Hindawi Publishing Corporation
Advances in Acoustics and Vibration
Volume 2008, Article ID 791050, 12 pages
doi:10.1155/2008/791050

*Research Article*

# Genetic Algorithm Applied to the Eigenvalue Equalization Filtered-x LMS Algorithm (EE-FXLMS)

**Stephan P. Lovstedt,[1] Jared K. Thomas,[2] Scott D. Sommerfeldt,[1] and Jonathan D. Blotter[2]**

[1] *Department of Physics and Astronomy, College of Physical and Mathematical Sciences, N283 ESC, Brigham Young University, Provo, UT 84602, USA*
[2] *Department of Mechanical Engineering, Ira A. Fulton College of Engineering and Technology, 435 CTB, Brigham Young University, Provo, UT 84602, USA*

Correspondence should be addressed to Scott D. Sommerfeldt, scott_sommerfeldt@byu.edu

Received 12 December 2007; Accepted 7 March 2008

Recommended by Marek Pawelczyk

The FXLMS algorithm, used extensively in active noise control (ANC), exhibits frequency-dependent convergence behavior. This leads to degraded performance for time-varying tonal noise and noise with multiple stationary tones. Previous work by the authors proposed the eigenvalue equalization filtered-x least mean squares (EE-FXLMS) algorithm. For that algorithm, magnitude coefficients of the secondary path transfer function are modified to decrease variation in the eigenvalues of the filtered-x autocorrelation matrix, while preserving the phase, giving faster convergence and increasing overall attenuation. This paper revisits the EE-FXLMS algorithm, using a genetic algorithm to find magnitude coefficients that give the least variation in eigenvalues. This method overcomes some of the problems with implementing the EE-FXLMS algorithm arising from finite resolution of sampled systems. Experimental control results using the original secondary path model, and a modified secondary path model for both the previous implementation of EE-FXLMS and the genetic algorithm implementation are compared.



## 1. INTRODUCTION

The most common control approach control (ANC) is the filtered-x least mean squares (FXLMS) algorithm [1, 2]. One of the limitations of the FXLMS algorithm is that it exhibits frequency-dependent convergence behavior that can lead to a significant degradation in the overall performance of the control system. Two types of noise will be discussed as they relate to this limitation.

(1) A single tone with time-varying frequency, such as engine noise, where the engine firing frequency changes along with the speed of the engine in revolutions per minute (rpm) during operation. It is assumed that the signal power of the tone in the reference remains the same, independent of frequency. This type of noise will be referred to as "swept tone noise."

(2) Noise containing multiple quasistationary tones, such as helicopter cabin noise, where multiple rotating parts contribute strong tones that do not vary significantly in frequency during normal operation. This type of noise will be referred to as "multiple tone noise."

Various adaptations to the FXLMS algorithm have been developed in an effort to overcome the performance loss due to its frequency-dependent convergence behavior. The normalized FXLMS algorithm [3] has a variable convergence parameter that changes with the power of the input for noise containing a single tone. Clark and Gibbs and Lee et al. [4, 5] developed a method to process tonal components of a multiple tone noise problem separately allowing for a different convergence parameter for each tone. More uniform convergence and increased overall attenuation of all tones are achieved at the expense of more computational complexity. Kuo et al. improved convergence for multiple tone noise by optimizing the magnitude of internally generated reference signals as the inverse of the secondary path magnitude [6]. This approach requires that the user have control over the reference tone amplitudes. The drawback of most of these approaches is that they increase the computational burden of the algorithm, increase the algorithm's complexity, or are not applicable to one of the two types of noise considered here.

Elliot and Cook preconditioned the input to the LMS update by using a second filter that was the inverse of the minimum phase part of the secondary path estimate, thus "whitening" the input and making convergence independent of resonances in the secondary path [7].

Prior research by the authors proposed the eigenvalue equalization filtered-x least mean squares (EE-FXLMS) algorithm [8]. This algorithm improves performance without increasing the computational burden or complexity of the algorithm. The development of the algorithm came from focusing on the eigenvalues of the autocorrelation matrix of the filtered-x signal, which relate to the dynamics or time constants of the modes of the system. Typically, there is a large spread in the eigenvalues of this matrix, corresponding to fast and slow modes of convergence. If the variance in the eigenvalues of the autocorrelation matrix is minimized, convergence properties will be more uniform and controller parameters could be optimized for all frequencies leading to increased performance (faster convergence speed and additional noise attenuation) of the controller.

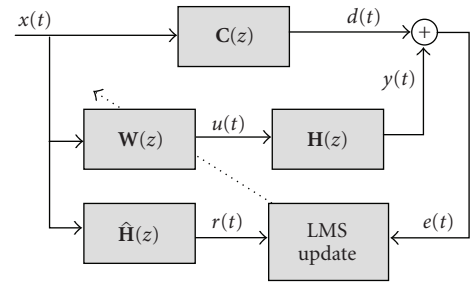
For the EE-FXLMS algorithm, adjustments to the secondary path estimate are made in the frequency domain. The phase of the original secondary path transfer function estimate is preserved while the magnitude coefficients are adjusted to have the inverse trend of tones in the reference signal. The new magnitude coefficients are combined with the original phase response and transformed back into the time domain, giving a new FIR estimate of the secondary path to filter the reference signal. This is intended to equalize the power of tonal components in the filtered-x signal, which in turn would equalize the eigenvalues of the filtered-x autocorrelation matrix.

Previously, the EE-FXLMS was implemented for swept tone noise by making each secondary path transfer function coefficient flat (equal amplitude) over frequency because, as noted, the power of the reference signal was independent of frequency. For multiple tones, the trend of the magnitude coefficients was made to be the inverse trend of the amplitudes of the tones in the reference signal. For both cases, this led to more uniform eigenvalues (of the filtered-x autocorrelation matrix), faster convergence times, and additional attenuation at the error sensor [8].

This paper revisits the EE-FXLMS implementation to modify the magnitude coefficients as motivation for investigating improved methods of adjusting the magnitude coefficients. In this work, a genetic algorithm is used to find the optimal magnitude coefficients for a limited frequency range, swept tone noise, and for a specific set of reference tones for multiple tone noise. Experimental results for ANC in a mock cabin enclosure for these control implementations are presented and compared.

## 2. BACKGROUND

The FXLMS algorithm involves adaptively filtering a reference signal taken from the noise source to create a control signal that attenuates the unwanted noise. The LMS update is used to change the control filter coefficients such that the measured residual noise is minimized. The measured residual is called the error signal and for this research, it will be utilized to minimize a squared pressure (SP) quantity. The mean squared error is a quadratic function of the filter coefficients with a unique global optimum. The LMS update is a gradient descent search method. It follows the path of steepest descent on the error surface toward the optimum filter weights. A block diagram for a single-channel implementation of the FXLMS algorithm is shown in Figure 1. In Figure 1, and in all equations presented, the variable $t$ is a discrete time index and the variable $z$ a discrete frequency domain index.




$d(t)$: Desired signal
$u(t)$: Control signal
$y(t)$: Output signal
$x(t)$: Reference signal
$e(t)$: Error signal
$r(t)$: Filtered-$x$ signal
$\mathbf{C}(z)$: Plant
$\mathbf{W}(z)$: Adaptive filter
$\mathbf{H}(z)$: Actual secondary path
$\hat{\mathbf{H}}(z)$: Secondary path estimate

Figure 1: Block diagram of the FXLMS algorithm.

### 2.1. Secondary path transfer function

The FXLMS algorithm derives its name from the filtered-x signal, $r(t)$, which is the convolution of the reference signal $x(t)$, with $\hat{\mathbf{h}}(t)$, a finite impulse response (FIR) estimate of the secondary path transfer function. The secondary path transfer function (shown in Figure 1 as $\mathbf{H}(z)$) includes the effects of digital-to-analog and analog-to-digital converters, filters, audiopower amplifiers, loudspeakers, the acoustical transmission path, error sensors, and other signal conditioning.

The secondary path model, $\hat{\mathbf{H}}(z)$, is estimated through a process called system identification (SysID). The SysID process is performed offline (before ANC is started) for the fastest convergence of the algorithm where the secondary path does not change significantly during operation of the system. Band-limited white noise is played through the control speaker(s) and the output is measured at the error sensor(s). The measured impulse response is obtained as an FIR filter, $\hat{\mathbf{h}}(t)$, that represents $\hat{\mathbf{H}}(z)$. The coefficients of $\hat{\mathbf{h}}(t)$ are stored and used to prefilter the input signal to the LMS update to run control. While inclusion of $\hat{\mathbf{H}}(z)$ is necessary for stability, the FXLMS algorithm is robust to errors in its estimation. The algorithm will converge (slowly) as long as phase errors are less than 90° [1] and phase errors less

than 40° do not significantly affect convergence [9]. The gain applied to the reference signal by filtering it with $\hat{\mathbf{h}}(t)$ does not affect the stability of the algorithm and is usually compensated for by modifying the convergence parameter $\mu$.

### 2.2. FXLMS convergence and eigenvalues of filtered-x autocorrelation matrix

The time constants for the modes of convergence of the ANC system are determined by the eigenvalues of the autocorrelation matrix of the filtered-x signal [10]. While the convergence parameter, $\mu$, can be optimized to give fast convergence for one mode, others will converge more slowly. For swept tone noise, $\mu$ can be optimized for a given frequency in the range of the sweep, but not for all frequencies in the range. When the algorithm is controlling a tone at a frequency other than that for which it was optimized, convergence will be slower and attenuation less. For multiple tone noise, the algorithm will be able to attenuate portions of the total noise quickly while other tones in the noise will linger and take longer to converge.

The properties of the filtered-x signal, and hence the autocorrelation matrix, are a function of the magnitude response of $\hat{\mathbf{H}}(z)$ and the spectrum of the reference signal. The autocorrelation matrix of the filtered-x signal is defined as

$$\mathbf{R} = E[\mathbf{r}(t)\mathbf{r}^T(t)], \tag{1}$$

where $E[\cdot]$ denotes the expected value of the operand which is the filtered-x signal vector, $\mathbf{r}(t)$, multiplied by the filtered-x signal vector transposed $\mathbf{r}^T(t)$. In general, it has been shown that the FXLMS algorithm (or any of its variations) will converge (in the mean) and remain stable as long as the chosen $\mu$ satisfies the following equation [9]:

$$0 < \mu < \frac{2}{\lambda_{\max}}, \tag{2}$$

where $\lambda_{\max}$ is the maximum eigenvalue of the autocorrelation matrix.

In practice, it is computationally demanding to obtain a real-time estimate of the autocorrelation matrix, so the optimal $\mu$ is often selected through experimentation. In this work, the structure of the eigenvalues of a given ANC problem is explored using an offline estimate of the autocorrelation matrix. This is done in a numerical analysis program by taking an actual $\hat{\mathbf{h}}(t)$ model from a mock cabin enclosure, convolving this with a reference signal for the given noise application, computing the autocorrelation matrix, and getting the eigenvalues. If a single frequency reference signal is used, $\lambda_{\max}$ can be computed for that frequency. If the simulation is repeated over a range of frequencies, $\lambda_{\max}$ for a single tone at each frequency in that range can be found. For control of a single tone, $\lambda_{\max}$ is the only eigenvalue of interest since it will determine the convergence of the algorithm for that frequency. Figure 2 (solid line) shows an offline simulation using an actual $\hat{\mathbf{h}}(t)$ from the mock cabin enclosure, and equal amplitude tonal inputs from 0–300 Hz. The disparity in $\lambda_{\max}$ over frequency shows how the convergence of the algorithm will change as it controls a single tone swept through this range. The range of interest from 0–300 Hz was selected because the experimental hardware was set with a cutoff frequency at 400 Hz. The eigenvalues in the figure have been normalized to the largest eigenvalue in the range.

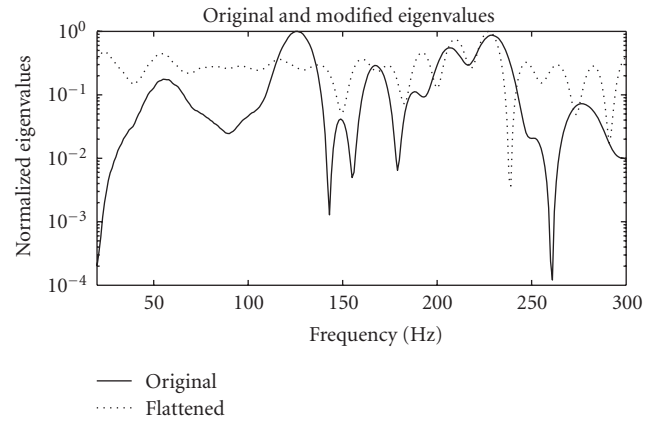



Figure 2: Plot of normalized maximum eigenvalues over frequency for original and modified (flat magnitude) eigenvalues.

The largest eigenvalue for a single tone occurs at about 125 Hz. This location corresponds to the largest $\mu$ that is stable for the entire frequency range from 0–400 Hz as given by (2). All other frequencies have a smaller eigenvalue and could use a larger $\mu$, and still be stable, if just that particular frequency was targeted for control. Frequencies at the valleys of the solid line in Figure 2 have the smallest eigenvalues and could use the largest $\mu$'s and still be stable, again if they were the only frequencies targeted for control. The larger $\mu$'s are especially desirable for nonstationary noise as they lead to faster convergence and increased attenuation.

For multiple tone noise that is stationary, the eigenvalues are not computed for individual tones as before, but for the composite reference signal containing all tones to be controlled. In this case, the disparity among all of the nonzero eigenvalues, not just $\lambda_{\max}$, gives information about how different spectral components of multiple tone noise will converge.

## 3. EIGENVALUE EQUALIZATION—PRIOR METHODS

If the variance in the eigenvalues of the autocorrelation matrix was minimized, a single-convergence parameter could then be chosen that would be nearly optimal for all frequencies targeted for control and the algorithm would converge at nearly the same rate at all frequencies or for all modes of convergence. Additionally, "misadjustment" errors that prevent the algorithm from converging to the true optimal solution depend on the eigenvalues of the autocorrelation matrix. Misadjustment error is larger when there is large disparity in the eigenvalues [11]. Misadjustment (and hence attenuation) can be improved by making these eigenvalues equal.

As previously stated, the autocorrelation matrix is directly dependent on the filtered-x signal, which is computed by filtering the input reference signal, $x(t)$, with $\hat{\mathbf{H}}(z)$. Any attempt at equalizing the eigenvalues must be done by altering either the reference signal or the secondary path model. Adjusting the power of the reference signal has been shown to be an effective way of doing this [6]; however, in many applications this amount of control over the reference signal is not feasible. We focus on making changes to $\hat{\mathbf{H}}(z)$ only. The span, defined as $\lambda_{\max}$ divided by $\lambda_{\min}$, is used as a metric to quantify any improvement in the eigenvalue disparity. This ratio is the most important property, as any change in the actual magnitude of the eigenvalues is compensated for by making a complementary adjustment to the magnitude of the convergence parameter $\mu$.

### 3.1. Eigenvalue equalization applied to swept tone noise

For swept tone noise, it has been shown that flattening the magnitude coefficients of $\hat{\mathbf{H}}(z)$, while preserving the phase reduces the variance in the eigenvalues [8]. Figure 2 shows both the original eigenvalues (solid line) and the modified eigenvalues (dotted line) when the magnitude coefficients of $\hat{\mathbf{H}}(z)$ are flattened. In the figure, the eigenvalues for both the original and modified cases have been normalized by the largest of the original eigenvalues. The span for the original eigenvalues in this range (0–400 Hz on the plot) is $1.385 \times 10^5$ and the span for the flattened magnitude $\hat{\mathbf{H}}(z)$ is 162.3. These modifications to $\hat{\mathbf{H}}(z)$ make a noticeable improvement in the performance of the algorithm [8]. The more uniform rate of convergence of all modes of the system is beneficial as it speeds up the overall convergence of the error signal. For dynamic signals, this increased rate of convergence equates to greater attenuation, as it also results in more rapid tracking.

The eigenvalues are much more uniform, but still not perfectly uniform. This is due to the finite resolution of the digital system and of the sampled secondary path estimate. The shape of the magnitude response, $\hat{\mathbf{H}}(z)$, can only be constrained to some value at its respective frequency bins; there is no guarantee that the response of $\hat{\mathbf{H}}(z)$ is also flat between frequency bins. As an example, a 128 coefficient $\hat{\mathbf{H}}(z)$ model sampled at 2000 Hz will have a frequency resolution of 15.625 Hz. For swept tone noise, the system may be excited at any frequency in the range of the application. An estimate of the "analog" or continuous response of $\hat{\mathbf{H}}(z)$ between frequency bins can be made by zero padding the 128-coefficient model before computing the fast Fourier transform (FFT). The original, flattened, and zero-padded flattened magnitude coefficients of $\hat{\mathbf{H}}(z)$ from a mock cabin are shown in Figure 3. The discrete magnitude response is indeed flat at the frequency bin values, but the zero-padded model shows that the true response deviates from flat in between bins. This magnitude variation between frequency bins in $\hat{\mathbf{H}}(z)$ contributes to the residual variation seen in the modified eigenvalues for the range.

Another source of variation may come from frequency leakage when the reference signal gets downsampled before being convolved with $\hat{\mathbf{h}}(t)$. Before being convolved with $\hat{\mathbf{h}}(t)$, the reference signal is downsampled with the same sampling frequency as was used to find $\hat{\mathbf{h}}(t)$; for this example, 2000 Hz was used. In addition, only $n$ number of samples are kept of the reference signal at a given time, where $n$ is the number of coefficients in $\hat{\mathbf{h}}(t)$; for this example, 128 was used. This downsampling process causes amplitude estimation error in the frequency domain due to leakage. Thus if the original reference signal is assumed to be equally weighted at each frequency, as was done to create the eigenvalue simulations shown in Figure 2, the actual reference signal used in those simulations is no longer equally weighted over frequency. This also contributes to residual variation in the eigenvalues.

### 3.2. Eigenvalue equalization applied to multiple tone noise

When multiple noise sources are present, a reference signal may be obtained from each and combined into a single-reference signal. In some cases, the reference signal will contain a fundamental frequency and harmonics from a single-noise source. In either case, the combined tones in the reference signal will in general have different amplitudes. This weighting of the reference tones will be specific to each application and depends on how the reference signals are conditioned and combined. This frequency-dependent weighting of the reference tones as well as the gain applied by the secondary path estimate contributes to the eigenvalue disparity for multiple tone noise. For multiple tone noise in this research, an arbitrary (but specific, for consistency) weighting was applied to the reference tones. The amplitude of the reference signal tones was defined by

$$\text{Tonal Amplitude} = -0.0036^{*}(\text{Tonal Frequency}) + 1.18. \tag{3}$$

This gave a decreasing trend in amplitude for increasing frequency that ranged from 1.0 at 50 Hz to 0.1 at 300 Hz. All tones used for the multiple tone noise were in this range.

To equalize the eigenvalues for this case, a trend line connecting the peaks of the tones (on a power spectrum plot) in the reference is drawn. The inverse of this line gives the desired trend for the magnitude coefficients in $\hat{\mathbf{H}}(z)$, which here corresponds to the inverse of (3). Since the tonal amplitudes for the test case were specified, obtaining the inverse trend line was straightforward. In actual implementation, an offline "Ref ID" process would also be required. This would entail recording the reference signal under normal operating conditions for the system at the sampling frequency used by the controller. The desired magnitude trend for the modified Sys ID filter could be obtained from the fast Fourier transform (FFT) or power spectrum plot of the reference.

This type of modified $\hat{\mathbf{H}}(z)$ is designated as an "X-inverse" model. Figure 4 shows the trend line for the amplitude of tones in the reference as given by (3), the desired magnitude response for $\hat{\mathbf{H}}(z)$, and the zero-padded response of the 256 coefficients X-inverse model. All curves have been normalized in the figure. As before, the response of the filter between bins deviates from the trend assigned

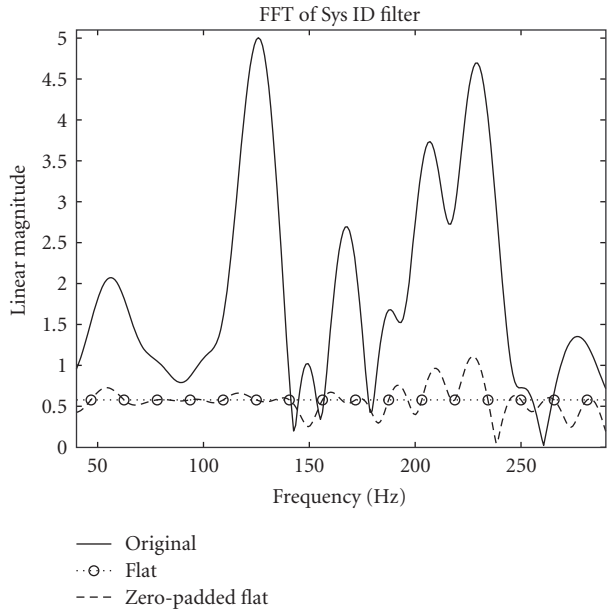



Figure 3: Plot of original-, flattened-, and zero-padded flattened magnitude coefficients of $\hat{\mathbf{H}}(z)$.

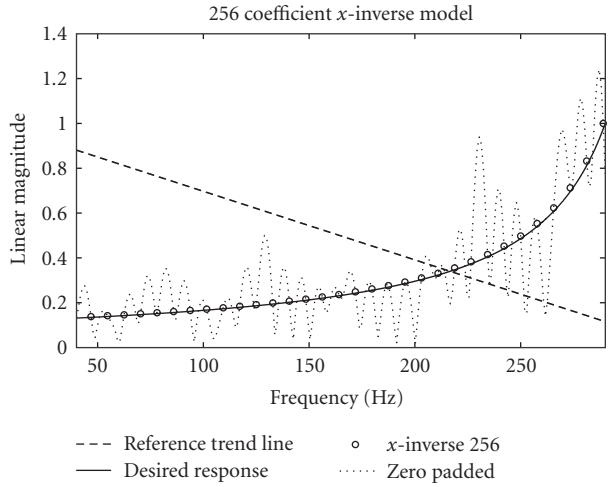



Figure 4: Reference tone amplitude trend line for multiple tone noise signals with desired trend and zero-padded X-inverse model magnitude responses.

the coefficients. Increasing the coefficients from 128 to 256 makes the magnitude response match the desired curve at more points, but does not improve the variation between bins. The same is true for the phase response.

This method will reduce the eigenvalue variation only for some cases. If the tones in the reference are chosen to correspond exactly to frequency bin values, the eigenvalues are much more uniform using the X-inverse model than using the original model. However, if the tones lie off these frequency bin values, the eigenvalue span can be worse than for the unmodified $\hat{\mathbf{H}}(z)$.

Table 1: Comparison of eigenvalue span for original and X-inverse models for multiple tones with frequencies on and off frequency bin values.

| | Sys ID length | Original | X-inverse |
|---|---|---|---|
| Tones on bin values | 128 | 4350 | 2.0 |
| | 256 | 2162 | 2.0 |
| Tones off bin values | 128 | 217 | 16 |
| | 256 | 239 | 991 |

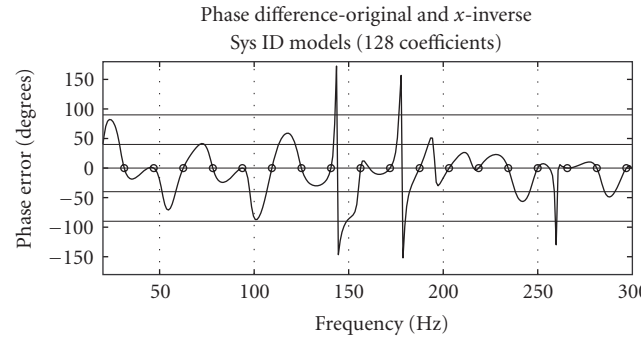



Figure 5: Phase difference between 128-coefficient original and X-inverse models with reference tones off frequency bin values. Dashed lines indicate tonal frequencies.

Two reference signals containing six tones were made for comparison; one with all six tones on frequency bins (62.5, 93.75, 125, 171.875, 203.125, and 296.875 Hz) and the other with these tones shifted slightly to lie between bin values (50, 100, 130, 180, 200, and 280 Hz). The length of the FIR filter model of $\hat{\mathbf{H}}(z)$ was increased from 128 to 256 to double the resolution in an attempt to constrain the magnitude response between bins to follow more closely the desired trend. The eigenvalue span for these reference signals with the original and X-inverse $\hat{\mathbf{H}}(z)$ models of different lengths were calculated. The results of these comparisons are shown in Table 1. When the tones lie on the frequency bins, the X-inverse model gives a significant improvement in the eigenvalue span. For offbin frequencies the X-inverse model is better than the original for the 128-coefficient filter, but not as good as when the tones are on bins. When the filter length is increased to 256, the span for the X-inverse model was worse than the original model for offbin tones. The span for the X-inverse model with 256 coefficients went from 239 to 991 likely because the magnitude response of the X-inverse model goes almost to zero at 200 Hz (see Figure 4). Increasing the resolution by using a longer filter does not (at least in some cases) improve the eigenvalue span. This gives the desired magnitude response at a larger number of points, but the deviation from the desired trend in between these points is not necessarily improved.

The eigenvalue span for the 128-coeffficient X-inverse model and tones at offbin frequency values was reduced significantly over the original model from 217 to 16. However, Figure 5 shows that the phase errors introduced exceed stability limits near several tones in the reference.

Horizontal lines mark 40° and 90° of phase error between the original and modified phase response and vertical dashed lines show the positions of offbin reference tones. Phase errors introduced into regions where no tonal components of the noise are being controlled will not affect the stability or performance of the system. For helicopters, the tones in the noise are very stable and do not shift in frequency significantly, however, phase errors very near those tonal frequencies are potentially problematic for both stability and overall performance of the ANC system.

In this case, even though the eigenvalue span was improved, the X-inverse model would not work well if used in ANC since instability and poor performance would result from the phase issues. This is another reason the X-inverse method is inadequate.

The inability to control the magnitude and phase response of the secondary path estimate in between frequency bin values and the unpredictable changes that occur in each when the magnitude coefficients of the original model are modified make the X-inverse method of eigenvalue equalization inadequate. A genetic algorithm approach was developed to optimize the magnitude coefficients, and which can overcome these difficulties for multiple tone noise.

## 4. EIGENVALUE EQUALIZATION—GENETIC ALGORITHM

A genetic algorithm was used to investigate the possibility of getting more uniform eigenvalues over narrow bands of frequencies for swept tone noise and for specific multiple tone noise cases. Optimizing the magnitude coefficients of $\hat{\mathbf{H}}(z)$ in ways other than those described previously may lead to improved eigenvalue span, but are not intuitive. Genetic algorithms (GAs) [12, 13] have gained considerable popularity in recent years for their ability to solve problems with a large number of design variables, multiple local minima and maxima, nondifferentiable functions, or some combinations of these. They can work well for both discrete- and real-valued problems. GA's mimic the natural selection process found in nature that allows individuals with the best "fitness" to survive. Parents are chosen from the most fit individuals of a population of randomly generated designs. These parents are then sent through a reproduction process to exchange and pass on genetic information to new designs (children). As in nature, mutations are introduced occasionally to provide for random variation. Parents and children compete to be included in the next generation. As the generations progress, the random designs converge to a design that has the best fitness.

### 4.1. Genetic algorithm cycle

The genetic algorithm cycle used to optimize the magnitude coefficients of $\hat{\mathbf{H}}(z)$ can be broken down into nine steps. A brief description of each step is now given. It should be noted that other GA's with different cost functions could be investigated. The purpose of this work is to present one such GA and compare the results to other easily implemented techniques.

*(1) Determine a coding for the design*

Each design in a GA consists of a number of independent variables chosen by the designer. Each independent variable is called a "gene," a set of genes giving one design, or "chromosome." As the desired result of the algorithm was to obtain an optimized impulse response model, $\hat{h}(t)$, that could be used in physical experimentation, a 128 or 256 coefficient $\hat{h}(t)$ for the mock cabin described in Section 6 was obtained by the SysID process described in Section 2.1. The FFT of $\hat{h}(t)$ was then taken, and the phase information of $\hat{H}(z)$ was preserved in a vector. The magnitude information of $\hat{H}(z)$ was discarded, as the GA was implemented to find the optimal magnitude coefficients by making each unknown magnitude coefficient a gene. Each design then contained 64 or 128 genes, which were the unknown 128 or 256 magnitude coefficients of $\hat{H}(z)$ (since they are mirrored about the Nyquist frequency).

*(2) Generate an initial population*

Once the coding scheme for a single design was established, a population of N designs was randomly generated. This was done by randomly assigning a value between a minimum value of 0.01 and a maximum of 10 for each gene (magnitude coefficient) in the design. This range was chosen based on some trial and error. If the minimum was set to zero, the GA would make all the magnitude coefficients zero giving a trivial solution of all zero eigenvalues. The maximum value was set to 10 so that the generated designs were close to the overall magnitude values for the original model. The process was repeated N times to generate the entire population. In general, designs with many genes require large population sizes to maintain adequate diversity. The population size was 500.

*(3) Calculate fitness for each design*

After the initial population was randomly generated, each design was evaluated and assigned a fitness value. Each randomly generated set of magnitude coefficients was recombined with the stored phase information, and the inverse FFT was taken to get a new unique model for the impulse response, $\hat{\mathbf{h}}(t)$. This new model was used to compute the eigenvalues of the filtered-x autocorrelation matrix in the same manner as explained in Section 4. For swept tone noise, the eigenvalues were computed over a specified frequency range and then normalized by the largest of the eigenvalues. As the ideal normalized eigenvalue at each frequency would be one, the fitness value was chosen as a sum of the squared errors between the actual value of each eigenvalue in the frequency range and one, as shown in (4):

$$\text{fitness} = \sum_{f_{\text{start}}}^{f_{\text{end}}} (1 - \lambda_k)^2. \tag{4}$$

The fitness value for multiple tone noise was simply the span ($\lambda_{\max}$ divided by $\lambda_{\min}$) of all nonzero eigenvalues.

In addition, a penalty was applied to any design whose phase response was in error by more than 40° in a range of +/− 5 Hz around each of the tonal frequencies in the multiple tone noise. This was done to decrease the design's sensitivity to tonal frequencies shifting. Constraining the phase in this way ensures that the algorithm will remain stable for small changes in the tonal frequencies. Designs whose performance would be hindered by the phase error introduced by altering the magnitude response were assigned a poor fitness value.

*(4) Selection of parents*

A tournament selection process was used to choose parent designs from the population. A specified number of designs were randomly selected to compete in the tournament. The design with the best fitness wins the tournament and was made a parent design. This process was repeated until enough parents had been selected to make N children; a set of two parent designs producing a single-child design.

*(5) Perform crossover*

A process called crossover exchanged traits from each parent design and created children designs. In this way, new designs were made that had traits from each parent. For this work, blend crossover was used. In blend crossover, genes from both parents are blended to make two new children genes. This occurs gene by gene. First, a random number between zero and one is chosen for each gene to determine whether crossover will occur. If the random number is larger than the user defined crossover probability, no crossover occurs. The genes for the children, $c_1$ and $c_2$, are equal to the parent genes, $p_1$ and $p_2$, respectively, so that if no crossover occurs for any genes in the design, the children will be identical to the parents. If the random number is less than the user specified crossover probability, another random number is chosen. If it is <0.5, the blend parameter, $a$, is calculated by

$$a = \frac{(2r)^{1/\eta}}{2}, \tag{5}$$

and if the random number is ≥0.5, the blend parameter is

$$a = 1 - \frac{(2 - 2r)^{1/\eta}}{2}. \tag{6}$$

The children genes $c_1$ and $c_2$ are created from the parent genes $p_1$ and $p_2$ by

$$\begin{aligned} c_1 &= (a)p_1 + (1 - a)p_2, \\ c_2 &= (1 - a)p_1 + (a)p_2. \end{aligned} \tag{7}$$

The value of $\eta$ is chosen by the user. As $\eta \to 0$, the crossover becomes uniform, meaning that $c_1 = p_2$ and $c_2 = p_1$. As $\eta \to \infty$, $a \to 1/2$ and the children's genes are the average of the parent's gene values.

The crossover probability was chosen to be 50% and $\eta$ was 0.5.

*(6) Perform mutation*

After crossover, some of the genes in the children designs are mutated. Mutation provides for diversity and occasionally introduces new beneficial information into a design. Higher mutation probability maintains more diversity in the designs as the generations progress and can help the algorithm avoid converging on a local optimum in the design space. Mutation can be made dynamic allowing for high diversity initially, keeping the algorithm from settling prematurely in a local optimum. In later generations, mutation is constrained allowing the algorithm to randomly make fine adjustments to the design once it is near what is hoped to be the global optimum. Initially, mutation can cause the gene to become any value in the allowable range for that gene. By the last generation, when mutation occurs the new value for the gene is only allowed to have a new value that is very close to the original. The probability of mutation occurring does not change, only how different the mutated gene is allowed to be from its premutation value. This is done by introducing a dynamic mutation parameter $\alpha$:

$$\alpha = \left(1 - \frac{n - 1}{N}\right)^{\beta}, \tag{8}$$

where $n$ is the current generation number and $N$ is the total number of generations. The exponent, $\beta$, is a user defined parameter that weights the dynamic function of $\alpha$. If $\beta = 0$, $\alpha$ will always be one and the amount of mutation allowed will be uniform for all generations. If $\beta$ is greater than zero, the amount of mutation allowed decreases as the generation number increases.

A random number is chosen to determine whether mutation will occur for each child gene. If the random number is less than the user-specified mutation probability, another random number, $c_{\text{mut}}$, is chosen within the allowable range for that gene. If $c_{\text{mut}}$ is less than the current value for the gene, the new gene value is

$$c_{\text{new}} = c_{\min} + \left(c_{\text{mut}} - c_{\min}\right)^{\alpha}\left(c - c_{\min}\right)^{(1-\alpha)}, \tag{9}$$

and if $c_{\text{mut}}$ is greater than the current value for the gene, the new gene value is

$$c_{\text{new}} = c_{\max} - \left(c_{\max} - c_{\text{mut}}\right)^{\alpha}\left(c_{\max} - c\right)^{(1-\alpha)}. \tag{10}$$

The mutation probability was chosen to be 50% and $\beta$ was set to 0.5.

*(7) Measure fitness of children*

Once all of the children were created through crossover and mutation, the fitness value of each child was computed in the same way as described in Step 3.

*(8) Perform elitism*

Once each child design has a fitness value, parents are made to compete with children in a process called elitism. All of the parents and children are sorted by their fitness value, and the N number of designs with the best fitness value becomes the starting generation for the next iteration of the algorithm.

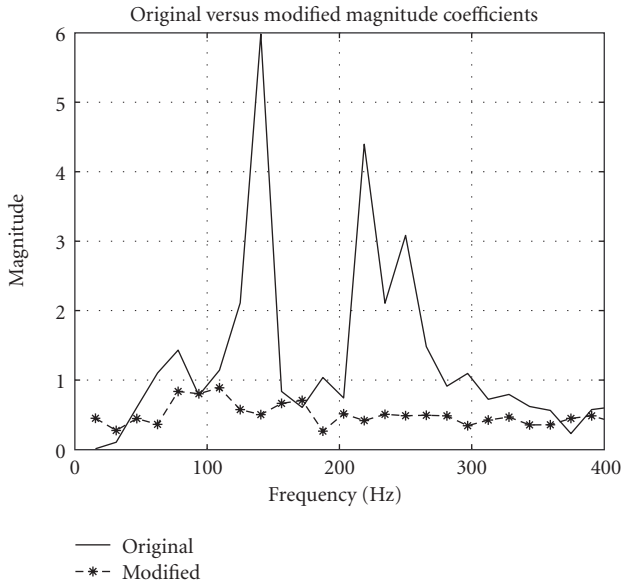



Figure 6: Original and modified magnitude coefficients of $\hat{H}(z)$ for genetic algorithm run from 60–90 Hz.

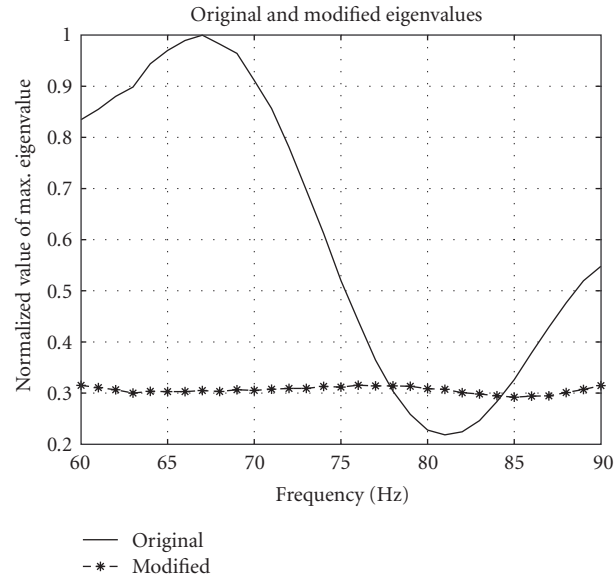



Figure 7: Normalized original and modified eigenvalues for genetic algorithm run from 60–90 Hz.

*(9) Repeat Steps 4–8 for $M$ number of generations*

Steps 4–8 were repeated for M number of generations. The number of generations needed to be large enough to allow the algorithm to converge on an optimum design. For the work reported here, M was chosen to be in the range of 50–200.

The optimization performed by the genetic algorithm is all done offline as part of the setup of the ANC system after the offline system identification routine. It is not run in real-time and so is currently limited to use with systems where the secondary path does not change significantly and can be characterized offline.

### 4.2. Genetic algorithm results

*(1) Swept tone noise*

The GA was unable to produce a design that had lower eigenvalue span than the flattened magnitude design when optimizing for the entire range of frequencies from 0–400 Hz. There were not enough degrees of freedom in the design variables to get a better result. The frequency range for swept tone noise was reduced to a much smaller range to see if the genetic algorithm could improve the span of the eigenvalues in a smaller range. The GA was run for swept tone noise in the range 60–90 Hz with 128 filter coefficients. The results for 60–90 Hz are shown in Figures 6 and 7. Figure 6 shows the original- and new-modified magnitude coefficients, and Figure 7 shows the resulting eigenvalues. As before, the eigenvalues in both the original and modified case have been normalized by the largest of the original eigenvalues. The eigenvalues from the genetically optimized magnitude coefficients are more uniform. The eigenvalue span for the genetic algorithm model approach was 1.08,

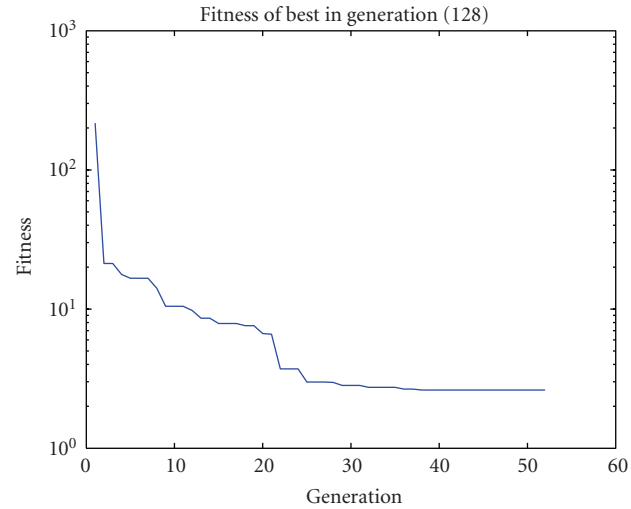



Figure 8: Fitness history for genetic optimization of 128-coefficient Sys ID model.

which is improved over both the eigenvalue span of 4.578 and 2.45 from the original and the flattened models, respectively.

*(2) Multiple tone noise*

The GA was also run for the reference signal containing six offbin tones, as described in Section 4.2, for 128 and 256 filter coefficients. The genetic algorithm was able to find a magnitude response that is unlike the X-inverse model and reduces the eigenvalue span to 5.8 for 128 coefficients and 5.3 for 256 coefficients. These values can be compared to the results for the other methods shown in Table 1, which have span values typically several orders of magnitude higher. Figure 8 shows the fitness history of the best design in each

Table 2: Comparison of eigenvalue span for original and GA magnitude coefficients for offbin tones shifted in frequency from the values for which the GA optimized the model.

| | f shift | −10 | −6 | −4 | −2 | 0 | 2 | 4 | 6 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Original model | 128 span | 193.5 | 209.3 | 217.5 | 220.7 | 216.8 | 213 | 201.1 | 202.2 | 392.7 |
| | 256 span | 64.4 | 265.5 | 444.0 | 361.7 | 238.5 | 169.6 | 174.8 | 500.0 | 1240.1 |
| Genetic model | 128 span | 9.0 | 5.1 | 4.3 | 5.0 | 5.8 | 7.6 | 9.9 | 12.8 | 24.4 |
| | 256 span | 23.9 | 11.3 | 28.2 | 13.0 | 5.3 | 12.9 | 51.4 | 132.0 | 1309.2 |

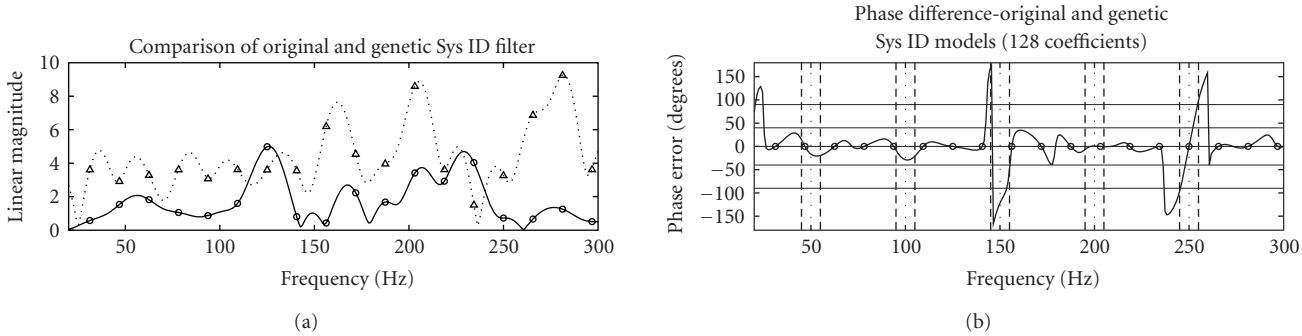



Figure 9: Magnitude of original and genetic 128-coefficient models along with the phase error in the genetic model.

generation of the 128-coefficient optimization and gives an idea of the dynamics of the genetic optimization. The fitness is the span plotted on a logarithmic scale.

The optimized magnitude coefficients found by the genetic algorithm are specific to the noise problem given to the genetic algorithm. If the tones shift in frequency or change in amplitude, the result is no longer guaranteed to be an optimum result. To see how sensitive the genetic algorithm model is to shifts in the tonal frequencies, the eigenvalue span for reference signals with all tones shifted by +/− 2 Hz up to +/− 10 Hz was calculated. This is compared to the original model in Table 2. In general, the farther the tones get from the frequencies for which the magnitude was optimized, the worse the span gets. It is difficult to predict how sensitive a genetic model will be for any given application without first performing the optimization. The sensitivity will depend on how much the magnitude response of the genetic model varies near the frequencies for which it was optimized. The phase is guaranteed to be within acceptable error +/− 5 Hz from the tonal frequencies by the GA and so the design remains robust in terms of stability for changes within this range. Where more shift in the tones is anticipated the GA can be constrained accordingly. The magnitude of both the original and genetic 128-coefficient models is plotted together in Figure 9 along with the phase error in the genetic model (difference between the two).

## 5. EXPERIMENTAL RESULTS

Experiments were performed to verify that the reduction in eigenvalue span demonstrated in Section 4 also leads to better ANC performance. First, the experimental setup will be explained, then ANC results for swept sine noise over the three ranges (60–90 Hz, 90–120 Hz, and 120–150 Hz) and for multiple tone noise (at offbin frequencies) will be shown.

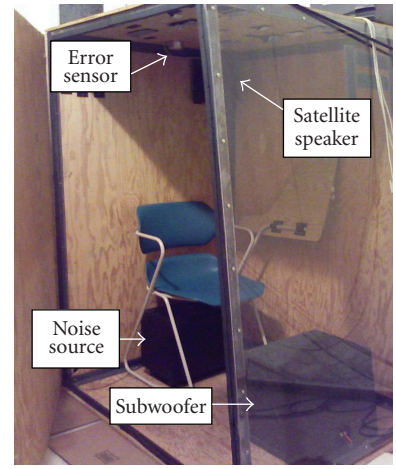



Figure 10: Photo of inside of mock cab.

### 5.1. Experimental setup

The experiments were conducted inside a mock cabin enclosure with nominal dimensions of 1.0 m × 1.5 m × 1.1 m. The cabin has a steel frame, 0.01 m thick plywood sides, and a 0.003 m thick Plexiglas front panel. A speaker placed under a chair served as the primary sound source, and two loudspeakers were setup in a single channel control configuration. A crossover circuit routed the low-frequency content (below 90 Hz) to a subwoofer on the floor of the cab, and the high-frequency content (above 90 Hz) to a satellite

Table 3: Comparison of control performance using original, flattened, and genetic $\hat{h}(t)$ models for swept tone noise.

| Frequency range | Type of $\hat{h}(t)$ model | $\mu$ ($0.1^{*}\mu_{\max}$) | Error Mic avg. reduction (dB) | Reproducibility (dB) | Additional reduction compared to original (in dB) |
|---|---|---|---|---|---|
| 60–90 Hz | Original | 1e-8 | 14.8 | 0.16 | |
| | Flattened | 3e-8 | 21.9 | 0.37 | 7.1 |
| | Genetic | 5e-8 | 20.8 | 0.17 | 6.0 |

speaker mounted in the top corner of the cab, near the back. An error microphone was placed on the ceiling near where an operator's head would be. The performance of the algorithms will be reported at the error sensor. Figure 10 shows the cab, error sensor, and speakers.

The adaptive control filter consisted of 32 taps for swept tone noise and 100 taps for multiple tone noise. Secondary path transfer functions were modeled with either 128 or 256 taps. The convergence coefficient, $\mu$, was determined experimentally by finding the largest stable value for the noise signal under test and then scaling it back by a factor of ten to ensure stability. All input channels were simultaneously sampled at 2 kHz, and all input and output signals had 16 bits of resolution. Fourth-order Butterworth low-pass filters (400 Hz cutoff) provided antialiasing and reconstruction of input and output signals, respectively.

### 5.2. Experimental results—swept tone noise

Each $\hat{h}(t)$ model was tested for swept tone noise over the frequency ranges 60–90 Hz, 90–120 Hz, and 120–150 Hz. A test signal was created for each frequency range that consisted of a sine wave being swept up and down over the frequency range at a rate of 2 Hz/sec. The time-averaged sound pressure level (SPL) over the entire duration of the test signal was measured with and without control running. Each measurement was performed three times for computation of an average and to give a sense of the measurement's reproducibility. The attenuation (the difference in SPL with control off and on) using all three $\hat{h}(t)$ models is shown in Table 3. The "reproducibility" shown in Table 3 was calculated in the same manner as a standard deviation, although it is recognized that the small sample size precludes referring to the result as a statistically valid standard deviation.

In the range from 60–90 Hz, the SPL before running control was about 95 dB (computed over the entire frequency range) and about 73 dB with control for the flattened magnitude model. Figure 11 shows a plot of the frequency spectrum for both control on and control off for the 60–90 Hz range.

The data show that control with the genetic and the flattened models significantly outperformed control with the original $\hat{h}(t)$ model. For the range 60–90 Hz, control with the genetic and flattened models outperformed the original control by 6-7 dB, with control with the flattened model providing 1 dB more control than the genetic model. Experiments for other frequency ranges were also done with similar results.

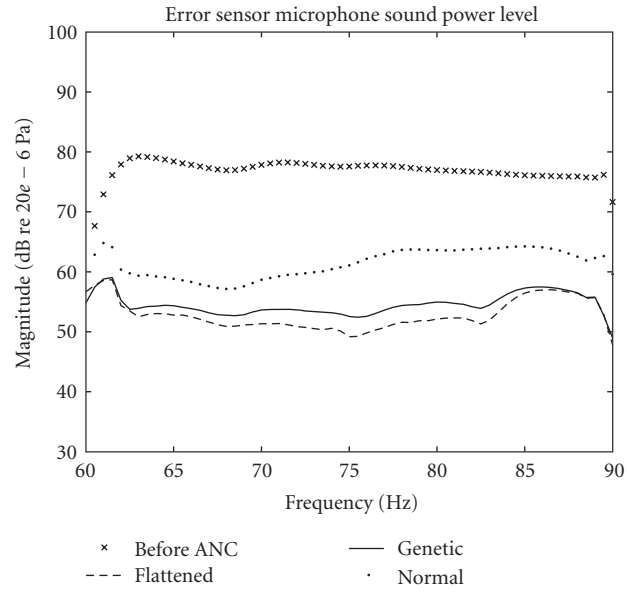



Figure 11: Sound pressure level (SPL) at the error sensor for 60–90 Hz.

### 5.3. Experimental results—multiple tone noise

Multiple tone noise ANC experiments in the mock cab were done using 128 and 256-coefficient original, X-inverse, and genetic secondary path models with the reference signal consisting of multiple offbin frequency tones, as described in Section 4.2. For these tests, three 10-second time records of the error signal were taken as follows:

(1) stationary error signal with control off;

(2) converging error signal from the time control was turned on;

(3) stationary error signal after the algorithm had converged to its eventual steady state level.

The measured performance for each test case was the eventual amount of attenuation (in dB) at the error sensor, calculated from the first and third time records and the convergence time in seconds from the second-time record. The convergence time was taken to be a measure of how long it took the error signal, from the time that control was enabled, to reach $1/e$ of its initial value (about 9 dB attenuation), where $e$ is the base of the natural logarithm. The reason for choosing this was that the convergence time essentially becomes a measure of the rate of attenuation,

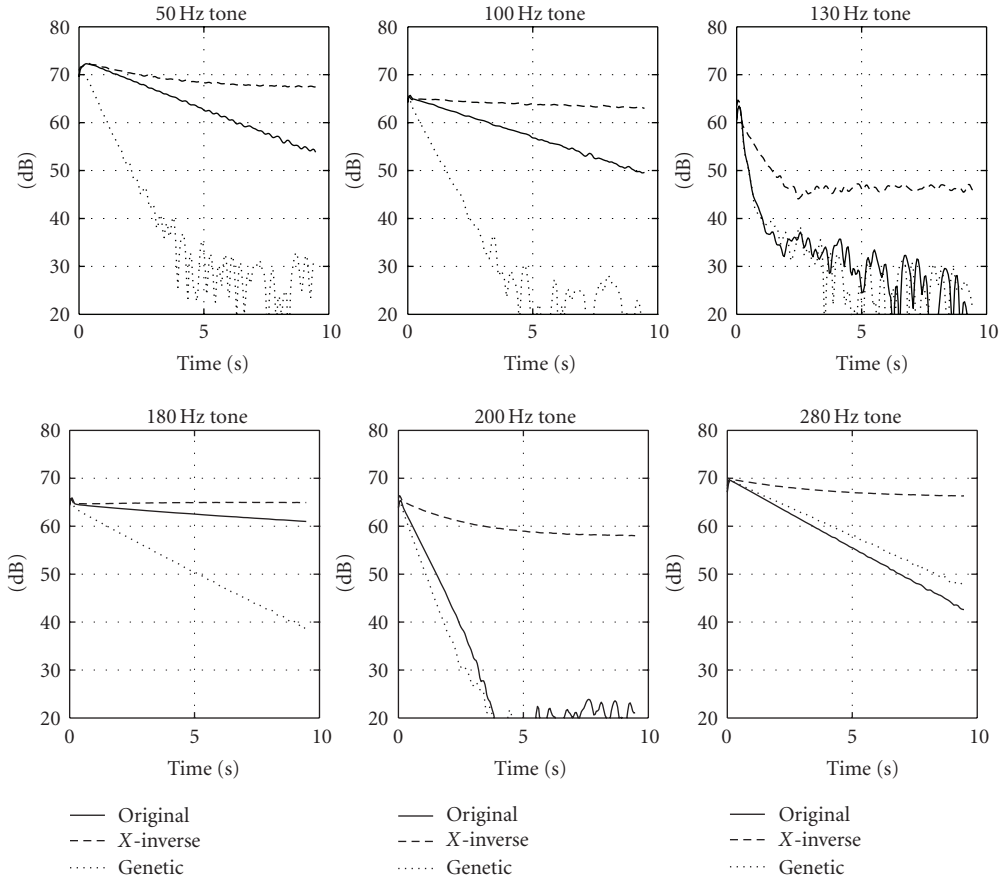



Figure 12: Learning curves for individual tones of multiple noise test case for 128-coefficient original, X-inverse, and genetic models.

Table 4: Comparison of control performance using original, X-inverse, and genetic $\widehat{h}(t)$ models for multiple tone noise.

| Sys ID type | Convergence time (sec) | Eventual reduction (dB) | Eignvalue span |
|---|---|---|---|
| Normal 128 | 3.9 | −19.5 | 217 |
| Normal 256 | 6.3 | −16.9 | 239 |
| X-inverse 128 | 10+ | −4.6 | 16 |
| X-inverse 256 | 10+ | −2.6 | 991 |
| Genetic 128 | 1 | −25.9 | 2.3 |
| Genetic 256 | 2.2 | −26.1 | 2.4 |

which was felt to be useful when comparing cases where the overall level of attenuation may be significantly different. These results are summarized in Table 4. When a signal did not converge to $1/e$ of its initial value during the second-time capture, it is reported as 10+ seconds. The actual convergence time for these measurements was not calculated. The genetic models for both 128 and 256 coefficients were better than both the original and X-inverse models for both measures of performance. Models that gave lower eigenvalue span performed better with the exception of the 128-coefficient X-inverse model whose performance was worse than expected.

Based on the eigenvalue span, the 128 coefficient X-inverse model should have been a significant improvement over the original model. The reason for the poor performance is a result of phase errors introduced by modifying the magnitude values. As for the magnitude coefficients, preserving the phase coefficients guarantees the phase response will be the same at frequency bins, but changing the magnitude coefficients does have an influence on the response in between these bins. As noted previously, as long as the phase response modeled by the secondary path estimate is within 90° of the true phase response, the algorithm will be stable. Comparing the zero-padded phase response for the original 128-coefficient model and the X-inverse model reveals that the phase difference between the two at 100 Hz (one of the tonal frequencies) approaches that limit. While the errors in the original model of the secondary path are not known, it is assumed that it is a better estimate and the X-inverse model deviating from it by close to 90° is the cause for the poor performance.

Figure 12 shows learning curves for the individual tones in the multitone test case for the 128-coefficient model. These are slices along tonal frequencies from a spectrogram of the converging error signal. These plots show the different rates of convergence for the individual tones in the noise. The genetic model converges faster and to a lower level than the other models at all tones except 280 Hz where performance is similar for the original model. This is the fastest converging mode for the algorithm for all secondary path models. The

X-inverse model shows that the 180 Hz tone is diverging due to the >90° phase error at that frequency (see Figure 5).

## 6. CONCLUSIONS

Use of a genetic algorithm to find optimum values for the magnitude coefficients of the secondary path estimate for the FXLMS algorithm while preserving the phase of $\hat{\mathbf{H}}(z)$ has been shown to reduce the variation in the eigenvalues of the filtered-x autocorrelation matrix.

ANC in a mock cab using control with both the flattened and genetic $\hat{\mathbf{h}}(t)$ models provided as much as 6-7 dB additional attenuation over control with the original $\hat{\mathbf{h}}(t)$ model. For these specific swept tone noise tests, the genetically optimized $\hat{\mathbf{h}}(t)$ algorithm did not provide any additional benefit over the flattened model, even though the eigenvalues were more uniform. It is possible that the improved eigenvalues resulting from the genetically optimized model could lead to better performance in other applications.

When considering the more general case for multiple tone noise, with tonal frequencies not corresponding exactly to frequency bins, control with the X-inverse $\hat{\mathbf{h}}(t)$ models performed worse than the original $\hat{\mathbf{h}}(t)$ model. Genetic $\hat{\mathbf{h}}(t)$ models were shown to give 6–9 dB additional attenuation with faster convergence times.

Use of a genetic algorithm as an optimization method in implementing the EE-FXLMS algorithm extends its utility and increases the potential benefit of its use over the FXLMS algorithm. With this method, the eigenvalue disparity can be reduced while assuring performance limiting phase errors are not introduced.

The optimization performed on the secondary path estimate in the EE-FXLMS algorithm in this paper is limited to applications where the secondary path model (at least the phase response) is relatively stable since the secondary path is only characterized and optimization performed as part of the setup of an ANC system. Further work could be done to implement the EE-FXLMS with genetic optimization for a changing secondary path with an online Sys ID routine [14]. The secondary path can be characterized online periodically and the eigenvalue equalization performed in the background while control is running. Everytime a newly optimized secondary path model becomes available, it can be updated and used to run control. The time it would take to get a new-optimized model for the secondary path estimate would be set by the time it takes for the genetic algorithm to execute.

## REFERENCES

[1] D. R. Morgan, "An analysis of multiple correlation cancellation loops with a filter in the auxiliary path," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 454–467, 1980.

[2] J. C. Burgess, "Active adaptive sound control in a duct: a computer simulation," *Journal of the Acoustical Society of America*, vol. 70, no. 3, pp. 715–726, 1981.

[3] S. M. Kuo and D. R. Morgan, "Adaptive transversal filters," in *Active Noise Control Systems: Algorithms and DSP Implementations*, J. G. Proakis, Ed., pp. 33–35, chapter 2, John Wiley & Sons, New York, NY, USA, 1996.

[4] R. L. Clark and G. P. Gibbs, "A novel approach to feedforward higher-harmonic control," *Journal of the Acoustical Society of America*, vol. 96, no. 2, pp. 926–936, 1994.

[5] S. M. Lee, H. J. Lee, C. H. Yoo, D. H. Youn, and I. W. Cha, "An active noise control algorithm for controlling multiple sinusoids," *Journal of the Acoustical Society of America*, vol. 104, no. 1, pp. 248–254, 1998.

[6] S. M. Kuo, X. Kong, S. Chen, and W. Hao, "Analysis and design of narrowband active noise control systems," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '98)*, vol. 6, pp. 3557–3560, Seattle, Wash, USA, May 1998.

[7] S. J. Elliott and J. G. Cook, "A preconditioned LMS algorithm for rapid adaptation of feedforward controllers," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '00)*, vol. 2, pp. 845–848, Istanbul, Turkey, June 2000.

[8] J. K. Thomas, S. P. Lovstedt, J. D. Blotter, and S. D. Sommerfeldt, "Eigenvalue equalization filtered-x algorithm for the multi-channel active noise control of stationary and non-stationary signals," submitted to *Journal of the Acoustical Society of America*.

[9] C. C. Boucher, S. J. Elliott, and P. A. Nelson, "Effect of errors in the plant model on the performance of algorithms for adaptive feedforward control," *IEE Proceedings F*, vol. 138, no. 4, pp. 313–319, 1991.

[10] S. J. Elliott and P. A. Nelson, "Active noise control," *IEEE Signal Processing Magazine*, vol. 10, no. 4, pp. 12–35, 1993.

[11] B. Widrow and S. D. Stearns, *Adaptive Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1985.

[12] L. Davis, *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, NY, USA, 1991.

[13] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[14] S. M. Kuo and D. R. Morgan, "On-line secondary path modeling techniques," in *Active Noise Control Systems: Algorithms and DSP Implementations*, J. G. Proakis, Ed., pp. 213–240, chapter 7, John Wiley & Sons, New York, NY, USA, 1996.