CHAPTER 2

FEED FORWARD ACTIVE NOISE CONTROL SYSTEM USING WAVELET DENOISE

2.1 INTRODUCTION

The most popular adaptation algorithm used for Active Noise Control (ANC) applications (both broadband and narrowband) is Filtered -x- Least Mean Square (FxLMS), which is a modified version of LMS algorithm. Although the FxLMS algorithm is computationally simple, its convergence speed is slow and the performance of FxLMS based ANC system is degraded when there is large measurement of noise in the reference and error signals. To solve this problem many approaches have been developed by employing different structures. Rupp and Sayed (1995) proposed two modifications of the FxLMS algorithm with improved convergence behaviour albeit at the same computational cost. An optimal choice of the step size parameter in order to guarantee faster convergence and conditions for robustness are also derived.

Tobias et al (2000) proposed a stochastic analysis of the Filtered-x LMS algorithm. The Wiener solution is determined explicitly as a function of the input statistics and the impulse responses of the primary and secondary signal paths. It is shown that the steady-state mean weights for the Filtered-x LMS algorithm converge to the Wiener solution only if the estimate of the secondary path is without error.
Hyuck-Jae Lee et al (2000) proposed an effective active noise control system based on which a lattice-structure adaptive filter is developed for the control of exhaust noise. Akhtar et al (2003) proposed an algorithm based on adaptive filtering with averaging (AFA) and used a similar structure as that of the FxLMS ANC system. The proposed algorithm, which we call FxAFA algorithm, uses averages of both data and correction terms to find the updated values of the tap weights of the ANC controller. It is shown that the proposed algorithm gives fast convergence as compared with the FxLMS algorithm, for both broadband and narrowband noise signals. In comparison to the FxLMS algorithm the proposed algorithm achieves faster convergence only at the expense of slightly increased computational complexity.

Min-Chun Pan and Ji-Yao Liu (2005) proposed a transient and nonstationary high-Sound Pressure Level (SPL) shock noise. A full-size acrylic model is built up for mimicking the sound field of the control room of a rocket-launching vehicle. For achieving the noise reduction, an adaptive algorithm called the FxLMS is applied to implement the active noise control system. The convergence speed and stability of the FxLMS-type systems depend on the step-size values mainly. They are usually difficult to be decided for fast converging, and keeping the system stable. In the study, the variable step-size algorithms are implemented to automatically select an appropriate step size varying with the system status for desired performance.

Several approaches have been introduced in literature for active noise control (ANC) systems. Since FxLMS algorithm appears to be the best choice as a controller filter, researchers tend to improve performance of ANC systems by enhancing and modifying this algorithm. The proposed method uses a similar structure as that of the FxLMS ANC system. Here the concept of FxLMS is combined with wavelet denoise method. The wavelet in signal
and image processing has been found to be a very useful tool for solving various engineering problems, where in de-noising is one of them. The classical methods based on spectral subtractions (Boll 1979) are effective for this purpose; however they introduce artificial noise and alter the original signal. Wavelet thresholding (shrinking) is introduced as a powerful tool in denoising signals degraded by additive white noise (Donoho 1995). Dai-fei Guo et al (2000) proved that the variances and amplitudes of the white noise at the various levels decrease regularly as the level increase. On the other hand, the amplitude and variances of wavelet transform of the available signal are not related to the change of scale. According to the properties of wavelet transform of the noise and the available signal, noise can be weakened and even removed. The most known thresholding methods in the literature are the soft and hard thresholding. It can be expected that the technique of soft thresholding would introduce more error or bias than hard thresholding (Hai-Tao Fang 2004). But on the other hand, soft thresholding is more efficient in de-noising.

In this thesis, threshold value is found based on residual error signal using soft threshold method. The threshold value is used to update the step size of the algorithm. A small step size is used here initially and later its value is increased according to the threshold. This improved performance is achieved at the cost of a slightly increased computational complexity. Several approaches have been introduced in literature for active noise control (ANC) systems. Since FxLMS algorithm appears to be the best choice as a controller filter, researchers tend to improve the performance of ANC systems by enhancing and modifying this algorithm. In this thesis, modification is done in the existing FxLMS algorithm that provides a new structure for improving the performance and convergence rate.
2.1.1 Introduction to Active Noise Control System

Conventional methods of suppressing acoustic noise using passive sound absorbers generally do not work well at low frequencies. This is because at low frequencies the acoustic wavelengths become large when compared to the thickness of a typical acoustic absorber. A sound wave of frequency 100Hz, for example, will have a wavelength of about 3.4 meters in air under normal conditions. It is also difficult to stop low frequency sound being transmitted from one space to another unless the intervening barrier is very heavy. For these reasons, a number of practically important acoustic noise problems are dominated by low frequency contributions. These problems are sometimes difficult to solve using passive methods since the solutions are expensive in terms of weight and bulk.

Active noise control exploits the long wavelengths associated with low frequency sound. It works on the principle of destructive interference between the sound fields generated by the original “primary” sound source and that due to other “secondary” sources, whose acoustic outputs can be controlled. The most common type of secondary source is the moving coil loudspeaker, although mechanical excitation of structural components or even a modulated compressed air stream has been used as secondary sources. In each of these cases, the acoustic output of the source is controlled by an electrical signal. It is the generation and control of the electrical signal (to best reduce the acoustic field) that is the signal processing task associated with active noise control. This task presents a number of interesting and challenging problems which do not arise in the direct control of an electrical signal, as in electrical “adaptive noise cancellation” (Widrow et al 1975).

There are two approaches for controlling acoustic noise: passive and active. The traditional approach to acoustic noise control uses passive techniques such as enclosures, barriers, and silencers to attenuate the
undesired noise. Passive silencers use either the concept of impedance change caused by a combination of baffles and tubes to silence the undesired sound (reactive silencers) or the concept of energy loss caused by sound propagation in a duct lined with sound-absorbing material to provide the silencing (resistive silencers). Reactive silencers are commonly used as mufflers on internal combustion engines, while resistive silencers are used mostly for duct-borne fan noise. These passive silencers are valued for their high attenuation over a broad frequency range. However, they are relatively large, costly and ineffective at low frequencies, making the passive approach to noise reduction often impractical. Furthermore, these silencers often create an undesired back pressure if there is airflow in the duct.

In an effort to overcome these problems, considerable interest has been shown in active noise control. The active noise control system contains an electro acoustic device that cancels the unwanted sound by generating an anti sound (antinoise) of equal amplitude and opposite phase. The original, unwanted sound and the antinoise acoustically combine, resulting in the cancellation of both sounds.

The successful application of active control is determined on the basis of its effectiveness compared with passive attenuation techniques. Active attenuation is an attractive means to achieve large amounts of noise reduction in a small package, particularly at low frequencies (below 600 Hz). At low frequencies, where lower sampling rates are adequate and only plane wave propagation is allowed, active control offers real advantages.

2.1.2 Types of Active Noise Control Systems

Two types of acoustic noise exist in the environment viz. broadband and narrowband. Broadband noise cancellation requires knowledge of the noise source (the primary noise) in order to generate the anti noise signal. The measurement of the primary noise is used as a reference input to
the noise canceller. Primary noise that correlates with the reference input signal is cancelled downstream of the noise generator (a loudspeaker) when phase and magnitude are correctly modelled in the digital controller. For narrowband noise cancellation (reduction of periodic noise caused by rotating machinery), active techniques have been developed that are very effective and that do not rely on causality (having prior knowledge of the noise signal). Instead of using an input microphone, a tachometer signal provides information about the primary frequency of the noise generator. Since all the repetitive noise occurs at harmonics of the machine’s basic rotational frequency, the control system can model these known noise frequencies and generate the anti noise signal. This type of control system is desirable in a vehicle cabin, because it will not affect vehicle warning signals, radio performance, or speech, which is not normally synchronized with the engine rotation. Active noise control systems are based on one of two methods. (i). Feed forward control in which a coherent reference noise input is sensed before it propagates past the cancelling speaker. (ii). Feedback control in which the active noise controller attempts to cancel the noise without the benefit of an upstream reference input. Feed forward ANC systems are the main techniques used today. Systems for feed forward ANC are further classified into two categories:

- Adaptive broadband feed forward control with an acoustic input sensor
- Adaptive narrowband feed forward control with a nonacoustic input sensor

2.1.3 The Broadband Feedforward System

A considerable amount of broadband noise is produced in ducts such as exhaust pipes and ventilation systems. A relatively simple feed forward control system for a long, narrow duct is illustrated in Figure 2.1.
A reference signal $x(n)$ is sensed by an input microphone close to the noise source before it passes a loudspeaker. The noise canceller uses the reference input signal to generate a signal $y(n)$ of equal amplitude but $180^\circ$ out of phase. This antinoise signal is used to drive the loudspeaker to produce a cancelling sound that attenuates the primary acoustic noise in the duct.

Figure 2.1  Single-channel broadband feed forward ANC system

The basic principle of the broadband feed forward approach is that the propagation time delay between the upstream noise sensor (input microphone) and the active control source (speaker) offers the opportunity to electrically reintroduce the noise at a position in the field where it will cause cancellation. The spacing between the microphone and the loudspeaker must satisfy the principles of causality and high coherence, meaning that the reference must be measured early enough so that the anti-noise signal can be generated by the time the noise signal reaches the speaker. Also, the noise signal at the speaker must be very similar to the measured noise at the input microphone, meaning the acoustic channel cannot significantly change the noise. The noise canceller uses the input signal to generate a signal $y(n)$ that is of equal amplitude and is $180^\circ$ out of phase with $x(n)$. This noise is output to a loudspeaker and used to cancel the unwanted noise. The error
microphone measures the error (or residual) signal \( e(n) \), which is used to adapt the filter coefficients to minimize this error. The use of a downstream error signal to adjust the adaptive filter coefficients does not constitute feedback, because the error signal is not compared to the reference input.

### 2.1.4 The Narrowband Feedforward System

The block diagram of a narrowband feed forward active noise control system is shown in Figure 2.2. The non acoustic sensor signal is synchronous with the noise source and is used to simulate an input signal that contains the fundamental frequency and all the harmonics of the primary noise. This type of system controls harmonic noises by adaptively filtering the synthesized reference signal to produce a cancelling signal. In many cars, trucks, earth moving vehicles, etc., the revolutions per minute (RPM) signal is available and can be used as a reference signal. An error microphone is still required to measure the residual acoustic noise. This error signal is then used to adjust the coefficients of the adaptive filter. Generally, the advantage of narrowband ANC systems is that the nonacoustic sensors are insensitive to the cancelling sound, leading to very robust control systems.

![Figure 2.2 Narrowband feed forward ANC system](image_url)
2.1.5 Finite Impulse Response Adaptive Filters

In contrast to IIR or recursive adaptive filters, FIR filters are normally used in adaptive filtering applications that range from adaptive equalizers in digital communication systems to adaptive noise control systems. There are several reasons for the popularity of FIR adaptive filters. First, stability is easily controlled by ensuring that the filter coefficients are bounded. Second, there are simple and efficient algorithms for adjusting the filter coefficients. Third, the performance of these algorithms is well understood in terms of their convergence and stability, and finally FIR adaptive filters very often perform well enough to satisfy the design criteria.

An FIR filter for estimating a desired signal \( d(n) \) from a related signal \( x(n) \) is (Hayes 1996)

\[
\hat{d}(n) = \sum_{k=0}^{p} w_n(k)x(n-k) = w_n^T x(n)
\]  

(2.1)

Here it is assumed that \( x(n) \) and \( d(n) \) are non stationary random process and the goal is to find the coefficient vector \( w_n \) at time \( n \) that minimizes the mean-square error,

\[
\xi(n) = E\{|e(n)|^2\}
\]  

(2.2)

where

\[
e(n) = d(n) - \hat{d}(n) = d(n) - w_n^T x(n)
\]

The solution to this minimization problem may be found by setting the derivative of \( \xi(n) \) with respect to \( w_n^*(k) \) equal to zero for \( k = 0,1,\ldots, p \). The result is
\[ E\{e(n)x^*(n-k)\} = 0; \quad k = 0,1,\ldots,p \quad (2.3) \]

Substituting equation (2.3) into equation (2.2) we have
\[ E\{[d(n) - \sum_{i=0}^{p} w_n(l)x(n-l)]x^*(n-k)\} = 0; \quad k = 0,1,\ldots,p \quad (2.4) \]

After rearranging the terms,
\[ \sum_{i=0}^{p} w_n(l)E\{x(n-l)x^*(n-k)\} = E\{d(n)x^*(n-k)\}; \quad k = 0,1,\ldots,p \quad (2.5) \]

Equation (2.5) is a set of \( p+1 \) linear equations in the \( p+1 \) unknowns \( w_n(l) \). However, unlike the case of an FIR wiener filter where it was assumed that \( x(n) \) and \( d(n) \) is jointly Wide Sense Stationary (WSS), the solution to these equations depends on \( n \). It may be expressed in vector form as follows
\[ R_x(n)w_n = r_{dx}(n) \quad (2.6) \]

where
\[
R_x(n) = \begin{bmatrix}
    E\{x(n)x^*(n)\} & E\{x(n-1)x^*(n)\} & \cdots & E\{x(n-p)x^*(n)\} \\
    E\{x(n)x^*(n-1)\} & E\{x(n-1)x^*(n-1)\} & \cdots & E\{x(n-p)x^*(n-1)\} \\
    \vdots & \vdots & \ddots & \vdots \\
    E\{x(n)x^*(n-p)\} & E\{x(n)x^*(n-p)\} & \cdots & E\{x(n-p)x^*(n-p)\}
\end{bmatrix}
\]

is a \( (p+1) \times (p+1) \) Hermitian matrix of autocorrelation and
\[ r_{dx}(n) = [E\{d(n)x^*(n)\}, E\{d(n)x^*(n-1)\}, \ldots, E\{d(n)x^*(n-p)\}]^T \quad (2.7) \]
is a vector of cross correlations between \( d(n) \) and \( x(n) \). It is to be noted that in the case of jointly WSS processes, equation (2.6) reduces to the Wiener-Hopf equations, and the solution \( w_n \) becomes independent of time. Instead of solving equation (2.6) for each value of \( n \), which would be impractical in most real-time implementations, hence consider an iterative approach that is based on the method of steepest descent.

### 2.1.5.1 The steepest descent adaptive filter

In the designing of an FIR adaptive filter, the goal is to find the vector \( w_n \) at time \( n \) that minimizes the quadratic function

\[
\xi(n) = E\{|e(n)|^2\} \quad (2.8)
\]

Although the vector that minimizes \( \xi(n) \) may be found by setting the derivatives of \( \xi(n) \) with respect to \( w^*(k) \) equal to zero, another approach is to search for the solution using the method of the steepest descent. The method steepest descent is an iterative procedure that has been used to find the extreme of nonlinear functions.

The basic idea of this method is as follows. Let \( w_n \) be an estimate of the vector that minimizes the mean-square error \( \xi(n) \) at time \( n \). At time \( n + 1 \), a new structure is formed by adding a correction to \( w_n \) that is designed to bring \( w_n \) closer to the desired solution. The correction involves taking a step size \( \mu \) in the direction of maximum descent down the quadratic surface. Thus the updated equation for \( w_n \) is

\[
w_{n+1} = w_n - \mu \nabla \xi(n) \quad (2.9)
\]

where \( \nabla \xi(n) \) is the gradient vector and it is given by
The step size $\mu$ affects the rate at which the weight vector moves down the quadratic surface and must be a positive number. For very small values of $\mu$, the correction to $w_n$ is small and the movement down the quadratic surface is slow and as $\mu$ is increased, the rate of descent increases. However, there is an upper limit on how large the step size may be. For values of $\mu$ that exceed this limit, the trajectory of $w_n$ becomes unstable and unbounded. The steepest descent algorithm may be summarized as follows:

1. Initialize the steepest descent algorithm with an initial estimate, $w_0$, of the optimum weight vector $w$.

2. Evaluate the gradient of $\xi(n)$ at the current estimate, $w_n$, of the optimum weight vector.

3. Update the estimate at time $n$ by adding a correction that is formed by taking a step size $\mu$ in the negative gradient direction $w_{n+1} = w_n - \mu \nabla \xi(n)$

4. Go back to (2) and repeat the process.

Let us now evaluate the gradient vector $\nabla \xi(n)$. Assuming that $w$ is complex, the gradient is the derivative of $E\{|e(n)|^2\}$ with respect to $w^*$. Hence
\[ \nabla \xi(n) = \nabla E \{ |e(n)|^2 \} = E \{ \nabla |e(n)|^2 \} = E \{ e(n) \nabla^* (n) \} \]  
(2.11)

and

\[ \nabla e^*(n) = -x^*(n) \]

it follows that

\[ \nabla \xi(n) = -E \{ e(n)x^*(n) \} \]

Thus, with a step size of \( \mu \), the steepest descent algorithm becomes

\[ w_{n+1} = w_n + \mu E \{ e(n)x^*(n) \} \]  
(2.12)

To see how this steepest descent updates equation for \( w_n \) performs, let us consider what happens in the case of stationary processes. If \( x(n) \) and \( d(n) \) are jointly WSS then

\[ E \{ e(n)x^*(n) \} = E \{ d(n)x^*(n) \} - E \{ w_n^T x(n)x^*(n) \} = r_{dx} - R_x w_n \]  
(2.13)

and the steepest descent algorithm becomes

\[ w_{n+1} = w_n + \mu (r_{dx} - R_x w_n) \]  
(2.14)

Note that if \( w_n \) is the solution to the Wiener-Hopf equations, i.e. \( w_n = R_x^{-1} r_{dx} \), then the correction term is zero and \( w_{n+1} = w_n \) for all \( n \). Of greater interest, however, is how the weights evolve in time, beginning with an arbitrary weight vector \( w_0 \). The following property defines what is required for \( w_n \) to converge to \( w \). For jointly wide-sense stationary processes, \( d(n) \) and \( x(n) \), the steepest descent adaptive filter converges to the solution to the Wiener-Hopf equations

\[ \lim_{n \to \infty} w_n = R_x^{-1} r_{dx} \]
If the step size satisfies the condition

\[ 0 < \mu < \frac{2}{\lambda_{\text{max}}} \]

where \( \lambda_{\text{max}} \) is the maximum eigen value of the autocorrelation matrix \( R_x \).

Although for stationary Wiener-Hopf equations, when \( \mu < \frac{2}{\lambda_{\text{max}}} \), this algorithm is primarily of theoretical interest and finds little use in adaptive filtering applications. The reason for this is that in order to compute the gradient vector, it is necessary that \( E\{e(n)x^*(n)\} \) be known. For stationary processes this requires that the autocorrelation matrix of \( x(n) \) and correlation between \( d(n) \) and \( x(n) \) be known. In most applications, these ensemble averages are unknown and must be estimated from the data.

### 2.1.6 Least Mean Square (LMS) Adaptive Algorithm

The Least Mean Square (LMS) algorithm was first developed by Widrow and Hoff (1960) through their studies of pattern recognition (Haykin 1991). From there it has become one of the most widely used algorithms in adaptive filtering. The LMS algorithm is known as stochastic gradient-based algorithms as it utilizes the gradient vector of the filter tap weights to converge on the optimal wiener solution. It is well known and widely used due to its computational simplicity. It is this simplicity that has made it the benchmark against which all other adaptive filtering algorithms are judged (Haykin 1991).

With each iteration of the LMS algorithm, the filter tap weights of the adaptive filter are updated according to the following formula (Farhang-Boroujeny 1999).

\[ w_{n+1} = w_n + 2\mu e_n x_n \quad (2.15) \]
Here \( x(n) \) is the input vector of time delayed input values, 
\[
x(n) = [x(n)x(n-1)x(n-2)\ldots x(n - N + 1)]^T.
\]
The vector \( w(n) = [w_0(n)w_1(n)w_2(n)\ldots w_{N-1}(n)]^T \) represents the coefficients of the adaptive FIR filter tap weight vector at time \( n \). The parameter \( \mu \) is known as the step size parameter and is a small positive constant. This step size parameter controls the influence of the updating factor. Selection of a suitable value for \( \mu \) is imperative to the performance of the LMS algorithm, if the value is too small, the time the adaptive filter takes to converge on the optimal solution will be too long; if \( \mu \) is too large, the adaptive filter becomes unstable and its output diverges.

### 2.1.6.1 Derivation of LMS algorithm

Assume an FIR filter structure with \( N \) co-efficient, the error \( e_k \) between the wiener filter output and the primary signal \( y_k \) is given by (Hayes 1996)

\[
e_k = y_k - w_n^T x_n
\]  

(2.16)

where \( x_n \) and \( w_n \) are the input signal vector and weight vector respectively. The square of error is given by

\[
e_n^2 = [y_n - w^T x_n]^2
\]  

\[
e_n^2 = y_n^2 - 2 y_n x_n^T w + w^T x_n x_n^T w
\]  

(2.17)

The mean square error (MSE), \( J \) is obtained by taking the expectation of both sides of above equation

\[
J = E[e_n^2] = E[y_n^2] - 2E[y_n x_n^T w] + E[w^T x_n x_n^T w]
\]  

(2.18)
where $E[y_n^2] = \sigma^2$ the variance of is $y_n$, $E[y_n x_n] = \rho$ is the N-length cross-correlation vector and $E[x_n x_n^T] = R$ is the $N \times N$ autocorrelation matrix. The equation (2.18) can be written as

$$J = \sigma^2 + 2\rho^T w + w^T R w$$

(2.19)

The gradient of the performance is defined as

$$\nabla \xi(n) = \frac{dJ}{dw} = \frac{d(\sigma^2 + 2\rho^T w + w^T R w)}{dw}$$

$$\nabla \xi(n) = -2\rho + 2R w$$

(2.20)

$$= -2x_n y_n + 2x_n x_n^T w_n$$

$$= -2x_n \{y_n - x_n^T w_n\}$$

$$\nabla \xi(n) = -2e_n x_n$$

(2.21)

Substitute the above in equation (2.9), it becomes

$$w_{n+1} = w_n + 2\mu e_n x_n$$

(2.22)

The equation (2.22) is called as the updating the co-efficient of the filter using LMS algorithm. The LMS algorithm requires approximately $2N + 1$ multiplication and $2N + 1$ addition for each new set of input and output samples.

2.1.6.2 Implementation of LMS algorithm

The computational procedure for the LMS algorithm is summarized below
(i) Initially, set each weight $w_n(i), i = 0, 1, 2, \ldots, N - 1$, to an arbitrary fixed value. For each subsequent sampling instant $k = 1, 2, 3, \ldots$ carry out steps (ii) to (iv) given below.

(ii) Compute filter output

$$\hat{n}_k = \sum_{i=1}^{N-1} W_k(i) x_{k-i}$$

(iii) Compute the error estimate

$$e_n = y_n - w_n^T x_n$$

(iv) Update the next filter weights

$$w_{n+1} = w_n + 2\mu e_n x_n$$

2.1.6.3 Convergence of the LMS algorithm

In estimating the ensemble average $E\{e(n)x^*(n)\}$ with a one-point sample average $e(n)x^*(n)$, the LMS algorithm replaces the gradient in the steepest descent algorithm

$$\nabla \tilde{\xi}(n) = -E\{e(n)x^*(n)\}$$

with an estimated gradient

$$\hat{\nabla} \tilde{\xi}(n) = -e(n)x^*(n)$$

When this is done, the correction that is applied to $w_n$ is generally not aligned with the direction of the steepest descent. However, since the gradient estimate is unbiased,
\[ E\{\hat{\xi}(n)\} = -E\{e(n)x^*(n)\} = \nabla \hat{\xi}(n) \] 

(2.25)

Then the correction applied is, on the average, in the direction of the steepest descent. Since \( w_n \) is a vector of random variables, the convergence of the LMS algorithm must be considered within a statistical framework. Therefore, one will begin by assuming that \( x(n) \) and \( d(n) \) are jointly wide-sense stationary processes and will determine when the coefficients \( w_n \) convergence in the mean to \( w = R_x^{-1}r_{dx} \), i.e.,

\[
\lim_{n \to \infty} E\{e(n)\} = w = R_x^{-1}r_{dx}
\]

(2.26)

Now the LMS weight update equation becomes

\[
w_{n+1} = w_n + \mu[d(n) - w_n^T x(n)]x^*(n)
\]

(2.27)

Taking the expected value, the equation (2.27) becomes

\[
E(w_{n+1}) = E(w_n) + \mu E[d(n)x^*(n)] - \mu E[x^*(n)x^T(n)w_n]
\]

(2.28)

In the above equation the last term is not easy to evaluate. It may be simplified by using the independence assumption that the data \( x(n) \) and the LMS weight vector \( w_n \) are statistically independent. Under this assumption, it may be expressed as

\[
E(w_{n+1}) = E(w_n) + \mu E[d(n)x^*(n)] - \mu E[x^*(n)x^T(n)]E[w_n]
\]

\[
= (1 - \mu R_x)E[w_n] + \mu r_{dx}
\]

(2.29)

This is same as that of weight vector in steepest descent algorithm. For jointly wide sense stationary processes, the LMS algorithm converges in the mean if \( 0 < \mu < \frac{2}{\lambda_{\text{max}}} \) and the independence assumption is satisfied. It is
generally acknowledged that the upper bound is too large to ensure stability of
the LMS algorithm since it is not sufficient to guarantee that the coefficient
vector will remain bounded for all \( n \). Since the upper bound is expressed in
terms of the largest eigenvalue of \( R_x \), using this bound requires that \( R_x \) be
known. If this matrix is unknown, then it becomes necessary to estimate \( \lambda_{\max} \).
One way around this difficulty is to use the fact that \( \lambda_{\max} \) may be upper bound
by the trace of \( R_x \),

\[
\lambda_{\max} \leq \sum_{k=0}^{p} \lambda_k = tr(R_x) \quad (2.30)
\]

Therefore if \( x(n) \) is wide sense stationary, then \( R_x \) is Toeplitz and the trace becomes

\[
tr(R_x) = (p + 1)r_x(0) = (p + 1)E\{ |x(n)|^2 \} \quad (2.31)
\]

Hence the LMS algorithm mean be rewritten as

\[
0 < \mu < \frac{2}{(p + 1)E\{ |x(n)|^2 \}} \quad (2.32)
\]

Similarly \( E\{ |x(n)|^2 \} \) can be estimated using average as follows

\[
\hat{E}\{ |x(n)|^2 \} = \frac{1}{N} \sum_{k=0}^{K-1} |x(n - k)|^2 \quad (2.33)
\]

2.2 METHODS

2.2.1 FxLMS Algorithm

2.2.1.1 Basic principles

Transfer function of the secondary path has a crucial role in
generating anti-noise in ANC applications as it is non-linear and introduces
delay causing instability problem to the standard LMS algorithm. The instability problem can be resolved using the FxLMS algorithm as it uses estimation of the secondary path. This algorithm can be applied to both feedback and feed forward structures. The most popular adaptation algorithm used for ANC applications is the FxLMS algorithm, which is a modified version of the LMS algorithm (Widrow 1985). The schematic diagram for a single-channel feed forward ANC system using the FxLMS algorithm is shown in Figure 2.3.

![Figure 2.3 Block diagram of feed forward ANC system using FxLMS algorithm](image)

The acoustic path between the reference noise source and the error microphone is called a primary path and is denoted by \( P(z) \). The reference noise signal is filtered through the primary path \( P(z) \) and appears as a primary noise signal at the error microphone. The objective of the adaptive controller \( W(z) \) is to generate an appropriate anti-noise signal \( y(n) \) propagated by the secondary loudspeaker. This anti-noise signal combines with the primary noise signal to create a zone of silence in the vicinity of the error.
The error microphone measures the residual noise $e(n)$ which is used by $W(z)$ for its adaptation to minimize the sound pressure at error microphone. Here $\hat{S}(z)$ accounts for the model of the secondary path $S(z)$ between the output $y(n)$ of the controller and that of the error microphone $e(n)$. The secondary path $S(z)$ comprises the digital-to-analog (D/A) converter, reconstruction filter, power amplifier, loudspeaker, acoustic path from loudspeaker to error microphone, error microphone, preamplifier, anti-aliasing filter and analog-to digital (A/D) converter. The filtering of the reference signal $x(n)$ through the secondary-path model $\hat{S}(z)$ is demanded by the fact that the output of the adaptive controller $W(z)$ is filtered through the secondary path $S(z)$ (Widrow 1985).

2.2.1.2 Derivation of FxLMS algorithm

Block diagram of a feed forward FxLMS ANC system is Figure 2.3. Here $P(z)$ is the primary path, the acoustic response from the reference noise source to the error sensor; and $S(z)$ represents the secondary path. In this figure, $\hat{S}(z)$ is estimation of $S(z)$.

The expression for the residual error $e(n)$ is given as

$$e(n) = d(n) - y'(n)$$  \hspace{1cm} (2.34)

where $y'(n)$ the controller output $y(n)$ is filtered through the secondary path $S(z)$, then $y'(n)$ and $y(n)$ are computed as

$$y'(n) = s^T(n)y(n)$$  \hspace{1cm} (2.35)

and  \hspace{1cm} $$y(n) = w^T(n)x(n)$$
where \( w(n) = [w_0(n)w_1(n)......w_{L-1}(n)]^T \) is tap weight vector, 
\( x(n) = [x(n)x(n-1)......x(n-L+1)]^T \) is the reference signal picked by the reference microphone, \( L \) is the filter order and \( s(n) \) is impulse response of secondary path \( S(z) \). It is assumed that there is no acoustic feedback from secondary loudspeaker to reference microphone. The FxLMS update equation for the coefficients of \( W(z) \) is given as:

\[
w(n+1) = w(n) + \mu e(n)x'(n)
\]

(2.36)

where \( x'(n) \) is the reference signal \( x(n) \) filtered through secondary path model \( \hat{S}(z) \)

\[
x'(n) = \hat{s}^T(n)x(n)
\]

(2.37)

Here \( \hat{s} = [\hat{s}_0\hat{s}_1......\hat{s}_{L-1}]^T \) is impulse response of the secondary path model \( \hat{S}(z) \).

Although the FxLMS algorithm is computationally simple, its convergence speed is slow and signal dependent. The performance of FxLMS based ANC system is degraded when there is a large measurement noise in the reference and error signals. This has motivated researchers to look for fast convergence algorithms. One such approach towards this view is the modified FxLMS algorithm, where variable step size is introduced to achieve improved noise reduction and convergence rate.

### 2.2.2 Proposed Modified FxLMS (MFxLMS) Algorithm based on Wavelet Denoise

The proposed method provides a new structure of FxLMS algorithm for improving the noise reduction and convergence rate performance. In modified FxLMS, secondary signal \( y'(n) \) is soft thresholded dynamically with respect to error signal by wavelet transform to improve the
convergence performance. In this method, the estimated output error is an important component for the calculation of threshold value. This threshold value decides the dynamic step size. The proposed method leads to significant gain in SNR (signal-to-noise ratio), thus visibly reduces the output error and greatly improves the convergence speed. This method also achieves faster convergence rate as well as smaller mean square error (MSE).

2.2.2.1 Denoising by wavelet thresholding

Wavelet transforms have gained wide acceptance as a valuable tool for common signal processing tasks. The most important difference between the wavelet transforms and transforms such as a Fourier transform is that the wavelets are localized in both frequency and time. This makes it possible to better localize properties of the analyzed signal. The result is a well known ability of the wavelet transforms to pack the main signal information into a very small number of large wavelet coefficients. Another advantage of wavelet transforms is that there is an indefinite number of basis functions. An appropriate wavelet can then be chosen for a specific signal which makes the transform adjustable and adaptable. Because of these excellent properties, wavelet transforms have been used with great success in many different applications, such as signal denoising and compression or feature detection.

In this thesis, the focus is on the application in signal denoising, where wavelets are used extensively. The principle under which the wavelet thresholding operates is similar to the subspace concept, which relies on the fact that for many real life signals, a limited number of wavelet coefficients in the lower bands are sufficient to reconstruct a good estimate of the original signal. Usually these coefficients are relatively large compared to other coefficients or to any other signal (especially noise) that has its energy spread over a large number of coefficients. Therefore, by shrinking coefficients smaller than a specific value, called threshold, noise can be eliminated nearly while preserving the important information of the original signal.
The proposed denoising algorithm is summarized as follows:

i) Compute the discrete wavelet transform for noisy signal.

ii) Based on an algorithm, called thresholding algorithm and a threshold value, shrink some detail wavelet coefficients.

iii) Compute the inverse discrete wavelet transforms.

Figure 2.4 shows the block diagram of the basic wavelet thresholding for signal denoising. Wave shrink, which is the basic method for denoising by wavelet thresholding, shrinks the detail coefficients because these coefficients represent the high frequency components of the signal and it supposes that the most important parts of signal information reside at low frequencies. This assumption is true for a large group of signals; this idea is also based on the sparsity characteristic of wavelet transform. Therefore, the assumption is that in high frequencies the noise can have a bigger effect than the signal. In other words, wave shrink supposes that at high frequencies, noise forms a bigger part of coefficients in comparison with low frequencies.

![Figure 2.4 Block diagram of the basic wavelet thresholding for signal denoising](image-url)
Denoising by wavelet is performed by a thresholding algorithm in which the wavelet coefficients smaller than a specific value, or threshold, will be shrunk or scaled (Kuo and Vijayan 1994) and (Donoho 1995). The standard thresholding functions used in the wavelet based enhancement systems are hard and soft thresholding functions (Ghanbari and Karami 2006), which can be reviewed before the introduction of a new thresholding algorithm that offers improved performance for a signal.

2.2.2.2 Thresholding rules

Hard thresholding

Hard thresholding is similar to setting the components of the noise subspace to zero. The hard threshold algorithm is defined as

$$\delta^H_\lambda(y) = \begin{cases} 0 & |y| \leq \lambda \\ y & |y| > \lambda \end{cases}$$  \hspace{1cm} (2.38)

In this hard thresholding algorithm, the wavelet coefficients less than the threshold $\lambda$ will are replaced with zero which is represented in Figure 2.5 (a).

![Figure 2.5 (a) Hard thresholding algorithm](image)
Soft thresholding

In soft thresholding, the thresholding algorithm is defined as follows (Figure 2.5 (b)).

\[
\delta^S_\lambda(y) = \begin{cases} 
0 & |y| \leq \lambda \\
\text{sign}(y)(|y| - \lambda) & |y| > \lambda 
\end{cases} 
\]  

(2.39)

Figure 2.5 (b) Soft thresholding algorithm

Soft thresholding goes one step further and decreases the magnitude of the remaining coefficients by the threshold value. Hard thresholding maintains the scale of the signal but introduces ringing and artifacts after reconstruction due to a discontinuity in the wavelet coefficients. Soft thresholding eliminates this discontinuity resulting in smoother signals but slightly decreases the magnitude of the reconstructed signal.

2.2.2.3 Variable threshold step size

In the proposed method, the secondary signal of FxLMS is denoised by wavelet. This is performed by a thresholding algorithm, in which the wavelet coefficients smaller than a specific value or threshold, will be
shrunk or scaled. The signal \( y'(n) \) can be soft thresholded because this eliminates the discontinuity and results in smoother signal.

The wavelet transform using soft thresholding algorithm for signal \( y'(n) \) is defined as follows:

\[
\delta^S_{\lambda'} = \begin{cases} 
0 & |y'| \leq \lambda' \\
\text{sign}(y')(|y'| - \lambda') & |y'| > \lambda'
\end{cases}
\]  

(2.40)

where \( y' = s^Ty \) is the secondary path signal given in equation (2.35).

The equation for variable threshold is as follows

\[
\lambda' = \frac{\lambda_0}{1 + \text{abs}(e(n))}
\]  

(2.41)

It has been noted that initially the error of the system is large, initial threshold \( \lambda_0 \) is set to a constant value (0.04). When the error is large, threshold is small and vice-versa. As the number of iteration continues, the error of system will decrease. Finally, it retains the original threshold value.

### 2.2.2.4 Dynamic variable step size

The step size of the FxLMS algorithm is varied with respect to threshold value. Since threshold value at the beginning is small, the step size of the algorithm is large. This in turn increases convergence rate. As the iteration progresses, the error will simultaneously decrease. Finally, the step size will be reduced.

Figure 2.6 shows the block diagram for the proposed method. Thus the convergence rate of the FxLMS algorithm is improved by varying the step-size with respect to threshold value. From the Figure, the expression for the residual error \( e(n) \) is given as
Initially the error rate is high, threshold in the system is small. So large step size is selected. Then the step size is varied with respect to the threshold value $\lambda'$. Finally the error is reduced greatly by the implementation of the dynamic step size algorithm.

\[ e(n) = d(n) - \delta_s \quad (2.42) \]

\[ w(n + 1) = w(n) + \mu(n)e(n)x'(n) \quad (2.43) \]

where,

\[ \mu(n) = \frac{\mu_0}{1 + abs(\lambda'(n))} \quad (2.44) \]

$\mu_0 = 0.0004$
Thus the equation (2.41) and equation (2.44) is called as modified FxLMS algorithm for improving the performance of the existing algorithm.

The performance of modified FxLMS algorithm is improved in comparison with the FxLMS algorithm and the results are shown in the simulation. In the search for further improvement, genetic algorithm is identified. Genetic algorithms (GAs) are classified as a global searching method to derive the exact or approximate optimal solution to searching problems (Goldberg 1989, Mitchell 1998). GAs also belongs to evolutionary computations that adopt schemes inspired by evolutionary biology. Among the several operations of a typical GA include encoding, evaluation of the fitness function, reproduction, crossover, mutation, and replacement (Zhou and Debrunner 2007, Tan et al 2008). Its major features include parallel computation, robustness, multiple objectives, multimodality, number representation and constraints. GAs is extensively used in many applications across a large and growing number of disciplines. Also, GAs lend themselves easily to pipelining and parallelization.

2.3 OPTIMIZATION

2.3.1 Genetic Algorithm

Genetic Algorithm (GA), first introduced by John Holland in the early seventies, is becoming a flagship among various techniques of machine learning and function optimization. Algorithm is a set of sequential steps to be executed in order to achieve a task. A GA is an algorithm with some of the principles of genetics included in it. The genetic principles “Natural Selection” and “Evaluation Theory” are main guiding principles in the implementation of GA. The GA combines the adaptive natural genetics and search is carried out through randomized information exchange.
There is a multitude of search techniques. Among them, Calculus-based, Enumerative, and random search techniques are mostly used. The first two techniques i.e., Calculus-based and Enumerative are capable of arriving at reasonable good solutions for search spaces of smaller sizes. But once they are confirmed with search spaces of enormous size and wide variation from point to point in their precinct, like all practical systems, their efficiency in delivering solutions is drastically low. These are insufficiently robust solution techniques for complex problems involving huge search space due to their lack of ability to overcome the local optimum point. In order to overcome this local optimum point one uses the random search techniques. It is important to note that this randomized search is not a directionless search. The search is carried out randomly and information gained from a search is utilized in guiding the next search. Genetic Algorithm is an example of such search techniques.

Genetic Algorithm surpasses all the above limitations of conventional algorithm by using the basic building blocks that are different from those of conventional algorithms. It is different from them in the following aspects:

1. GA works with a coding of the parameter set and not the parameters themselves.
2. GA searches from a population of point and not from a single point like conventional algorithms.
3. GA uses objective function information, not derivative or other auxiliary data.
4. GA uses probabilistic transition rules by stochastic operands, not deterministic rules.
The very first step of a GA is the random selection of initial search points from the total search space. Each and every point in the search space corresponds to one set of values for the parameters of the problem. Each parameter is coded with a string of bits. The individual bit is called “gene”. The content of each gene is called “allele”. The total string of such genes of all parameters written in a sequence is called a “chromosome”. So there exists a chromosome for each point in the search space. The set of search points selected and used for processing is called a “population” i.e. population is a set of chromosomes. The number of chromosomes in a population is called “population size” and the total number of genes in a string is called “string length”. The population is processed and evaluated through various operations of GA to generate a new population and this process is carried out till global optimum point is reached. The two parts of this process are called “Generation” and “Evaluation”.

In the Evaluation of GA, fitness function has been defined and the fitness for each chromosome of the population has been evaluated. This fitness is an indication of the suitability of the values of the parameters, as represented by that chromosome, as a solution to the optimization problem under consideration. This fitness is used as a bias for selecting the parents and generating a new population from the existing one.

As obviously evident, a significant number of solutions are tested here simultaneously since each chromosome represents a solution. This is referred to as “Implicit Parallelism”. GA is the only search technique that enjoys implicit parallelism.

2.3.1.1 Phases of genetic algorithm

Genetic Algorithm consists of a string representation of points in the search space, a set of genetic operators for generating new search points
and a stochastic assignment to control the genetic operations. The simplicity of operation and the power of effect are two main attractions of the GA approach. It typically consists of three phases.

1. Initialization
2. Evaluation
3. Genetic Operation

1. Initialization

Initialization is the generation of initial population of chromosomes i.e. initial search points. Two parameters “population size” and “string length” need to be judiciously selected before this job is performed. The size of the population, i.e. number of chromosomes in a population, is a direct indication of effective representation of whole search space in one population. The population size affects both the ultimate performance and efficiency of GA. If it is too small, the chance that the members of a population cover the entire search space is low. This results in difficulty of obtaining the global optimal solution and leads to a local optimum solution. It is important to note that this stagnation at local optimum is a result of premature. So large population size is preferable to avoid this premature convergence and to reach a global optimum point. But a too large population size decreases the rate of convergence and in the worst case may lead to divergence. So based on the size of search, the population size needs to be selected.

The selection of “string length” depends on the accuracy of the optimization problem. The higher the string length, the better will be the resolution and accuracy. But this leads to slow convergence. Also the number of parameters in the optimization problem will have a direct effect on the number of bits in a chromosome i.e. string length. For particular solution and
accuracy requirements, problem with more number of parameters will have a large string. Keeping all the above constraints in mind, the string length is chosen appropriately.

After the selection of string length and population size, the initial population is generated as a set of strings of bits either 0 or 1. Random number generation techniques are used to accomplish this task. These strings of bits contain the information related to the parameters of the optimization problem in encoded format. Any of the encoding techniques can be used but binary encoding is convenient and mostly used.

From the initial population, chromosomes are decoded and all parameters of the optimization problem are calculated for each chromosome. This results in a set of solutions whose size is equal to population size. These steps of coding and decoding are reported in the implementation section.

2. Evaluation

In the evaluation phase, suitability of each of the solution from the initial set as the solution of the optimization problem is determined. For this, a function called “fitness function” is defined. This is used as a deterministic tool to evaluate the fitness of each chromosome. The optimization problem may be of minimization or maximization type. In the case of maximization type, the fitness function can be a function of variables with direct proportionality relationship with the objective function. For minimization type problems, fitness function can be a function of variables that bear inverse proportionality relationship with the objective function or can be reciprocal of a function of variables with direct proportionality relationship with the objective function. In either case, fitness function is so selected that the most fit solution is the nearest to the global optimum point. The programmer of GA
is allowed to use any fitness function that adheres to the above requirements. This flexibility with the GA is one of its fortes.

On the whole, for a typical optimization problem, evaluation phase consists of calculation of individual parameters, testing of any equality or inequality constraints that need to be satisfied, evaluation of objective function, and finally evaluation of fitness from fitness function. This evaluation is discrete in nature vis-à-vis some genetic operators which operate on more than one chromosome at a time.

3. Genetic Operation

In this phase, the objective is the generation of new population from the existing population with the examination of fitness values of chromosomes and application of genetic operators. These genetic operators are reproduction, crossover and mutation. This phase is carried out if we are not satisfied with the solution obtained earlier. The GA utilizes the notion of survival of the fittest by transferring the highly fit chromosome to the next generation of string and combining different strings to explore new search points.

a) Reproduction

Reproduction is simply an operator where by an old chromosome is copied into a mating pool according to its fitness value. Highly fit chromosomes receive higher number of copies in the next generation. Copying chromosomes according to their fitness means that the chromosomes with a higher fitness value have higher probability of contributing one or more offspring in the next generation.
b) Crossover

It is recombination operation. Here the gene information contained in the two selected parents is utilized in a certain fashion to generate two children who bear some of the useful characteristics of parents and expected to be fitter than their parents.

There are various techniques that are used for performing this crossover. First two parents should be picked from the existing population to perform crossover. Some of the selection schemes are

1. Rank-based fitness assignment
2. Roulette wheel selection
3. Stochastic universal sampling
4. Local selection
5. Truncation selection
6. Tournament selection

Among the above existing selection schemes, tournament selection technique used in the proposed work. In the tournament selection technique, a number tour of individuals is chosen randomly from the population and the best individual from this group is selected as parent. This process is repeated as often choose individuals. These selected parents produce random offspring uniformly. The parameter for tournament selection is the tournament size tour. Tour takes values ranging from 2-Nind (number of individuals in population). In the present case the tournament size tour is 2.

Obviously through this tournament selection more reproductive chances are given to those population members that are the fit. Thus the
picking of chromosomes as parents is according to their objective function values.

Now crossover is carried out using any of the following three methods

1. Simple or Single point crossover
2. Multi point crossover
3. Uniform crossover

1. Single Point Crossover

In this method a crossover is carried out at a single point. This is illustrated in the following example. Let Par1 and Par2 be the two parents selected for crossover. Assume

The strings Par1 and Par2 as below

Par1: 11000101  Par2: 10110111

Now, a crossover site is selected randomly as an integer between 1 and string length. For illustration, take the string length as 8. Let this crossover site is 4. Then the children Child1 and Child 2 are generated as below.

Child 1: 11000111

<Par1> | <Par2>

Child 2: 10110101

<Par2> | <Par1>
2. Multi Point Crossover

This method is similar to single point crossover except that more than one crossover site is randomly selected and the contents of Child1 and Child2 are selected alternatively from parent1 and parent 2 with change from one parent to another at the crossover sites.

Par1: 11000101        Par2: 10110111

Here four crossover points are taken and the resultant child’s are

Child1: 11110111
<Par1><Par2><Par1><Par2>

Child2: 10000101
<Par2><Par1><par2><Par1>

3. Uniform Crossover

In this method, crossover is performed over the entire length of string of bits. For this a “mask” is generated randomly. This “mask” is a string of bits of value 0 or 1 and its size is same as the chromosome string length. The generation of children from this crossover is shown below.

Par1: 11001011
Par2: 01000100
Mask: 00101101

Child1: 11000110(If mask=0, Child1=Par1 and if mask=1, Child2 = Par2)
Child2: 01001001(If mask=1, Child1= Par2 and if mask=0, Child2=Par1)
Here there is a need to generate a mask for each crossover but there is no need to store them. So number of masks needed is equal to the number of crossover need to be performed. Generate them as and when required and discard them thereafter.

Thus by observation each crossover resulted in two children. So the number of crossover required to be performed for next generation depends on the number of children one need. Usually it is a general practice to copy some of the best parents as it is into the next generation, the required strings as children. This phenomenon of copying best parents into the next generation is called ‘Elitism’ and the number of parents so copied is indicated by a parameter of GA called “percentage of Elitism” (Pe). This is nothing but the percentage of parents so copied of the total number of parents. This Elitism is basically carried out not to lose the best strings obtained so far which otherwise may be lost.

In order to control the crossover also there is a parameter called “Crossover probability (Pc)”’. This probability is used as a decision variable before performing the crossover. This is done as follows. A random number between 0 and 1 is generated and if that number is than Pc, crossover is performed. The randomly generated number is greater than Pc, crossover is not performed and Child1 and Child 2 are directly selected as Par1 and Par2. This is equivalent to the case of crossover where crossover site is equal to the string length.

c) Mutation

This operation is capable of creating new genetic material in the population to maintain the population diversity. It is the random alteration of a bit value at a particular bit position in the chromosome. The following example illustrates the mutation operation.
Original string: 10110011
Mutation site: 4 (assumption)
String after mutation: 10100011

“Mutation Probability (Pm)” is a parameter used to control the mutation. For each string a random number between 0 and 1 is generated and compared with the Pm. If it is less than Pm, mutation is performed on the string. Sometimes mutation is also performed bit-by-bit instead of strings. This result in substantial increase in time but the performance of GA will not increase to a acceptable extent.

2.3.1.2 Implementation of tuning of threshold using genetic algorithm

The GA differs from classical optimization techniques in that it works on a population of solutions and searching are on a bit string encoding of the real parameters rather than parameters themselves. The algorithm starts from an initial population generated randomly. This population undergoes three genetic operations: selection, crossover and mutation to produce a new generation after duly considering the fitness of the strings, which corresponds to the objective function for the concerned problem. A trial solution for the problem requires the selection of number populations for a generation and a number for several such generations in order to find best fitness of the strings (best objective function) in that trial. Several such values of the strings is dependent on the number of population in a generation, the number of generations and the number of trials solving the problem through GA.

In this thesis, the objective is to reduce the mean square error value, i.e. fitness function of genetic algorithm. This can be achieved by tuning and finding the optimum value of threshold. In a typical run of the genetic algorithm, an initial population is randomly generated. This initial population
is referred to as the $0^{\text{th}}$ generation. Each individual in the initial population has an associated performance index value. Using the performance index information, the genetic algorithm then produces a new population.

### 2.3.1.3 Working principle of the proposed algorithm

The following steps explain how the genetic algorithm works:

**Step 1.** **Set the parameter:** Generate an initial population of chromosomes. Each chromosome consists of genes and each of these genes represents threshold value (Th).

**Step 2.** **Perform the fitness function:** Evaluate the objective function based on the threshold value.

$$MSE = \frac{1}{N} \sum_{n=1}^{N} [x(n) - \hat{x}(n)]^2$$

**Step 3.** **Reproduce the population:** Identify the Chromosomes parents that will go to the mating pole for producing the next generation using tournament selection technique. A number tour of the individuals is chosen here randomly from the population and the best individual from this group is selected as parent. This process is repeated as often to choose individuals. These selected parents produce uniform random offspring.

**Step 4.** **Perform crossover operations:** The crossover rate is an important parameter in a GA. It is a recombination operation. Here the gene information contained in the two selected parents is utilized in a certain fashion to generate two children who bear some of the useful characteristics of parents and expected to be fit than parents. Crossover operations are normally divided into two classes: “one-point”- and “two-point”-type crossovers. The parents are randomly
selected based on step 3. Additionally, a randomly generated crossover point is selected. Two chromosomes beyond this point are swapped to form the offspring in the one point-type crossover. Moreover, the two-point-type crossover resembles the one-point crossover in that two crossover points are randomly generated for two chromosomes, and then, the contents of two points are swapped to form the offspring. The two-point type crossover is performed in the proposed method.

Step 5. Implement the mutation operation: Perform genes Mutation for the meeting pool parents after been crossovered. Mutation is also an important parameter in a GA. Although a larger mutation rate transforms the GA into a purely random search algorithm, the typical mutation rate only uses the fixed value within 0.005-0.05. This method adopts the mutation rate according to the fitness function. This operation is capable of creating the new genetic material in the population to maintain the population diversity. It is nothing but random alteration of a bit value at a particular bit position in the chromosome.

Step 6. Identify the Best Chromosome that has the minimum Fitness value (Here optimization problem is a minimization problem) and store it

Step 7. In each time identify the best chromosome and compare its fitness with the stored one, if it is better (less mean square error) replace the best chromosome with this new one.

Step 8. The loop of generation is repeated until the best chromosome in term of minimum mean square error is identified.

The flow chart of the proposed method can be summarized as follows and given in Figure 2.7.
Figure 2.7 Flow chart for proposed method
Table 2.1 shows the comparison results of different methods with various parameters. The performance of the modified FxLMS method has been improved when compared with the existing FxLMS method. With the help of genetic algorithm the performance of modified FxLMS algorithm is improved. The performance comparisons of different step sizes are represented in Table 2.1.

**Table 2.1 The performance of different methods using various step sizes**

<table>
<thead>
<tr>
<th>Method/Parameter</th>
<th>Noise Reduction in dB (RdB)</th>
<th>Signal to Noise Ratio in dB (SNR)</th>
<th>Mean Square Error (MSE)</th>
<th>Relative Modeling Error in dB(RME)</th>
<th>Percent Root Mean Square Difference (PRD) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>FxLMS</td>
<td>2.6</td>
<td>1.14</td>
<td>78.8</td>
<td>-13.3</td>
<td>87.6</td>
</tr>
<tr>
<td>Modified FxLMS</td>
<td>4.7</td>
<td>2.25</td>
<td>66.1</td>
<td>-15.12</td>
<td>77.1</td>
</tr>
<tr>
<td>Modified FxLMS using Genetic Algorithm</td>
<td>10.10</td>
<td>4.38</td>
<td>238.2</td>
<td>-10.17</td>
<td>60.3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method/Parameter</th>
<th>Noise Reduction in dB (RdB)</th>
<th>Signal to Noise Ratio in dB (SNR)</th>
<th>Mean Square Error (MSE)</th>
<th>Relative Modeling Error in dB(RME)</th>
<th>Percent Root Mean Square Difference (PRD) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>FxLMS</td>
<td>6.75</td>
<td>2.7</td>
<td>49.03</td>
<td>-14.7</td>
<td>73.2</td>
</tr>
<tr>
<td>Modified FxLMS</td>
<td>9.7</td>
<td>3.99</td>
<td>47.22</td>
<td>-17.06</td>
<td>63.1</td>
</tr>
<tr>
<td>Modified FxLMS using Genetic Algorithm</td>
<td>20.68</td>
<td>9.14</td>
<td>78.83</td>
<td>-11.63</td>
<td>34.8</td>
</tr>
</tbody>
</table>
Table 2.1 (Continued)

<table>
<thead>
<tr>
<th>Method/Parameter</th>
<th>Noise Reduction in dB (RdB)</th>
<th>Signal to Noise Ratio in dB (SNR)</th>
<th>Mean Square Error (MSE)</th>
<th>Relative Modeling Error in dB(RME)</th>
<th>Percent Root Mean Square Difference (PRD) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>FxLMS</td>
<td>10.45</td>
<td>4.39</td>
<td>31.34</td>
<td>-16.18</td>
<td>60.2</td>
</tr>
<tr>
<td>Modified FxLMS</td>
<td>12.9</td>
<td>5.27</td>
<td>29.27</td>
<td>-19.3</td>
<td>54.4</td>
</tr>
<tr>
<td>Modified FxLMS using Genetic Algorithm</td>
<td>28.6</td>
<td>13.77</td>
<td>27.06</td>
<td>-14.59</td>
<td>20.4</td>
</tr>
<tr>
<td><strong>Step size (mu=0.0008)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FxLMS</td>
<td>12.3</td>
<td>5.35</td>
<td>29.9</td>
<td>-19.8</td>
<td>53.9</td>
</tr>
<tr>
<td>Modified FxLMS</td>
<td>15.1</td>
<td>6.55</td>
<td>26.39</td>
<td>-21.2</td>
<td>46.9</td>
</tr>
<tr>
<td>Modified FxLMS using Genetic Algorithm</td>
<td>29.8</td>
<td>15.89</td>
<td>17.13</td>
<td>-22.62</td>
<td>16.0</td>
</tr>
<tr>
<td><strong>Step size (mu=0.001)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FxLMS</td>
<td>11.9</td>
<td>5.18</td>
<td>38.5</td>
<td>-12.10</td>
<td>55.0</td>
</tr>
<tr>
<td>Modified FxLMS</td>
<td>12.19</td>
<td>5.29</td>
<td>30.8</td>
<td>-13.9</td>
<td>54.3</td>
</tr>
<tr>
<td>Modified FxLMS using Genetic Algorithm</td>
<td>26.08</td>
<td>13.8</td>
<td>27.4</td>
<td>-9.31</td>
<td>20.3</td>
</tr>
</tbody>
</table>
Table 2.1 (Continued)

<table>
<thead>
<tr>
<th>Method/Parameter</th>
<th>Noise Reduction in dB (RdB)</th>
<th>Signal to Noise Ratio in dB (SNR)</th>
<th>Mean Square Error (MSE)</th>
<th>Relative Modeling Error in dB(RME)</th>
<th>Percent Root Mean Square Difference (PRD) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>FxLMS</td>
<td>8.9</td>
<td>3.8</td>
<td>43.4</td>
<td>-11.8</td>
<td>63.9</td>
</tr>
<tr>
<td>Modified FxLMS</td>
<td>10.3</td>
<td>4.7</td>
<td>32.15</td>
<td>-12.4</td>
<td>59.5</td>
</tr>
<tr>
<td>Modified FxLMS using Genetic Algorithm</td>
<td>22.75</td>
<td>11.87</td>
<td>42.7</td>
<td>-8.76</td>
<td>25.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method/Parameter</th>
<th>Noise Reduction in dB (RdB)</th>
<th>Signal to Noise Ratio in dB (SNR)</th>
<th>Mean Square Error (MSE)</th>
<th>Relative Modeling Error in dB(RME)</th>
<th>Percent Root Mean Square Difference (PRD) %</th>
</tr>
</thead>
<tbody>
<tr>
<td>FxLMS</td>
<td>3.5</td>
<td>1.43</td>
<td>73.0</td>
<td>-8.6</td>
<td>84.7</td>
</tr>
<tr>
<td>Modified FxLMS</td>
<td>5.1</td>
<td>2.2</td>
<td>50.1</td>
<td>-9.2</td>
<td>77.1</td>
</tr>
<tr>
<td>Modified FxLMS using Genetic Algorithm</td>
<td>20.2</td>
<td>10.4</td>
<td>59.2</td>
<td>-8.39</td>
<td>29.9</td>
</tr>
</tbody>
</table>

2.4 RESULTS AND DISCUSSION

This section presents the simulation performed to verify the effectiveness of the proposed method. The performance of the proposed modified FxLMS with genetic algorithm is compared with Modified FxLMS algorithm and FxLMS algorithm on the basis of five performance measures; 1) Noise reduction \( R(dB) \), 2) Relative modelling error \( \Delta S(dB) \), 3) Mean square error, 4) Signal to noise ratio and 5) Percent root mean square difference, being defined as
\[ R(dB) = -10 \log \left( \frac{\sum e^2(n)}{\sum d^2(n)} \right) \]  
\[ (2.45) \]

\[ \Delta S(dB) = 10 \log \left( \frac{\sum_{j=0}^{M-1} [s_j(n) - \hat{s}_j(n)]^2}{\sum_{j=0}^{M-1} [s_j(n)]^2} \right) \]  
\[ (2.46) \]

\[ MSE = \frac{1}{N} \sum_{n=1}^{N} [x(n) - \hat{x}(n)]^2 \]  
\[ (2.47) \]

\[ SNR = 10 \log \left\{ \frac{\sum_{n=1}^{N} [x(n)]^2}{\sum_{n=1}^{N} [x(n) - \hat{x}(n)]^2} \right\} \]  
\[ (2.48) \]

\[ PRD = 100 \sqrt{\left\{ \frac{\sum_{n=1}^{N} [x(n) - \hat{x}(n)]^2}{\sum_{n=1}^{N} [x(n)]^2} \right\}} \]  
\[ (2.49) \]

For this simulation, the primary path \( P(z) \) and secondary path \( S(z) \) of the experimental data provided in (Kuo 1996) is used. The primary path \( P(z) \) and the secondary path \( S(z) \) are FIR filters of tap weight length 48 and 16, respectively. The control filter \( W(z) \) and modelling filter \( \hat{S}(z) \) are FIR filters of tap-weight length \( L = 32 \) and \( M = 16 \) respectively. Here the reference signal is a tonal of 300 Hz and sampling frequency of 2 kHz. The variance of this signal is adjusted to 2.0 and a zero-mean white Gaussian noise is added with SNR of 30 dB. Extensive simulations are performed to show the effectiveness of the modified FxLMS with the genetic algorithm method (proposed method) compared with that of modified FxLMS and FxLMS algorithm.
Case 1

First the analysis of the performance of Modified FxLMS and FxLMS algorithm on the basis of noise reduction $R(dB)$ for various step sizes 0.0002, 0.0004, 0.0006, 0.0008, 0.001, 0.0012 and 0.0014 are seen in Table 2.1 and the modified FxLMS algorithm method seems better than FxLMS algorithm. This is because the modified FxLMS method uses variable step size that is controlled by the secondary path variable threshold value. Due to this process, the residual error is reduced. This in turn improves the noise reduction. It is also seen that convergence occurs at the step size value of 0.0008, after which the reduction in noise reduces in both the methods.

Case 2

In this case, the performance of Relative modelling error for modified FxLMS and FxLMS for various step sizes 0.0002, 0.0004, 0.0006, 0.0008, 0.001, 0.0012 and 0.0014 are analyzed (Table 2.1). The modified FxLMS method can reduce the modelling error more than that of FxLMS method. After the occurrence of convergence (step size value of 0.0008) modelling error increases in the above two methods.

Case 3

The analysis of convergence rate for modified FxLMS and FxLMS for step sizes 0.0008 and 0.001 are seen. The speed of convergence is faster in modified FxLMS than the FxLMS. It is also seen that convergence occurs at step size value of 0.0008 (Figure 2.8(a)) after which the convergence rate reduces as seen in Figure 2.8(b).
Case 4

In this case, the noise reduction $R(dB)$ for various step sizes using modified FxLMS with genetic algorithm is analyzed. The performance of modified FxLMS with genetic algorithm method achieves best performance than modified FxLMS and FxLMS methods and can reduce the residual noise signal at a much faster rate than the modified FxLMS and FxLMS method. This is because the method uses fitness function, and based on the fitness function, the threshold value is tuned. This in turn improves the performance
on noise reduction (see Table 2.1). The noise reduction for various step sizes is shown in Figure 2.9(a). Here the convergence occurred at step size value of 0.0008 i.e. the point of maximum noise reduction. The noise reduction reduces after this convergence point as seen in Figure 2.9(b).

![Graph](image)

**Figure 2.9 (a)** Noise reduction versus iteration time \((n)\) for various step sizes using genetic algorithm (Before convergence)

![Graph](image)

**Figure 2.9 (b)** Noise reduction versus iteration time \((n)\) for various step sizes using genetic algorithm (After convergence)

**Case 5**

The analysis of the performance of Relative modelling error for modified FxLMS using genetic algorithm for various step sizes 0.0002,
0.0004, 0.0006 and 0.0008 are done. Here the performance of modified FxLMS using genetic method can reduce the modelling error at a faster convergence rate for different step sizes. The relative modelling error for various step sizes are shown in Figure 2.10 (a), which shows convergence occurred at the step size value of 0.0008. The modeling error increased after the convergence point (0.0008) as shown in Figure 2.10(b).

**Figure 2.10 (a) Relative modelling error versus iteration time \((n)\) for various step sizes using genetic algorithm (Before convergence)**

**Figure 2.10 (b) Relative modelling error versus iteration time \((n)\) for various step sizes using genetic algorithm (After convergence)**
Case 6

The residual noise signal for Modified FxLMS using genetic, Modified FxLMS and FxLMS method are compared. The graph shows Modified FxLMS using genetic (proposed method) achieves the best performance among the above methods. This is because the threshold value is tuned based on the fitness function, which reduces residual noise efficiently.

Figure 2.11  Residual error signal $e(n)$ versus iteration time $n$ for various methods (Step size=0.0008)
Case 7

This section presents the simulation of various parameters performed to verify the effectiveness of modified FxLMS using genetic algorithm method. The simulation results of mean square error for various methods are represented in Figure 2.12 (a). The mean square error obtained for modified FxLMS using genetic method is initially large, reduces drastically after the threshold tuning, reaches minimum on convergence and again increases after the convergence. The signal to noise ratio of modified FxLMS using genetic method increases when compared to modified FxLMS and FxLMS up to the point of convergence(0.0008) and then reduces as represented in Figure 2.12 (b). Similarly the comparison of the percent root mean square difference (PRD) shows that the PRD value is reduced more in modified FxLMS using genetic method than modified FxLMS and FxLMS methods. It reaches maximum at convergence after which reduces as shown in Figure 2.12(c). Finally the bar chart for the performance of noise reduction $R(dB)$ and Relative modelling error for the above methods, confirms again that modified FxLMS using genetic is the best, where both reaches their best at the point convergence and then performance degrades as shown in Figure 2.12 (d) and Figure 2.12 (e).

![Figure 2.12 (a) Mean square error versus step size for various methods](image)
Figure 2.12 (b) Signal to noise ratio versus step size for various methods

Figure 2.12 (c) Percent root mean square difference (PRD) for various methods
Figure 2.12 (d) Noise reduction versus step size for various methods

Figure 2.12 (e) Relative modeling error versus step size for various methods
2.5 CONCLUSION

In the proposed research, a modified structure of FxLMS Algorithm using wavelet denoise concept in secondary path is implemented and the step size of the update equation is altered which improves the fundamental trade-off between the speed of convergence and minimum MSE. The step size is decided by threshold which in turn depends on the estimated output error. Thus significant gain in SNR (signal-to-noise ratio) and reduced output error is achieved. For further improved performance, the genetic algorithm is used, where a new fitness function to optimize the threshold value is defined.

The results obtained on performance like noise reduction, convergence, mean square error, signal to noise ratio and Percent Root Mean Square Difference are compared for FXLMS, modified FXLMS and modified FxLMS using genetic algorithm. The present study confirms that a significant improvement in performance is achieved only in modified FxLMS using genetic algorithm than the other methods.