Short explanations of the models

DC removal:

Theory

DC removal eliminates the direct current (DC) component from a signal, leaving only the alternating current (AC) components. A high-pass filter (HPF) achieves this by allowing high

frequencies to pass while blocking low frequencies, including DC. In a simple RC high-pass filter, the transfer function is H(s) = sRC / (1 + sRC), where R is the resistance and C is the capacitance. The cutoff frequency, which determines the transition point, is given by f_c = 1 / (2πRC).

Implementation

Example Code: dc_removal.ipynb

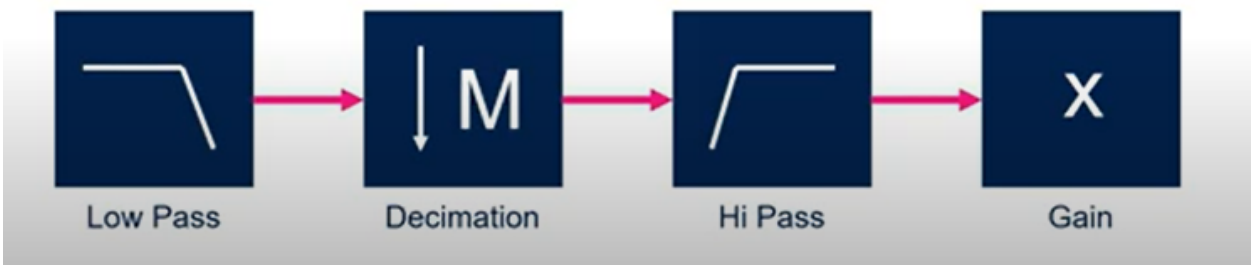
Pdm2pcm:

Theory

Converting Pulse-Density Modulation (PDM) to Pulse-Code Modulation (PCM) involves filtering, decimation, high-pass filtering, and gain adjustment. PDM signals are high-frequency bitstreams, with pulse density representing signal amplitude. The process includes low-pass filtering to remove high-frequency noise, decimation to reduce the bit rate (using:

$$y[m] = \frac{1}{N} \sum_{k=0}^{N-1} x[mN + k]$$

), high-pass filtering to eliminate DC offset and low-frequency noise, and gain adjustment to normalize signal amplitude. These steps ensure the PCM output accurately reflects the original analog signal.



Implementation:

The **practical** process includes reading the PDM file, applying a CIC filter (integration, decimation, and combination stages), and then converting the processed signal to PCM. Finally, the PCM signal is normalized and saved as a WAV file, ensuring the output accurately reflects the original analog signal.

References:

Good video explanation from Youtube: https://www.youtube.com/watch?v=5lH-tQw0tlU

Noise reduction:

Theory

Noise reduction involves minimizing unwanted noise from a signal to enhance its quality. Noise reduction in the time-frequency domain using spectral subtraction aims to restore the power or magnitude of a signal corrupted by noise. The process involves estimating the noise spectrum N_hat(f) by selecting noise-only segments, computing their Fourier transforms, and averaging these spectra to get

$\hat{N}(f) = \frac{1}{M} \sum_{i=1}^{M} N_i(f)$

The noisy signal Y(f) is the sum of the clean signal X(f) and the noise N(f): Y(f)=X(f)+N(f). The clean signal spectrum X_hat(f) is obtained by X_hat(f)=Y(f)−N_hat(f). The extent of the subtraction can be varied by applying a scaling factor α : X_hat(f)=Y(f)−αN_hat(f).

Implementation

Our **practical** implementation involves loading a wav file, normalizing it, extracting the first 0.25 seconds to sample the noise, performing Short-Time Fourier Transform (STFT), subtracting the noise magnitude from each calculated magnitude, clamping negative values to zero, reconstructing the signal using the original phases and Inverse STFT (ISTFT), normalizing the output, and converting it back to a wav file.

References:

[1] M. Berouti, R. Schwartz and J. Makhoul, "Enhancement of speech corrupted by acoustic noise," *ICASSP '79. IEEE International Conference on Acoustics, Speech, and Signal Processing*, Washington, DC, USA, 1979, pp. 208-211, doi: 10.1109/ICASSP.1979.1170788

[2] Noise Reduction Based on Modified Spectral Subtraction Method, Ekaterina Verteletskaya, Boris Simak, February 2011, IAENG International Journal of Computer Science 38(1) https://www.researchgate.net/publication/50264702_Noise_Reduction_Based_on_Modified_Spectral_Subtraction_Method (introduction gives a good overview of the approach)

Transmission and reception:

Theory

Transmission involves sending a signal using only half the bandwidth, achieved through Single Side Band (SSB) modulation. We start by reading the signal and determining its frequency, then create the SSB signal using a Hilbert transform to obtain the analytic signal xa(t), which is multiplied by a complex exponential to shift the frequency:

$\mathrm{SSB}(t) = \mathrm{Re}\{x_a(t) \cdot e^{i2\pi ft}\}$

The SSB signal is then demodulated by multiplying it with a cosine function, reducing its amplitude by half, and filtered using a Low Pass Filter (LPF) to remove high-frequency components: y(t) = LPF { SSB(t) · cos(2πft) }.

Implementation

The **implementation** involves using NumPy for mathematical operations, SoundFile for reading and saving audio files, and SciPy for the Hilbert transform and LPF. The input is an audio signal, and the output includes the original, SSB, and recovered signals, as well as a reconstructed audio file.

Pitch Estimation

Theory

Pitch estimation is the process of identifying the fundamental frequency of a periodic signal. There are three main methods for pitch estimation: time-domain, frequency-domain, and cepstrum-domain. Our approach utilizes the time-domain method due to its relatively fast computation time, which is suitable for real-time applications. Specifically, we employ the autocorrelation function (ACF), given by:

$$\mathrm{ACF}_{\mathrm{time}}(\tau) = \sum_{n=0}^{N-1} \mathrm{wav}(n) \cdot \mathrm{wav}(n+\tau)$$

where wav is the input signal, N is the number of samples, and τ\tauτ is the delay. The peak in the ACF indicates the fundamental period of the signal, helping to identify the pitch.

Implementation

See Noam's file, implementation group described this part well.

Acoustic gain control

Theory

Acoustic gain control addresses the issue of significant audio level variations within a file, which can cause loss of detail or distortion due to excessively high volume. It manages dynamic range differences in audio files to ensure consistent volume levels and prevent distortion. The algorithm reads a WAV file, converts audio data to a Numpy array, and calculates the RMS amplitude:

$$A_{\mathrm{RMS}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}$$

and peak amplitude Apeak=max(|x1|,|x2|,…,|xN|). It then adjusts the gain
G= Vtarget / Vcurrent, where Vtarget and Vcurrent are the desired and current dBFS levels. This ensures clear and consistent output before saving the processed data to a new WAV file.

Voice Ativity Detector (VAD)

Theory

Voice Activity Detection (VAD) aims to distinguish speech segments from silence or background noise in an audio file. The algorithm divides the audio signal into frames and analyzes each frame to determine if it contains speech or silence using the formula d[i]=**VAD**(x[i:i+N−1]), where x[i : i+N−1] represents a frame of the audio signal and d[i] is the binary value for that frame. The result of this analysis is a binary vector where 1 indicates speech and 0 indicates silence or noise. This process enhances the efficiency of speech recognition and audio indexing by focusing on the relevant parts of the signal.

Common approach for calculating **VAD** is using energy-based formula. This algorithm identifies speech by checking if the energy of a frame exceeds a certain threshold, which can be adjusted based on the noise level and desired sensitivity, Steps of the algorithm:

1. Calculate the short-term **energy** of each frame:

$$E[i] = \sum_{n=0}^{N-1} x[i+n]^2$$

   x[i+n] is the sample value at position i+n within the frame, and N is the number of samples in the frame.

2. Compare the energy E[i] of each frame to a predefined **threshold** T to determine if the frame contains speech or silence, resulting in binary decision d[i], where 1 indicates speech and 0 silence or noise:

$$d[i] = \begin{cases} 1 & \text{if } E[i] > T \\ 0 & \text{if } E[i] \leq T \end{cases}$$

References

[1] Faneuff J.J, Brown D. R. “Noise Reduction and Increased VAD Accuracy
Using Spectral Subtraction”, Processing of the Global Signal Processing
Exposition and International Signal Processing Conference (ISPC’ 03).
Dallas, Texas. April 2003.

[2]Jongseo Sohn and Wonyong Sung, "A voice activity detector employing soft decision based noise spectrum adaptation," *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181)*, Seattle, WA, USA, 1998, pp. 365-368 vol.1, doi: 10.1109/ICASSP.1998.674443.

Short Time Fourier Transform (STFT)

Theory

STFT (Short-Time Fourier Transform) is a method used to analyze the frequency content of a signal over time. It achieves this by dividing the signal into overlapping segments and applying the Fourier Transform to each segment. Mathematically, this can be expressed as

$$STFT(x(t)) = X(\tau, \omega) = \int_{-\infty}^{\infty} x(t)w(t-\tau)e^{-j\omega t}dt$$

where x(t) is the signal in time, w(t) is the window function, tau represents the time shift, and ω is the angular frequency. The result is a complex-valued matrix representing the magnitude and phase information of the signal at different frequencies and time intervals.

Decimation and interpolation

Theory

Decimation and interpolation are two fundamental techniques in digital signal processing. Decimation involves reducing the sampling rate of a signal by removing some of its samples, effectively downsampling the data. Interpolation, on the other hand, increases the sampling rate of a signal by adding new samples between the existing ones, effectively upsampling the data. Both processes are essential for tasks such as resizing images, converting audio sampling rates, and improving the efficiency of data transmission and storage.

The **decimation** process can be expressed by the formula: y[n]=x[nM] where y[n] is the decimated signal, x[n] is the original signal, and M is the decimation factor.

The **interpolation** process can be expressed by the formula:

$$y[n] = \sum_{k=0}^{L-1} h[k] \cdot x\left[\frac{n-k}{L}\right]$$

where y[n] is the interpolated signal, x[n] is the original signal, Lis the interpolation factor, and h[k] represents the interpolation filter coefficients.

Speed down / up

Theory

Speeding up or slowing down a signal involves changing its time scale. For a given signal x(t), speeding up the signal by a factor of α\alphaα is done by x(αt), where α > 1. Conversely, slowing down the signal by a factor of α is achieved by x(t/α), where α > 1. In speech processing, this technique is used to adjust the playback speed without altering the pitch.

Code and notes:

Code: The referenced **code** can be found in GitHub under the research section.
Publications: search https://scholar.google.com/, then use https://sci-hub.se/ with the article doi to read the text.