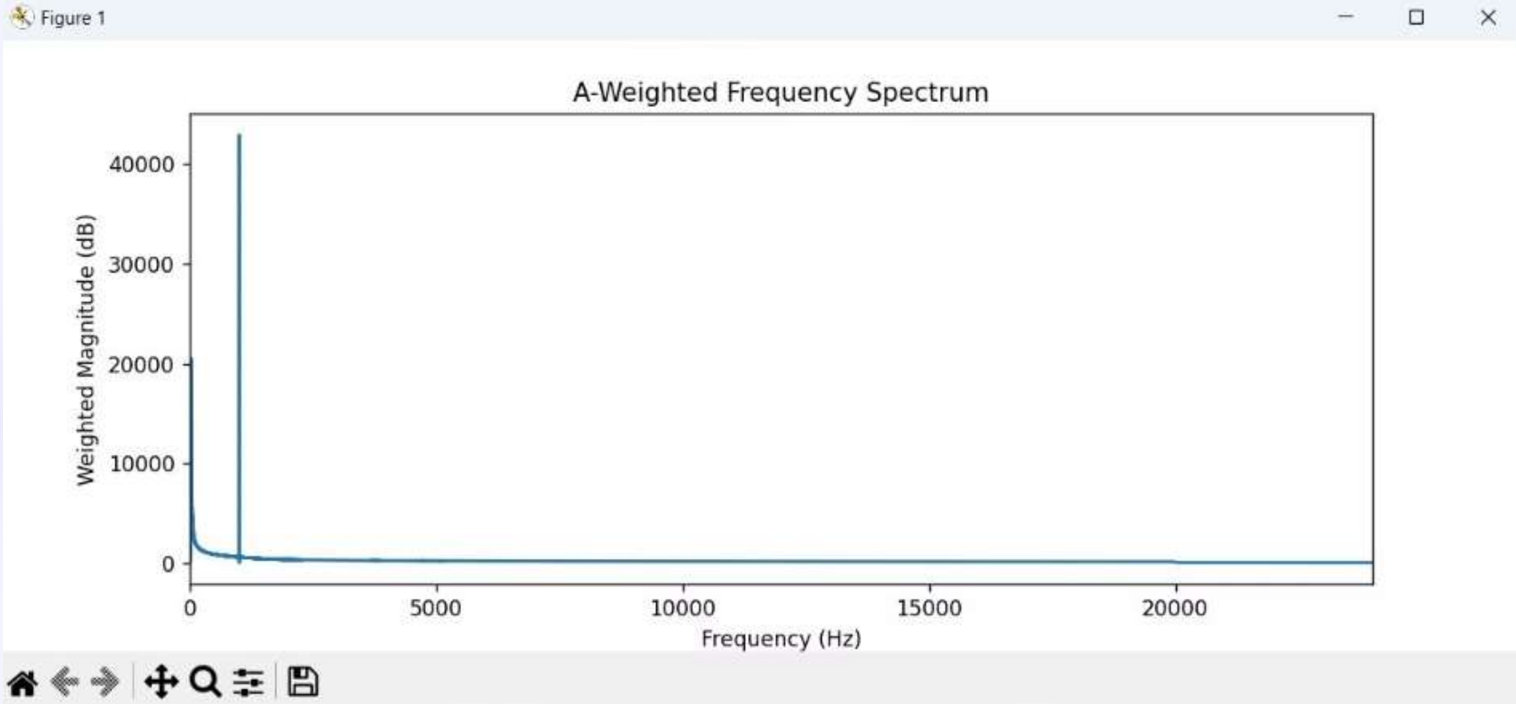




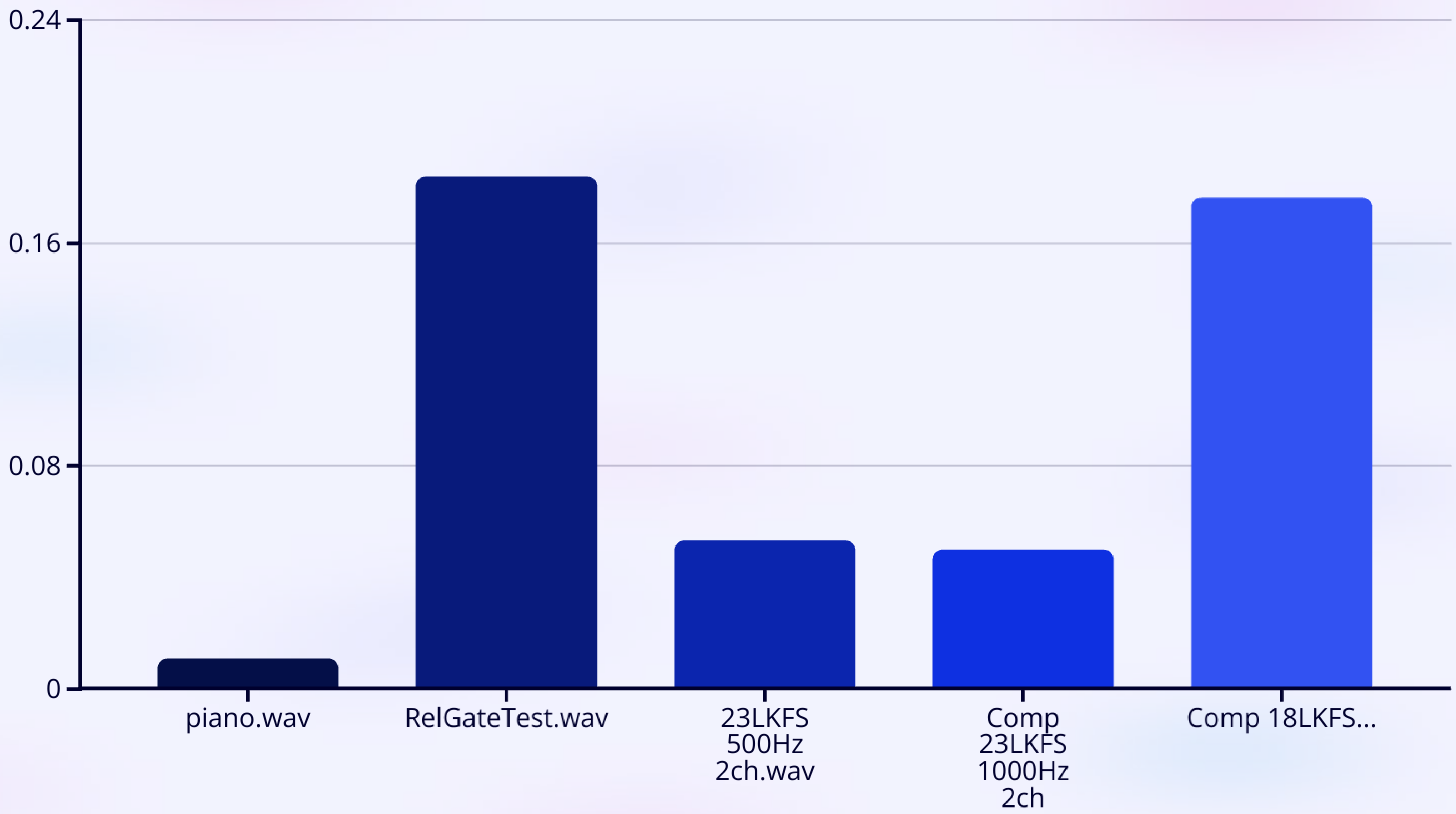
FrequencySweep.wav





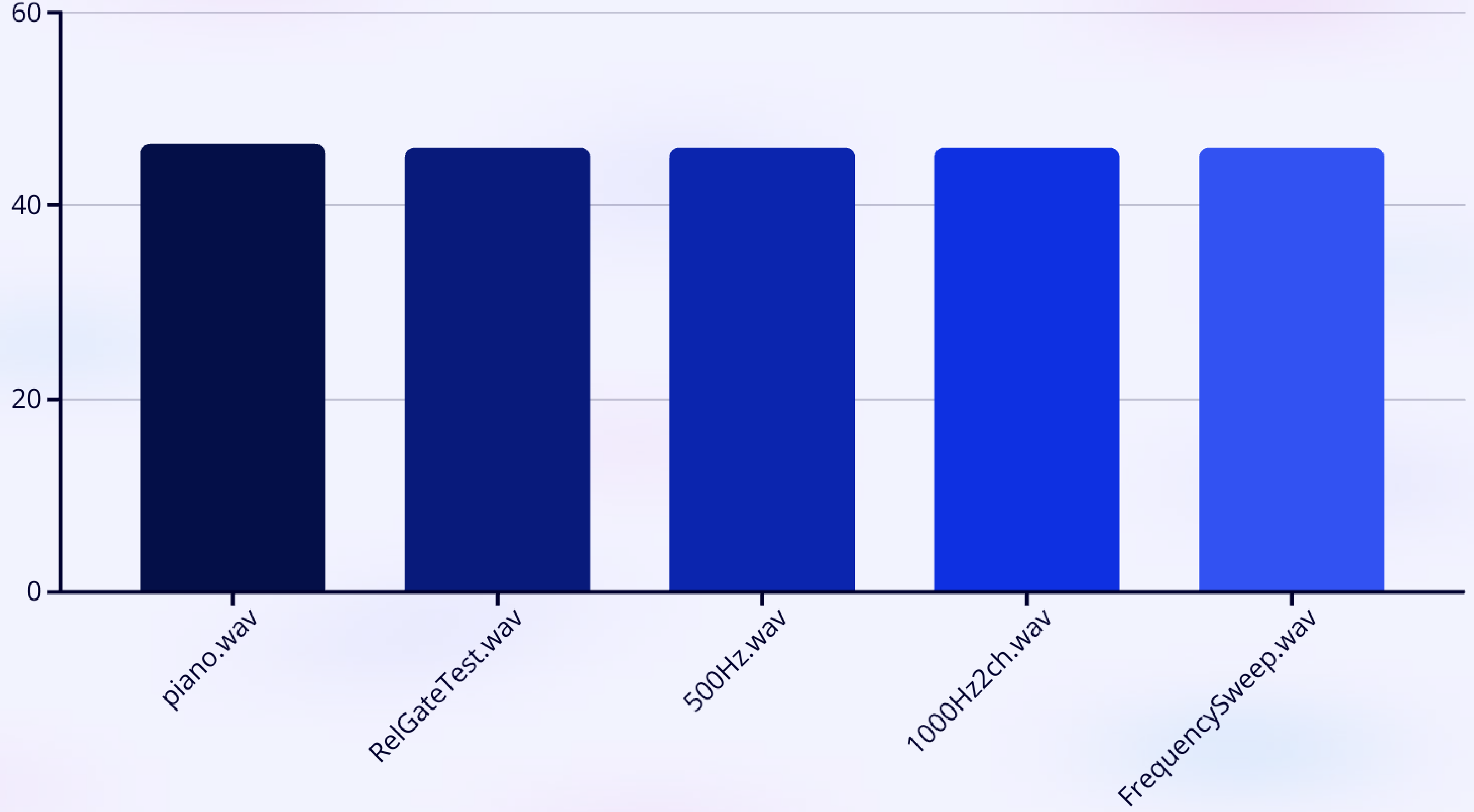


RMS



0.24
0.16
0.08
0
piano.wav
RelGateTest.wav
23LKFS 500Hz 2ch.wav
Comp 23LKFS 1000Hz 2ch
Comp 18LKFS...

A-weighting



60
40
20
0
piano.wav
RelGateTest.wav
500Hz.wav
1000Hz2ch.wav
FrequencySweep.wav