DESIGN AND IMPLEMENTATION OF AN EFFICIENT ACTIVE NOISE CONTROL SYSTEM

ABHISHEK SAHOO (108EC010)

Department of Electronics & Communication Engineering
National Institute of Technology Rourkela

DESIGN AND IMPLEMENTATION OF AN EFFICIENT ACTIVE NOISE CONTROL SYSTEM

A Thesis submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology in "Electronics & Communication Engineering"

By

ABHISHEK SAHOO (108EC010)

Department of Electronics & Communication Engineering
National Institute of Technology
Rourkela-769008 (ODISHA)
May-2012

DESIGN AND IMPLEMENTATION OF AN EFFICIENT ACTIVE NOISE CONTROL SYSTEM

A Thesis submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology in "Electronics & Communication Engineering"

By

ABHISHEK SAHOO (108EE016)

Under guidance of

Prof. POONAM SINGH

Department of Electronics & Communication Engineering
National Institute of Technology
Rourkela-769008 (ODISHA)
May-2012

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA

ODISHA, INDIA-769008

CERTIFICATE

This is to certify that the thesis entitled "**Design and implementation of an efficient Active Noise Control system**", submitted by **Abhishek Sahoo(Roll. No. 108EC010)** in partial fulfilment of the requirements for the award of **Bachelor of Technology** in **Electronics & Communication Engineering** during session 2011-2012 at National Institute of Technology, Rourkela. A bonafide record of research work carried out by them under my supervision and guidance.

The candidates have fulfilled all the prescribed requirements.

The Thesis which is based on candidates' own work, has not been submitted elsewhere for a degree/diploma.

In my opinion, the thesis is of standard required for the award of a bachelor of technology degree in Electrical Engineering.

Place: Rourkela

Dept. of Electronics & Communication Engineering
National institute of Technology
Rourkela-769008

Prof. Poonam Singh
Associate Professor

ACKNOWLEDGEMENTS

On the submission of our thesis entitled "**Design and implementation of an efficient Active Noise Control system**", I would like to extend my gratitude & our sincere thanks to my supervisor **Prof. Poonam Singh**, Associate Professor, Department of Electronics & Communication Engineering for her constant motivation and support during the course of my work in the last one year. I truly appreciate and value her esteemed guidance and encouragement from the beginning to the end of this thesis. Her knowledge and company at the time of crisis would be remembered lifelong.

I am very thankful to my teachers **Prof. Ganapati Panda, Prof. Sarat Kumar Patra, Prof. Ajit Kumar Sahoo** for helping me in my research works. I am very thankful to **Nithin V George**, PhD scholar of School of Electrical Sciences, IIT Bhubaneswar for providing solid background in MATLAB software for simulation of my project works. They are great sources of inspiration to me and I thank them from the bottom of our hearts.

At last but not least, i would like to thank the staff of Electronics & Communication engineering department for constant support and providing place to work during project period. I would also like to extend our gratitude to our friends who are with us during thick and thin.

Abhishek Sahoo

B.Tech (Electronics & Communication Engineering)

Dedicated to

My beloved parents

ABSTRACT

.

Noise is an undesired, but unavoidable phenomenon. We cannot stop its generation at the source level, but control it at the listener level up to some extent either by passive or active method. In our work we are only concerned about the active method. In this paper we have proposed several types of adaptive algorithms for updating the weights of the digital filter, which acts as the controller. First we have proposed Filtered-X LMS algorithm which is very simple to implement and easy to understand. Then some amount of nonlinearity is introduced in the primary path and the performance is seen to be degraded. So this problem is sorted out by assuming a nonlinear controller using nonlinear algorithms like Volterra series method, Back propagation method for multi-layer perceptron, FLANN filter. After studying these algorithms we introduce a completely different type of algorithm known as evolutionary computing methods, which is based on the population based searching techniques. In this field we have studied 3 algorithms. i.e. Genetic Algorithm, Particle Swarm Optimization, Differential Evolution. A brief comparison is made between them and also the performance is studied in presence of nonlinearity.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background of noise

Sound is the sensation produced at our ear due to small pressure fluctuation in the surrounding medium. Due to the pressure fluctuation, a vibration is induced in the ear drums which produce sensation. Due to the fluctuation, a sound field is created in the atmosphere. This is also termed as acoustic field and the pressure as acoustic pressure. Noise can be basically defined as unwanted sound. It is a very subjective term; what is music to our ears might be noise to some other person.

Like all physical quantities, sound can also be measured. It is often quantified by the pressure of the acoustic field because it is very simple to measure and it is a scalar quantity. The standard unit for its measurement is Pascal, abbreviated as Pa. In a qualitative way, sound pressure fluctuations are very small and human ears are very sensitive to detect it. In fact our ear is incredibly sensitive to relative changes in the fluctuations. For an example, a change of pressure from 0.0005 Pa to 0.0007 Pa will be as noticeable as a change from 5 Pa to 7 Pa. Due to the range and sensitivity of human ear; we quantify the pressure fluctuations by another unit known as decibel (dB), where a reference amplitude of pressure fluctuation (p_{ref}) is needed to define this new unit. Here the p_{ref} is 20 microPascals, which is the threshold of hearing.

$$L_p = 10 \log_{10}(p^2/p^2_{ref}) = 20 \log_{10} (p) – 20 \log_{10} (p_{ref}) \quad (dB)$$

$$L_p = 20 \log_{10} (p) + 94 \quad (dB)$$

Now we are able to quantify sound in an absolute manner and the value beyond which the sound starts to cause pain and irritation in ear, that sound is called noise and it is highly undesired. From survey, it is studied that sound pressure level from 60-80 dB are included as

noise; from 80-100 dB sound is called high noise and beyond that value it becomes painful for human ear.

1.2 Sound waves

A vibrating source always makes its surrounding vibrate and the process continues. In our context the same thing happens, where the surrounding is the air and due to compression and rarefaction of the air molecules an acoustic field is created in the atmosphere, which causes pressure fluctuation and it is sensed by our ear.

From this point, we will use the term noise instead of sound as we are interested in only the noise and its cancellation. The propagation of noise from its source to our ear takes place through the atmosphere and thus the properties of atmosphere control the residual noise, which we hear. Noise is the unwanted sound, which we want to reduce at the level of listener. So our basic aim is to minimize the residual noise at our ear and this can be done in two ways as stated above. One is to modify our characteristics of the environment in such a way that the effect of noise seems to be nullified. It is known as the passive noise control method. But there are some constraints, where we are not allowed to manipulate our surroundings. In that case the manipulation part is done in electronic domain; an electronically generated anti-noise, which has equal magnitude and opposite phase as the noise is introduced into the acoustic domain and the noise field is controlled. This is called the active method. Both of the methods are discussed below in great details.

1.3 Passive noise control

Modification of the environment where noise propagates is the basic idea of passive noise control method. It is a traditional method, where we aim to reduce the effect of noise by changing the path of energy flow away from humans. This can be accomplished by a numerous ways. One of the methods is to use some materials, which absorb the sound form

of energy and convert it into heat. Another method is to reduce the volume velocity by attenuating vibration. Some other techniques are to isolate our place of interest by a wall or barrier. But in this way we isolate ourselves from noise, but we have exposed the whole world to a large amount of noise and it is highly undesired. So research works have been done for more than 70 years on the active control of noise as described below.

1.4 Active Noise Control

In this method an electronically synthesized anti-noise, which is equal in magnitude and opposite in phase with noise, is superimposed with it so that destructive interference takes place and the original sound field is exactly cancelled. This is the basic idea of active noise control method. Here we do not modify the environment, rather our work is confined to electronic domain only and this is the main area of interest of our research work.

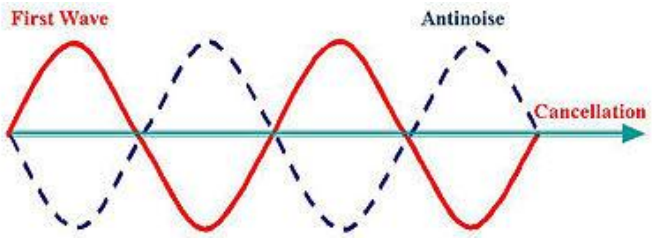

Fig. 1.1 Destructive interference of noise and anti-noise

Chapter 2

Active Noise Control

2.1 Physical Mechanisms

In this chapter we will discuss about the active noise control system in a more detail. The cancelling signal is called the anti-noise, which has equal magnitude and opposite phase with the noise. This is generated electronically and introduced into the acoustic domain so that there is a destructive interference between noise and anti-noise and we get a silent zone. In this method the noise level may be reduced at our desired locations, but according to law of conservation of energy at some other places the noise level is increased considerably. This is known as "Local cancellation". For example if we have designed an active headset, then near our ear the sound field is cancelled actively, but at other locations the sound level is high.

Except cancellation there are also some mechanisms by which noise can be controlled actively. Due to introduction of anti-noise sources the acoustic radiation impedance of the undesired noise source is changed. The noise field may be absorbed or reflected by the anti-noise sources in some other methods.

2.2 Structures of ANC system

An Active Noise Control (ANC) system basically consists of four components.

- **Reference microphone: -** It is a transducer whose job is to convert an acoustic signal to its equivalent electronic form. It samples the noise signal at each instant and the sampled signal is forwarded to the controller.
- **Controller: -** This is heart of the ANC system that actually synthesizes the anti-noise.
- **Loudspeaker: -** Electronically generated anti-noise must be converted to its equivalent acoustic form for noise cancellation.
- **Error microphone: -** the residual noise field is sensed by this and feedback to the controller for its improved performance.

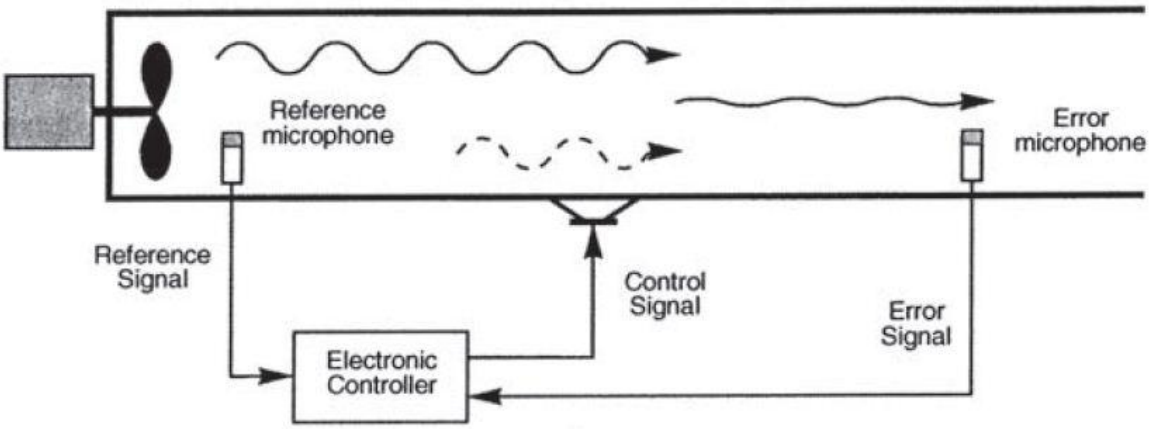

Fig. 2.1 Feedforward active noise control system

Ideally the ANC systems are suited for cancellation of noise below the range of 500 Hz. For higher range of frequency the performance degrades for active method, because of more complex vibration and radiated sound field. Also high sampling rate is required for higher frequency noise. So for noise control we implement active control for low frequencies noise and passive method for high frequencies. From this point onwards our discussion will be focused on the controller only as it is the one which actually generates the anti-noise.

2.3 The electronic controller

The electronic controller is nothing but a digital filter, which takes the noise sampled by the reference microphone and filters it to generate the anti-noise. Noise signal from its source propagates through the acoustic path, also known as the primary path. The job of the controller is to optimize its weights according to transfer function of the primary path so that anti-noise thus synthesized is equal in magnitude and opposite in phase as the noise.

In our research works, we have taken an FIR adaptive filter as our controller. From basics of signal processing we know that Finite Impulse response (FIR) is one where the output is a function of the present as well as some past values of the input only. For simplicity we have taken an FIR filter. The term “adaptive” signifies that the filter weights

are not fixed, rather they are adjustable. Here in our particular problem of noise cancellation we do not know the primary path transfer function and hence we are unable to synthesize our anti-noise directly. So we chose an adaptive filter as controller; randomly initialize the weight values and then update its weights according to an optimization algorithm.

2.4 Digital Filters

Our controller is nothing but an adaptive FIR filter. For simplicity we have assumed our controller to be FIR and adaptive filter is used to adjust its weights according to any variation in external environment. Suitable optimization algorithms are implemented for weight adaptation for efficient noise cancellation.

A digital FIR filter is a combination of delay blocks, multiplication blocks and a summer. The output of the filter is a linear combination of input signal sampled at that instant as well as some past values also. So we can model it by taking the input and passing it through a number of delay blocks, which are also called the taps. Then these are multiplied by respective coefficients also known as weights. Then output is obtained by simply adding all the weighted input signals. The complete architecture of an FIR filter is given below.

In the diagram below the weight coefficients are constant, but in an adaptive filter the values are updated using suitable algorithm to get an optimum value of weight vectors. Now having studied the structures of the controller filter our objective is to find an adaptive algorithm that will update the filter weights.

In our work we have proposed different algorithms for noise cancellation. Broadly those can be categorized into two types. The first type of algorithm tries to minimize the mean of the square of the error. The second one is the stochastic algorithm, which is based on the population based searching technique or the evolutionary computing method.

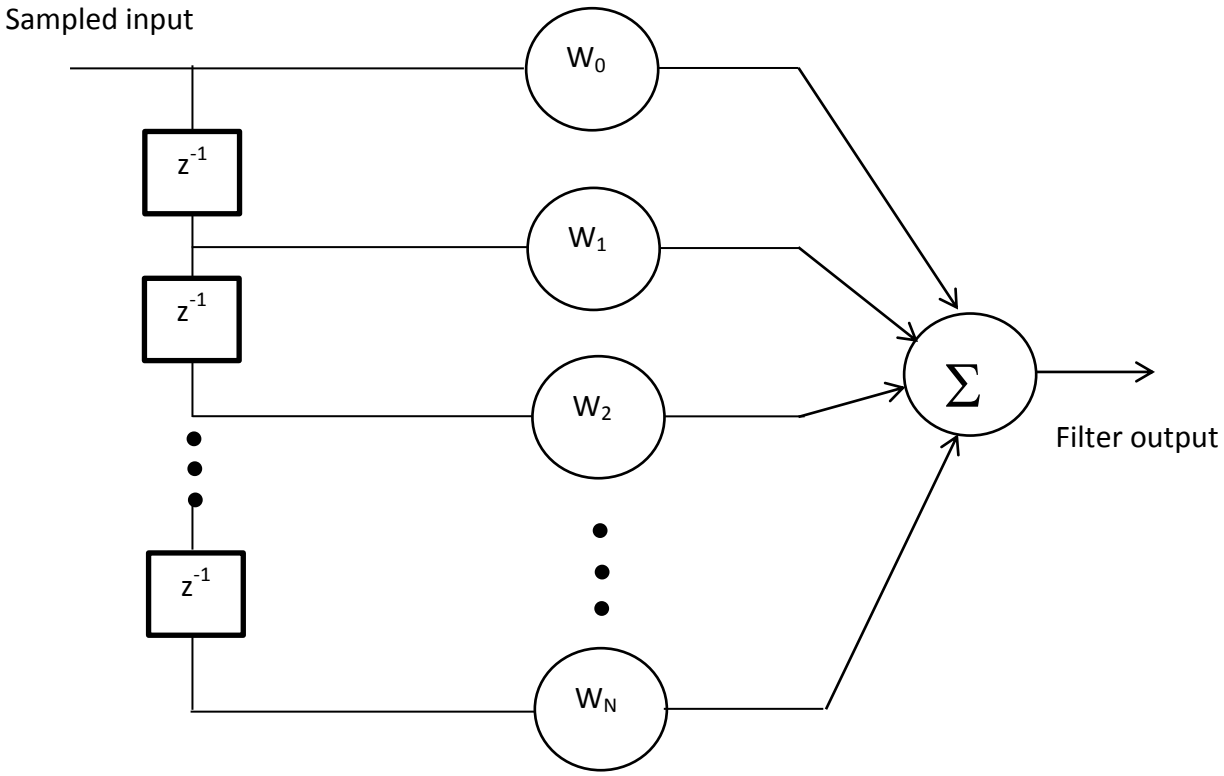

Fig. 2.2 FIR filter architecture

Some nonlinear algorithms have also been proposed for better performance of the controller in presence of nonlinearity in the primary acoustic path. All the algorithms, their performance and implementation complexity are compared in the subsequent chapters.

Chapter 3

Adaptive algorithms in Active Noise Control

3.1 Adaptive control

The working of adaptive algorithm is completed in two steps. The first step is the filtering step where the output is obtained in a usual manner by filtering the input. In the second step, also called the weight adaptation step the filter output is compared to the desired output and the error signal thus obtained is fedback to the controller to update the weights.

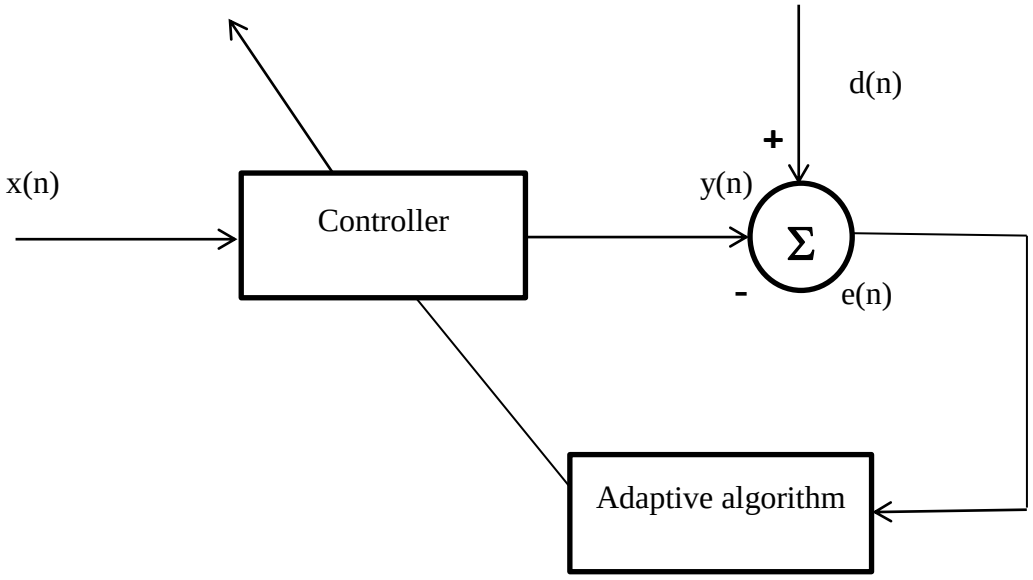

Fig. 3.1 Working principle of adaptive algorithm

3.2 The LMS algorithm: -

From basics of signal processing we know that the wiener filter is an optimum filter that minimizes the mean square error of our model if the autocorrelation properties of the reference signal and the cross correlation between the reference and desired signal is being given. For this a huge amount of computation is needed as a large amount of information about past values of the signal is required. To avoid this complexity another approach is to make the system adaptive. Instead of using the past values of the signal for estimation of correlation functions, instantaneous data are used sequentially so that the algorithm gradually proceed in a direction of minimum mean square error. After some sampling instants our

algorithm will converge to its optimal solution and this time period is defined as the convergence time. For an efficient algorithm the convergence time should be very small.

3.3 Steepest Descent algorithm

For all algorithms to work there is a cost function which is to be optimized. Here in our problem the cost function is the mean of the square of the error. This algorithm suggests that if a filter coefficient is adjusted by a small amount which is proportional to the negative of the gradient of the cost function with respect to the filter coefficient, then it approaches towards its global solution. Simultaneous adjustment of all the filter coefficients can be represented in vector form. The governing equation is written below

$$\mathbf{w}\ (\text{new}) = \mathbf{w}\ (\text{old}) - \mu\ \frac{\partial \boldsymbol{J}}{\partial \boldsymbol{w}}\ (\text{old})$$

where μ is the convergence factor or the step size

J is the cost function, which is defined as the mean of the square of the error

$$\mathbf{J} = E\ [\ e^2\ (n)\ ]$$

where the residual error is given by

$$e(n) = d(n) - y(n) = d(n) - \mathbf{x}^T(n)\ \mathbf{w}$$

$$\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{w}} = 2\ E[\ \mathbf{x}(n)\ \mathbf{x}^T(n)\ \mathbf{w} - \mathbf{x}(n)d(n)\ ]$$

$$\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{w}} = -\ 2E[\ \mathbf{x}(n)e(n)\ ]$$

The expectation value of the product of the error signal with the reference signal vector is computed by time averaging a large segment of data, which increases the computational

complexity of the system. So the expectation value is simply approximated by the instantaneous value and the resulting algorithm is known as the LMS algorithm.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\,\mu\,\mathbf{x}(n)\,e(n)$$

This is a very important algorithm widely used in a variety of applications. For our control system design we use the above equation directly or with some modifications as will be discussed in the following sections.

3.4 Filtered Reference LMS algorithm

In our noise cancellation problem the anti-noise introduced from the loudspeaker into the acoustic field cancels the noise and residual error signal is sampled by the error microphone. Now the small acoustic path from the loudspeaker to the error microphone, also known as the secondary path has some transfer function and it needs to be taken into account during implementation. The new error signal sampled at the error microphone is given by,

$$e(n) = d(n) - \mathbf{w}^T\,\mathbf{r}(n)$$

where $\mathbf{r}(n)$ is the input vector filtered by the secondary path (g).

$$\mathbf{r}(n) = \mathbf{x}^T(n)\,\mathbf{g}(n)$$

So filtered-reference LMS or filtered-x LMS (FxLMS) algorithm requires the identification of the secondary acoustic path for updating its weights. In practice the filtered reference signal is generated by passing the reference signal by the estimated version of the secondary path.

$$\mathbf{w}(n+1) = \mathbf{w}(n) + 2\,\mu\,\mathbf{r}(n)\,e(n)$$

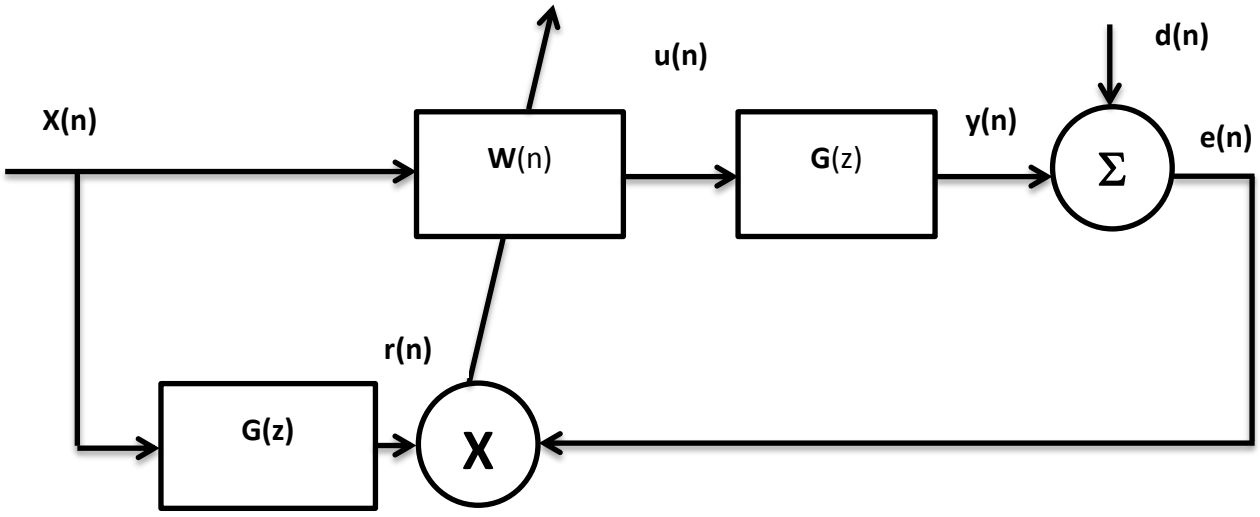

Fig. 3.2 Block diagram of the practical implementation of the FxLMS algorithm

3.5 Nonlinear systems

Till now our controller is assumed to be a linear one, because the primary path through which noise propagates is itself is linear. But in practical cases, the atmosphere or in technical language the primary path is never linear. Due to humidity or wind effect or anything the primary path is nonlinear. By a nonlinear system, we understand that it does not obey the principle of superposition. Nonlinearity can be divided into two types. A system may be weakly nonlinear, where the system behaves as linear upto some extent and beyond that value it exhibits nonlinearity. But a system is said to be a chaotic if it is highly nonlinear. So a proper anti-noise can be synthesized by a nonlinear controller if the primary path itself is nonlinear. In a much generalized sense if the output contains any higher order terms of the input, then that system is said to be nonlinear. Sometimes the acoustic path may exhibit trigonometric nonlinearity. Keeping these two types of nonlinearities in mind we propose two algorithms in the following sections for different situations.

3.6 The Volterra FxLMS algorithm

For discrete time system the volterra series can be described as

$y(n) = h_0 + \sum_{k1=0}^{\infty} h1(k1)x(n-k1) + \sum_{k1=0}^{\infty}\sum_{k2=0}^{\infty} h2(k1,k2)x(n-k1)x(n-k2) + \ldots$

where the terms $h_1(k_1)$ and $h_2(k_2)$ are called the first and second order volterrs kernels and in principle the series can be extended to any order.

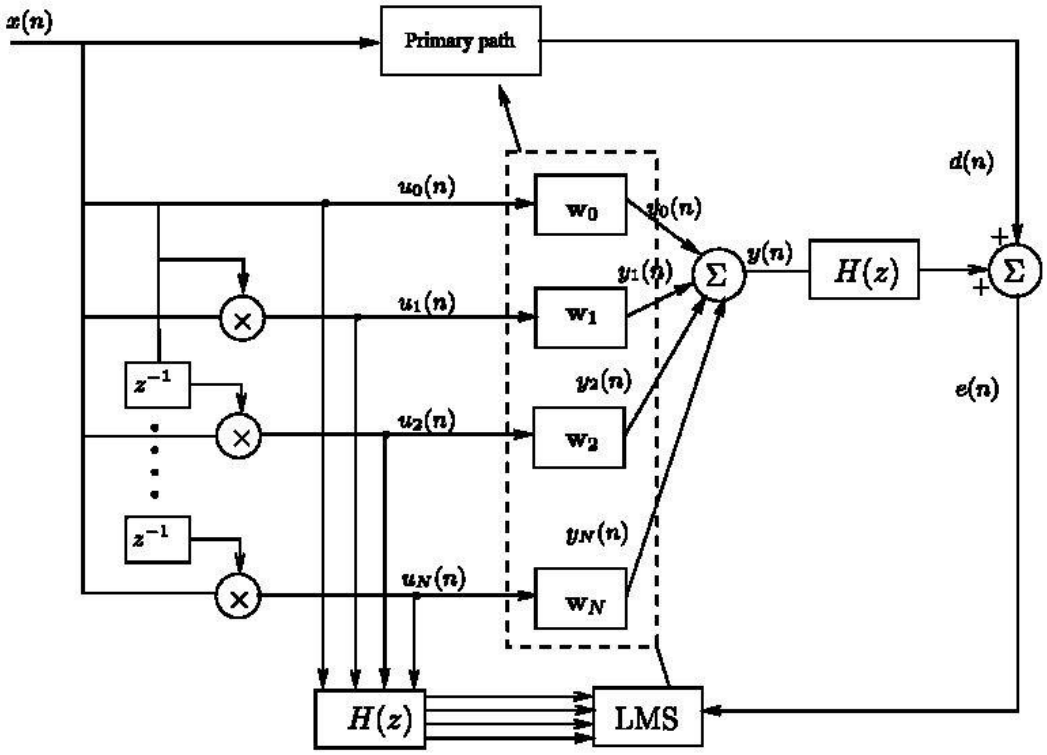

Fig. 3.3 Block diagram of the second order VFxLMS algorithm

For h_0 and second and higher kernels to be zero the series simply becomes the linear convolution. For a memoryless system for k_1, k_2 etc. > 0 the kernels become zero and the series reduces to a power series. In our work we have considered only upto second order kernels. In the quadratic part we have not considered the crossterms. i.e.- k_1 and k_2 are assumed to be same.

So our series becomes

$$y(n) = h_0 + \sum_{k=0}^{N} h1(k)x(n-k) + h_2(k)x^2(n-k)$$

where N = order of the FIR filter

The input vector without considering the corssterms is written as

$\mathbf{x} = [\ x(n)\ x(n-1)\ x(n-2)\ \dots\ x(n-N+1)\ \ x^2(n)\ x^2(n-1)\ x^2(n-2)\ \dots\ x^2(n-N+1\ ]^T$

If we include the crossterms then our input vector will be as below

$\mathbf{x} = [\ x(n)\ x(n-1)\ x(n-2)\ \dots\ x(n-N+1)$

$x^2(n)\ x(n)x(n-1)\ x(n)x(n-2)\ \dots\ x(n)x(n-N+1)$

$x^2(n-1)\ x(n-1)x(n-2)\ x(n-1)x(n-3)\ \dots\ x(n-1)x(n-N+1)$

$\dots$

$x^2(n-N+1)\ x(n-N+1)x(n-N)\ x^2(n-N+1)\ ]^T$

After getting the input vector we initialize the corresponding weight vector of same size as that the input vector. Now the input vector is expanded nonlinearly as a Volterra series. So the rest thing is the weight adaptation, which can be completed by using the mathematical equation of FxLMS. This is called the Volterra FxLMS algorithm.

Introducing nonlinearity to the primary path, the performance of general FxLMS is compared with the proposed second order VFxLMS.

3.7 Trigonometric FLANN filter

FLANN: - Functional Link Artificial Neural Network

Neural network is a field, which is inspired by the working principle of the human brain. Our brain is a parallel processor which is highly nonlinear, distributed and yet very efficient in performance. So to model human brain we have proposed an artificial neural network, where the input is allowed to expand nonlinearly. Here we are only interested in trigonometric expansions. Here our filter includes the point-wise trigonometric expansion at the same time instant and also the products of samples of different time shifts (cross terms).

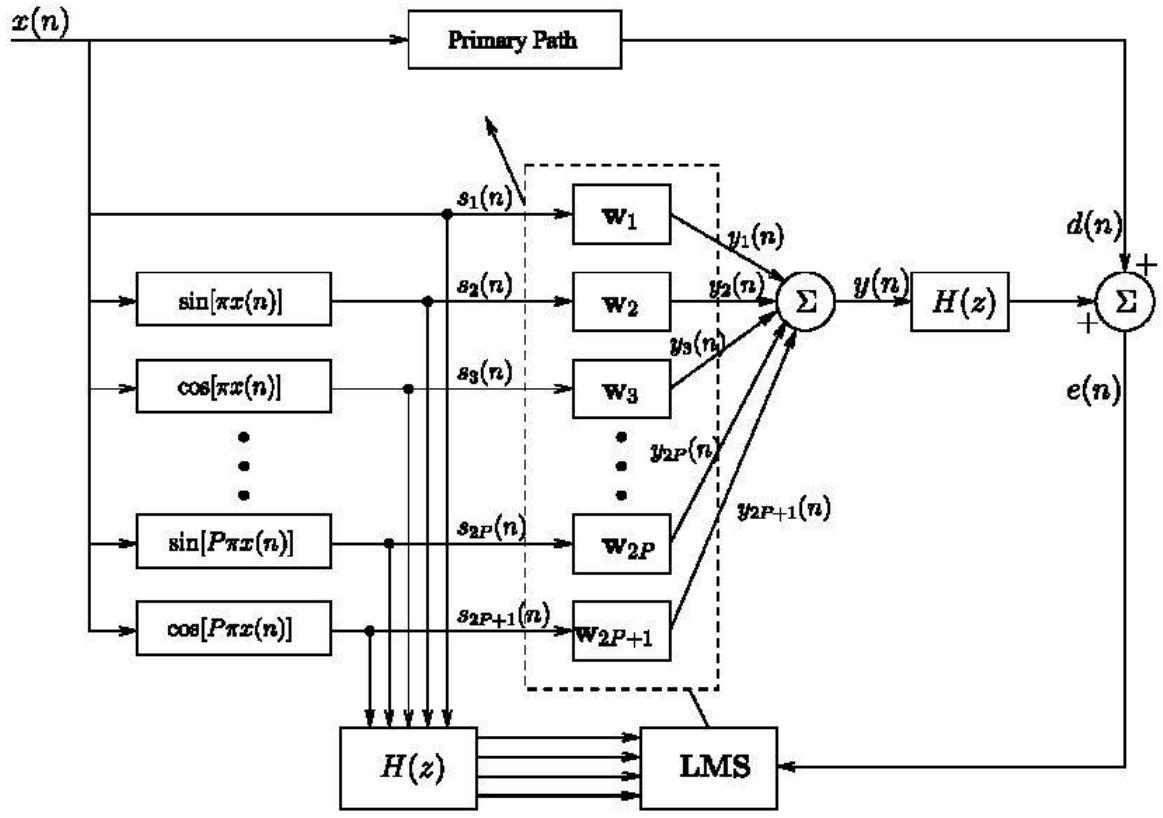

Fig. 3.4 Block diagram of the trigonometric FLANN filter

The above block diagram does not include any crossterms. As we can see the input vector has a linear part as well as nonlinear expansions in terms of sine and cosine functions.

A FLANN filter using trigonometric nonlinear expansion of order ‘P’ without crossterms is first taken into consideration. For this we can write the input vector as below

$$\mathbf{x} = [\ x(n)\ x(n-1)\ x(n-2)\ \ldots\ x(n-N+1)$$

$$\sin[\Pi x(n)]\ \sin[\Pi x(n-1)]\ \sin[\Pi x(n-2)]\ \ldots\ \sin[\Pi x(n-N+1)]$$

$$\cos[\Pi x(n)]\ \cos[\Pi x(n-1)]\ \cos[\Pi x(n-2)]\ \ldots\ \cos[\Pi x(n-N+1)]$$

$$\sin[2\Pi x(n)]\ \sin[2\Pi x(n-1)]\ \sin[2\Pi x(n-2)]\ \ldots\ \sin[2\Pi x(n-N+1)]$$

$$\cos[2\Pi x(n)]\ \cos[2\Pi x(n-1)]\ \cos[2\Pi x(n-2)]\ \ldots\ \cos[2\Pi x(n-N+1)]$$

$$\ldots$$

$$\sin[P\Pi x(n)]\ \sin[P\Pi x(n-1)]\ \sin[P\Pi x(n-2)]\ \ldots\ \sin[P\Pi x(n-N+1)]$$

$$\cos[P\Pi x(n)]\ \cos[P\Pi x(n-1)]\ \cos[P\Pi x(n-2)]\ \ldots\ \cos[P\Pi x(n-N+1)]\ ]^{T}$$

if crossterms are taken into account, then it will be very difficult to all the terms here. For a simple demonstration we have assumed the order of the filter (P) to be 1. Now the input vector can be written as

$$\mathbf{x} = [\ x(n)\ x(n-1)\ x(n-2)\ \ldots\ x(n-N+1)$$

$$\sin[\Pi x(n)]\ \sin[\Pi x(n-1)]\ \sin[\Pi x(n-2)]\ \ldots\ \sin[\Pi x(n-N+1)]$$

$$\cos[\Pi x(n)]\ \cos[\Pi x(n-1)]\ \cos[\Pi x(n-2)]\ \ldots\ \cos[\Pi x(n-N+1)]$$

$$x(n-1)\sin[\Pi x(n)]\ x(n-2)\sin[\Pi x(n-1)]\ x(n-3)\sin[\Pi x(n-2)]\ \ldots\ x(n-N+1)\sin[\Pi x(n-N+2)]$$

$$x(n-1)\cos[\Pi x(n)]\ x(n-2)\ \cos[\Pi x(n-1)]\ x(n-3)\cos[\Pi x(n-2)]\ \ldots\ x(n-N+1)\cos[\Pi x(n-N+2)]$$

$$x(n-N+1)\sin[\Pi x(n)]\ \ x(n-N+1)\cos[\Pi x(n)]\ ]^{T}$$

After expanding our input vector according to the above methods we randomly initialize the weight vector, whose size is same as the input vector. If any nonlinearity is present in the system, then they are exactly cancelled by the output of the nonlinearly expanded input vector. Then the weight vector is updated using the equation of the FxLMS algorithm.

A trigonometric expansion based FLANN filter is designed and under same conditions of primary path nonlinearity the performance of the general FxLMS and the proposed method of FLANN filter is compared. Again the performance comparison of a simple FLANN filter without using crossterms and including suitable crossterms is done.

Chapter 4

Evolutionary computation

Till now all the algorithms discussed are modified form of LMS algorithm, which are based on steepest descent procedure. This means the weights are updated in such a way that the small change in weight value is directly proportional to the negative gradient of the cost function with respect to the weight vector. In an error surface the weight is updated in a direction of maximum decrement of cost function. But in this section we will introduce a completely different type of algorithm, which is known as evolutionary computation algorithm.

This type of algorithms is inspired by the natural process of biological evolution. It is natural process which is based on the principle of "survival of the fittest". It means the fittest individuals only survive and appear in the next generation. Those who cannot cope up with the external environment do not survive; they simply die and do not appear in the next generation. In this way the individuals who can adapt themselves according to a variable environment are considered as the fittest individuals. The same analogy will be used for adaptive algorithms.

The evolutionary computation methods are also known as the population based searching techniques as the optimum solution of a problem is found out by a population of potential solutions, which follow the principle of biological evolution. As the process of evolution proceeds in a direction to get fittest individuals, similarly our adaptive algorithm will update its weights in a direction of cost function optimization. Here the objective is to minimize the mean square error.

This method is also called the stochastic algorithm, because all the steps and processes involved here are random and based on a probability factor as will be seen in the following sections.

4.1 Genetic algorithm

Genetic algorithm (GA) is completely inspired by the process of natural biological evolution, which was first proposed by John Holland in the early seventies. First a population of weight vector is initialized randomly, which represent the potential solutions and then the existing parents in the current population are allowed to mate with each other. Then according to laws of natural genetics and various genetic operators like crossover and mutation new individuals are produced. Then both the parents as well as the children are evaluated against an unknown environment, i.e. - the mean square error of all the individuals in the population is calculated and the best individuals are selected. If the population size is assumed to be P then after reproduction stage the total size becomes 2P, because of additional P numbers of children. During evaluation and selection, the best P individuals survive in the next generation and others are simply discarded. This process continues and after some generations it reaches the global solution. All the steps of genetic algorithms are described in a sequential manner.

4.1.1 Population Initialization: -

First P numbers of controller weights are randomly initialized. Each individual is a weight vector and for GA first the weight values must be coded into a fixed length of binary strings. The length of the string depends on domain of parameters and the precision of computation. For an example if the domain of the solution is [0,1] and the precision of computation is taken upto four places after the decimal point, then the domain [0,1] should be divided into 10000 equal sizes.

$$8192 = 2^{13} < 10000 < 2^{14} = 16384$$

For our problem the length of the string is taken 14. So during population initialization we take P numbers of 14 bit strings, where the bit values are randomly assigned.

The decoding of the binary string <b_{13} b_{12} … b_1 b_0> to corresponding real numbers is shown below.

$$x' = \sum_{i=0}^{13} b_i \, 2^i$$

The corresponding real number is given by,

$$x = x' / (2^{14} - 1)$$

4.1.2 Evaluation

All the individuals are nothing but the potential solutions of the weight vectors of our control system. In this step, all the individuals are evaluated by passing the input signal through all the controllers one by one and then calculating the error. From the error we get the fitness function according to the relation, $f_j = 1 / e_j$. The individual having highest fitness value is called the fittest individual and has a better probability of getting selected for next generation.

4.1.3 Crossover

The individuals of the existing population are now called parents, who are allowed to reproduce among themselves to produce children. The crossover can be of various types. It can be single-point, two-point, uniform crossover etc. The single-point crossover operation can be realized by randomly choosing two weight vectors as parents, specifying a crossover point randomly and then interchanging the bit values of the chromosomes beyond the crossover point. In two point operation the chromosome bits are interchanged between the two specified points. In uniform operation the points at which the two parents will interchange their bits will be determined randomly. In this operation the two parents will reproduce two children. So a population of size P will be now 2P.

4.1.4 Mutation

Mutation is a very peculiar phenomenon in the evolution process, which accounts for variation in the process. Here the process of mutation can be realized by taking the children chromosomes, randomly choosing the mutation points according to the probability of mutation, which determines the number of chromosomes bits undergoing mutation and then reversing the bit values at the mutation points. In the problem of noise cancellation, this is a very important step, which helps the system converge to its global solution if it has struck in a local solution.

4.1.5 Selection

Now all the individuals including the parents as well as the children are evaluated and their fitness values are calculated. Accordingly best P individuals survive in the next generation and rest are discarded. This process continues until we get a global solution.

Though due to a large population size the controller has become very huge, its computational complexity has been reduced because we do not need the secondary path to be identified. Also the performance is comparatively good as the system always converges to its global solution as a large number of individuals are involved in the searching process.

4.2 Particle Swarm Optimization (PSO)

It is an evolutionary computation technique, which is inspired by the social behaviour of the swarms. In an analytical way, we compare the swarms and their positions as our controller weight vectors. The potential solutions, also called as particles are randomly initialized and in the searching space they fly towards the global solution with a velocity which is dynamically adjusted according to the flying experience of their own as well as other particles in the space. The position and velocity of the i^{th} particle at t^{th} iteration are given by $x_i(t)$ and $v_i(t)$. The best previous position for i^{th} particle is recorded and its fitnees value is denoted as $pbest_i$ and the position as $xpbest_i$. The index of the best $pbest_i$ is recorded and denoted as *gbest* (global best) and its location as *xgbest*. Now the velocity and the position of the swarms can be adjusted according to their personal best as well as their global best as given below.

$$v_i(t) = \phi v_i(t\text{-}1) + r_1 c_1(xpbest_i - x_i(t)) + r_2 c_2(xgbest - x_i(t))$$

$$x_i(t) = x_i(t - 1) + v_i(t)$$

where ϕ is the inertia constant

c_1, c_2 are the acceleration constants

r_1 and r_2 are random numbers generated between [0,1]

In each time step the particles are evaluated and the corresponding error values and hence the fitness values are recorded. If at any time the current fitness value is greater than $pbest_i$ value then it gets stored in $pbest_i$. If the personal best value is greater than the global value, then the *gbest* value is replaced by it. in this way the personal and global best values and the particle locations are updated. In the next iterations the velocity and the position of each particle are adjusted according to the above equations.

4.3 Differential Evolution algorithm

Differential Evolution (DE) is an algorithm that is much similar in structure to GA. As the name suggests the difference vector of two random vectors in the search space are used to update the weight vectors for next generation. Randomly initialized population of weight vectors are allowed to crossover and mutation. Then the best individuals get selected for which we get a better fitness value. The summary of all the steps are described below.

4.3.1 Population initialization

In the search space, the weight vectors of dimension N are randomly initialized uniformly within the lower (x^l) and upper boundary (x^u).

$$\mathbf{X} = [\, \mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_P} \,]^T \text{ is called the target vector}$$

where $\mathbf{x_i} = [\, x_{i,1}, x_{i,2}, \ldots, x_{i,N} \,]$ is the i^{th} weight vector of order N ,where i = 0,1,…,P

$$x_{i,j} = x^l + \text{rand}(0,1)\,(x^u - x^l)$$

where each coefficients of the individual vectors are initialized according to the above equation and the role of rand(0,1) generates random numbers within [0,1] uniformly.

4.3.2 Mutation operation

The concept of mutation is same as it was in GA. This operation does not allow the system to converge to its local minima. This process is completed by taking differential vector of random weight vectors for obtaining the mutant vector and hence satisfies its name. At a generation G for each target vector $\mathbf{x_{i,G}}$ a mutant vector $\mathbf{v_{i,G}}$ is calculated.

Where $\mathbf{v_{i,G}} = [\, v_{1,i,G}, v_{2,i,G}, \ldots, v_{N,i,G} \,]$ and it can be generated by using any one of the following strategies.

"DE/rand/1": $\mathbf{v_{i,G} = x_{r1,G} + F\,(x_{r2,G} - x_{r3,G})}$

"DE/best/1": $\mathbf{v_{i,G} = x_{best,G} + F\,(x_{r1,G} - x_{r2,G})}$

"DE/current to best/1": $\mathbf{v_{i,G} = x_{i,G} + F\,(x_{best,G} - x_{i,G}) + F\,(x_{r1,G} - x_{r2,G})}$

"DE/best/2": $\mathbf{v_{i,G} = x_{best,G} + F\,(x_{r1,G} - x_{r2,G}) + F\,(x_{r3,G} - x_{r4,G})}$

"DE/rand/2": $\mathbf{v_{i,G} = x_{r1,G} + F\,(x_{r2,G} - x_{r3,G}) + F\,(x_{r4,G} - x_{r5,G})}$

Where r_1, r_2, r_3, r_4, r_5 are random and mutually different integers generated in the range [1, P]

F is the mutation constant and $\mathbf{x_{best,G}}$ is the individual vector with best fitness value in the generation G.

4.3.3 Crossover operation

After the mutation phase, crossover operation is done between the target vector ($\mathbf{x_{i,G}}$) and the mutant vector ($\mathbf{v_{i,G}}$) to produce a trial vector.

$\mathbf{u_{i,G}} = [\, u_{1i,G}\, ,\, u_{2i,G}\, ,\, \dots\, ,\, u_{Ni,G}]$ is the trial vector which is obtained by the following formula.

$$u_{j,I,G} = \begin{cases} vji, G\ , if\,(rand(0,1) < CR)\ or\ (j = jrand) \\ xji, G \qquad\quad , \qquad\qquad\qquad otherwise \end{cases}$$

where CR is the crossover constant. jrand is random integer in the range [1,P] to ensure that the trial vector will be different from the target vector at least by one parameter.

Now the upper and lower bounds of the trial vector are checked. If the vectors are out of the boundary values, then we randomly scale it within the boundary values.

4.3.4 Selection operation

Now both the target as well as trial vectors have been generated. These are taken as the controllers one by one sequentially; input is applied to the system and after getting the output the error is calculated for all the target and trial vectors and also the fitness values. Then fitness of each target vector is compared to its trial vector. If the trial vector has more fitness value than the target vector then the trial vector is selected for the next generation. This process continues until a global solution is achieved.

The parameters F and CR control the speed of the convergence. If these are not chosen properly there is a chance of divergence of the algorithm. For better performance the DE algorithm may be modified so that the values of F and CR are not constant, rather these are adaptively changing from one generation to the next.

Implementation of DE algorithm in noise cancellation system shows better performance and also helps the system converge to its global solution. It also does not require the identification of the secondary path, which makes the system very simple.

Chapter 5

Results and Conclusion

5.1 Simulation 1: Performance of FxLMS algorithm

In this experiment, the noise signal (x (n)) is represented by random numbers generated in the range of [-0.5, 0.5] uniformly. The primary path is given by P (z) = z^{-3}-0.3z^{-4}+0.2z^{-5} and the secondary path by S (z) = z^{-2}+0.5z^{-3}. u (n) = x (n) * p (n) , where p (n) is called the primary path impulse response function. Now the primary noise sensed by the error microphone is given by, d (n) = u (n-2) + d_1u^2 (n-2) + d_2u^3 (n-1). The factors d_1 and d_2 are the measure of the strength of the primary path nonlinearity. For simulation, the model is assumed to be an FIR filter of 15 taps. For weight adaptation the using the FxLMS algorithm value ofμ, the learning rate parameter is taken to be 0.1. Now the training period consists of 2000 samples and the complete procedure is repeated for 50 times. For the above experimental setup we have conducted 3 experiments. First the primary path is assumed to be completely linear and this is done by taking the values of d_1 and d_2 to be zero. In the second experiment the nonlinearity introduced in the primary path is very low. i.e. - d_1 = 0.08 and d_2 = 0.04. In the third experiment the primary path nonlinearity is very high. i.e. - d_1 = 0.8 and d_2 = 0.4.

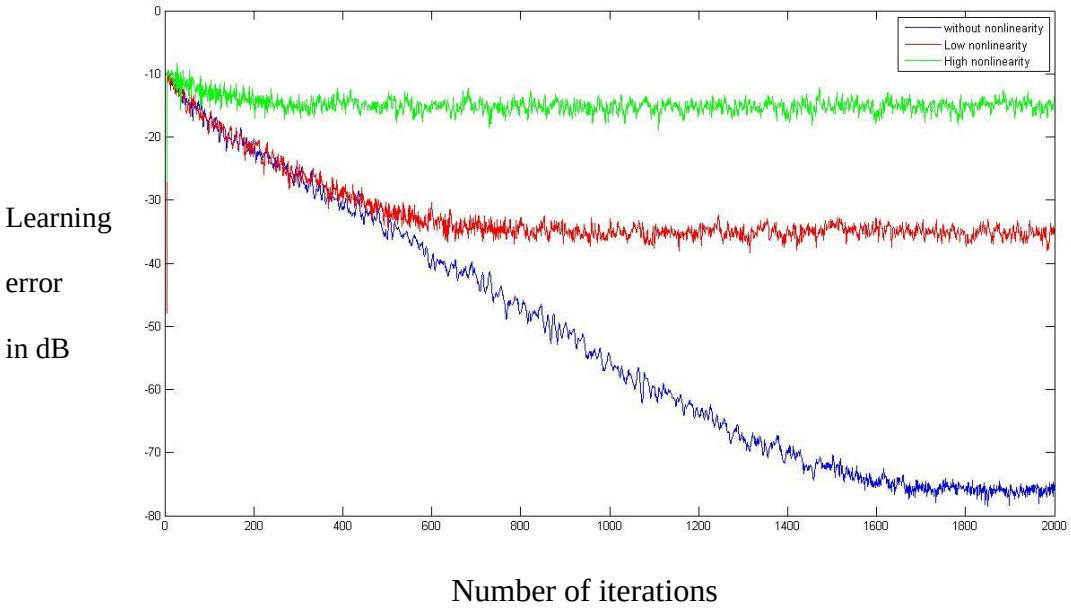

Fig. 5.1 Performance of FxLMS algorithm in presence of nonlinearity

5.2 Simulation 2: Performance of Volterra FxLMS (VFxLMS) algorithm

In the previous experiment, we observed that the performance of the filter using FxLMS algorithm is very good in the absence of primary path nonlinearity. But as the nonlinearity in the path increase the performance of the algorithm degrades. So the performance of the proposed VFxLMS algorithm is simulated in this experiment. The VFxLMS algorithm does not include any crossterms, which makes the implementation of the algorithm simpler. The same experimental setup is used here. As previously done, the nonlinearity is increased and the performance in terms of noise cancellation is recorded.

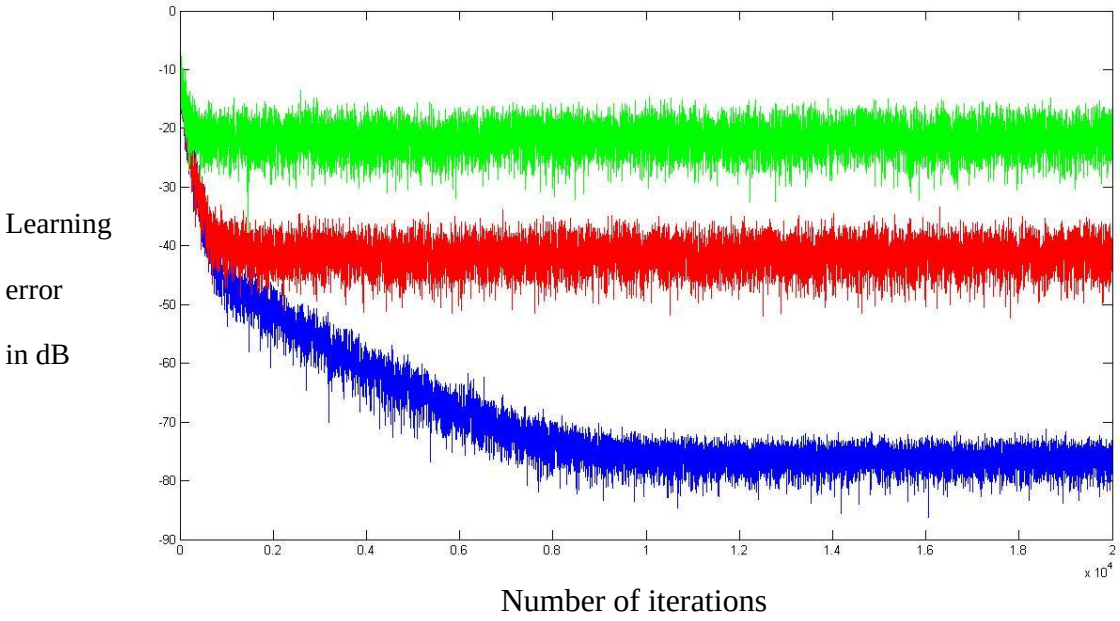

Fig. 5.2 Performance of VFxLMS algorithm in presence of nonlinearity

We observe that the performance of the VFxLMS also degrades in presence of nonlinearity. The reason of the performance degradation may be the exclusion of crossterms of the input signal.

5.3 Simulation 3: Performance of FLANN filter using trigonometric nonlinear expansion

The FLANN filter is designed by trigonometric nonlinear expansion of the input signal. We have taken upto 4^{th} harmonics of sine as well as cosine terms for nonlinear expansions. The algorithm performance is observed by increasing the primary path nonlinearity. As we go on increasing the nonlinearity the performance degrades, but is better than proposed FxLMS.

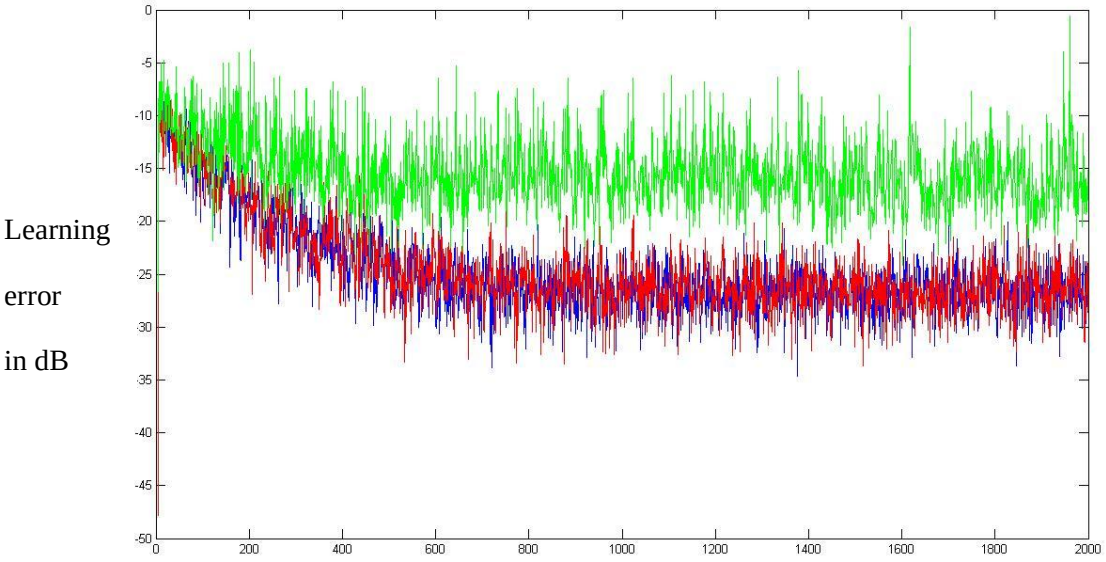

Fig. 5.3 Performance of FLANN filter in different nonlinearities

In presence of low level of nonlinearities the performance of FLANN filter using trigonometric expansions the system performance is not disturbed. But presence of high nonlinearity degrades the efficiency in terms of noise cancellation.

5.4 Simulation 4: Performance of PSO algorithm

In this experiment the primary path transfer function is taken to be P (z) =0.23+ .96z^{-1} + 0.23z^{-2} and the secondary path as S (z) = 0.02 + 0.09z^{-1} +0.02z^{-2}. Here the numbers of particles initialized is 10. The inertia constant (ϕ) is taken to be 0.9. The acceleration constants c_1 and c_2 are taken to be 0.4 and 0.6 respectively. First the weight vectors are evaluated by taking the error. For a better result the mean error of 50 input samples are taken then fitness value of each weight vector is calculated and then accordingly the weights are updated one by one. This process is repeated for 100 times and performance of the algorithm is observed. The same experiment is conducted for low as well as high level of nonlinearity in the primary path.

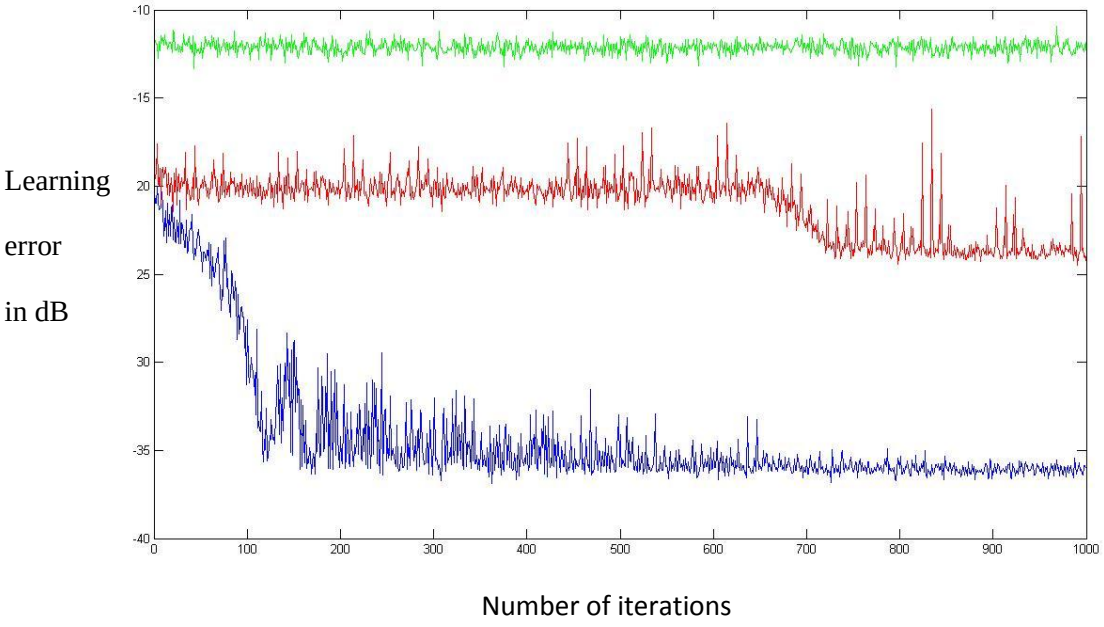

Fig. 5.4 Performance of PSO algorithm in presence of nonlinearity

5.5 Simulation 5: Performance of DE algorithm

In this experiment the same primary path and secondary path are taken. Both the mutation constant (F) and the crossover ratio (CR) are taken to be 0.8. 10 numbers of weight vectors are randomly initialized and are evaluated according to their fitness values, which is obtained from the corresponding error. The error is calculated by taking the maximum of the error of 50 input samples. The process is repeated for 15 generations. Then conducting this experiment for 10 times we take the mean value of the learning error curve. This complete procedure is done, first by considering the primary path to be linear and then adding low and then high level of nonlinearity.

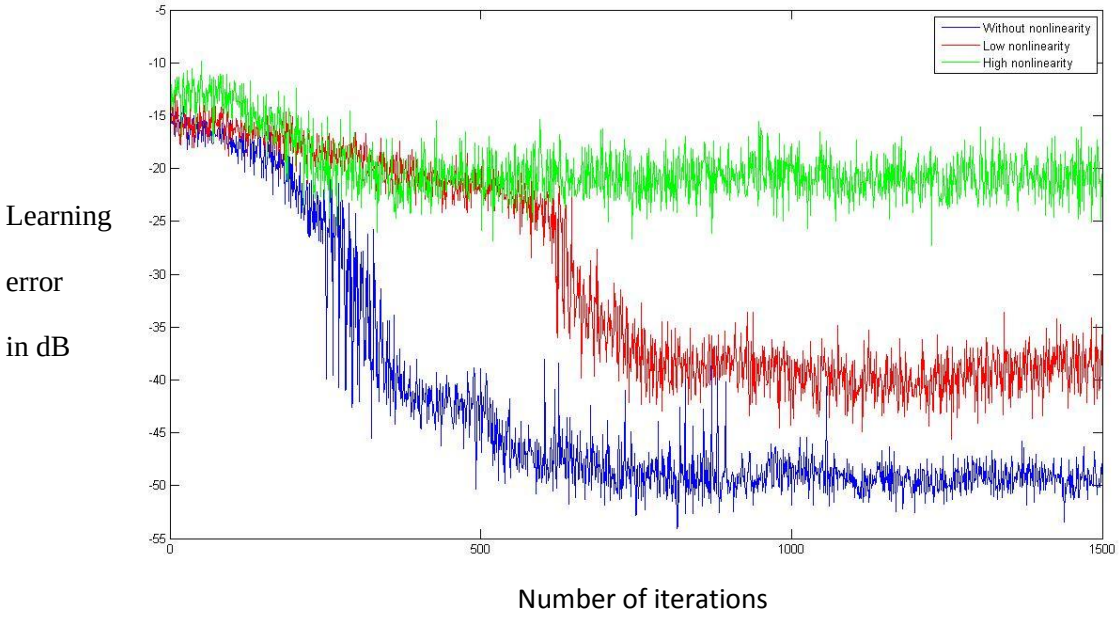

Fig. 5.5 Performance of DE algorithm in presence of nonlinearity

5.6 Simulation 6: Performance comparison of FxLMS, VFxLMS, FLANN

In this experiment, the three algorithms are compared by taking different nonlinearities. The value of learning arte parameter is taken to be 0.01.

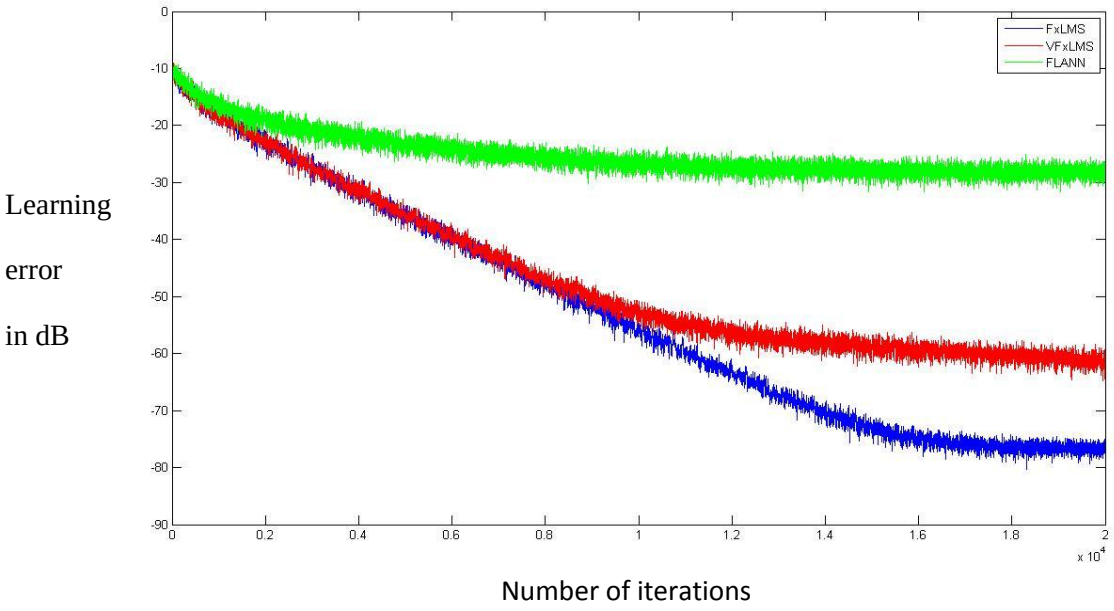

Fig. 5.6.1 Performance of FxLMS, VFxLMS, FLANN without of nonlinearities

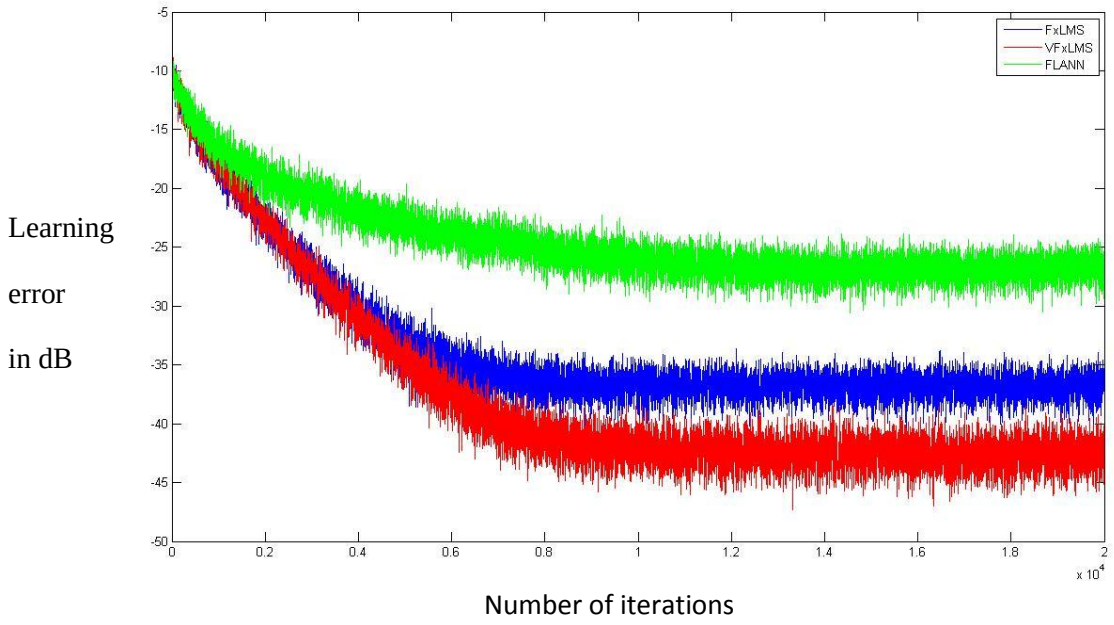

Fig. 5.6.2 Performance of FxLMS, VFxLMS, FLANN with low level of nonlinearities

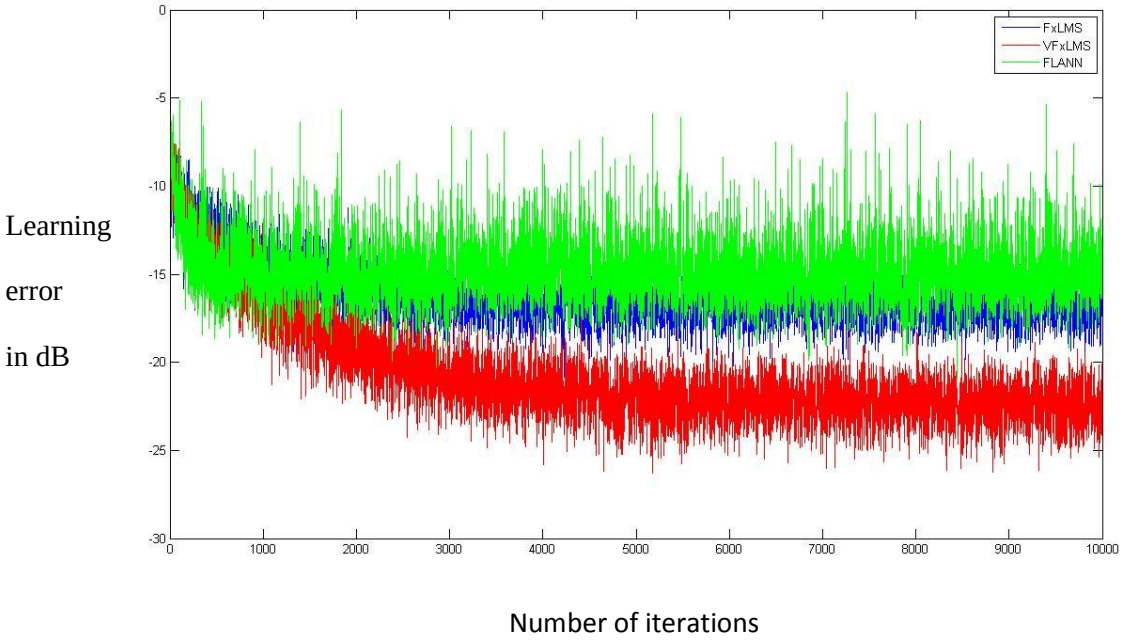

Fig.5.6.3 Performance of FxLMS, VFxLMS, FLANN with high level of nonlinearities

The performance of VFxLMS using second order terms is the best in presence of nonlinearity. The poor performance of the proposed FLANN filter may be due to the assumed second and third order nonlinearity in the primary path. The trigonometric expansions in the FLANN filter are not able to cancel the higher order terms of nonlinearity.

5.7 Simulation 7: Performance comparison of PSO & DE algorithm

After simulating and studying the behaviour of PSO and DE algorithm, we now compare both of them. The same constant values are used, which were in the previous experiments. Three simulations are being carried out. First by considering the primary path to be linear and then adding nonlinearity to it.

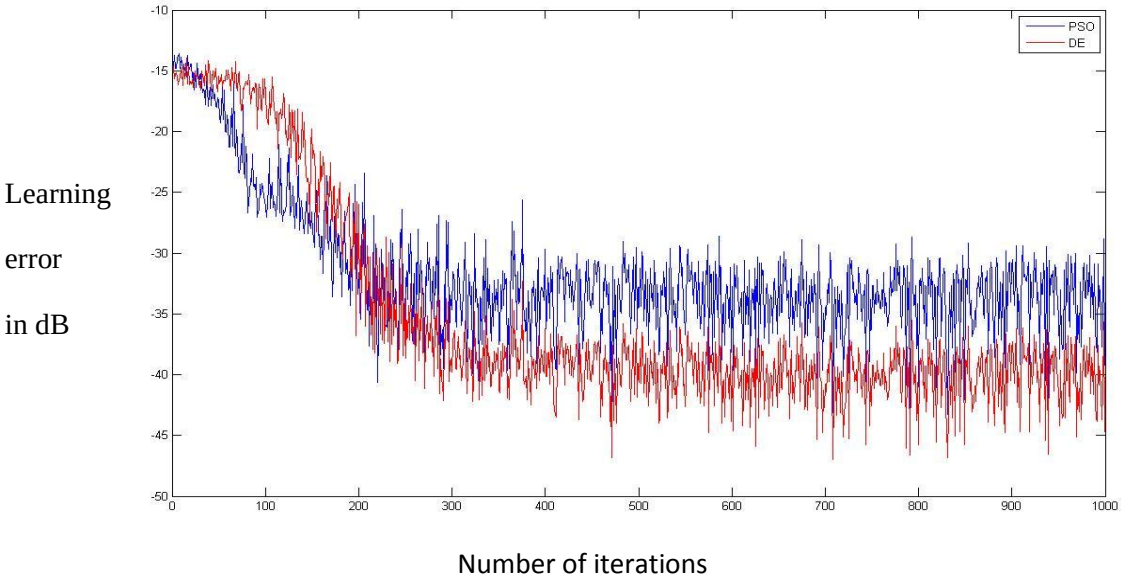

Fig.5.7.1 Performance comparison of PSO & DE without primary path nonlinearity

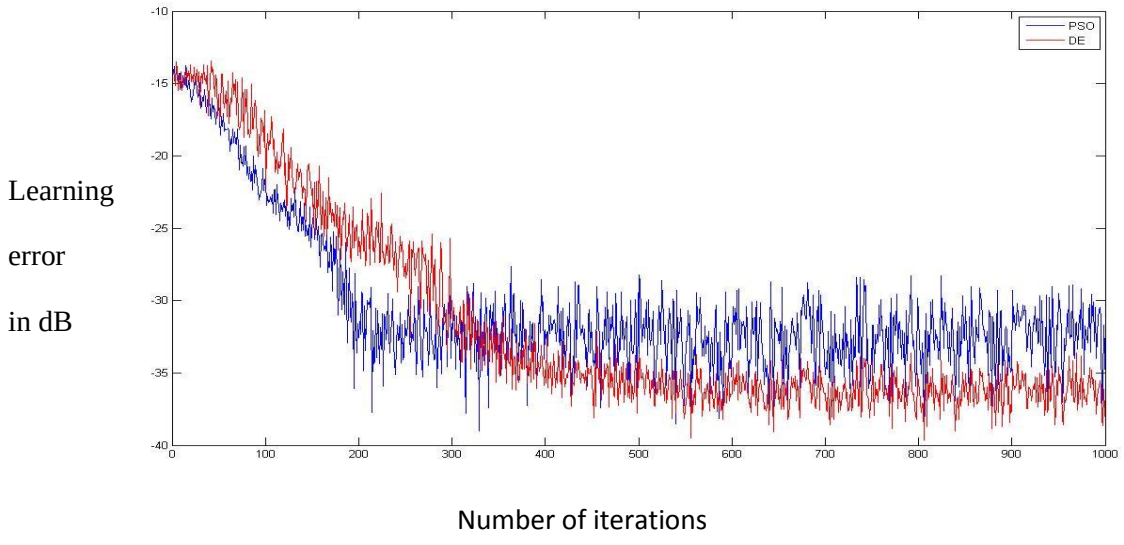

Fig.5.7.2 Performance comparison of PSO & DE with low level of primary path nonlinearity

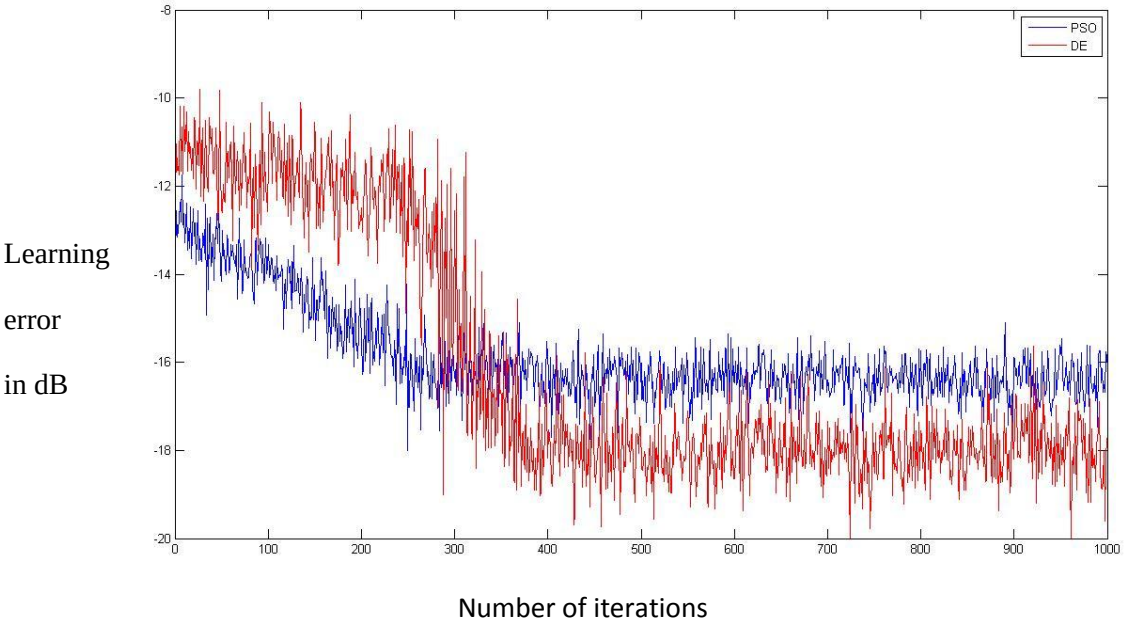

Fig.5.7.3 Performance comparison of PSO & DE with high level of primary path nonlinearity

As we can observe the performance of DE is better than PSO in all the cases of nonlinearities. But during implementation DE requires target vectors, mutant vectors and the trial vectors, whereas in PSO it requires only calculation of position vectors and velocity vectors only. So in terms of computation, PSO is simpler.

Performances in terms of noise cancellation using different algorithms under different values of nonlinearities are shown in the table below. All the numerical values are presented in dB.

Table 5.1 Comparison of different algorithms

	FxLMS	VFxLMS	FLANN	PSO	DE
Without NL	75.3782	73.3782	27.9849	36.2458	48.3729
Low NL	34.4426	42.0346	27.2383	23.4384	41.7392
High NL	15.1071	22.4938	17.8329	12.3874	18.2712

5.8 Conclusion

From all the experiments we see that the performance of all the algorithms degrade in presence of nonlinearities. To compensate the nonlinearities we have to implement nonlinear algorithms and the noise cancellation efficiency of different algorithms are also compared. The VFxLMS and FLANN filter based on trigonometric expansion happen to be good filters those can be used in presence of primary path nonlinearities. But these above algorithms need online secondary path identification to get the filtered reference signal. In some cases there is probability of convergence of the system to a local solution.

To avoid the above two difficulties we propose evolutionary computing methods, where online secondary path identification is not required and as it is a population based searching technique the system always converges to its global solution. The noise cancellation efficiency of both PSO and DE are very good. From the two, DE shows better performance as well as faster convergence. In presence of low level of nonlinearities the DE shows very good performance in cancelling noise. So it can also be designed as a nonlinear controller.

References

[1] S. J. Elliot, "*Signal Processing for Active Control"*, London, U.K.:Academic ,2001.

[2] S. D Snyder, "Active Noise Control Primer", Modern Acoustics and Signal Processing, 2000.

[3] C. H. Hansen, "Understanding Active Noise Cancellation",Taylor & Francis group, 2001.

[4] J. G. Proakis, D. G. Manolakis, "Digital Signal Preocessing", Pearson Publication, 2007.

[5] G. L. Sicuranza, A. Carini: "A Generalized FLANN Filter for Nonlinear Active Noise Control", IEEE Trans. on Audio, speech, and language Processing, Vol. 19, 2011.

[6] C. Y. Chang and D. R. Chen, "Active noise cancellation without secondary path identification by using an adaptive genetic algorithm," *IEEE Trans. Instrum. Meas.*, vol. 59, no. 9, pp. 2315–2327, Sep. 2010.

[7] N. K. Rout, D. P. Das, G. Panda, "Particle Swarm Optimization Based Active Noise Control Algorithm without Secondary Path Identification", *IEEE Trans. Instrum. Meas.*, vol. 61, 2012.

[8] N. V. George, G. Panda, "A robust evolutionary feedforward active noise control system using Wilcoxon norm and particle swarm optimization algorithm", *Expert Systems with Applications*, vol. 39, no. 8, pp. 7574-7580, Jun. 2012.

[9] N. V. George, G. Panda, , "Development of low complexity evolutionary computing based nonlinear ac-tive noise control systems", *in Proc. of ICEAS 2011*, Bhubaneswar, India, Dec 2011.

[10] H. Modares, A. Ahmadyfard, M. Hadadrazif, "A POS approach for nonlinear Active Noise Cancellation", 6th WSEAS International conference, Lisbon, 22nd September 2006.

[11] A. K. Qin, P. N. Suganthan, Self-adaptive Differential Evolution Algorithm for Numerical Optimization, 2005.

[12] LI Gao-Yang, LIU Ming-Guang, “The Summary of Differential Evolution Algorithm and its Improvements”, 3rd international Conference on Advanced Computer Theory and Engineering (ICACTE), 2010.