Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

A Mini Project Report

on

"Implementation of FIR filter for Noise Reduction in Python"

Digital Signal Processing (Comp 407)

(For partial fulfillment of IV Year/ I Semester in Computer Engineering)

Submitted by

Dikshyant Adhikari (04)

Nitin Ghimire (18)

Submitted to

Mr. Rupesh Dahi Shrestha

Department of Computer Science and Engineering

05th July, 2025

Abstract

This project demonstrates the implementation of a Finite Impulse Response band-pass Filter (FIR) and Wiener filter for noise reduction in audio signals using Python. The objective is to reduce unwanted noise components such as fan noise and white noise from recorded audio files, enhancing signal clarity. The filter design involves creating a band pass and a wiener filter, applying them and evaluating their effectiveness through visual and auditory analysis.

Keywords : *FIR filter, Noise reduction, Band-pass Filter, Wiener Filter*

Introduction

Noise reduction is a crucial preprocessing step in many audio processing applications such as speech recognition, hearing aids, telecommunications, and audio restoration. In real world environments, audio signals are often contaminated with unwanted noise such as fan hum, environmental interference, or white noise, which can degrade the performance of downstream systems like automatic speech recognizers or audio codecs.

Finite Impulse Response (FIR) filters are widely used in digital signal processing due to their inherent stability, finite-duration response, and linear phase characteristics, making them ideal for applications where phase distortion must be minimized. FIR filters work by convolving the input signal with a predefined impulse response, selectively allowing specific frequency bands to pass while attenuating others.

Mathematically:

$$y[n] = b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N]$$
$$= \sum_{i=0}^{N} b_i \cdot x[n-i],$$

where:

- x [n] is the input signal,
- y [n] is the output signal,
- N is the filter order;
- bi is the value of the impulse response at the i'th instant

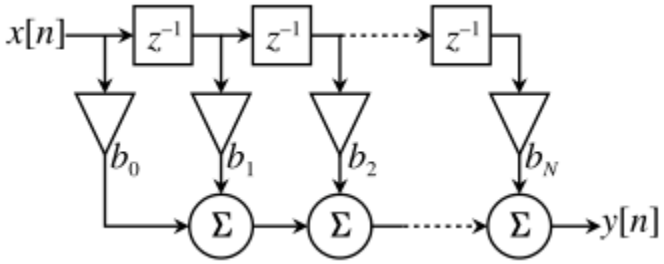

Figure 1 : FIR filter

This project involves the use of bandpass FIR filtering to isolate desired frequency ranges to remove fan hum, additionally Wiener filtering is applied as a statistical method to adaptively suppress remaining white noise by estimating local signal and noise power.

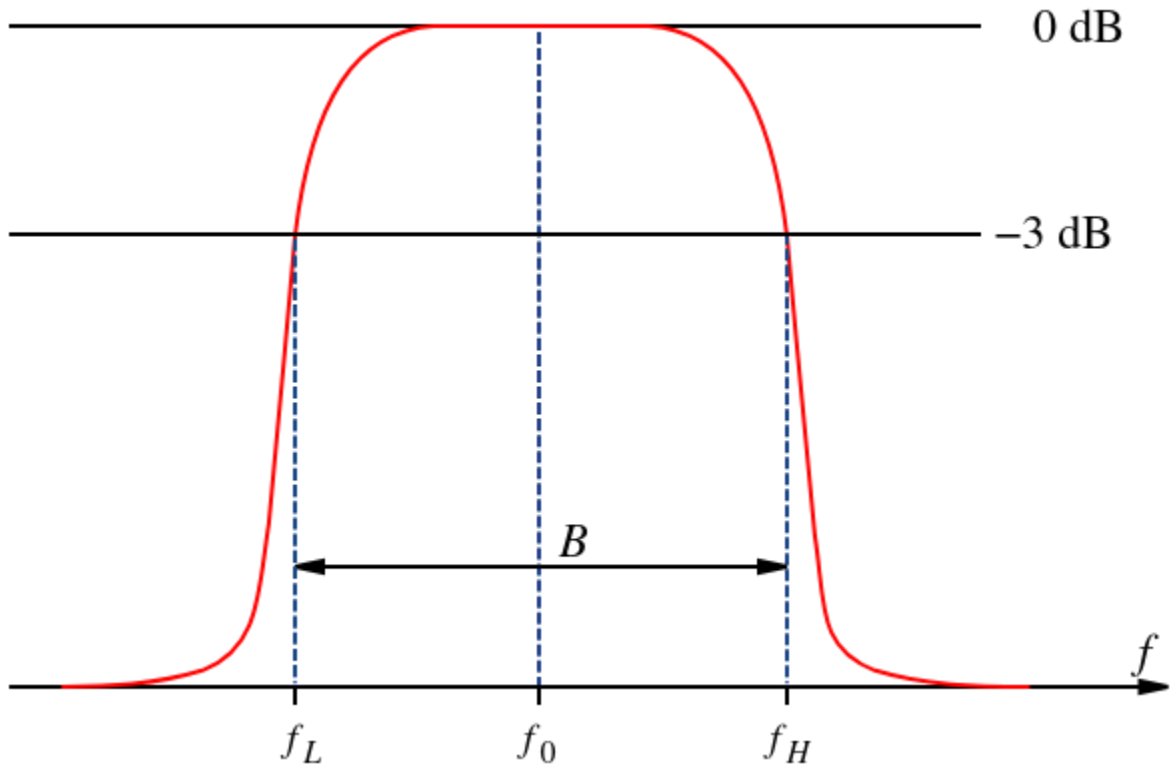

Figure 2 : Bandpass Filter

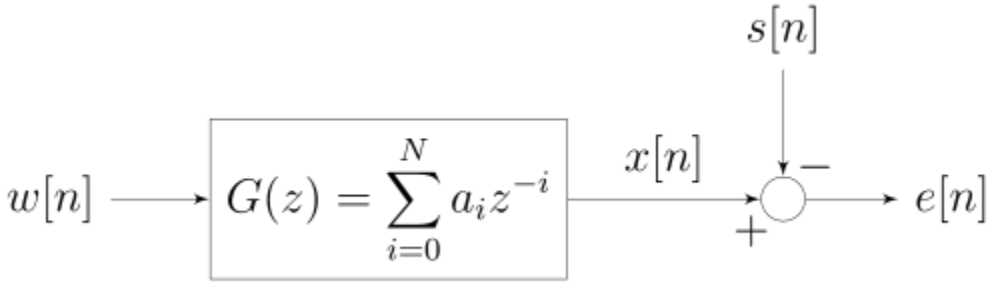

Figure 3 : Block diagram view of the FIR Wiener filter for discrete series

The implementation is done using Python and open-source libraries like scipy.signal, numpy, and matplotlib for signal processing, visualization, and analysis. This project demonstrates how classical DSP techniques like FIR filtering and Wiener filtering can be effectively used in modern, high-level programming environments for real-world audio processing tasks.

Methodology

1. Audio Data Acquisition

Audio files containing noise such as fan hum was recorded. The *.wav* format was chosen for ease of processing.

```
fs, audio = wavfile.read("resource/output.wav")
```

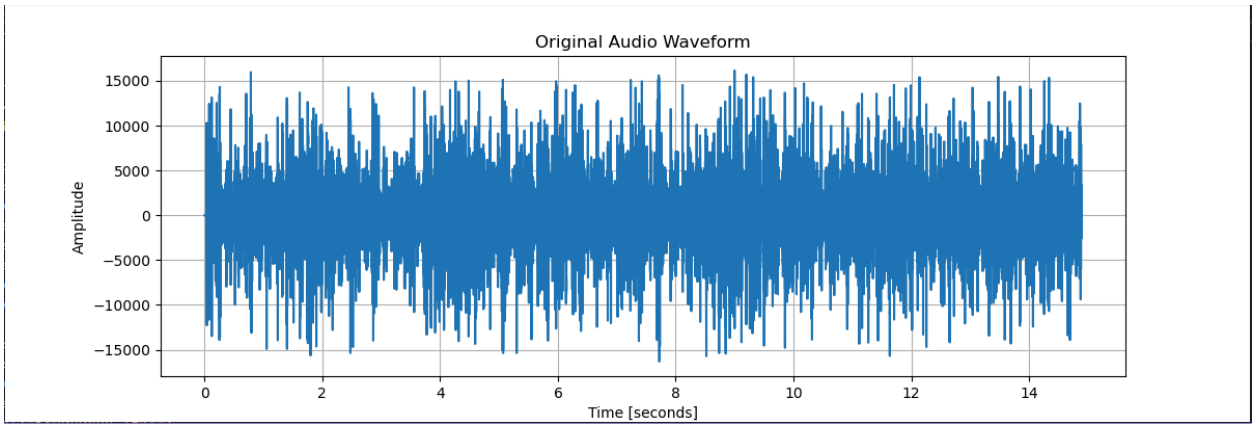

Figure 4 : Audio Waveform

2. FIR Filter Design

Using the *firwin* function from *scipy.signal*, band-pass FIR filters was designed with specified cutoff frequencies corresponding to the desired signal frequency band. Filter length (number of taps) was selected based on the trade off between filter sharpness and computational complexity.

```
#  Creating FIR filter
fir_coeff = firwin(numtaps, band, pass_zero=False)
```

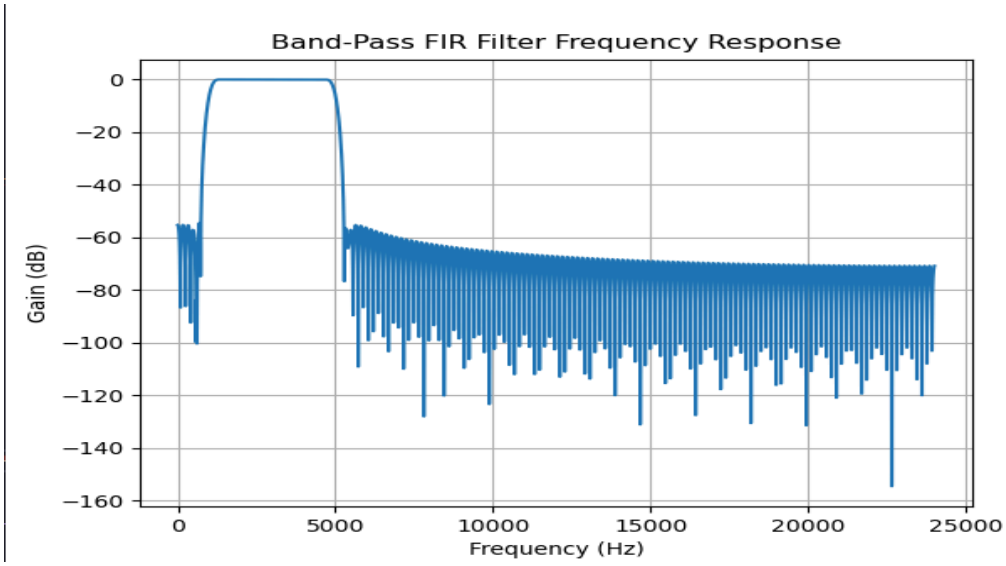

Figure 5 : BandPass FIR Filter Frequency Response

The FIR filter design process employs the windowing method to create optimal filter coefficients. The algorithm first normalizes the cutoff frequencies to the Nyquist frequency and validates the filter specifications. It then generates an ideal brick-wall frequency response using *sinc* functions, which represents the perfect rectangular frequency response in the time domain. However, since ideal filters are not realizable, a window function is applied to the impulse response to create a practical filter with finite length. The windowing process smooths the abrupt transitions of the ideal response, reducing spectral leakage and controlling the trade-off between main lobe width and side lobe suppression. Finally, the filter coefficients are scaled to ensure unity gain at the desired frequency, resulting in a linear-phase FIR filter suitable for noise reduction applications.

3. Filter Application

The designed FIR filter was applied to the audio signals using *lfilter*, which implements convolution between the input signal and filter coefficients.

```
#Applying the filter to the audio signal
filtered_audio = lfilter(fir_coeff, 1.0, audio)
```

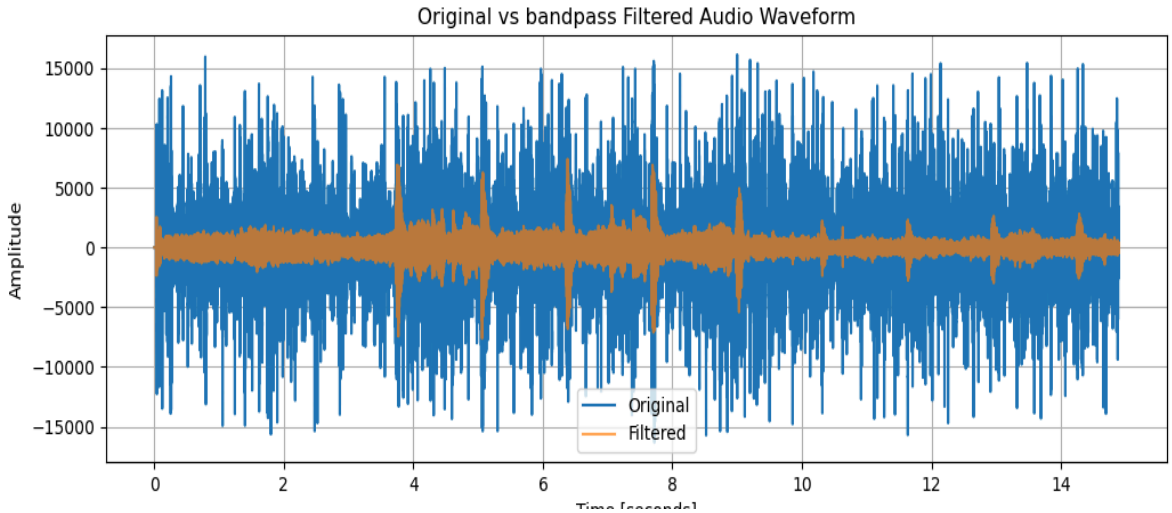

Figure 6

The filtering process applies the designed filter coefficients to the input audio signal through digital convolution. The algorithm processes each sample by computing the weighted sum of current and previous input samples using the filter coefficients. This convolution operation effectively multiplies the frequency domain representations of the signal and filter, resulting in the desired frequency-selective filtering. The linear convolution preserves the signal's temporal characteristics while attenuating unwanted frequency components, producing a filtered output with reduced noise content.

4. Wiener Filtering

To further reduce residual noise, Wiener filtering was applied to the band-pass filtered audio. The wiener function from *scipy.signal* provides noise reduction based on local variance estimation.

```
filtered_audio = wiener(audio, mysize=20)
```

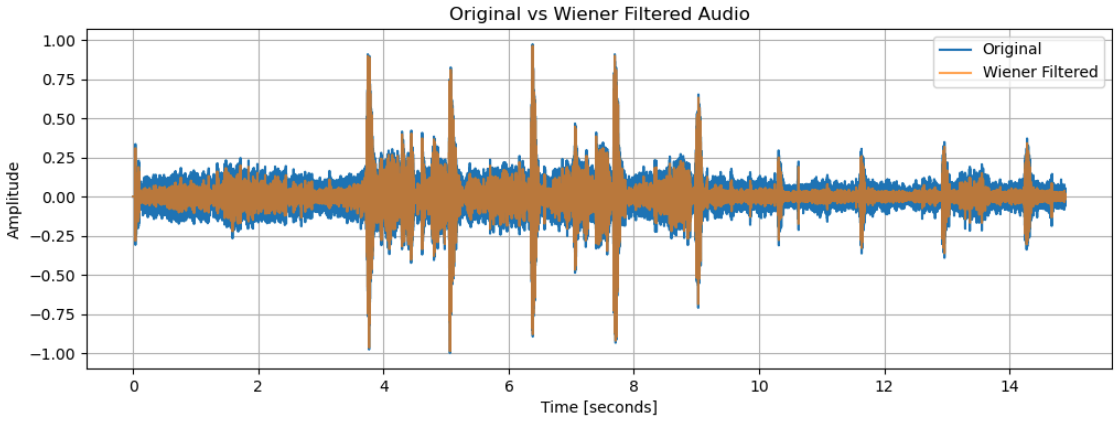

Figure 7

The Wiener filter is an adaptive noise reduction technique that estimates the local mean and variance within a moving window to distinguish between signal and noise. It assumes the presence of additive noise and aims to minimize the mean squared error between the filtered and original signals. Scipy.signal.wiener automatically adapts to local signal characteristics, effectively smoothing out noise while preserving important audio features.

Results and Discussion

The implemented FIR band-pass and Wiener filtering techniques were tested on real-world noisy audio. The following results summarize the visual, auditory, and technical evaluation of the filters.

1. FIR Filter Effectiveness

The FIR band-pass filter, with a passband between 1000 Hz and 5000 Hz and 301 taps, effectively attenuated frequency components outside the desired range. This significantly reduced both low-frequency fan hum and high-frequency noise, which are common in indoor recordings. As shown in the frequency response plot (Figure 5), the filter sharply attenuates frequencies outside the passband while preserving the central frequencies important for speech and acoustic information.

2. Visual Observation of Waveform

Visual inspection of the waveform plots before and after filtering (Figures 6) revealed:

- Smoother waveform with fewer erratic oscillations after FIR filtering.
- Reduction in low-frequency amplitude modulations caused by background hum.
- Clearer signal envelope consistent with spoken or tonal content.

The zoomed-in waveform comparison clearly highlights that FIR filtering eliminates much of the noise while preserving the original signal shape.

3. Wiener Filtering Results

To suppress residual white noise, the Wiener filter was applied as a post-processing step. The result (Figure 7) shows a further reduction in high-frequency graininess and smoother transitions between waveform sections.

The Wiener filter adaptively estimated and suppressed noise based on local variance, improving clarity without distorting the waveform significantly. This

stage is particularly helpful in handling background hiss and other high-frequency noise that FIR filters may not fully eliminate.

4. Results:

- The fan hum was noticeably reduced after FIR filtering.
- The Wiener filter enhanced clarity by reducing background hiss and fine noise textures as well as white noise caused from FIR filtering.
- The overall listening experience improved, with less distraction and better intelligibility of the signal.

5. Filter Parameters:

- Cutoff frequencies: Choosing appropriate low and high cutoffs was essential to avoid unintentionally removing parts of the signal (e.g., music content below 1 kHz).
- Filter length (numtaps): A longer filter (301 taps) provided a sharper transition band and better noise rejection but increased computational delay and group delay (~150 samples).
- Wiener window size: The size of the Wiener filter window (mysize = 20) had a trade-off between smoothing and detail retention. Larger sizes resulted in over-smoothing, while smaller sizes failed to suppress noise effectively.

Conclusion

The project successfully implemented an FIR filter-based noise reduction system in Python. Combining band-pass FIR filtering with Wiener filtering improved audio quality by attenuating unwanted noise while preserving important signal components. This approach demonstrates the applicability of classical DSP techniques for practical audio denoising tasks.

Bibliography

[1] A. V. Oppenheim and R. W. Schafer, Discrete-Time Signal Processing, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2009.

[2] R. G. Lyons, Understanding Digital Signal Processing, 3rd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2011.

[3] S. W. Smith, The Scientist and Engineer's Guide to Digital Signal Processing. California Technical Publishing, 1997. [Online]. Available: http://www.dspguide.com/

[4] J. Proakis and D. Manolakis, Digital Signal Processing: Principles, Algorithms, and Applications, 4th ed. Pearson Education, 2007.

[5] SciPy Documentation: scipy.signal.firwin – [Online]. Available: https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.firwin.html

[6]Plett, G.; Vetterli, M. "EE264: Lecture 12 - Wiener Filtering" (PDF). *Stanford University*. Retrieved 2025-03-20.

[7]Oppenheim, A. V.; Verghese, G. C. "Signals, Systems and Inference, Chapter 11: Wiener Filtering" (PDF). MIT OpenCourseWare. Retrieved 2025-03-20.