Speech Enhancement using Wiener Filtering

Sirish Hublikar

1 Introduction

In many speech communication scenarios, recordings are contaminated by environmental noise, reducing intelligibility and perceived quality. The goal of speech enhancement is to recover an estimate of the clean speech signal from such noisy observations. Among various methods, the Wiener filter offers an optimal linear solution in the mean-square error (MSE) sense, assuming statistical knowledge of both speech and noise signals.

This report presents the theoretical formulation of the Wiener filtering problem and its underlying assumptions. The Wiener filter is then derived as the optimal estimator under these conditions, followed by a discussion of how the required power spectral densities (PSDs) can be estimated from real data. Finally, the filter is evaluated under two noise conditions, stationary speech-shaped noise and non-stationary babble noise, to assess how its assumptions affect practical performance.

2 Methodology

2.1 Problem formulation and model assumptions

We consider a single-microphone recording in which the observed signal $x(n)$ consists of the superposition of a desired speech component $d(n)$ and additive noise $v(n)$:

$$x(n) = d(n) + v(n). \tag{1}$$

The following assumptions are made:

- $v(n)$ is a zero-mean, gaussian noise.
- $d(n)$ is a WSS random process in short lengths.
- Speech and noise are uncorrelated: $E[d(n)v(n)] = 0$.

The goal is to design a linear time-invariant filter $W(z)$ that produces an estimate $\hat{d}(n)$ of $d(n)$ by minimizing the mean-square error:

$$W(z) = \arg\min_{W} E\left[(d(n) - \hat{d}(n))^2\right]. \tag{2}$$

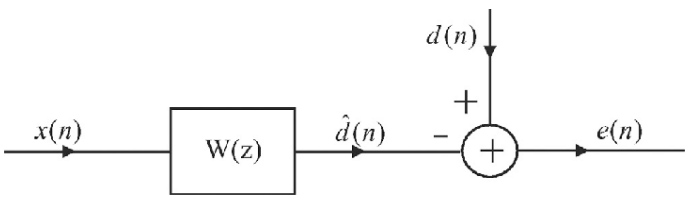


Figure 1: Block diagram of the Wiener filtering problem.

2.2 FIR Wiener filter

The FIR Wiener filter provides the optimal linear estimator that minimizes the mean-square error between the desired signal $d(n)$ and the estimate $\hat{d}(n)$. For jointly wide-sense stationary (WSS) signals, the optimal filter coefficients satisfy the Wiener–Hopf equations:

$$\mathbf{R}_x \mathbf{w} = \mathbf{r}_{dx} \tag{3}$$

where $\mathbf{R}_x$ is the Hermitian Toeplitz autocorrelation matrix of the input, $\mathbf{w}$ is the coefficient vector, and $\mathbf{r}_{dx}$ is the cross-correlation vector between desired and input signals. The matrix $\mathbf{R}_x$ has dimensions $p \times p$, with p the filter order.

This solution achieves minimum MSE under the WSS assumption, which is only approximately valid for speech. In practice, the signal is divided into short frames that can be treated as locally stationary, and the filter is computed per frame. If the desired signal and noise are uncorrelated, then $\mathbf{r}_{dx} = \mathbf{r}_d = \mathbf{r}_x - \mathbf{r}_v$, and

$$[\mathbf{R}_d + \mathbf{R}_v]\,\mathbf{w} = \mathbf{r}_d \tag{4}$$

If the desired signal statistics are unavailable, the coefficients can be computed from

$$\mathbf{R}_x \mathbf{w} = \mathbf{r}_x - \mathbf{r}_v \tag{5}$$

where the noise statistics are estimated from a noise-only segment. Although optimal in theory, frame-wise FIR implementation is computationally expensive and may introduce boundary artifacts. For this reason, a frequency-domain formulation is typically preferred.

2.3 Noncausal IIR Wiener filter

The IIR Wiener filter follows the same MSE minimization principle as the FIR case, but allows an infinite impulse response $h(n)$. This leads to a formulation in terms of power spectral densities (PSDs). Without additional constraints, the solution is noncausal and therefore suited for offline analysis.

The estimated output is

$$\hat{d}(n) = \sum_{l=-\infty}^{\infty} h(l)\, x(n-l) \tag{6}$$

with estimation error

$$e(n) = d(n) - \sum_{l=-\infty}^{\infty} h(l)\, x(n-l). \tag{7}$$

Minimizing the mean-square error

$$\xi = E\{|e(n)|^2\} \tag{8}$$

leads to the orthogonality condition

$$E\{e(n)x^*(n-k)\} = 0. \tag{9}$$

Substituting and rearranging yields the Wiener–Hopf equations:

$$\sum_{l=-\infty}^{\infty} h(l)\, r_x(k-l) = r_{dx}(k) \tag{10}$$

where $r_x(k)$ is the autocorrelation of $x(n)$ and $r_{dx}(k)$ the cross-correlation between $d(n)$ and $x(n)$.

In the frequency domain, convolution becomes multiplication:

$$H(e^{j\omega})\, P_x(e^{j\omega}) = P_{dx}(e^{j\omega}) \tag{11}$$

so that the optimal frequency response is

$$H(e^{j\omega}) = \frac{P_{dx}(e^{j\omega})}{P_x(e^{j\omega})}. \tag{12}$$

Equivalently, in the z-domain,

$$H(z) = \frac{P_{dx}(z)}{P_x(z)}, \tag{13}$$

where $P_x(z)$ is the PSD of $x(n)$ and $P_{dx}(z)$ the cross-PSD between $d(n)$ and $x(n)$.

This frequency-domain expression provides a direct analytical form of the optimal Wiener filter.

Mean-Squared Error

Once the optimal filter $h(n)$ is obtained, the minimum mean-square error (MSE) is given by

$$\xi_{\min} = r_d(0) - \sum_{l=-\infty}^{\infty} h(l)\, r_{dx}^*(l) \tag{14}$$

Using Parseval's theorem, this expression can be written equivalently in the frequency domain as

$$\xi_{\min} = r_d(0) - \frac{1}{2\pi} \int_{-\pi}^{\pi} H(e^{j\omega})\, P_{dx}^*(e^{j\omega})\, d\omega \tag{15}$$

Since the autocorrelation at lag zero can be expressed as

$$r_d(0) = \frac{1}{2\pi} \int_{-\pi}^{\pi} P_d(e^{j\omega})\, d\omega, \tag{16}$$

the minimum MSE becomes

$$\xi_{\min} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left[P_d(e^{j\omega}) - H(e^{j\omega})\, P_{dx}^*(e^{j\omega}) \right] d\omega \tag{17}$$

Wiener Smoothing Filter

The observed signal $x(n)$ can be modeled as the sum of a desired speech component $d(n)$ and additive noise $v(n)$:

$$x(n) = d(n) + v(n) \tag{18}$$

Assuming that $d(n)$ and $v(n)$ are uncorrelated, zero-mean random processes, their autocorrelation and power spectral densities satisfy

$$r_x(k) = r_d(k) + r_v(k) \tag{19}$$

$$P_x(e^{j\omega}) = P_d(e^{j\omega}) + P_v(e^{j\omega}) \tag{20}$$

The cross-correlation and cross-power spectral density between $d(n)$ and $x(n)$ are

$$r_{dx}(k) = E\{d(n)x^*(n-k)\} = r_d(k) \tag{21}$$

$$P_{dx}(e^{j\omega}) = P_d(e^{j\omega}) \tag{22}$$

Substituting these relations into the Wiener–Hopf formulation yields the frequency response of the Wiener smoothing filter:

$$H(e^{j\omega}) = \frac{P_d(e^{j\omega})}{P_d(e^{j\omega}) + P_v(e^{j\omega})} \tag{23}$$

At frequencies where $P_d(e^{j\omega}) \gg P_v(e^{j\omega})$, the signal-to-noise ratio (SNR) is high and $|H(e^{j\omega})| \approx 1$, so the speech component passes with minimal attenuation. Conversely, when $P_d(e^{j\omega}) \ll P_v(e^{j\omega})$, the SNR is low and $|H(e^{j\omega})| \approx 0$, resulting in strong noise suppression.

Substituting this expression into the MSE formulation gives

$$\xi_{\min} = \frac{1}{2\pi} \int_{-\pi}^{\pi} P_d(e^{j\omega}) \, \frac{P_v(e^{j\omega})}{P_d(e^{j\omega}) + P_v(e^{j\omega})} \, d\omega \tag{24}$$

This formulation illustrates that the residual error depends on the relative magnitudes of the speech and noise PSDs: the higher the noise power in a frequency band, the greater the residual error and attenuation applied by the filter.

2.4 Estimation of PSDs

In practice, the PSDs are not directly available since they are ensemble expectations. They must be estimated from finite data. The noise PSD $\hat{P}_v(e^{j\omega})$ is estimated from a segment known to contain only noise (here, the first 0.25s of the recording).

The speech PSD is then approximated as

$$\hat{P}_d(e^{j\omega}) = \hat{P}_x(e^{j\omega}) - \hat{P}_v(e^{j\omega}), \tag{25}$$

assuming that speech and noise are uncorrelated.

These estimations are inherently affected by bias and variance, which depend on factors such as the stationarity of the noise and the length of the analysis window. Shorter windows improve temporal resolution but increase spectral variance, whereas longer windows provide smoother estimates at the cost of reduced adaptability to non-stationary conditions.

2.5 Short-time spectral implementation

Since real speech and noise are not strictly stationary, the Wiener filter is applied in the short-time frequency domain. In this approach, the noisy signal is divided into overlapping, windowed frames, and each frame is processed independently in the spectral domain. The filter gain $H(e^{j\omega})$ is applied to each short-time Fourier transform (STFT) of the noisy signal, and the enhanced speech $\hat{d}(n)$ is reconstructed using the inverse STFT and overlap-add synthesis. This short-time formulation allows the filter to adapt to local variations in speech and noise statistics. The detailed MATLAB implementation of this method is described in Section 3.

2.6 Linear vs. circular convolution and zero-padding

In time-domain filtering, the convolution between the input signal and the filter impulse response corresponds to a linear convolution:

$$y(n) = x(n) * h(n). \tag{26}$$

When filtering is implemented in the frequency domain using the discrete Fourier transform (DFT), the multiplication of spectra corresponds instead to a circular convolution in time:

$$\text{IFFT}\{X(k)H(k)\} = x(n) \circledast h(n) = \sum_{k=0}^{N-1} x(k) \, h\big((n-k) \bmod N\big). \tag{27}$$

Unlike linear convolution, circular convolution considers samples in the boundaries as belonging to the same time index, adding them up, which produces time-domain aliasing if the signal and filter are not properly extended. In the short-time Fourier transform (STFT) implementation of the Wiener filter, this distinction is important: without correction, circular convolution can cause unwanted overlap between consecutive frames when reconstructing the signal using the overlap-add method.

To make circular convolution equivalent to linear convolution, zero-padding is applied to both the signal frame and the filter before taking the FFT. If L_x and L_h denote the lengths of the signal segment and the filter, respectively, the FFT length N must satisfy

$$N \geq L_x + L_h - 1. \tag{28}$$

Zero-padding ensures that the wrapped-around portion of the circular convolution contains only zeros, so that the IFFT result corresponds exactly to the desired linear convolution. This guarantees that the STFT-based Wiener filtering produces the same output as direct time-domain convolution, without introducing aliasing artifacts.

3 Implementation in MATLAB

The Wiener filter was implemented in MATLAB using the short-time spectral approach described in Section 2.5. The implementation consists of four main stages: preprocessing, noise estimation, spectral filtering, and signal reconstruction.

3.1 Preprocessing and noise addition

The clean speech and two noise signals (stationary speech-shaped noise and non-stationary babble noise) were first loaded and resampled to a common sampling rate. Each noise type was then scaled to achieve a specified input signal-to-noise ratio (SNR) of -5, 0, 5, and 10 dB before being added to the clean speech, producing the noisy observation $x(n) = d(n) + v(n)$. The scaling factor was computed from the ratio of average signal and noise powers to ensure an accurate SNR level.

3.2 Spectral analysis and noise estimation

The noisy speech was processed frame by frame using the short-time Fourier transform (STFT) with a Hamming window of 1024 samples, 50% overlap, and an FFT length of 2048 points. Zero-padding was automatically added by MATLAB since the FFT length (2048) exceeded the window length (1024), ensuring that the frequency-domain multiplication was equivalent to linear convolution in time, as explained in Section 2.6.

The noise power spectral density (PSD), $\hat{P}_v(e^{j\omega})$, was estimated from a short segment of the recording (the first 0.25 s) assumed to contain only noise. The noisy-speech PSD, $\hat{P}_x(e^{j\omega})$, was computed from the STFT coefficients of $x(n)$. The speech PSD was then approximated by subtraction:

$$\hat{P}_d(e^{j\omega}) = \max\left[\hat{P}_x(e^{j\omega}) - \hat{P}_v(e^{j\omega}),\, 0\right], \tag{29}$$

where negative values are truncated to zero to prevent numerical artifacts.

3.3 Wiener filter computation

The Wiener filter gain was computed for each time–frequency bin as

$$H(e^{j\omega}) = \frac{\hat{P}_d(e^{j\omega})}{\hat{P}_d(e^{j\omega}) + \alpha\, \hat{P}_v(e^{j\omega})}, \tag{30}$$

where α is an empirically chosen parameter controlling the amount of noise suppression. A value of $\alpha = 3.5$ provided the best trade-off between noise attenuation and speech distortion in this experiment. The estimated clean-speech spectrum was then obtained as

$$\hat{D}(e^{j\omega}) = H(e^{j\omega})\, X(e^{j\omega}). \tag{31}$$

3.4 Signal reconstruction and evaluation

The enhanced signal $\hat{d}(n)$ was reconstructed by inverse STFT with the same window and overlap parameters, using the overlap-add method to recombine the frames. Finally, input and output SNRs were computed to quantify performance:

$$\mathrm{SNR}_{\mathrm{in}} = 10\log_{10}\left(\frac{\sum d^2(n)}{\sum(x(n)-d(n))^2}\right), \tag{32}$$

$$\mathrm{SNR}_{\mathrm{out}} = 10\log_{10}\left(\frac{\sum d^2(n)}{\sum(\hat{d}(n)-d(n))^2}\right), \tag{33}$$

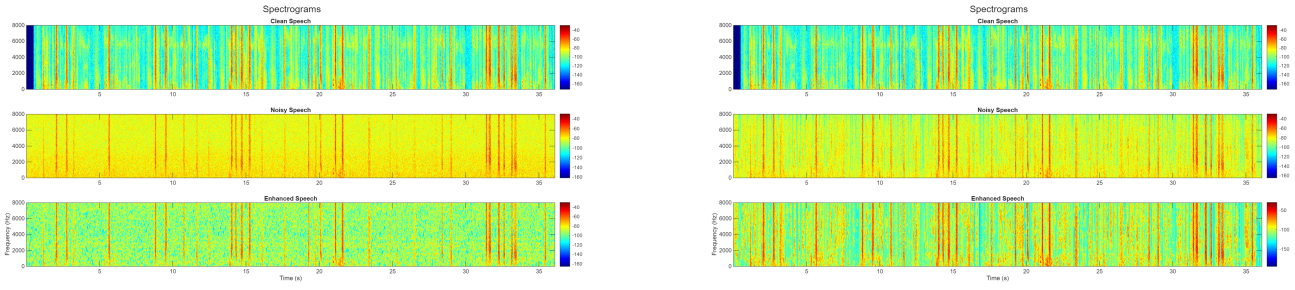
and the improvement was measured as $\Delta\mathrm{SNR} = \mathrm{SNR}_{\mathrm{out}} - \mathrm{SNR}_{\mathrm{in}}$. The MSE is also included for comparison.

4 Results

The output SNR was calculated to evaluate performance under both noise conditions. Results are summarized in Table 1.

Input SNR (dB)	**Output SNR (dB)**		**MSE**	
	Speech-shaped	**Babble**	**Speech-shaped**	**Babble**
-5	3.84	1.63	0.001198	0.001993
0	7.71	6.07	0.000491	0.000717
5	11.60	10.36	0.000201	0.000267
10	15.49	14.56	0.000082	0.000101

Table 1: Output SNRs and mean-square errors (MSE) obtained for stationary (speech-shaped) and non-stationary (babble) noise across different input SNR levels.



(a) Stationary speech-shaped noise

(b) Babble (non-stationary) noise

Figure 2: Spectrograms of enhanced speech for the two noise conditions (with input SNR = 5dB).

5 Discussion

The results in Table 1 show that the filter improves the SNR in all conditions, confirming its effectiveness as a linear noise suppressor. However, the improvement is consistently higher for stationary speech-shaped noise than for babble noise. This difference arises because the PSD of babble noise changes rapidly, violating the stationarity assumption on which the Wiener gain is derived. As a result, the noise estimate becomes less accurate within each frame, leading to residual noise and reduced enhancement performance.

Although the STFT-based implementation updates the PSDs across frames, it cannot fully adapt to rapid spectral changes. Consequently, the filter performs reliably under stationary conditions but struggles with non-stationary noise, as reflected in the lower output SNR values.

6 Conclusion

This report presented the theoretical foundation and practical implementation of the Wiener filter for speech enhancement. The filter was applied to noisy speech corrupted by stationary (speech-shaped) and non-stationary (babble) noise to evaluate its effectiveness under different conditions.

The results showed a consistent improvement in output SNR for all test cases, demonstrating that the Wiener filter can effectively reduce additive noise while preserving speech content. However, the enhancement was stronger for stationary noise, as the filter's assumptions of wide-sense stationarity and uncorrelated signal–noise components are only approximately satisfied in real speech and non-stationary environments.

The Wiener filter is therefore the best linear solution when the underlying processes are stationary or slowly varying and their power spectra can be accurately estimated. Under rapidly changing noise conditions, its performance degrades due to lack of adaptability.

Future research could focus on adaptive or time-varying Wiener filtering methods that update the PSD estimates continuously, or on combining statistical filtering with modern data-driven approaches such as deep neural networks for improved robustness in non-stationary acoustic environments.

References

[1] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*, John Wiley & Sons, 1996.

[2] Philipos C. Loizou, *Speech Enhancement. Theory and Practice*, CRC Press. Taylor & Francis Group, 2013.