ADAPTIVE EQUALIZATION BASED ON PARTICLE SWARM OPTIMIZATION TECHNIQUES

BY

ADIL HUMAYUN KHAN

A Thesis Presented to the

DEANSHIP OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

1963   ١٣٨٣

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

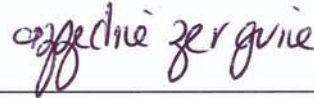## ELECTRICAL ENGINEERING DEPARTMENT

APRIL 2013

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
DHAHRAN 31261, SAUDI ARABIA

DEANSHIP OF GRADUATE STUDIES

This thesis, written by **ADIL HUMAYUN KHAN** under the direction of his thesis adviser and approved by his thesis committee, has been presented to and accepted by the Dean of Graduate Studies, in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE IN ELECTRICAL ENGINEERING**.
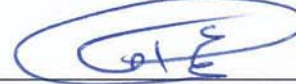
Thesis Committee

_____
Dr. Azzedine Zerguine (Adviser)

_____
(Co-adviser)

_____
Dr. Ali T.Al-Awami (Member)

_____
Dr. Ali Ahmad Al- Shaikhi (Member)

_____
(Member)

_____
Dr. Ali Ahmad Al- Shaikhi
Department Chairman

_____
Dr. Salam A. Zummo
Dean of Graduate Studies

_____
10/4/13
Date

Dedicated to the two most important persons in my life
Mom and Dad

ACKNOWLEDGMENTS

me during the initial stages of my thesis work and for his continuous guidance and support. And in the end but none the less, i would like to thank my friend Danish Sattar, for all his help during my research.

I would like to thank my friends and colleagues for their support and awareness they have given me since I arrived at KFUPM. We certainly enjoyed doing our thesis during the same time, sharing our experiences, spending long nights together doing our thesis work and celebrating our achievements. My deepest gratitude goes to my resilient and loving parents and family. I am eternally indebted to my parents for all their prayers, concern, love and understanding.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AE   Adaptive Equalization

BER   Bit Error Rate

EA   Evolutionary Algorithms

EMSE   Excessive Mean Square Error

GA   Genetic Algorithms

HPSO   Hybrid Particle Swarm Optimization

HPSO-LS   Hybrid Particle Swarm Optimization using Local Search

HPSO-TV   Hybrid Particle Swarm Optimization using Train and Verify

LMS   Least Mean Square

LS   Local Search

MSE   Mean Square Error

PSO   Particle Swarm Optimization

PSO-AW   Particle Swarm Optimization using Adaptive inertia weights

PSO-CCF   Particle Swarm Optimization using Constant Constriction Factor

PSO-VCF   Particle Swarm Optimization using Variable Constriction Factor

PSO-W   Particle Swarm Optimization using linearly inertia weights

RLS   Recursive Least Squares

SNR   Signal to Noise Ratio

TV   Train and Verify

THESIS ABSTRACT

NAME: Adil Humayun Khan

TITLE OF STUDY: Adaptive Equalization Based on Particle Swarm Optimization Techniques

MAJOR FIELD: Electrical Engineering

DATE OF DEGREE: April 2013

Adaptive equalization made it possible for digital data transmission over radio and telephone channels, as it mitigates the distortions caused by these channels. Different algorithms have been used in adaptive equalization, e. g., the least mean square (LMS) and the recursive least square (RLS) algorithms. Recently, particle swarm optimization (PSO) technique was introduced and turned out to be very effective in handling problems having non linear behaviour. Different versions of the PSO algorithm were proposed, to name a few, the PSO using linearly time decreasing inertia weight (PSO-W) and the PSO using constant constriction factor (PSO-CCF). However, these algorithms still suffer the problem of stagnation and can become less effective in a situation when the solution hits a local minimum. We will address such issues here. In this thesis we have implemented a new al-

gorithm for adaptive equalization, PSO using adaptive inertia weight (PSO-AW). A new algorithm, Hybrid Particle Swarm Optimization (HPSO), is also proposed for adaptive equalization. In the end two new methodologies, named Local Search (LS) and Train and Verify (T&V), are used to reduce the number of computations. PSO-AW uses adaptive inertia weights, instead of linearly decreasing inertia weights, to improve the convergence rate and secure better steady state error simultaneously. HPSO will incorporate three different techniques. These techniques includes re-randomization of particles to improve the search capacity of the swarm, second one is to introduced more socialized behaviour among particles, so that there is less chance of getting trap in to some local minimum values. And the third one is adaptively inertia weight assignment to the particles. This hybrid algorithm secured the minimum steady state error as compared to all previously used PSO algorithms as well as the LMS algorithm in both non-linear and linear channels. In order to complete the process with minimum number of computations, our proposed algorithm will be incorporated with two new techniques as well, LS and T&V. While using these techniques, although there is slight effect on convergence rate, but the reduction in number of PSO operations is remarkable. Significant improvements in BER and convergence rate, obtained using these algorithms. Extensive simulation results are conducted to confirm the consistency in performance of these algorithms in different scenarios.

خلاصة أطروحة

الاسم : عادل همايون خان

عنوان الدراسة : معادلة التكيف استناداً إلى تقنيات سرب الجسيمات المتحسن

الميدانية الرئيسية : الهندسة الكهربائية

التاريخ الدرجة : ابريل ٢٠١٣م

لقد جعلت معادلة التكيف انه من الممكن نقل البيانات الرقمية عبر قنوات الراديو والهاتف، كما أنها تخفف من التشوهات و التشويشات الناجمة عن هذه القنوات. استخدمت خوارزميات مختلفة في معادلة التكيف كأقل متوسط مربع (لاست مان سقار) و خوارزمية عودية اقل مربع (رخرسف لاست سقار). في الآونة الأخيرة، تم تقديم تكنولوجيا سرب الجسيمات المتحسن واتضح أنها فعالة جداً في التعامل مع مشاكل وجود السلوك الغير خطي. واقترحت إصدارات مختلفة من خوارزمية سرب الجسيمات المتحسن كاستخدام تناقص الوقت للوزن الراكد خطيا (لنارلي تم دئراسنگ انرتا واگهت) باستخدام عامل التقليص الثابت. ومع ذلك، هذه الخوارزميات لا تزال تعاني من مشكلة الركود ويمكن أن تصبح أقل فعالية في حالة عندما الحل يكون الحد الأدنى المحلي. وسوف نطرح بعض من هذه القضايا هنا. وقد نفذنا في هذه الأطروحة خوارزمية جديدة لمعادلة التكيف، باستخدام ركود الوزن التكيفي. خوارزمية جديدة، تسمى هجين سرب الجسيمات الأمثل، ايضا مقترحه لمعادلة التكيف. في نهاية المطاف تستخدم منهجيتان جديدتان، تسميان البحث المحلي (مَل صارئَه) و درب وتحقق (ةرين اند طرفي)، للتقليل من عدد العمليات الحسابية. يستخدم جهاز سرب الجسيمات المتحسن لأوزان الجمود التكيفية، بدلاً من تناقص الوقت للوزن الراكد خطيا، إلى تحسين معدل التقارب وتأمين أفضل حالة مستقرة خطأ في وقت واحد. وسيتضمن هجين سرب الجسيمات الأمثل ثلاث تقنيات مختلفة. هذه التقنيات تشمل إعادة توزيع عشوائي

للجزيئات للتحسين من قدرات البحث عن السرب، و ثانيا عرض عدد أكثر من السلوك اجتماعيا بين الجسيمات، حتى تكون هناك فرصة أقل لحصر السرب بين قيم محلية دنيئه. والثالث هو بتكييف تحديد وزن الركود للجزيئات. استخدام خوارزمية الهجين هذه ضمنت الحد الأدنى من الأخطاء في حالة مستقرة مع المقارنة مع كل ما سبق من الخوارزميات ، فضلا عن خوارزمية أقل متوسط مربع في القنوات الغير خطية، والخطية. من أجل استكمال العملية مع الحد الأدنى لعدد من العمليات الحسابية، ستدمج لدينا الخوارزمية المقترحة مع اثنتين من التقنيات الجديدة كذلك، البحث المحلي و درب وتحقق. في حين استخدام هذه التقنيات، على الرغم من أن هناك تأثير طفيف على معدل التقارب، ولكن الانخفاض في عدد من عمليات سرب الجسيمات المتحسن لافت للنظر. تحسينات كبيرة في معدل بيغ والتقارب، التي تم الحصول عليها باستخدام هذه الخوارزميات. نتائج المحاكاة واسعة النطاق تجري للتأكد من الاتساق في أداء هذه الخوارزميات في سيناريوهات مختلفة.

CHAPTER 1

INTRODUCTION

Adaptive Channel Equalization, made it possible the efficient use of all the radio and telephone channels, as it compensate for all the distortions, added by channel. Due to its significant importance, a productive research is carried out which produced a very beneficiary work [1]. Adaptive equalizers are used to mitigate the different distortions added by the channel in to any transmitted signal, and one of these distortions, is inter symbol interference (ISI). The main reason for ISI is the dispersion of the signal due to multipath in time varying channels. Adaptive equalization is used in communication systems which have high speed of data transmission and especially when these systems do not use frequency division multiplexing or differential modulation schemes. An equalizer is the most important and expensive part of the demodulator for any system as it usually takes more than 80% of the whole computation required to demodulate any signal[2], ultimately adaptive equalization gained a lot of attraction for the research in this field and contributed a rich body of literature in this area.

In this chapter, we will go through the background of channel equalization and adaptive channel equalization with their usage in different applications, then we will turn our attention to Particle Swarm Optimization (PSO) techniques in channel equalization. There will be brief literature overview on adaptive equalization and PSO, then main thesis contributions and thesis layout are stated.

1.1 Background of Equalization

High speed data networks and transmissions, over different channels of limited bandwidth, for example voice bandwidth channels, have been able to meet the requirements of the swiftly growing needs of different networks for communications. The rapid use of common carrier in digital transmission has also been applied to different technologies, like line of sight terrestrial radio and satellite communications. As these analogue channels do not have ideal behaviour, so these will introduce different impairments is the signal which was used as a input to the modulator. These impairments might include the statistical corruption which may be multiplicative or additive due to thermal or any other source of noise, like impulse noise which will make the signal fade. These different types of impairments introduced by the channels are basically linear, non-linear or harmonic distortions, and also time dispersions. In telephone lines the frequency response of ideal channel should have constant amplitude with constant and linear phase delay, but due to time dispersion the response of these channels will deviates from it.

Therefore in order to tackle all these distortions and impairments of the channel response, Equalizers are used on the receiver side. Equalization was initially used for the loading of coils to enhance the performance of the twisted pair telephone cables, which is used for voice channels in telephony communication. The transmitter side will take the input data in form of bits then it will encode them at some specified signalling rate. Different modulation schemes are used on transmitter, for example in case of Pulse Amplitude Modulation (PAM), in which every signal will be mapped on a pulse amplitude. And every symbol is defined for a specific time interval, but as these symbols are transmitted over these time dispersive channels, they will not remain limited to their time interval but will be extended in the time intervals of the other symbols, and this type of distortion is known as inter symbol interference. In any type of communication system, this ISI is one of the major problems while retrieving back the original signal and this impairment is the great hindrance in the way of high speed data communication [3]. It can be stated here that equalization will be applied to any signal processing device in which the main goal is to reduce this inter symbol interference. For high speed data communication, like at a rate of more than 4800 bits per second, over telephone voice channels, equalization will surely be required to tackle the inter symbol interference added by these telephone channels. As due to importance of this research area, equalization, so many researches proposed some useful literature on it, and still the research on this is continue.

1.2 Adaptive equalization

As stated earlier that equalizers are used in the high speed data communication networks to avoid inter symbol interference, which is caused by the time dispersive behaviour of the channels. Still there is another problem related to characteristic of the channel, over which data has to sent, that characteristics of these channels are unknown and it varies with the time, so a simple equalizer at the receiver side will not be enough to re construct the original signal and we have to take on account this time varying behaviour of the channel. Therefore instead of equalization, we used Adaptive Equalization, in which equalizer's behaviour will vary as the characteristics of the channel varies, here equalizer will keep on tracking the channel. For medium speed communication networks, like up to 2400 bits per second, conventional methodologies might work but the channels which are used in switched telephony networks, where variation of the channels cannot be tackled by conventional methods, adaptive equalizations is used. Now it is like a universal rule of using adaptive equalizer in high speed data networks, more than 2400 bits per second [4].

Inter symbol interference in the channels of under water and radio links is due to the multipath propagations on these channels, and the transmission over these channels can be seen as a simultaneous transmission over different channels which have different delays and amplitude and characteristics[5]. The adaptive equalizers are able to respond efficiently for this inter symbol interference in multipath propagation, just like in the telephone voice channels. In different scenarios of

radio channels, adaptive equalizers plays important role to cancel out the effect of interference, jamming sources and diversity combining [6]. In almost every radio channel we can have time varying fading, so adaptive equalizers which applied over these channels should be able to control this issue as well. Time varying fading is also one of the major impairment causes, and it is widely observed in radio channels. Tropospheric microwave digital radio channels, ranges from 4 to 11 GHz, endure slow fading. Therefore in order to mitigate all these effects added by the channels, adaptive equalizers are used [7].

1.2.1   Functionality of an Adaptive Channel Equalizer

As the name shows that the behaviour of such channel equalizers will be channel dependent, means the filter taps of the equalizer will vary accordingly to channel response. We will explain briefly working of a linear channel equalizer and the cost function which is normally used to find the optimum values for the taps of equalizer. The basic block diagram of the adaptive channel equalizer is shown in figure 1.1. Let the input signal $s(k)$ is passed through the channel, and some noise $v(k)$ is added in to it and the resultant signal will be denoted as $u(k)$, then this degraded signal will be passed through our adaptive equalizer, whose tap length is M. So the vector of adaptive equalizer taps will be,

$$f^T(k) = [f_0(k) \quad f_1(k) \quad f_2(k)..... \quad f_{M-1}(k)]$$

5
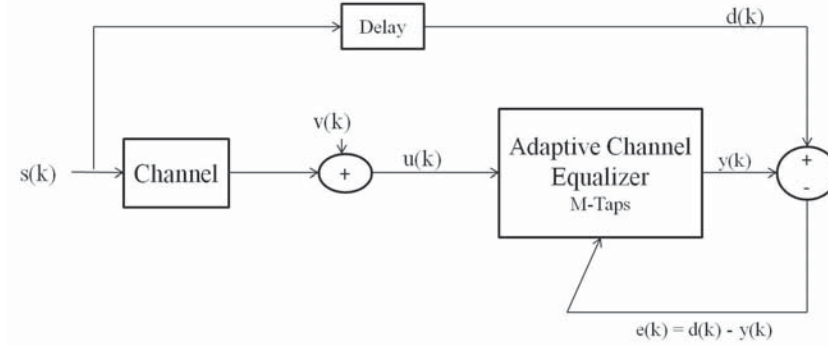
Figure 1.1: Set up of an Adaptive Channel Equalizer

And the input to this adaptive equalizer is $u(k)$,

$$u(k) = [u_0(k) \quad u_1(k) \quad u_2(k)..... \quad u_{M-1}(k)]$$

$$u(k) = [u_0(k) \quad u_0(k-1) \quad u_0(k-2)..... \quad u_0(k-M+1)]^T$$

And at the output of the equalizer we will have the signal $y(k)$, which can be shown as following,

$$y(k) = \sum_{i=0}^{M-1} f_i(k) * u_0(k-i) = f^T(k)u(k)$$

Now we have to calculate the error in the recovered signal $y(k)$ and on the basis of this error we will vary the taps of equalizer, which is actually the adaptive nature of such equalizers. But we cannot directly compare the output of equalizer with the input signal, because during all the processing of the input signal, like passing through the channel then passing through the equalizer, so there will be some delay added in the input signal to make it a fair comparison with the recovered

6

signal. We say $d(k)$, desired response, a delayed version of the input signal, the we have to subtract the output of equalizer from this desired response to calculate the error term $e(k)$. So the error signal will be,

$$e(k) = d(k) - y(k) \tag{1.1}$$

$$e(k) = d(k) - f^T(k)u(k)$$

While using this error function we have to device a way to judge the performance of different algorithms. There are many methods which are used to find the steady state analysis of the algorithms, such as Mean Square Error (MSE), Mean Square Deviation (MSD) or Excessive Mean Square Error (EMSE) are commonly used for the steady state analysis of the algorithms.

The most commonly used method, which is also used in the rest of this literature for analysis and the comparison with our proposed algorithms, is Minimum Mean Squared Error (MMSE). The cost or objective function of MMSE can be defined as following [8],

$$J = E[e(k)^2]$$

$$J = E[d(k)^2 + y(k)^2 - 2d(k)y(k)]$$

$$J = E[d(k)^2] + E[f^T(k)u(k)u(k)^T f(k)] - 2E[d(k)f^T(k)u(k)]$$

Then we will take the gradient of the above equation and equate it to zero to find the minimum optimum value for the taps of equalizer. We will not go in to the

details of this whole procedure. The so far discussion was just a brief overview of the analysis that how things work here. There are many algorithms working currently for adaptive channel equalization, like Least Mean Square Estimation (LMS), Recursive Least Squares (RLS) and Steepest Descent etc. A detailed literature review on these and other algorithms has been done in next section. There are different ways of handling the cost function for each algorithm, like LMS will not use the expected value of the objective function rather it will use the instantaneous values, and this is the most commonly used algorithm for channel equalization, and we will use it as well for the comparison with our proposed algorithms.

1.2.2 Convergence of Adaptive channel Equalizer

The exact convergence analysis of any algorithm, used for adaptive channel equalization, is very difficult to comprehend, but for the algorithms like LMS etc, it can be defined as the dependency of convergence speed on step size in algorithm. Basically the convergence of algorithm reveals the fact that how quickly any algorithm will approach to its optimum values. And convergence speed is very important parameter in adaptive channel equalization.

Convergence speed can be increased for any algorithm, like in LMS, steepest descent etc, by assigning higher value to step size of that specific algorithm. As nothing is free so, if we assign higher values to step size in order to get better convergence rate, there is the chance that our algorithm will stuck in to some local

minima and this thing will cause a higher error, of course if step size is greater than the tracking capability of algorithm will be greater. Another parameter, which is used for the analysis of algorithm is excess mean square error (EMSE), is affected by step size. This EMSE will have higher values of error if filter taps will keep roving around the optimum values of filter taps. So a trade of will be done here between higher convergence rate with better tracking analysis and the excess mean square error.

In Particle Swarm Optimization (PSO) algorithm, this convergence speed will not depend on some step size, rather here it will depending on fine tuning of the parameters, which are used in this algorithm. Different parameters will be used in different versions of PSO algorithm, and the sensitivity analysis of these will be taken place to attain any specific convergence speed.

1.2.3   Literature review on Adaptive Equalizer

In 1928, telegraph transmission theory has been presented by Nyquist [9], in which he made the basis of transmission of pulse over analogue and band limited channels. Least mean square (LMS), which is an adaptive filtering algorithm and it has been in used in this research area for more than last decay and even in present days, it was presented by Widrow and Hoff, in 1960 [10]. However, in 1960 the research on PAM systems regarding equalization in adaptive sense, was more in theoretical knowledge of trapped delay equalizers. These equalizers are also known as zero forcing or transversal equalizers, they performed equalization

in presence of symbol interval tap spacing [11, 12]. Time dispersive additive wide Gaussian noise channels, which are discussed in [13], such channels add so much distortions to the signals, the structure and design of filter which minimize mean square error for such channels, were proposed in [14, 15]. AS LMS showed great satisfactory results for such research problems, so in 1960, extensive research conducted in this topic which contributed productive literature [16, 17, 18]. All the proposed methodologies in these literatures contributed highly favourable results in field of adaptive channel equalization. After all these, abundance of research done on the structure of the non linear receivers, and it was defined for different optimum criteria, for example error probability etc, in [19, 20, 21]. All the research done on this specific area, set the base for the other researchers to produce some ground breaking results like some of the literature is [22, 23, 24, 25, 26, 27]. Due to this fruitful research, Maximum Likelihood Sequence Estimator was proposed in [28], in which they formed this ML estimator by viterbi algorithm [29]. Some researchers tried to continue their research in more simpler way, by doing research on a much more simpler sub-optimum receiver known as decision feedback equalizer [30, 31, 32, 33, 34, 35, 36]. There were also some research topics like linear feedback, infinite impulse response which were the strong options to implement the adaptive equalization but these were facing some serious issues like lack of quadratic performance surface, which do not guarantee the stability and almost negligible gain if we compare it with transversal equalizers. Different modulation schemes were used like single side band (SSB), vestigial side band (VSB) etc,

but researchers proved the superiority of the quadrature amplitude modulation (QAM) over other modulation schemes, so this provision required the modifications in the old PAM equalizers and give them more complex structure to make the signalling possible in the receiver of in phase [37, 38, 39, 40]. In early 1970, different equalizers such as decision feedback and transversal filters, in which tap spacing was less than symbol interval, were proposed in [41, 42]. And in same decade 1970, the practical use of these equalizers were made possible by using such equalizers in telephone lines and in military radio systems [43]. In literature [44, 45, 46, 47], the practical advantages of such equalizers over symbol spaced equalizers were discussed. After all the research done in this field, it was revealed by the researchers that it is useful to use non linear decision feedback receiver, which is also known as Anti-ISI filter, in which fractionally spaced equalizers used as a matched filter [48, 49].

Up to 1970, the most of the literature review in the field of adaptive channel equalization, was focussed in the structure of the equalizer and also the steady state analysis of the equalizer. Research work was also done on the transient analysis of it, but this work was less as compare to the steady state analysis, due to the difficulties in it. In following literature, [50, 51, 52], the research on the convergence of LMS algorithm in steady state analysis, for transversal equalizers, was done. The effect of channel characteristics on the convergence rate of LMS algorithm was also discussed in this literature. After abundance of research on LMS, different versions of LMS such as orthogonal LMS was proposed in [53, 54]. In order to

make a fast start up in these methods, cyclic and periodic training methods for adaptive equalization were used [55, 56, 57]. Godard, first time described the way of using Kalman filtering estimation method to estimate the filter taps in LMS equalization at every symbol interval [58]. Later the method of recursive least squares (RLS), was introduced and intense research started in this subject, and due to this research lattice and transversal algorithms were designed, which are presented in these literatures [59, 60, 61, 62]. Then advantage of such research was taken by applying these algorithms, in the adaptive equalization of signal which were sent on high frequency radio channels in which we have to track rapidly time varying conditions of these channels [63]. So in overall an extensive research have been done in this research area which cause the reduction in number of arithmetic calculations but at the coast of more complex and sophisticated structures with larger memories, but still it is the sole purpose of research to propose the methods with their advantages and drawbacks, and it depends on the users to implement the most suited one in their application.

1.3 Particle Swarm Optimization Algorithm

Particle swarm optimization is a newly developed algorithm which is nowadays making its way in almost every optimization research problem. This heuristic approach of finding optimum value for any cost function, lies under the category of Swarm Intelligence. Kennedy and Eberhart, first proposed this algorithm [64], they applied this algorithm initially to some simplified social model, here they

took the mental state of mind of every individual as the position of the particle, and each individual will posses his own mental state and attitude [65].

In this section the reasons for using PSO for adative equalization, instead of instead of conventional algorithms like LMS etc, will be explained. *The problem with the previously used algorithms, for channel equalization, is that the convergence speed of algorithms, like LMS, is too slow and it takes large number of iterations for the solution to converge on some optimum value. Although LMS gives a stable solution, if parameters of it are properly tuned, but still the convergence rate of it was slow. Therefore to make the convergence rate fast, PSO was applied to such research problems.*A detailed literature review on PSO has been done in next section.

PSO gives better solution to the problems which are multi dimensional and non liner or non differentiable in nature. It is quite robust, fast, have lower computational cost, provides swift convergence rate with reliable solution. And due to such abilities it has been used to solve such important research problems [66]. PSO basically works like a ad-hoc system, where each individual particle will take the decision on the basis of its individual observations and also modify this decision on the account of the cooperation which it gets from the remaining particles of the swarm. Hence every particle will take in to account these both things while making any decision regarding the problem. As there will be so much inter communication at the swarm level among particles so it will have a complex structure, but it will be able to solve quite complex optimization problems easily. Here we

will just briefly explain the working of this algorithm.

This algorithm works on the basis of its population members, which collectively known as swarm, and these particles work collectively to optimize the function, which should be real valued, have any specific number of dimensions. Each particle in the swarm will have a specific position which is known as a potential solution to any optimization problem in solution space. PSO will approach towards the most optimal solution by applying specific modifications on the existing set of solution, and these changes will be made through probabilistic and iterative modifications. Every particle will be having two basic parameters one is the position and second one is the velocity, which decides the speed of every particle through which it will approach to its optimum value. These both parameters are of immense importance, as these should not be too fast that the particles fly out from the solution space or they should not have smaller values that the convergence rate will be slower. These particles will be generated randomly, and just like any other optimization algorithm, it will be having an objective or cost function. For channel equalization PSO will use the following cost function,

$$J_n(k) = \frac{1}{N} \sum_{m=1}^{N} [e_{mn}(k)]^2 \qquad (1.2)$$

Where $e_{mn}(k)$ represents the mth error of the nth particle, and N is input data window size to the equalizer which is shown as $u(k)$ in the figure 1.1. And the error term will be calculated exactly in the same way as by equation 1.1, which is the desired value, $d(k)$, should be subtracted from the output of the equalizer, $y(k)$.

Afterwards we will apply this cost function to each and every particle, then we will look for the minima among all the particles that will be known as global minima or *gbest*. Every particle will be observed separately as well, and we will look for its minimum value achieved so far, means every particle will have a memory and it will also be storing the value of it before applying the cost function on it. Thus the minimum value the minimum value so far achieved by this particle will be known as local best or *pbest*. We will keep on changing the positions of particles in such a way that they approach to an optimized value. While applying the position and velocity updates on the particles, it has to keep under consideration that no particle should fly out from specified range. Figure 1.2 represents the flow diagram of the basic functionality of the PSO algorithm.
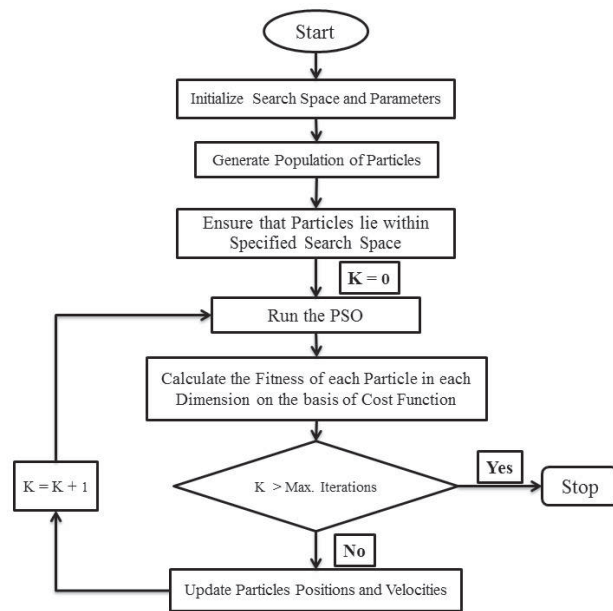
Figure 1.2: Flow chart of basic PSO Algorithm

The flow chart, in figure 1.2, clearly explain the basic functionality of PSO algorithm that in start we have to define the boundaries, then randomly generation of the particles and also make sure that particles are in specified range, then run this algorithm for any number of specified iterations, which is totally application dependent. Apply cost function on every particle, look for the minimum values as specified earlier, then update the position of every particle until a stopping criteria has arrived.

1.3.1   Literature review on PSO algorithm

Since after the introduction of this optimization technique, different variations were introduced by many researchers in it, in order to improve its performance in different aspects. In start, many researchers discussed the effect of inertia weights on the performance of PSO, as velocity is multiplied by this inertia weight factor before the velocity updating process so inertia weight actually controls the velocity of the particles [67]. Then a new version was introduced by Clerc in [68], where a constriction factor was used instead of inertia weight factor, and this constriction factor used to make sure that particles should be limited to their present velocity before updating. The main advantage of this algorithm is that it converges very swiftly for the uni-modal problems, but for the multi-modal problems, there is the chance that due to its swift convergence, it might trap in to some local minima. One solution to get rid of this problem is to first find out all local minima before making the decision for any global optimum value, but this procedure is

quite time consuming and complex. To tackle this premature convergence, a new technique was proposed in [69], here it was shown that the problem of this premature convergence can be resolved if we breed some of the particles, and this new technique was named as Hybrid-PSO. A similar idea was proposed in [70]. In this paper two new techniques were introduced, the combination of PSO and Genetic Algorithms, both in parallel and in series form, and it was proved with the help of simulation results that both these algorithms work better in order to find out global optimum solution.

The most important thing regarding PSO, is the sensitivity analysis regarding its different parameters, the proper value assignments to all its parameters and if these values are properly assigned then there is very less chance of the errors like premature convergence etc. There are many research papers, which have addressed the critical issue of parameters tuning for PSO algorithm, but the most prominent research papers are proposed by Shi and Eberhart [67, 71, 72, 73, 74, 75].

Kennedy in [76], proposed the idea of neighbourhood topology, where information was shared by the particles with neighbours in order to enhance the search capacity and get better results with better convergence rate. Here different convergence rate were achieved by making different topologies of the neighbours, as neighbour will share information with only specific neighbours only.

First time PSO was applied to the Neural Networks [65], after the achievement of better results in this research field, PSO used for other applications as well.

Another development was proposed in PSO, which reduced the training time, is known as Cooperative Particle Swarm Optimizer (CPSO) [77, 78]. In this methodology, the whole swarm will be divided in to different small swarms and each will be solved with separate PSO. The different effects of size of the swarm was discussed in [79]. Parsopoulos and Vrahatis in 2002 [80], used PSO first time to solve the problems which were multi objective. PSO solved many test problems which were also multi objective and comparison of it with genetic algorithms have been discussed by researchers as well.

PSO also performed well as hybrid with other techniques, like in [81] it was hybrid with genetic algorithms. PSO has another hybrid version which is based on Levenberg-Marquardt optimizer, and while estimating six parameters, it achieved high success rate [82]. Hybrid PSO with Cauchy mutation was proposed by Wang in 2007 [83], and in this technique the problem of particles getting trapped in to some local optimum was removed as in this technique particles will get closer to the best particle in much less time or number of iterations. Due to this Cauchy mutation best particle will lead remaining particles towards better optimum value. Different simulation results showed that this method secured better performance with multimodal functions. Another technique was proposed by Wei in [84], in which he used different benchmark functions with only one dimensional mutation operator to evade local optimum values. Along with one dimensional mutation he used variable inertial weight and the variation in this inertial weight is based on the diversity and the fitness of any specific particle.

Another version of PSO was proposed by Xu in 2005, named Extended PSO (EPSO) [85]. In this method, at each iteration while updating velocity equation for each particle, he used global and local both best positions, as previously only global positions were used for velocity updating process. This methodology combines the best effects of both local and global best simultaneously.

A very useful paper which enlist the developments and all the applications and resources of PSO algorithm, was presented by Eberhart and Shi in 2001 [86]. This paper also presents all the arrays functions and the different types of it. Then another very important modification done in [87], by making the inertia weight as a function of improvement, any particle achieve in present iteration as compare to previous one, in this research paper there are some other techniques were proposed to overcome some critical issues related to the PSO algorithm and some techniques of this paper will be used in our proposed algorithm. A very important paper on this critical issue of PSO was presented by Kennedy [88]. In this paper he describe in some informal way that true purpose of the research in PSO algorithm is not to make it complicated rather to make it straightforward in such way that it can be used in different applications without any ambiguity, and also to do some detailed research on its essentials rather than purposing its suboptimal methods.

1.4 Thesis Contribution

This thesis comprises three contributions, first the implementation of a new algorithm PSO-AW to adaptive channel equalization, mainly to enhance the convergence rate. Second contribution is the proposed algorithm HPSO for adaptive equalizations, mainly to improve steady state error. The third contribution is regarding reducing the number of computations or PSO operations, and to achieve this two new techniques, LS and T&V, are incorporated with the proposed algorithm HPSO.

First newly implemented algorithm, PSO-AW, will contribute in the sense of, improvement in convergence rate. In previously used PSO algorithm, one of those exhibits better convergence rate but with degraded steady state error. And the other one, have the ability to secure better steady state error but with slow convergence rate. This newly implemented algorithm exhibits these both properties, improved convergence rate with better steady state error. The proposed algorithm will use hybrid methodology, by combining two previously used methods along with a new methodology. In this hybrid algorithm, these all techniques will be incorporated together in such a way that it yield improved convergence rate by securing minimum steady state error as compare to all previously used conventional algorithms.

This hybrid proposed algorithm achieved remarkable improvement in steady state error as compare to previously used algorithms, for nonlinear system. These both algorithms showed great improvement in BER with respect to LMS. Optimized

parameters of these both algorithms will be provided as well, in order to get stabilized MSE curves.

The main concern in all types of communications is to perform the functionality with minimum efforts, in order to save time and power. In this thesis we have incorporated our proposed algorithm HPSO, with two new techniques. Although these techniques have slight less convergence rate as compare to PSO-AW and HPSO, but these techniques used minimum number of steps to complete the optimization process in much lesser time.

1.5 Thesis Layout

This thesis will consist of six chapters. In first chapter, we explained the functionality of the adaptive channel equalization, and also all the reasons were stated which brought this optimization technique alive. In this chapter we also explained the basic functionality of the PSO algorithm, and the reason to employ this algorithm over conventionally used other algorithms.

In second chapter we will clarify the origins of PSO algorithm, right from the beginning when it was simulated just as a social model. In this chapter we will state the examples through which different types of social and basic scientific optimizations problems were tackled by using PSO. PSO is the part of swarm intelligence techniques, and in this chapter we will compare it with the genetic algorithms which belong to evolutionary algorithms.

In third chapter we will explain the PSO algorithm, right from the scratch and

explain it from basic level. All the modifications which have been done in PSO, for adaptive equalization application, will also be explained. Here we will compare the PSO with conventionally used algorithm, LMS, with the help of simulation results.

In fourth and fifth chapter, we will explain everything about this newly implemented algorithm PSO-AW and our proposed algorithm HPSO. All the reasons which have led us to use these techniques will be explained as well. Their functionality with help of flow chart and equations will be explained. The comparison of these algorithms, with all previously used PSO algorithms and LMS, will be presented with help of simulation results. Sensitivity analysis, to find the optimized parameters for these both algorithms, will be done also. These both algorithms will also be compared with LMS for Bit Error Rate (BER) analysis. In this whole research everything will be proved with the help of simulations results and figures, to strengthen our point of view.

In sixth chapter, we will state two new techniques which are used to reduce, the number of computations, and processing time. We will make the tabular comparison of these techniques with respect to number of computations and values of different parameters, for both linear and nonlinear scenarios. Final conclusion for this thesis and future work will also be proposed in this chapter.

CHAPTER 2

PARTICLE SWARM

OPTIMIZATION

2.1 Introduction

Particle swarm discovered by simulating a simplified social model. In this chapter we will describe the ideas related to particle swarm optimization in provisions of its originator, and will also review the steps of its progress, from basic social simulation in to an optimizer. Different examples that employ this concept will be stated as well. We will also explain the basics of particle swarm in contents of its emergence from its root level. Then we will explain it with the help of different evolutionary examples. The particle swarm algorithm here introduced in terms of social and cognitive behaviour, although in engineering and computer sciences, it has been used as a problem solving method. The first version of the particle swarm which is presented here is designed to work in a binary search space. Later

in this chapter we will introduce more commonly used version, which operates in a space of real numbers.

As stated earlier that it was first introduced by Kennedy and Eberhart [66].The roots of particle swarm optimization can be found in two main component methodologies. But more precisely it has ties to artificial life (A-life) in general, like fish schooling, bird flocking and swarming theory in particular. The other related theory is evolutionary computation, genetic algorithms, and evolutionary programming, both have ties to it. These relationships are briefly reviewed in the chapter. In the end we will compare PSO, which is the part of Swarm Intelligence, with Evolutionary Algorithms(EAs), and among these all EAs, we will mainly compare PSO with Genetic Algorithm (GA). Although there are many other evolutionary algorithms but most commonly used technique is GA, therefore we choose this for the comparison with PSO.

2.2 Simulating Social Behaviour

The movements in organisms like in fish school or bird flocking, have been explained by many scientists through computer simulations. Movement in organism of bird flocking has been explained by Notably, Reynolds [89] and Heppner and Grenander [90]. Heppner, who was a zoologist, was trying to find the basics rules by which so many birds to flock synchronously, doing scattering and regrouping and suddenly changing directions etc. These all scientists tried to find the insight that local processes, which were modelled by cellular automata, might underlie

24

the unpredictable group dynamics of bird social behaviour. All models depend strongly on manipulation of inter-individual distances that is, the synchrony of flocking behaviour was considered to be a function of the efforts of birds to maintain a specific and optimum distance between him and his neighbours.

It is quite obvious that some similar rules underlie animal social behaviour, which includes herds, schools, and flocks, and also of humans. A sociobiologist E.0. Wilson [91] has written, in the context of fish schooling, In theory each individual members of the school can benefit by the discoveries and previous experiences of all other members of the school during the search for the food. This advantage can become disadvantage, whenever the food resources are unpredictably distributed in different patches. This hypothesis or statement suggests that social sharing of information between consecrates gives an evolutionary advantage. The fundamental development of particle swarm optimization is basically due to this hypothesis. The model human social behaviour was one motive for development of such simulation, which is obviously not similar to bird flocking or fish schooling. Abstractness is one important difference. Physical movement of birds and fish have adjusted to avoid predators, find food and mates, and also adjust and optimize their environmental parameters such as temperature, etc. The physical movement and adjustment of humans also depend on cognitive or experiential variables as well. We, human, in common practice do not walk in step and turn in at the same time (although different researches in human this type of behaviour shows that human are capable of it) rather, we used to change or adjust our beliefs and atti-

tudes to comply with those of our social peers. Now in order to make a computer simulation, a very major difference came across in the psychology of human and birds, and that is known as collision. Any to human being can have similar beliefs and attitude without colliding with each other, but any two birds cannot have similar position without banging together. It seems fair, in discussing social behaviour of human, to map their notion of change into the fish or bird homology of movement. This thing is persistent with the classic Aristotelians point of view for qualitative and quantitative change as types of movement. So, rather than, moving through the three-dimensional physical space, and obviate themselves from collisions, humans revert in abstract multidimensional space, collision-free. Informational inputs, obviously, affected by physical space, but this thing is arguably a niggling component of psychological experience. In general, in very early age, humans learn to avoid physical collision, but to avoid collision while navigating in n-dimensional psychosocial space, still requires so many years of practice .

The particle swarm optimizer is probably best presented by explaining its conceptual development. As mentioned above, the algorithm began as a simulation of a simplified social environment. Therefore we will just explain the basics of this swarm optimization and then we will explain it with the help of some social behaviour examples and a binary model.

2.3   Evaluate, Compare and Imitate

Adaptive Culture Model and particle swarms have been explained by the help of a very simple sociocognitive theory. The explanation of the the process of cultural adaptation has been done here in terms of a high-level component, which can be viewed in the formation of different patterns among individuals and the also the ability to solve problems, and a low-level component. Which we can say that it is the actual or self explanatory universal behaviours of every individual, these low-level components can be explained in terms of three pioneer principles,

- Evaluate

- Compare

- Imitate

These all will be briefly explained here.

2.3.1   Evaluate

It is the most common behavioural characteristic of living organisms, which is tendency to evaluate. Any individual can evaluate oneself in so many ways, for example to grade or rate itself as attractive or repulsive, as positive or negative etc. This thing can even be found in bacterium, when they become agitated and start running and tumbling, when the environment is poisonous. It is very common thing in all organisms that their learning cannot occur unless they evaluate and distinguish features of the environment that attract and also the features that

repel, means they learn that which one is good or bad for them and so on. From this analysis, we can say that this learning could even be explained as a change that enables every organism to enhance or improve the average assessment of its environment.

2.3.2   Compare

How people are used others as standard for measuring themselves, is comparison and there are so many social theories which describe the ways that are used by the people for this specific purpose. And also these theories explain that these comparisons to others also serve as a kind of motivation to learn and change. It is a very common practice in our common life, we can take the example of a school boy who always tries to compare his own routine with the routine of his class fellow how is a position holder in his class. In general there are many theories which became the basis for subsequent social-psychological theories. Almost in every phenomenon of life, we rate ourselves by the comparison with others, and this comparison could be in evaluating our looks, personality, education, intelligence, wealth or any other aspects of our ability or opinion.

Like these all models here in particle swarm, individuals in the adaptive culture model compare themselves with their neighbours, but by keeping this view in mind that the neighbour which they have chosen for the comparison should be superior to them, if not then it is very obvious that there is no benefit in comparing. And these standards for social behaviour, of being superior or inferior, are set by

comparison to others. It is just like the similarity of a school boy, that he will compare himself with the student who have a good position in class relative to him, but not with the students whose grades are even low relative to him, so it is like a common psychology of people.

2.3.3 Imitate

Imitation is an effective way to learn to do things and normally people think that it is everywhere in the nature. And through research many scientists have proved that not all animals are capable of doing real imitation, in fact, they pronounce that only some birds, fish and humans are capable of it. There are different variations of social mimicking and learning is found among other species, but none compare to the ability of human to copy one another. Human mimicking comprises taking the perception of the other person, not only copy behaviour but also look forward for the reasons and purposes of it then execute that specific behaviour when it is suitable. For example, any individual's utilization of an entity as a tool may be known as individual's attention to the entity, this second individual may use the same entity, but might be in a different way. True imitation is mainly essential to human sociality, and it is central to the acquirement and preservation of the mental abilities as well.

These three principles of evaluating, comparing, and imitating, which have been explained, may be combined, even in quite simple format to execute the computer programs, while enabling them to acclimatize to multi-dimensional and complex

environmental challenges, and solve tremendously difficult problems. Our view deviates from the cognitive viewpoint in that nothing moreover evaluation, comparison and imitation takes place within the individuals mind will not be found hidden, private chambers covered inside the individual, but still it will remain in existence in the open, it is more like a communal experience.

2.4 Examples explaining Swarm optimization

Here swarm optimization will be explained with the help of different examples.

2.4.1 Nearest neighbour velocity matching and craziness

These simulations which are quite satisfactory relied on two props: nearest neighbour velocity matching and craziness. A random population of birds was generated with a location for every bird on a pixel grid and representing their velocities by X and Y. To make our simulation reasonable we represent each bird by an agent. Then within a loop at each iteration this program will determine the nearest neighbour for each agent, then assign that agents X and Y velocities to the focused agent. By applying this simple rule in program, it was observed that it creates a synchrony of movement [68].

After running this simulation so many times it was observed that the flock rapidly settled on a common and fixed direction. So in order to introduce some random behaviour in our simulation, a stochastic variable called *craziness* was introduced. Within a loop at each iteration some random changes were added to the velocity

variables X and Y. This change in simulation reflects enough discrepancy into the system to give the it an exciting and lifelike emergence, although of course these whole variations were artificial.

2.4.2   The Cornfield Vector

To overcome the deficiencies of previous simulations here a dynamic force will be added in to it. In which new term introduced, roost, which is known as temporary rest place of birds, all birds will flock around it, and in programming sense it will be a position on the pixel screen that will attract the birds until they landed there eventually. Due to this addition, we will not be needed the variable like craziness to introduce stochastic behaviour in simulation. And the idea of a roost was interesting; and it raised another question which seemed even more motivating. This flock of birds in simulation knew the exact location of roost, but in real life it would not be the case, birds may land on any tree or telephone wire that meet their urgent requirements. Rather in real life, birds flock try to land where there is food. So the very next basic question will be that how do they find about it? If you put out a bird feeder, you will see that within hours a huge number of birds will be expected to find it, although they will not be having any previous knowledge of its location or appearance etc. So to answer this question that how the flock find it, there is likelihood that there might be something related to the flock dynamics which enables members of the flock to exploit on one anothers information.

Then another change has been added in simulation by defining a cornfield vector, which is a two dimensional vector of XY coordinates on the pixel plane [92]. Then in our simulation each agent will be programmed to calculate its present position by following equation,

$$P_{eva} = \sqrt{(present\_x - 100)^2} + \sqrt{(present\_y - 100)^2}$$

So that at the position (100,100) the value will be zero. Each agent will remember the best value of it so far and the coordinates which had resulted in that value. This value will be known as *pbest[]* and the positions *pbestx[]* and *pbesty[]*, here it is important to mention that braces shows that these are arrays with number of elements equal to number of birds or agents. So every agent will evaluate its position by moving across the whole pixel space, so in this manner X and Y velocities for every agent will be adjusted in a simple manner. If the position of agent is on the right side of its *pbestx*, then its X velocity, let call it vx, will be adjusted negatively by a random amount weight by following equation,

$$vx[] = vx[] - rand() * p - increment$$

If the position of agent is on the left side of its *pbestx*, then *rand()*p-increment* will be added to *vx[]*. Similarly for Y, velocities *vy[]* will be adjusted up and down, depending on whether the position was above or below *pbesty*.

Now to introduce some social behaviour in simulation it will be programmed in

such a way that every agent will know about the globally best position that any other member of the flock so far have found, and also its value. This phenomenon was introduced by simply assigning the array index of that agent with the best value so far, to a variable called *gbest*, so that *pbestx[gbest]* will be the groups best X position, and *pbesty[gbest]* will be the best Y position, and this information will be available to every agent of flock. Again, each agents *vx[]* and *vy[]* will be adjusted as follows, where g-increment is a system parameter.

$$if \; presentx[] > pbestx[gbest] \quad then \quad vx[] = vx[] - rand * g - increment$$

$$if \; presentx[] < pbestx[gbest] \quad then \quad vx[] = vx[] + rand * g - increment$$

$$if \; presenty[] > pbesty[gbest] \quad then \quad vy[] = vy[] - rand * g - increment$$

$$if \; presenty[] > pbesty[gbest] \quad then \quad vy[] = vy[] + rand * g - increment$$

With *p-increment* and *g-increment* will be assigned some high values, after it the flock seemed to be sucked aggressively into the cornfield. Within some iterations the entire flock of birds, usually 20 to 25 those, was seen to be clustered in form of tiny circles surrounding the goal. If we assign *p-increment* and *g-increment* low values, then flock will swirl around the goal, rationally approaching it, swinging out steadily with making sure that subgroups will be synchronized, and eventually landing to the target.

Now we will turn our attention from birds to a binary case.

2.4.3 A model of Binary decision

Let we consider any set individuals and all of them will have only one simple thing in mind, only single and fixed set of decisions to make, true/false or yes/no, these type of decisions are known as binary decisions but these are very slight decisions where choice of decision is not very easy. For every single decision, this simplified individual can have only one of the two stages either in yes state, which will be represented here by 1, or the other state will be represented by 0. As any single individual will be surrounded by other individuals which means that any single decision will be surrounded by other yes or no individuals, who are also trying to make the decisions. Every individual will obviously want to make the best choices, whether it would be no or yes [93].

Every individual will have two types of important information. The first is their own experience of choosing that is which state has been better so far, whether yes was more fruitful or no was. But these individual beings have second thought; they have awareness of how the other individual beings around them have performed so far. As these choices are so simple that they all know that the choices which have been made by their neighbours so far, have found most positive and how positive the best prototype of choices was. If these individual beings were like people, then they also would have known that how their neighbours so far have done by observing them and also by sharing with them about their experiences. The likelihood that the any individual will decide yes for any of the decisions is a function of how booming this choice has been for them in the past as compare

to the other choice. The choice will also be affected by some social weight, although there is no doubt in it that this rule is not so clear among human being. It has been proposed by social impact theory that the binary decision by every individual will tend to agree with the judgement held by the bulk of others and also subjective to the strength and proximity.

Different individuals will be connected to each other according to so many different schemes; there are two simple sociometric principles which have been used by most particle swarm. The first conceptual thing, which connects all members of the population with each other, is called *gbest*. Here it is worthy to mention that each particle will be influenced by the overall best performance of any member among whole population. The second parameter known as *lbest*, where g and l will be used for global and local, it will create a neighbourhood for each specific individual, which consist of its p nearest neighbours in the population and along with itself. For example, if $p = 2$, then each individual i will be effected by the best performance with in a group made up of particles i $1, i, and$ $i + 1$. Different kind of effects can be achieved by making different neighbourhood topologies. There are two components of theory here first one is that individual term, individual knowledge or approach towards any behaviour, and other is social term, which can be known as cultural approach. In many theories, these two kinds of concepts are found and can also be seen in our decision model as the two main things that made the changes in formula. Here strongly stated that the coexistence of these two modes of awareness, that is, knowledge and awareness gained

by the sanity through practice in the world and the knowledge gained from others, these two skills give human the rational gain, which is also the main source of our astuteness. There is also a third parameter which affects the individual's decision, besides their previous experiences and the feedback from the social environment, is their present point regarding that specific issue. There might be the case that current point of view regarding specific issue of any individual, may have a negative attitude but subsequent feedback from others might be positive experiences regarding that choice, but that individual still have a negative feeling about it. The positive feedback from others may compel that specific individual likely to select the positive alternative, but to make that individuals general behaviour towards the positive domain; the assessment verge would still have to be shifted upwards. If individual is already at extreme position initially, then likelihood of it to change is lower.

In order to make our simulation work properly, mathematically, we will introduce a model where the likelihood of an individuals decision to be yes or no, true or false, is a function of both social and cognitive factors [93],

$$P(x_{jd}(t) = 1) = f(x_{jd}(t-1), v_{jd}(t-1), p_{jd}, p_{gd})$$

- $P(x_{jd}(t) = 1)$ is the probability that individual j will choose 1, whereas probability of making the choice zero is *1 P*, for the bit at the dth site on the bit string

- x_{jd} (t) is the present state of the bit string d of the individual j

36

- t represents the current time step, and t 1 is the previous time step

- $v_{jd}(t-1)$ shows the measurement of the individuals tendency or present probability of deciding 1

- p_{jd} will be the best state or best choice taken so far, for example, its value will 1 if the individuals overall best success occurred when x_{jd} had the value of 1 and obviously zero if it was zero

- p_{gd} shows the social, neighbourhood, best, and again it will have the value of 1 if the best success achieved by any member of the neighbourhood was 1 or zero in the other case

Here we can state that overall decision is stochastic and all the forces are involved in taking any choice and it is hardly possible, or we can say impossible that any choice taken is based only on isolated facts and effecting directly to that decision all alone. More the randomness will make the exploration of new likelihood and possibilities, and by reducing this randomness will obviously force the members to settle for the best values achieved so far. So we have to create a balance among these two modes of search so that exploration of new possibilities and convergence in results, both should be there simultaneously.

2.5 Swarm and Particles

The swarm term used by many researchers in literature, like Millonas [94], he developed this models in such a way that it can be used in applications of artificial

life, and expressed five basic rules of swarm intelligence. First of these rules, is the immediacy principle, that population should be carried in easy way and time computation. Second rule is related to quality, that every member of population should respond to some quality factors, which will be defined in the surroundings. Third rule is related to various responses, that the members of population should not take their activities along extremely constricted channels. Fourth rule ensures the stability, that population members should not vary their mode of actions each time the surrounding change. The last rule is related to adaptability, that there should not be stiff behaviour in any member of the population and every member should be able to vary mode of its behaviour when it is worth to computational cost. It is clear that fourth and fifth rules are totally two opposite concepts but both have their own significance in specific situations.

The concepts of particle swarm optimization and all its details which have been presented here seem to stick to all five rules. To create the relation of these rules to all the discussion have been made so far, we proposed that the population members will respond to the quality factors *pbest* and *gbest*. The provision of responses among *pbest* and *gbest*, will be carried out. Every population member will change the mode of its behaviour, only when *gbest* will vary, so this proposition is similar to stability. The population members are adaptive in nature because it varies only when *gbest* will vary.

The term *particle*, which have been picked by researchers here is like a compromise, as it can be said here that the population members do not have mass or

volume, so they should have been stated as points. In this case the velocities and accelerations, which have been used many times here, are more fittingly related to particles instead of points, this will be appropriate even if every member have randomly small mass or volume. Also, Reeves [95] explained system of particles as members of clouds which models to disseminate some rules such as clouds, fire and smoke. Therefore the tag which have been chosen here to represent this optimization concept is **particle swarm**.

2.6   Swarm Intelligence

We have explained so far, particle swarm with the references of animal and human behaviour, of both kinds social and cognitive. Particle swarm is mainly lie under the category of swarm intelligence. In swarm intelligence there will not be any centrally controlled mechanism, means no one will be giving the orders to anyone, which actually happens in particle swarm, where every agent or particle will take the decision on the basis of some local information. And the overall swarm will be able to perform the different complex tasks which obviously cannot be done by any individual particle, because it also requires some social interaction to perform such complex tasks. This interaction among the particles will give the swarm a complex structure, but this will be required in order to tackle complex optimization functions.

The idea of swarm intelligence was also first purposed by Kennedy and Eberhart [64], in which they defined five basic working principles of swarm intelligence,

and following are these five principles,

1. Proximity principle: Simple and space, all types of calculations should be carried out by this swarm.

2. Quality principle: Swarm should vary its parameters in beneficiary way, in order to maintain some quality factors, as environment change.

3. Diverse response principle: The swarm should not be spending too much time during its processing at channels which are narrow.

4. Stability principle: In order to ensure the stability, swarm should not change its behaviour every time when environment change, until its beneficial.

5. Adaptability principle: But when it is worthy enough, with respect to computational price, swarm must change its behaviour.

And our PSO follows all these five principles, so we can entitled it as a intelligent system.

2.7   Comparison between GA and PSO

A chromosome of a population in genetic algorithm is similar to a particle in PSO, both particle and chromosome are representation of potential solution for any given optimization problem. As in genetic algorithms, the main operators like crossover and mutation can be implemented in different ways, similarly in PSO different operators can also be implemented in different ways. Although these

operators in both techniques are not the same, but still there are some similarities depending upon the implementation of operators. This association is normally affected by this reality that, effect of operators in genetic algorithm frequently changes by the time.

For example if we consider crossover operator, then the effect of it can vary with time. At the initial stages of formation a generation the population members are randomly initialized, then by applying this operator, the new born chromosomes may diverge in the solution space, but it is not the case every time, some new chromosomes can get a place near the optimized solution, and it also might be the case that some chromosomes will go out from the search space. Therefore in the end of formation of a new generation, the population members will converge towards the optimum solution, even if not every member, but most of them will have almost same structures. If we use the crossover operator less then effect on the population members of next generation will be minimum, so it means in order to optimize our problem we have to vary the crossover probability, and we have to keep it large in the start of simulation so that it can explore the whole search space, and should not get stuck in some local value. As process reaches towards end then this probability should be small so that it should converge [93].

As PSO do not have this operator, crossover, but this phenomenon is done in PSO by making sure that each particle of search space should be accelerated stochastically in the direction of its personal prior best position, so far, and also in the direction of global best position. The behaviour like crossover operator can also

be found in the particles of PSO, like the particles which emerge in the region of halfway between swarms of particles and the particles which cluster around local best positions or among consecutive global best positions. It looks like those particles are doing exploration, even for a while, in a section that represents the geometric mean of two prominent regions. And this geometric mean is basically the main analogy, which makes PSO similar to the crossover operator in genetic algorithm [96]. Normally, in genetic algorithm the crossover operator is used between arbitrarily chosen population members, which are the main reason that the progression of any specific population member implies the exchanging of genetic trait with other arbitrarily chosen population member. Whereas in PSO, any member or particle will not swap traits with other potential particle, but its movement will be influenced by the movement of the other particle, towards the best positions. Hence in this way, the movement of any specific particle will be inclined by its own prior best location and also by the global best location.

Mutation has opposite effect as compare to crossover operator in genetic algorithm. Mutation will have less effect at the starting of the simulation but it will have a very pronounced effect in the end. And the reason behind this is that in the start population will be generated arbitrarily so at this stage even we flip a bit, it will not alter the chromosome with greater effect as compare to that of in the end when even a single bit flip will cause an evident change in chromosome because at the end these chromosomes were converging towards some optimum value, so change at this point will show some large effects. Again it all depends

upon the requirement of the problem, to select the value of mutation rate, but usually this rate kept small in start and high in the end of run. Now there exist the analogy between mutation operator and the velocity of particles in PSO, as every particle in PSO will have some specific velocity, so we can say that the mutation like conduct is directional in PSO. In PSO the distance between the present location of the particle and the best location of that particle, achieved so far, is similar.

Hence this discussion proves that the effectiveness of these both techniques depend on the application in which these will be used. Still PSO is better than GA in a sense that here system is initialized by random solutions for the optima by updating generations but no mutation or crossover, here potential solutions, particles, fly through the problem by following the current optimum particle. The very important advantage of PSO over GAs is that it has memory, means each and every particle will remember the best position, means the best solution, it has achieved so far, and also it will remember the best position or solution of the group, known as global best. PSO is also more suitable to handle time varying problems. PSO has another advantage over GAs, that the number of members of population will remain fix through whole simulation, so different operators to apply on the population will not be required, in which process might become slow. Therefore we can say that in PSO algorithm we apply the productive collaboration among particles, which is not the case of other different artificial algorithms where the main idea is that only fittest members will survive [65].

CHAPTER 3

BLUEPRINT OF PSO

ALGORITHM

3.1   Introduction

In this chapter, the formation of Particle Swarm Optimization algorithm from its very early stages to its up to date changes in its basic structure, will be explained. The algorithm of PSO was originated from basic displacement equation of physics, so it will be explained first then there will be detailed overview about all the changes, which were afterwards made in to it, in order to improve the simulation results. In order to strengthen our argument we will also explain every change, which had been made in to it, by simulation different scenarios in both linear and non linear environment.

3.2   From Physics to PSO

First we will give the brief idea of the algorithm of Particle swarm Optimization, then we will show that how it has been derived from physics.

3.2.1   Basic Algorithm

As stated earlier that in PSO we generate a random potential solutions of any optimization problem and these solutions will known as particles. Like any other optimization problem it will be having an objective or cost function, which will be applied on particles, afterwards we will look for the minima among all particles that will be known as global minima or gbest. Then every particle will be analysed separately and we will see its minimum value on all of its so far iterations, means every particle will have a memory and it will be storing the value of it before applying the cost function on it. This value will known as local best or pbest, and then we will keep on changing the positions of particles in such a way that they approach towards an optimized value.

3.2.2   Derivation of Algorithm from Displacement equation

As we all know that the displacement equation from physics as follows,

$$\vec{x} = \vec{x_0} + \vec{v_0}t + \frac{1}{2}\vec{a}t^2 \tag{3.1}$$

Where $\vec{x}$ is new displacement and x_0 is previous position and v_0 is previous velocity, and here we will assume constant acceleration a overtime period t.

For the iteration version,

$$\vec{x}(k+1) = \vec{x}(k) + \vec{v}(k) + \frac{1}{2}\vec{a}(k)$$

We will be using time index k as previous iteration and k+1 as present iteration, and t will be the time difference of it so,

$$t = (k+1) - k = 1$$

That shows how t eliminated from equation.

As in the start we stated that random particles will be generated, so we need to index them. For any ith particle, the equation 3.2 will become,

$$\vec{x_i}(k+1) = \vec{x_i}(k) + \vec{v_i}(k) + \frac{1}{2}\vec{a_i}(k) \tag{3.2}$$

And all particles will follow the same rule.

These particles or potential solution to any given optimization problem will have the social influence and of cognitive, self intelligence, influence, and on the basis of these two things these all particles will decide for their future decisions. As we are explaining it in terms of physics so we will denote this influence term by acceleration of the particles and there would be two types of accelerations,

46

- Cognitive acceleration

- Social acceleration

First for cognitive acceleration, it will be having the influence of two things, or it is proportional to two parameters, first is its position difference from its local best so far it has achieved, also known as *pbest* or $\vec{p_i}(k)$, that is $\vec{p_i}(k) - \vec{x_i}(k)$, and second parameter is cognitive acceleration constant denoted by c1.

For social acceleration, it will be having the influence of two things as well, or it is proportional to two parameters, first is its position difference from global best so far whole population has achieved, also known as *gbest* or $\vec{g_i}(k)$, that is $\vec{g_i}(k) - \vec{x_i}(k)$, and second parameter is the social acceleration constant denoted by c2.

Hence the total acceleration will become,

$$\vec{a_i}(k) = c_1 * (\vec{p_i}(k) - \vec{x_i}(k)) + c_2 * (\vec{g_i}(k) - \vec{x_i}(k))$$

By substituting this value in equation-3.2

$$\vec{x_i}(k+1) = \vec{x_i}(k) + \vec{v_i}(k) + \frac{1}{2}\left(c_1 * (\vec{p_i}(k) - \vec{x_i}(k)) + c_2 * (\vec{g_i}(k) - \vec{x_i}(k))\right)$$

In order to introduce the stochastic behaviour in the algorithm we will introduce pseudo random numbers in each dimension with expected value of one half and ranging from zero to one, and we will replace these random variables in equation 3.3 in place of,

$\frac{1}{2} * c_1 = c_1 * rand()$ and $\frac{1}{2} * c_2 = c_2 * rand()$

In this case the equation-3.3 will become,

$$\vec{x_i}(k+1) = \vec{x_i}(k) + \vec{v_i}(k) + c_1 * rand()(\vec{p_i}(k) - \vec{x_i}(k)) + c_2 * rand()(\vec{g_i}(k) - \vec{x_i}(k))$$

Similarly we will from the velocity update equation, as all particles have to move towards a optimal solution with a specified velocity so similar update will be done for velocity update, and that equation will be,

$$\vec{v_i}(k+1) = \vec{v_i}(k) + c_1 * rand() * (\vec{p_i}(k) - \vec{x_i}(k)) + c_2 * rand() * (\vec{g_i}(k) - \vec{x_i}(k))$$

If we consider previous velocities as well, which sometimes grows too larger and our particles might flew out of bound from the domain or search space. In order to avoid this situation a new variable named *inertia weight*, which will control the position and velocity of all the particles to remain in some specified bound, will be introduced. Hence by introducing this weight factor w in to above equations, our velocity and position update equations will become,

Velocity update equation:

$$\vec{v_i}(k+1) = w\vec{v_i}(k) + c_1 * rand() * (\vec{p_i}(k) - \vec{x_i}(k)) + c_2 * rand() * (\vec{g_i}(k) - \vec{x_i}(k))$$

Position update equation:

$$\vec{x_i}(k+1) = \vec{x_i}(k) + \vec{v_i}(k+1)$$

So for every iteration we will update these two equation for whole population in each dimension.

3.3   Multidimensional PSO

We have derived PSO for only one dimension of each particle, but in most practical cases we have to search for an optimal solution while considering many aspects of that single solution. Therefore every single particle have to optimize itself in multi dimensions and the update equation for our particles will be multi dimensional. Let the ith particle have D dimensions, then it will be represented by,

$X_i = (x_i1, x_i2, x_i3, ..., x_{iD})$

Here every particle will follow its coordinates in multi-dimensional space, which were related to the most fit solution achieved so far. The fitness value, which have been achieved so far, of any specific particle i, known as (*pbest*), is also stored as,

$P_i = (p_i1, p, p_i3, ..., p_{iD})$ As it has been stated earlier that PSO also keeps tracks globally, so it will also keep on calculating the most best value globally, and that value is known as (*gbest*), and also location of this value, which have been secured by any particle in the whole population. At every iteration, PSO will keep on varying the velocity of each and every particle in the direction of its *pbest* and

gbest, by following formula [97],

$$v_{id} = w * v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}), \quad 1 \leq d \leq D$$

Here w is the inertia weight,c_1 and c_2 are acceleration constants, rand( ),is used to introduce the stochastic behaviour in it, is uniformly distributed random number between zero and one, and $pid = pbest$ and $pgd = gbest$.

And position update equation will from above given detail will be,

$$x_{id} = x_{id} + v_{id} \quad 1 \leq d \leq D$$

3.4 Effect of different parameters on algorithm

In this section we will discuss effects of all the parameters used in PSO, like inertia weight w and constants $c_1 and c_2$ etc, on the performance and efficiency of the algorithm. This discussion will give a brief idea of the basics of PSO algorithm and effect of parameters on it. The inertia weight w here controls the momentum expression,$w * v_{id}$, which denotes the effect of preceding velocity of the ith particle on its present velocity. In other words, the large value of w will make present velocityv_{id} large and due to this particles will search extensively in solution space which will help to find out the new solutions in the search spaces, but it will make the convergence rate slow. On the other hand if we keep the inertia weight

small then momentum term in the formula will be smaller which will be useful to discover solution in the present search space. This concludes that bigger the inertia weight will make particles to discover more solutions in the whole solution space, but on the cost of slow convergence rate and for smaller values of it, the convergence rate will be higher [98].

The rate, at which particles move in the direction of their local best values, is controlled by acceleration constant c_1, and c_2 is the acceleration constant which will control the movement of particles in the direction of global best values. By taking $c_1 = 0$, every particle will get global experience only, which means each and every particle will not have any cognitive control but will be effected by social weight only, and all particles will move freely in a swarm with the less probability to reach a global solution. If we make $c_2 = 0$, then every particle will endure only self experience, means it will make the decisions only by cognitive sense. The convergence rate in this case is quite higher but there will be higher possibility that particles will be trapped in to some local optimum value. And when $c_1 = c_2 = 0$, then all particles will not be having any kind of social or cognitive experience, and this will make the some sort of disordered movement of particles in swarm. Therefore a trade off has to be done among these parameters. In order to make sure that all particles should not fly out from the solution space, the velocities of every particle will be constrained in some selected range [vmax, vmax], which will be problem dependent. The vector of velocity for the ith particle will be shown as,

$V_i = (v_i1, v_i2, v_i3, ..., v_{iD})$ We will give some random weight to acceleration, and these random numbers will be produced separately, due to which particles will move towards *pbest* and *gbest*.

So far we have explained the basic PSO algorithm, in next section we will show some simulation results of previously used PSO algorithms and of LMS, for application of channel equalization, then we will identify the problems in the simulation results and propose the effective solution to those.

3.5 Modifications in PSO algorithm

As effect of different parameters on PSO algorithm has been explained, so different changes have been made in algorithm by altering different parameters. Here we will explain these two types of PSO algorithm, which are,

- PSO-W (using linearly decreasing inertia weight)

- PSO-CCF (using constant constriction factor)

- PSO-VCF (using variable constriction factor)

We will briefly explain these all types and in order to ensure that we will apply changes on perfectly working algorithm, we have to implement these algorithms first. Therefore we will implement these algorithms on any deterministic function just to see that it is working properly.

3.5.1 PSO-W

Here the first change in algorithm will be made, by making its inertia weight as a function of time. This first improvement ensures that in less number of iteration, process reach to its optimized value. The tuning of inertia weight will decide the performance of this customized algorithm. A linearly time decreasing inertia weight was proposed in this method [72] and it has been evaluated as well, by the following update equation 3.3,

$$w_n = (w_i - w_f) * \frac{m - n}{m - 1} + w_f \qquad (3.3)$$

Where w_i is initial weight, and w_f is final weight, m is maximum number of iterations and n is iteration variable index. Whereas position update equation will remain the same but velocity update equation will become,

$$v_{id} = w_n * v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}), \quad 1 \le d \le D$$

$$x_{id} = x_{id} + v_{id} \quad 1 \le d \le D$$

As explained earlier that inertia weight term is used to control the effect of velocities of previous particle on both the present velocity and the exploration of local and global solutions in search space. Therefore it can be stated here that the motive for using a linearly time decreasing inertia weight parameter, which is

large values of inertia weight will be used in the start of the simulation to make sure that the particles should discover globally the search space, and in the end of simulation assign it smaller values to make sure that at the end of simulation particles should explore only locally around a globally optimum value, and should converge on it.

3.5.2   PSO-CCF

Afterwards different literature were proposed to tackle the convergence rate and other problems of this algorithm and then instead of using inertia weight, constriction factor was used to make sure that PSO algorithm should converge in less number of iterations. A basic constriction factor was anticipated in [72], which is shown in equation 3.4,

$$K = \frac{k}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|} \tag{3.4}$$

Where k = 2, ϕ = c1 + c2, and $\phi > 4$. And our velocity update vector will become,

$$v_{id} = K \left[v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \right], \quad 1 \leq d \leq D$$

Here position update equation will remain the same. An analysis has been done in [72], for comparison of PSO using constriction factor and using linearly time

decreasing inertia weight. This analysis proves that the most suitable approach is, to use PSO using constriction factor and limit the maximum velocity *vmax* the variable range of the variable *xmax* in every dimension.

3.5.3 PSO-VCF

Many researchers started work on constriction factor, once faster convergence rate was achieved through this method. An idea of using variable constriction factor, instead of constant constriction factor, was proposed in [98]. This variable constriction factor is shown in equation 3.5,

$$k_n = k_{min} + (k_{max} - k_{min}) * \frac{m - n}{m - 1} \tag{3.5}$$

Where m is the maximum number of iterations and n is variable iteration index. The position update equation will remain same and velocity update equation is shown in following equation,

$$v_{id} = k_n \left[v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}) \right], \quad 1 \leq d \leq D$$

We will apply these all algorithm to any deterministic function, a simple non-linear function which is shown in equation 3.6,

$$f(x) = x^2 \tag{3.6}$$

And the simulation result is shown in figure 3.1.
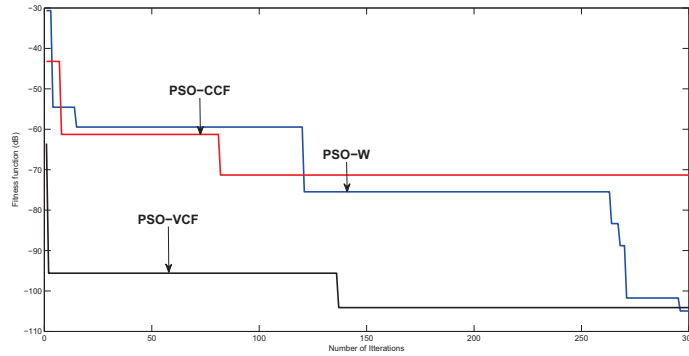
Figure 3.1: Simulation results of PSOW, PSO-VCF and PSO-CCF using deterministic function

From figure 3.1, it is evident that there is an improvement in convergence while using PSO-CCF, although steady state error is increased but still the purpose of getting higher convergence rate was achieved while using PSO-CCF. And with PSO-VCF we had both, improved steady state error and better convergence rate, at same time. Also this figure confirms that these algorithms are working properly and we can apply these algorithms on specified application.

3.6 Comparison of simulation results of PSO and LMS Algorithms

In order to compare the efficiency of PSO algorithm with respect to LMS algorithm, we will use the application of channel equalization. We will perform the channel equalization by first using LMS and then by using PSO, afterwards we

will compare the different versions of PSO through this application. For PSO-W, PSO-CCF and PSO-VCF, the optimal parameters will be, Xmin=-2, Xmax=2, the number of particles for every run is 40, input window size N will be 200, and number of iterations will be 500 and results will be averaged over 20 runs. For remaining parameters, for PSO-W, Vmax=0.07Xmax, Wmin=0.6 and Wmax=1, c1=c2=1.5. For PSO-CCF, Vmax=0.20Xmax, c1=c2=4, k=5, Vmax=0.20Xmax. For PSO-VCF, $c1 = c2 = 4$, kmin = 4, kmax = 6, and vmax = 0.20xmax. For LMS the step size will be 0.025 [98]. Two linear time-invariant channel models are used in the simulation and are described by their following transfer functions $H1(z) = 0.2602+0.9298z1 +0.2602z2$, and $H2(z) = 0.408 + 0.816z1 + 0.408z2$.

Simulation result comparison of LMS algorithm with PSO-W, PSO-VCF and
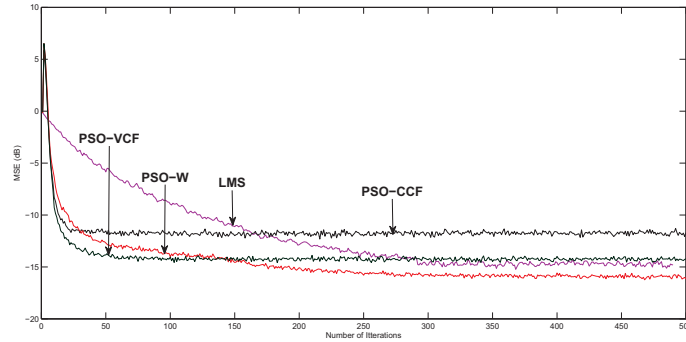
Figure 3.2: MSE curves for PSO-W, PSO-CCF, PSO-VCF and LMS using H1(z)

PSO-CCF are shown in figure 3.2 and figure 3.3. It is evident from these figures these PSO algorithms showed superior performance over LMS. If we are concerned about convergence rate then PSO-CCF is the algorithm of choice but steady state error is higher in this case, so it will be application dependent to choose any spe-
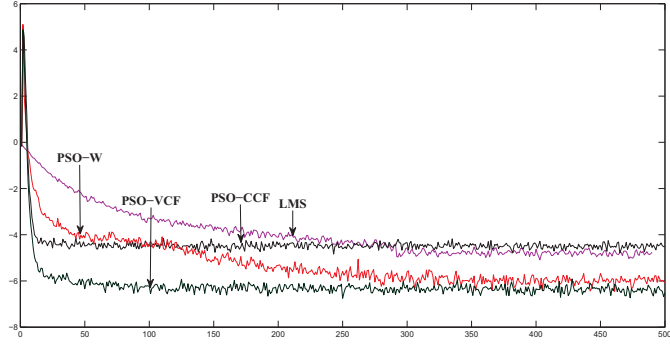
Figure 3.3: MSE curves for PSO-W, PSO-CCF, PSO-VCF and LMS using H2(z)

cific algorithm.

Previously shown simulation results were using linear systems, now we will compare the simulation results using non linear system which is shown in figure 3.4.

Where b1=1, b2=0.1, b3=0.05, and the channel used is H1(z) = 0.2602+0.9298z1
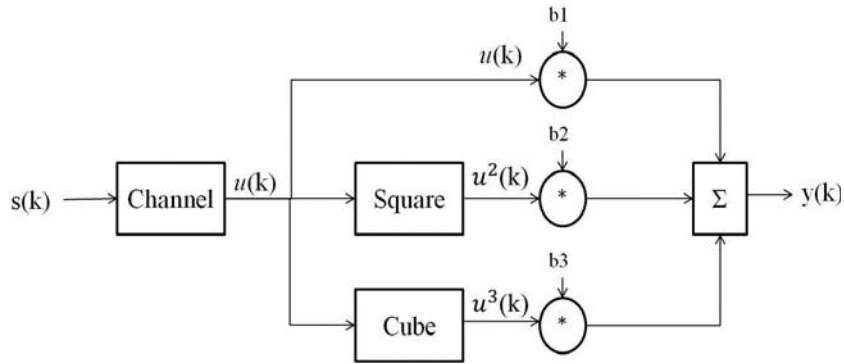
Figure 3.4: Nonlinear system

+0.2602z2.

Figure 3.5, shows the simulation results of comparison among these algorithms using nonlinear system, and here we can observe that now the improvement in steady state error is more for PSO as compare to the linear systems, which is

58
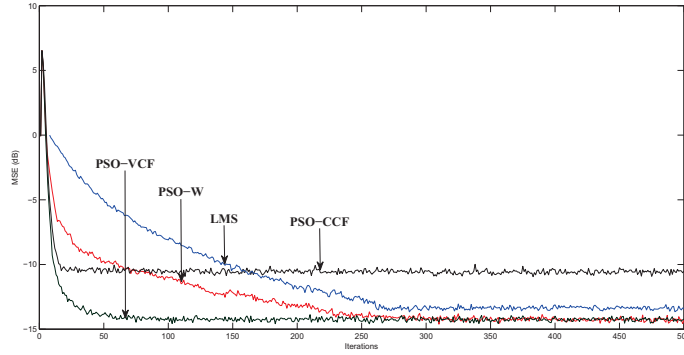
Figure 3.5: MSE curves for PSO-W, PSO-CCF, PSO-VCF and LMS using non-linear system

also due to the fact that all PSO algorithms perform better for nonlinear systems. And in most of the cases, in practical applications, we face nonlinear conditions. So PSO algorithms are more suitable option in such scenarios.

3.7 Conclusion

In this chapter we explained the working of PSO algorithms, right from the beginning. Then we explained all the modifications, which have been done with it. And after that we did the simulation based comparison of conventionally used algorithm LMS, with the so far used PSO algorithms for adaptive equalization. We did this comparison while using both linear time invariant channels and nonlinear systems. And through all these simulation results it can be concluded here that all these PSO algorithms showed better performance with respect to convergence rate as compare to LMS, in both linear and nonlinear system.

59

CHAPTER 4

IMPLEMENTATION OF PSO ALGORITHM BY ADAPTIVE INERTIA WEIGHT

4.1 Introduction

In this chapter, problems in all the previously used algorithms for adaptive channel equalization will be explained, for which the convergence rate or steady state error got effected. Then we will explain PSO algorithm with adaptive inertia weight and then applied to real scenarios. Simulation results presented here, showed the superiority of this algorithm over others. Since PSO more vulnerable to instability as compare to LMS, therefore to achieve stable and best performance, it is necessary to optimize every PSO algorithm with respect to every parameter involved in it. Therefore a sensitivity analysis for this algorithm will be carried out.

As steady state MSE is a good metric for ensuring the performance of algorithms under test, but still Bit Error Rate (BER) analysis is used for comparison with LMS.

4.2   Problems With The Conventional PSO Algorithms

In conventional PSO algorithms, from the simulation results of PSO-W, PSO-VCF and PSO-CCF provided in the previous chapter, it can be observed that it possess a disadvantage. Which is the stagnancy of whole population around any specific point, and due to this there is chance that it might search only in that specific region. This will be excellent if that point is optimum one, but if that point is local minima, then the probability that whole population will remain around that local minimum value will be higher. Sometimes such scenarios severely damage the results.

There are two main parameters through which the performance of any algorithm can be evaluated, first one is convergence speed or rate and other is search capacity. From the simulation results shown in figures-3.2, 3.3, 3.5, it can be observed that results using PSO-W and PSO-VCF, gave better results with respect to search capacity, but the convergence speed is slower. Whereas results achieved through PSO-CCF, showed great enhancement in convergence rate, but the error is quite higher which shows that the search capacity of this algorithm is not up to mark

as compare to PSO-W and PSO-VCF. And the convergence rate of PSO-W and PSO-VCF is less as compare to the convergence rate of PSO-CCF. Although these all PSO algorithms showed better results, with respect to convergence rate and search capacity, as compare to LMS.

Hence it can be stated that there is an issue of search capacity enhancement and convergence rate, with these two PSO algorithms. Now we will explain, how these two issues are causing the degradation in these algorithms and then their remedy will be explained.

4.2.1 Search Capacity Enhancement, Expected Problems and Solutions

When a new gbest is found all particles will start to move towards it in same general direction and due to this there is the chance that some regions, other than this new minima discovered, will be excluded from the search space. Particles closer to gbest will tend to converge on it in very short time and then there will be no update in their position or velocity, as they have already approached to optimum value, so these particles will become stagnant and will not contribute further in search. If we have very irregular surface with many local minima, then there is the chance that particles will get trap in to some local value, and they will never get the chance to approach toward the global optimum value.

So far the expected problems related to search capacity enhancement has been explained, now different possible solutions will be explained for this.

One possible solution is, increase the acceleration coefficients. In this way the acceleration will be increased and swarm will get more chance to explore the search space. As acceleration of the particles increases, the probability that particles will be trapped in to some local minima, will be reduced as well. Because these particles with higher acceleration, will not settle down quickly so the chance of getting trap in some local value will be minimized. As nothing comes free here, so due to this increased acceleration of the particles, the swarm will take more time to converge on some value and there will be a degradation in the performance regarding convergence rate.

The problem of stagnancy of the particles can be minimized if different parameters of particles can be varied after some specific number of iterations. This will have almost similar effect as created by mutation in genetic algorithms [87]. Although this mutation in the particles will have slight effect on the distance of particles from gbest. Because, the effect on the particle distance caused by velocity update equation is greater than this mutation, variation in different parameters of particles. Still it will degrade the convergence rate, which is highly undesirable for PSO algorithms, because the main concern of this PSO algorithm to have higher convergence rate as compare to other conventional algorithms.

The major source of degradation in search capacity is due to premature convergence of the swarm. In previously proposed solution mutation done over whole swarm of the particles, but it can be modified by changing the parameters for only portion of the particles. This variation can be carried out for some specific

number of the iterations. In this scenario we have to consider other issues as well, like to keep the track of the particles in such a way that these variations will be effected only in the start but after some time they should converge, so we have to evaluate all of them continuously for every iteration which will increase the number of operations, which is also not desirable.

4.2.2 Convergence Speed Enhancement, Expected Problems and Solutions

Degradation in convergence rate can be due to the higher acceleration of the particles, in this way they explore search space extensively but on the cost of slower convergence rate. Convergence rate also degraded due to loose conditions on the velocity and the range of the allowed position of the particles. This increase in convergence speed can be done by optimizing the acceleration constants or inertia weights or constrict the search space, and this can be done by replacing all or some of the pbest with the pbest which have the better fitness function value. By doing this so we are actually making the search confined by only concentrating in the area of interest. But for this method to work, we have to be sure that our area of interest includes the optimal value, otherwise there will premature convergence and degraded performance might happen [87]. And this evaluation has to be tested for every iteration that our swarm still includes the area of interest of the search space.

In this chapter we will take in to account the problem of convergence rate only,

and our newly implemented algorithm will tackle the issue of better convergence rate with minimum possible steady state error.

4.3 Newly Implemented PSO Algorithm using Adaptive Inertia Weight (PSO-AW)

Some of the techniques were stated above to enhance the convergence speed of PSO algorithm, but all of these have their own limitations. Here we are going to implement PSO algorithm with a new mechanism for adaptive equalization which will enhance the convergence speed of it with minimum steady state error, as compare to previously used algorithms.

The convergence speed of the algorithm usually controlled by the speed at which the particles move towards its best positions, if that best position is optimal for them and these particles move towards that with more speed then obviously, the convergence speed will be enhanced, but if that best position, which any particle is holding, not the optimal one and the speed of particle towards that position is still higher, then obviously it will decrease the convergence speed. Therefore we can conclude that convergence speed can be increased if particles move more quickly towards their best positions, and with less speed towards their non optimal positions. And the idea about that position can be taken through the fitness function, which has been stated in first chapter, equation- 3.3. Speed of the particle is related to inertia weight of the particles, so in order to increase the

convergence rate, we have to vary the inertia weight accordingly by looking at the fitness function of each particle in every iteration.

4.3.1 Functionality of the Algorithm

To implement above mechanism, we can adjust the inertia weight of each particle independently at each iteration, based on their new fitness evaluation, that whether it is better than previous one or not. If a particle attained a better position, then its inertia weight should be increased or maintained but if its present fitness is not better than previous one then there should be reduction in its inertia weight [87]. Through this mechanism we are making sure that particles are moving towards their optimal positions with more speed and for non optimal positions their speed is reduced. Now instead of using inertia weight, we will use **adaptive inertia weight**, means inertia weight is varying adaptively according to the environment. The relation of this adaptive inertial weight is given in equation 4.1 [87],

$$w_i(n) = \frac{1}{(1 + e^{\frac{-\Delta J_i(n)}{s}})} \tag{4.1}$$

Where $w_i(n)$ is the current inertia weight of the ith particle and $\Delta J_i(n)$ is the difference of the fitness values of the particle from its previous one, and can be represented by following equation 4.2, and S is used to control the expected fitness range.

$$\Delta J_i(n) = \Delta J_i(n-1) - \Delta J_i(n) \tag{4.2}$$

This relation of inertia weight will assign the values in the range between (0,1), with midpoint of 0.5. If the previous fitness level was better then it will assign the value less than 0.5, means the speed of the particle should be reduced as the new position is not better than the previous one. And if present fitness value is better then it will assign the value greater 0.5, which means now newly calculated position is better than the previous one so speed of the particle should be increased. This relation of inertia weight has been depicted in figure 4.1, for random data.

Velocity and position update equations can be shown in following equations 4.3
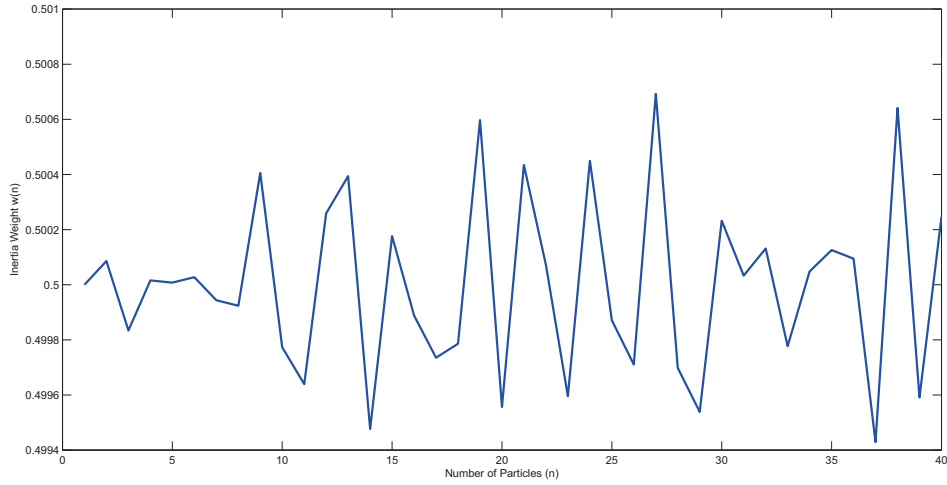
Figure 4.1: Inertia Weight Assignment for Particles

and 4.4.

$$v_{id} = w_i(n) * v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id}), \quad 1 \leq d \leq D \tag{4.3}$$

$$x_{id} = x_{id} + v_{id} \quad 1 \le d \le D \tag{4.4}$$

Where $w_i(n)$ in equation 4.3, is shown in equation 4.1. So through this mechanism we can see that when the fitness function of particles improves the speed of the particles will be increased and it will be reduced when fitness function degraded, so through this the convergence speed will be enhanced. We will name it as **PSO-AW**. Functionality of this algorithm can be explained easily with the help of its flow chart which is shown in figure 4.2. In the start we have to randomly generate
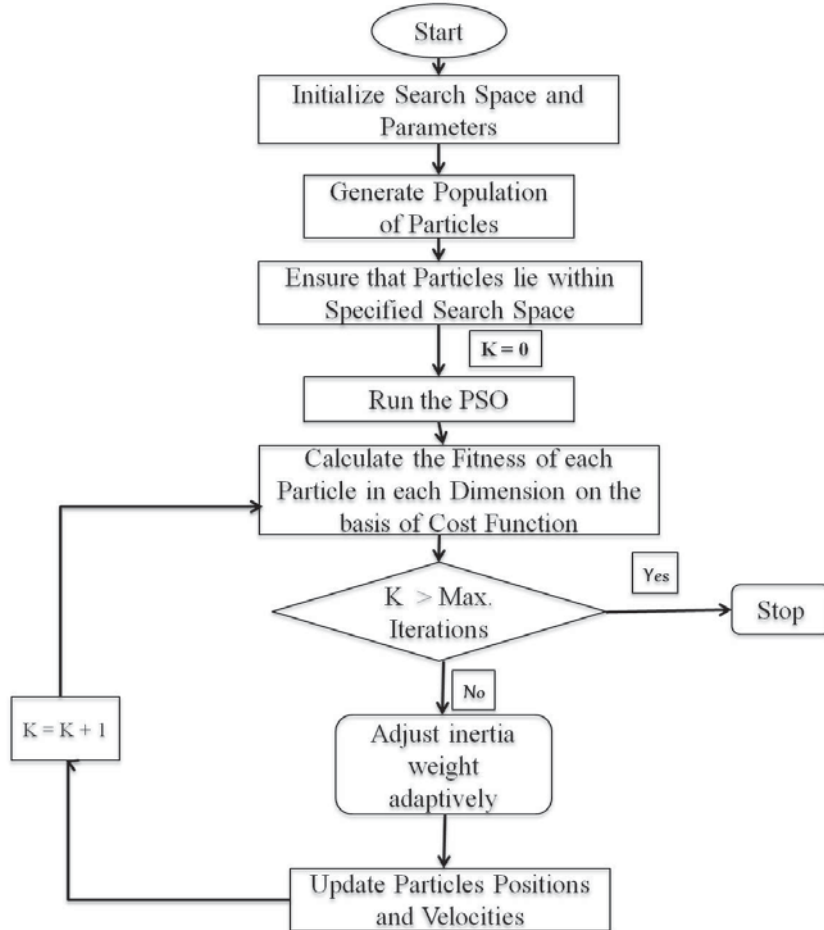
Figure 4.2: Flow chart of PSO-AW

68

all the particles, and we have to define the boundaries of the search space in which particles will move. It should be ensured that particles should lie in the search space so while generating the particles this issue has to be addressed. Then the processing will start, we have to calculate the fitness of each particle through the objective function defined in equation 1.2. then we will check the stopping criteria, if it meet this criteria, we will stop if not then adjust the adaptive inertia weights of each particle. For the first iteration, obviously we will not have the previous fitness of the particles so we will consider the adaptive inertia weight for every particle to be 0.5, and then afterwards we will follow the previously stated procedure. After assigning the adaptive inertia weight to whole population of the particles, we will apply the velocity and position update equation to each particle and this procedure will go on until we reach some stopping criteria.

4.4 Sensitivity Analysis of PSO-AW

There is the issue of stability for every PSO algorithms, means in order to get the proper results from these algorithms, it should be tuned perfectly with respect to every parameter involved in it. In most cases simulations will not give the desired or expected results due to instability of PSO algorithms and the main reason behind this instability is the values assigned to parameters involved in it are not tuned properly. Hence similar to other PSO algorithms, there has to be a complete sensitivity analysis of PSO-AW with respect to every parameter involve in it. And these parameters are, number of particles, acceleration constants, window size,

slope factor, number of taps of the equalizer and maximum allowed velocity to particles. With the help of simulation results, we will find optimized values for all these parameters, in order to make sure that PSO-AW should perform proper functionality in every environment.

We will compare these optimized values of the parameters with the previously used PSO algorithms, used parameters, in order to prove that this PSO-AW algorithm shows better performance.

4.4.1 Effect of the Slope Factor

This parameter S, is basically the slope factor, used in equation 4.1, which is the formula of assigning the inertia weight adaptively to the particles. This parameter controls the expected value of the range for the values which are assigned to the particles. If we are assigning smaller values to this parameter then the inertia weight which has to be assigned to the particles would be taken from a smaller range of values, which will effect the performance of particles.

As it was mentioned earlier that if the inertia weights of the particles are larger then they will get the chance to explore more from search space and steady state error might improve, and for less inertia weights they will not explore the whole search space. Hence this parameter S is behaving almost in a similar way here. If we assign S higher values then sready state error will be improved. For smaller values of S, particles will get less chance to explore the whole search space. Simulation results, shown in figure 4.3, present the effect of different values of

70

parameter S on its MSE curves. The effect on steady state error will be more
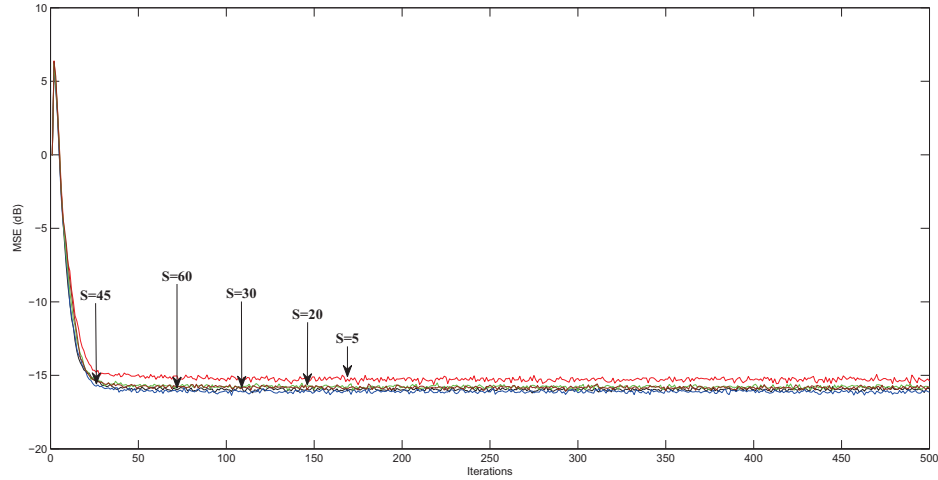
Figure 4.3: Effect of the slope factor, S, on Performance behaviour

pronounced if we are directly varying the values of inertia weights of the particles.

Here from the simulation results shown in figure 4.3, it is clear that this effect on

steady state error is not large. Still for the small values of S, steady state error is

increased. In this simulation MSE curves are plotted for S= 5, 20, 30 45 and 65.

But after S=45, there is not even observable change in the MSE curve and steady

state error. From here it can be concluded that 45 will be optimal value for this

parameter used in PSO-AW.

4.4.2   Effect of the Number of Particles

Number of particles are like number of potential solution in any search space. As

the number of particles increase, the chance of exploring more area of the search

space also increases, and the probability of achieving most optimum value, also

increases. Hence with greater number of particles we can secure better steady state error, as more particles will search the solution space.

Number of particles also have effect on convergence rate as well. As more particles will be involved in the process of searching the optimum value, there will be the possibility that these particles will find the optimum value in less number of iterations by making the convergence rate higher. If less number of particles are involved in the process then obviously it will take more iterations to converge on some optimum value, so it is better to have larger number of particles in search space.

The disadvantage of using large number of particles for search process is more overhead. As process will take more time to generate the results and larger number of iterations will be required to get results. Also after some specific number of particles, the increment in convergence speed, and improvement in steady state error will be negligible. After that there will be not even visible effect on MSE curves. Simulation results shown in figure 4.4, depicts the effect of number of particles on the convergence rate and steady state error. It is evident from simulation results shown in figure 4.4, that after 30 particles there is not consider able change, with respect to convergence rate and steady state error, in the MSE curves. For smaller number of particles like 10 and 20, the steady state error is -10dB and -12.5dB respectively, but with the number of particles 30, 40 and 60 we got almost the same steady state error of -16dB with same number of iterations taken for convergence. As there is no considerable variation with respect to
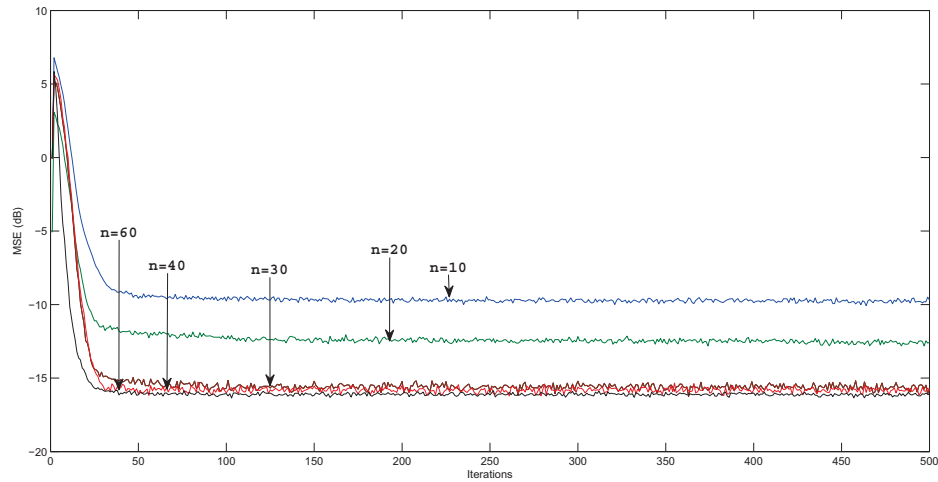
Figure 4.4: Effect of the Number of Particles, n, on the performance behaviour

convergence rate and steady state error after 30 particles so it can be concluded that n=30 particles will be optimum value for this parameter.

In previously used, PSO algorithms, the optimum value for number of particles was 40 [98]. Here in our newly implemented algorithm it takes less number of particles to reach the desired results which shows the superior performance of this algorithm, over conventionally used PSO algorithms. As this algorithm took less number of particles to generate similar results so we are getting here the advantage of fast processing as compare to conventional PSO algorithms.

4.4.3 Effect of the Acceleration Constants

The rate, at which particles move in the direction of their local best values, is controlled by acceleration constant c1. And c2 is the acceleration constant, which will control the movement of any particle in the direction of global best value.

From this it can be stated that these two parameters controls the speed of the particles, through which they move towards their best values achieved so far in the search space. By taking c1 = 0, will make every particle to have global experience only, which means each and every particle will not have any cognitive control but will be effected by social weight only, and all particles will move freely in a swarm with the less probability to reach a global solution. But if we make c2 = 0, the every particle will endure only self experience, means it will make the decisions only by cognitive sense.

These two parameters control both, the steady state error and also the convergence speed of the particles. If we assign smaller values to these parameters then particles will move with slow speed in the search space and explore more solutions. In this way there is quite less chance that they will get trap in to some local optimum value and therefore through this we can achieve better steady state error. By applying this, we are making the convergence speed slower and it will take more time by the swarm to converge on some optimum value. Hence a trade off has to done between better steady state error and better convergence rate.

On the other hand if we assign higher values to these parameters, then particles will move quickly and will settled down quickly on some optimum value. This will make the convergence rate higher but there is chance of getting trap in to some local minimum which will cause the steady state error to degrade. Therefore the values for these parameters will be application dependent.

Simulation results, shown in figure 4.5, present the effect of different values of

acceleration constants on its MSE curves. From the simulation results shown in
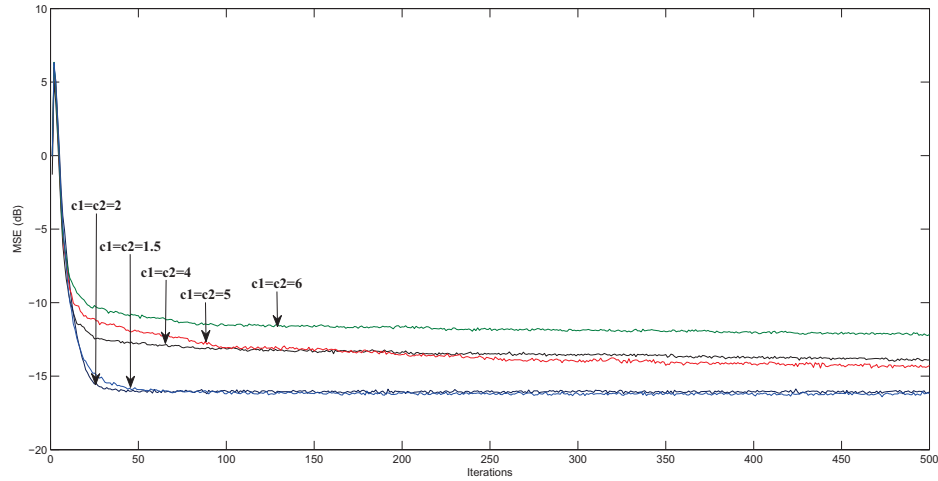
Figure 4.5: Effect of the Acceleration Constants,c1&c2, on the performance behaviour

figure 4.5, it is evident that for smaller values of acceleration constants we achieve

better steady state error. For c1=c2=1.5 and c1=c2=2, the steady state error

is almost -16.5dB. And as we increase in the values of acceleration constants the

steady state error will be increased. Like for c1=c2=4 and c1=c2=5, the steady

state error is almost -14dB and for c1=c2=6 it increases to -12 dB. Hence from

these curves shown in figure 4.5, it is evident that most optimum value for these

acceleration constants is c1=c2=2.

4.4.4   Effect of the Number of Taps of Equalizer

The number of taps of adaptive equalizer is a parameter which is purely application

dependent, and any algorithm modification will have almost negligible effect on

it. Here in PSO, for each algorithm, the number of taps of equalizer is equivalent to the dimensions of each particle. And it required by our newly implemented algorithm to cover more search space so higher value of number of taps should be assigned.

Simulation results, shown in figure 4.6, present the effect of different values of number of taps on its MSE curves. As it is also evident from the simulation

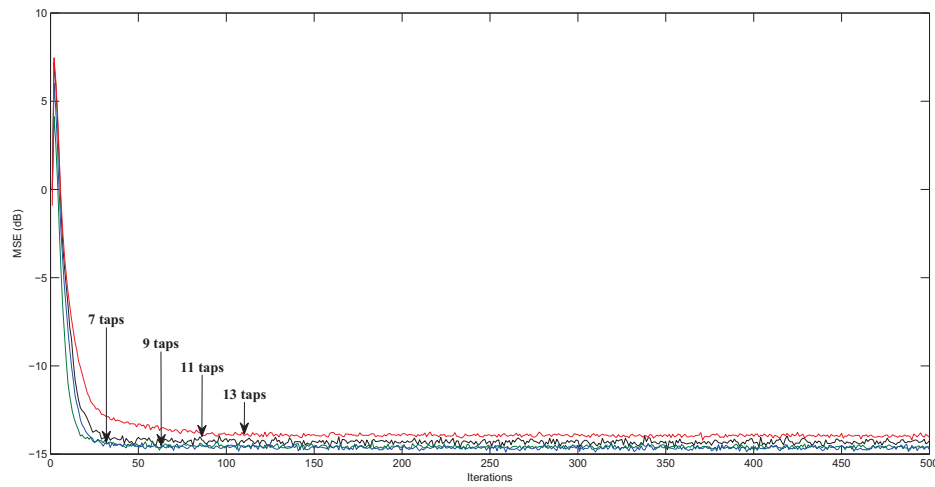Figure 4.6: Effect of the Number of Taps, d, on the performance behaviour

results shown in figure 4.6, that for this same application of adaptive channel equalization, the different values of number of taps have very slight effect on both areas of concern, steady state error and convergence rate. Hence from this simulation result it can be stated that the optimum value for the number of taps of adaptive equalizer will be 9.

4.4.5   Effect of the Maximum Velocity

This parameter of maximum velocity is actually controls the maximum allowed velocity to the particles, through which these particles move in search space. This parameter is related to maximum allowed position to any particle, means this maximum allowed velocity of any particle is equal to some number times of maximum allowed position to any particle, therefore these two parameters have combined effect.

This maximum allowed velocity is the parameter through which particles move towards their global optimum positions. If particles move towards their global optimum values with higher speed then there will be the chance for these particles to search extensively around the global value and the probability of getting better MSE will be higher. Therefore higher values of this parameter will have productive effects on MSE curves by giving minimum steady state error with better convergence. On other hand if we assign lower values to this parameter then there is chance that these particles might get trap around some local minimum values, and they will keep on searching only around that area. And due to this we might get higher steady state error and MSE curve can also take higher number of iterations to converge.

Simulation results, shown in figure 4.7, present the effect of different values of maximum allowed velocity on its MSE curves. From the simulation results shown in figure 4.7, it is evident that greater this parameter, secures minimum steady state error. As minimum value of this parameter, Vmax=0.02Xmax, gives the
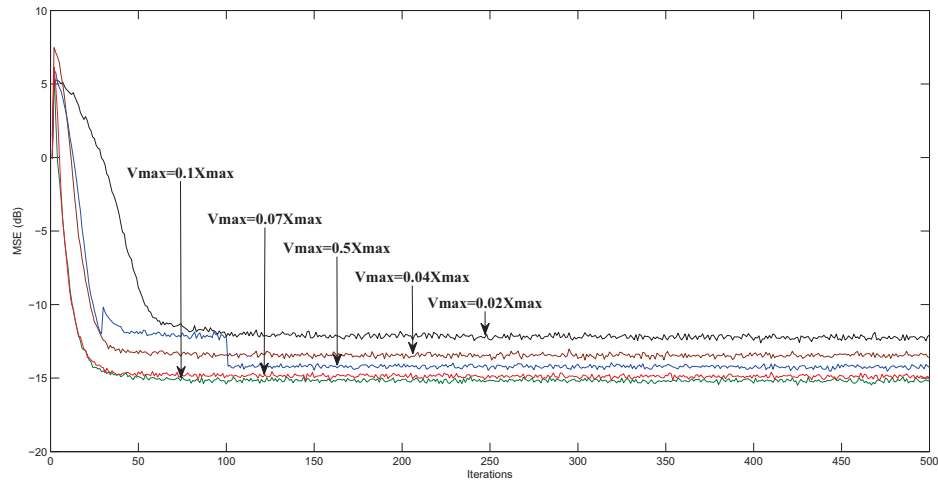
Figure 4.7: Effect of the Maximum Allowed Velocity, Vmax, on the performance behaviour

most degraded steady state error which is almost -12.5dB. By assigning higher values to this parameter we get better MSE curves, with minimum steady state error and better convergence rate. By increasing the value of this parameter after, Vmax=0.1Xmax, there is not even noticeable effect on MSE curves. Higher values were also tried in simulation, but those were not shown in this figure, and those values also gave the similar results. Hence it can be concluded here that the most optimized value for this parameter will be Vmax=0.1Xmax.

4.4.6 Effect of the Data Window Size

In most of the practical conditions we do not have the whole data at the same time. Most of the time we get the data in form of chunks for equalization, and we have to perform this equalization only on the subset of the data, and from that

78

subset we have to find the behaviour of the incoming data. Therefore to make our

simulations more like practical conditions, this parameter will be introduced.

If we assign higher values to this parameter then we are taking more data to

observe the statistics of the complete data set, which obviously will yield the

better results in sense of steady state error. And if we assign smaller values to

this parameter then we have smaller subset of the complete data, to understand

the statistics of the complete data. And in this case we might not get better

steady state error. This parameter has also effect on convergence rate, but this

effect on convergence rate of the MSE curves, is not of great extent.

Simulation results, shown in figure 4.8, presents the effect of different values of

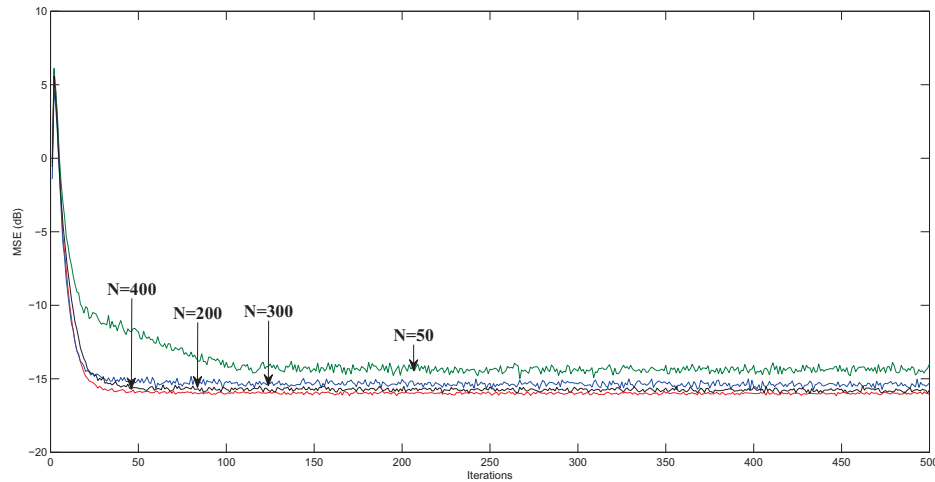data window size N, on its MSE curves. From the simulation results, shown in

Figure 4.8: Effect of the Data Window Size, N, on the performance behaviour

figure 4.8, it is evident that for smaller value of this parameter we have degraded

steady state error and also slow convergence rate. For N=50, it converges at almost

79

120 iterations with steady state error of -13.5dB. As we assign higher values to this parameter, convergence rate and steady state error, both improved. For N= 200, 300, 400, it converges after only 25 iterations with steady error of almost -16.5dB. After N=200, if we keep on increasing the value for this parameter, it will not have significant effect on MSE curves. So it can be concluded here that most optimum value for this parameter is N=200.

4.5 Simulation Results of Adaptive equalization using PSO-AW

Here the comparison of the simulation results for adaptive equalization, using different linear channels and non linear system, at different noise levels, will be conducted. We will compare the MSE for this newly implemented algorithm with the previously used PSO algorithms and LMS.

4.5.1 Simulation Comparison Using Linear Channels

Same previously used two linear time-invariant channels will be adopted here as well, described by their following transfer functions *H1(z)* = 0.2602+0.9298z1 +0.2602z2, and *H2(z)* = 0.408 + 0.816z1 + 0.408z2. The first channel *H1(z)* is more stable as compare to second channel *H2(z)*, because the second channel have more eigenvalue spread and it cause more damage to the signal. Hence we will make the comparison for both kind of environments. Following are the details

of different parameters which will be used in all these algorithms. For every PSO algorithm the optimal parameters will be, Xmin=-2, Xmax=2, the input window size N is 200 and the number of iterations is fixed to 500. For the remaining parameters, for PSO-W, Vmax=0.07Xmax, Wmin=0.6 and Wmax=1, c1=c2=1.5 and the number of particles will be 40. For PSO-CCF, Vmax=0.20Xmax, c1=c2=4, k=5, Vmax=0.20Xmax and the number of particles will be 40. For PSO-VCF, c1 = c2 = 4, kmin = 4, kmax = 6, and vmax = 0.20xmax. For this newly implemented algorithm PSO-AW, S=45, Vmax=0.1Xmax, Wmin=0.6 and Wmax=1, c1=c2=2 and the number of particles will be 30. We found the values of these parameters from the sensitivity analysis done in previous section. For LMS the step size will be 0.025. And these all results will be averaged over 25 runs. The SNR for all these algorithms will be 20dB. The number of taps for the adaptive equalizer will be 9. As it was stated earlier while explaining the functionality of the equalizer that there will be delay of processing after signal pass through the equalizer and in order to compare it properly with the original signal we have to introduce a delay factor in the original signal, and the value of this delay, D will be 11.

Figure 4.9 and 4.10 shows the simulation comparison of all these algorithms, using first and second linear time variant channels respectively. From these two figures 4.9 and 4.10, we can conclude that PSO-AW, showed better results as compare to all previously used algorithms with respect to both convergence rate and steady state error. While using, PSO-W and PSO-VCF, we achieved better
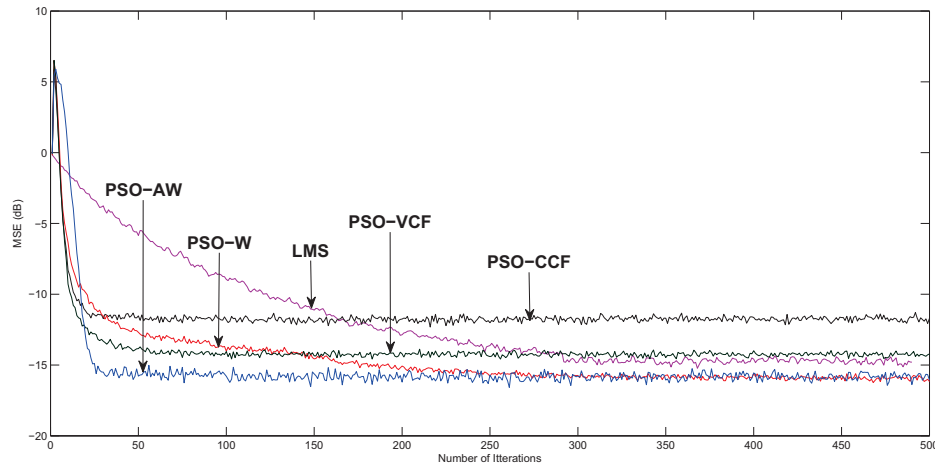
Figure 4.9: MSE curves for PSO-W, PSO-CCF, PSO-VCF, LMS and PSO-AW using H1(z)

steady state error only but with slow convergence rate and while using PSO-CCF we achieved higher convergence rate but the steady state error is highly degraded while using this algorithm. On other hand while using PSO-AW we achieved both higher convergence rate and better steady state error simultaneously, which proves the superior performance of this algorithm over conventional PSO algorithms.

In order to check the proper functionality of this newly implemented algorithm, we plotted it at different SNR values, as SNR increases the steady state error should be improved. The simulation of PSO-AW at different SNR values, while using H1(z), is shown in figure 4.11. From figure 4.11, it can be inferred that, steady state error improves as SNR increase which shows that this implemented algorithm is working properly.
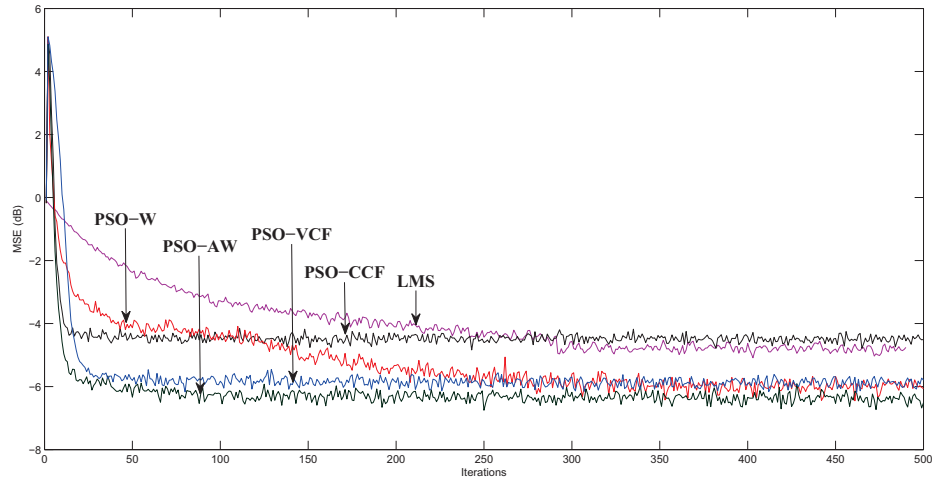
Figure 4.10: MSE curves for PSO-W, PSO-CCF, PSO-VCF, LMS and PSO-AW using H2(z)

4.5.2 Simulation Comparison Using Non Linear Systems

In most of the practical conditions, especially in wireless communication systems, channels do not behave linearly and it is more difficult to tackle the non linear channels. It was stated earlier that all PSO algorithms work more effectively than other conventional algorithms, for non linear channels. In fact the popularity of PSO algorithms for any application was due to the reason that these algorithms perform better for non linear systems. Hence in order to strengthen this statement, simulation comparison has been shown in figure 4.12, of all previously used algorithms with PSO-AW, using nonlinear system which is shown in figure 3.4, in it b1=1, b2=0.1, b3=0.05, and the channel used is H1(z) = 0.2602+0.9298z1 +0.2602z2. It is evident from the results, shown in figure 4.12, that all PSO algorithms shown better performance as compare to LMS, which shows that PSO has
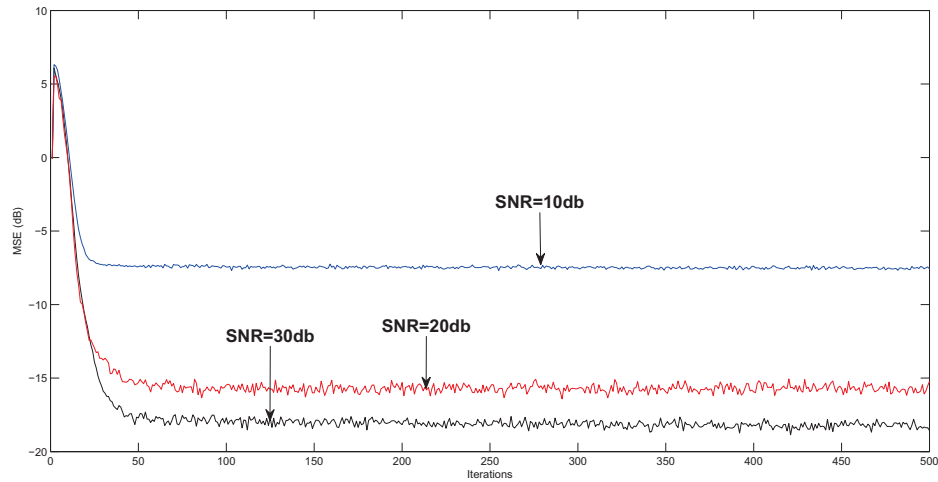
Figure 4.11: MSE curves for PSO-AW at different SNR values

better functionality in nonlinear systems. If we compare the statistics of simulation results achieved from non linear systems with that of linear system, it can be concluded that for non linear channel steady state error of PSO-AW is improved more, almost 3 dB as compare to PSO-W. Here for non linear system, these all PSO algorithms have better steady state error as compare to LMS and among all these PSO-AW have the minimum steady state error of -17dB. Which shows the better performance of all PSO algorithms than other conventional algorithms, and it strengthens the statement of better performance of PSO algorithms in non linear systems.
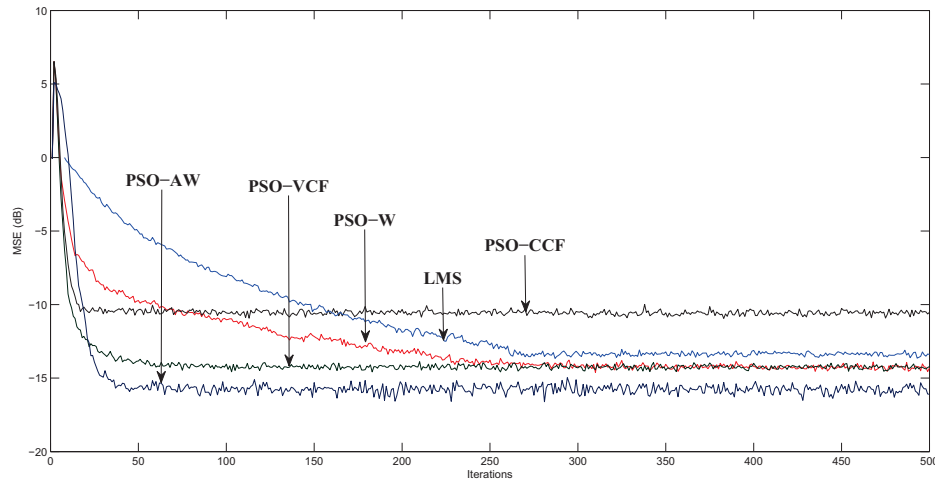
Figure 4.12: MSE curves for PSO-W, PSO-CCF, PSO-VCF, LMS and PSO-AW using nonlinear system

4.6 Comparison of BER between PSO-AW and LMS

The comparison for BER between our newly implemented algorithm PSO-AW and LMS will be done in this section. Among all previously used conventional algorithms like, RLS and Steepest Descent etc, LMS is the most commonly used algorithm. Therefore to make the comparison this LMS algorithm was selected. Also the performance of LMS algorithm is quite stable with respect to many aspects, so if our implemented algorithm shows better performance in sense of BER as compare to LMS, then it means it is a reasonable gain to use PSO-AW. We will compare, with respect to BER, while using both linear time invariant channel and nonlinear system.

85

4.6.1   Comparison of BER using LTI Channel

We will use the same linear time invariant channel which has been used earlier for the comparison of MSE curves. The transfer function of which is H1(z) = 0.2602+0.9298z1 +0.2602z2. Usually the analysis of BER curve is considered to be like this, that as we increase the Signal to Noise Ratio (SNR) then BER should decrease, while making the water fall curve.

Simulation results, shown in figure 4.13, present the effect of SNR on BER curve. It is evident from the results shown in figure 4.13, that as SNR increase BER for
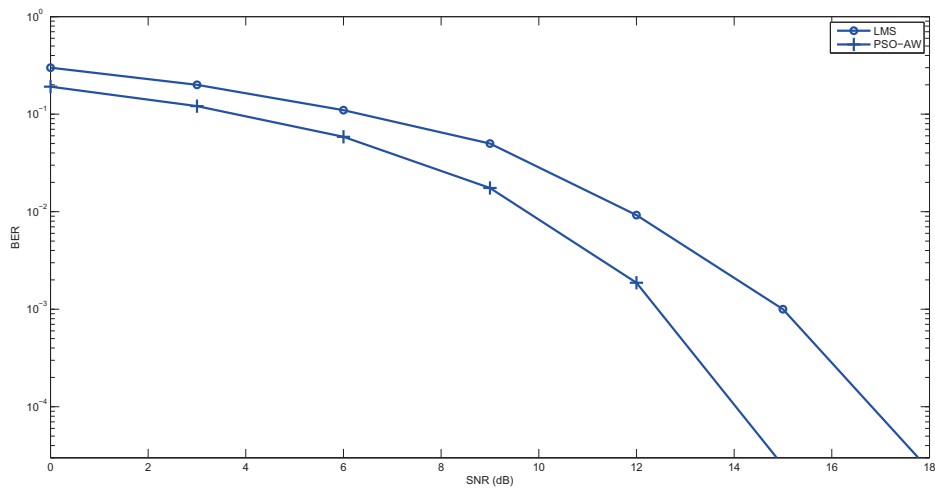
Figure 4.13: BER performance of LMS and PSO-AW while using LTI channel H1(z)

both the algorithms decreases. If we observe the comparison among these both algorithms, the BER while using LMS at SNR=18dB, this same BER is achieved at lower SNR=15dB while using PSO-AW. Means if we use PSO-AW and increase the SNR up to same level of 18dB, then BER will be less as compare to LMS at

that same SNR value. Hence it can be concluded here that while using PSO-AW, we can have less BER at same SNR value as compare to LMS.

4.6.2 Comparison of BER using Nonlinear System

In previous subsection comparison was conducted for linear system, now we will compare the BER performance of these both algorithms while using nonlinear system. The same nonlinear system will be used here, which is shown in figure 3.4, in which values of the coefficients will be b1=1, b2=0.1, b3=0.05, and the channel used is H1(z) = 0.2602+0.9298z1 +0.2602z2.

Simulation results, shown in figure 4.14, presents the effect of SNR on BER curve. From the simulation shown in figure 4.14, it is evident that while using PSO-AW
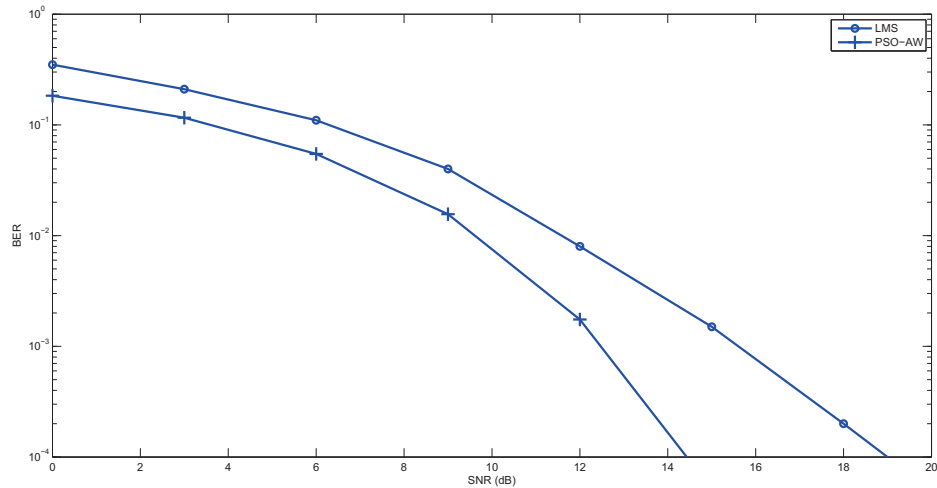
Figure 4.14: BER performance of LMS and PSO-AW while using Nonlinear System

we got better BER at even less SNR as compare to LMS. Like the same value of

BER was achieved, at SNR=14.5dB for PSO-AW, and for LMS we achieved this value of BER at SNR=19dB. It can be concluded here that for nonlinear systems as well, we achieved better BER while using PSO-AW as compare to LMS. From these simulation results, it is also evident that for nonlinear system, we have more improvement in BER as compare to linear system. Simulation results shown in figure 4.13, for linear system, PSO-AW showed improvement of almost 3dB as compare to LMS. Here in figure 4.14, it is clear for nonlinear system, PSO-AW has more than 4dB improvement as compare to LMS. Hence these results again proved that PSO algorithms showed better performance in nonlinear systems as compare to conventionally used algorithms.

4.7 Conclusion

In this chapter we described the main reasons to introduce PSO-AW algorithm, then we completely explained the working of this algorithm with the help of simulation results. Complete sensitivity analysis of this newly implemented algorithm has been performed. Then we compared all the simulation results with previously used algorithms while using linear and nonlinear systems. Analysis with respect to BER was also done. The main reason behind simulating this algorithm is that while using PSO-W and PSO-VCF we achieved better steady state error but convergence rate was not as quick as displayed by PSO-CCF. While using PSO-AW we achieved the same convergence rate like we got using PSO-CCF with the same steady state error as we secured with PSO-W and PSO-VCF. Hence in conclusion

it can be stated that PSO-AW, provides the faster convergence rate like PSO-CCF and improved steady state error like PSO-W and PSO-VCF. And we can secure these both, better convergence rate and better steady state error simultaneously, while using PSO-AW.

<center>

CHAPTER 5

HYBRID PSO ALGORITHM

FOR ADAPTIVE

EQUALIZATION

</center>

5.1   Introduction

Previously, the PSO-AW algorithm was implemented for adaptive equalization and it performed better than previously used algorithms. The PSO-AW algorithm achieves a faster convergence rate but not necessarily an improved steady-state error. Here, we propose a new algorithm that improves the steady-state error as well.

The new hybrid PSO (HPSO) algorithm is a hybrid of three techniques, namely, re-randomization, enhanced social effect and adaptive inertia weight for particles. A detailed comparison of the proposed algorithm with all previous algorithms has

<center>

</center>

been presented in this chapter. A complete sensitivity analysis of the proposed algorithm, with respect to all parameters involved, follows the comparative study. In the end, the BER performance is shown for both linear and nonlinear systems.

5.2   Another Eminent Issue with Previous PSO Algorithms

In previous chapter we have stated some of the issues in conventionally used PSO algorithms. Now we are going to propose a new algorithm, in which we have introduced two newly implemented techniques, re-randomization and increase social effect. Before introducing such techniques, we must know the problems which can be minimized by these techniques.

In previously used PSO algorithms, when a new *gbest* is found all particles will start to move towards it in same general direction and due to this there is the chance that some regions, other than this new minima discovered, will be excluded from the search space. Particles which are closer to *gbest* will tend to converge on it in very short time and then there will be no update in their position or velocity, as they have already approached to optimum value, so these particles will become stagnant and will not contribute further in search. In this way we can face the problem of stagnancy, which might cause in degrading the steady state error.

If the surface or search space in which optimization is to be carried out, have regular shape or very less number of local minimas, then there will be no problem

regarding this issue. On the other hand if we have very irregular surface with many local minimas in it, then there is the chance that particles will be trapped in to some local value.

5.3 Proposed Hybrid PSO (HPSO) Algorithm

We have simulated different PSO algorithms in previous chapters and from these simulation results we have perceived some idea that how can we overcome the issue of convergence rate and steady state error. And from this perceived knowledge, while observing all the issues that might happen with PSO algorithms and their possible solutions, we are proposing a new algorithm here. This algorithm will try to overcome all the previously stated problems that might can occur in PSO algorithms, to give us better steady state error with possible best convergence rate. This hybrid PSO algorithm will use following three techniques,

- Re-randomization around new *gbest*

- Increased social effect while introducing a parameter *lbest*

- Adaptive inertia weight for the particles

This adaptive inertia weight assignment to particles is same technique which has been implemented in previous chapter, while implanting PSO-AW. Explanation for remaining two techniques will be given here.

5.3.1 Re-randomization around new *gbest*

This technique of re-randomization will be used to overcome the issue of stagnancy in PSO algorithms. As it was mentioned earlier, every time a new *gbest* is found all particles will start to move towards it in same general direction, due to this reason there is a chance that these particles will become stagnant. Therefore to avoid such behaviour of the particles, we can re-randomize the particles around *gbest*, every time, when a new value of the *gbest* is found [87]. Now if we apply this phenomenon to the particles then the main purpose of having *gbest* will be mitigated. Because we want our particles to move towards *gbest*, but on the same time we do not want stagnancy around this *gbest*. As this re-randomization of particles might make the convergence rate too slow, so to avoid this problem we have to make sure that particles should not be initiated every time away from the new *gbest*. Hence to avoid this issue, we will make the variance of this re-randomization higher in the start of simulation and keep on decreasing the value of this variance as simulation progress. Following equation 5.1, shows the formula of variance used for the re-randomization.

$$variance(n) = \frac{-A}{(1 + e^{\frac{-n+M}{S}})} + A \tag{5.1}$$

Where A is the starting value of the effectiveness of the re-randomization, M is the midpoint, and S is the slope. Following figure 5.1 will provide more clear idea about these parameters. The value of variable S, will decide the slope of the curve in figure 5.1, smaller values of S will make the curve steeper and large values of
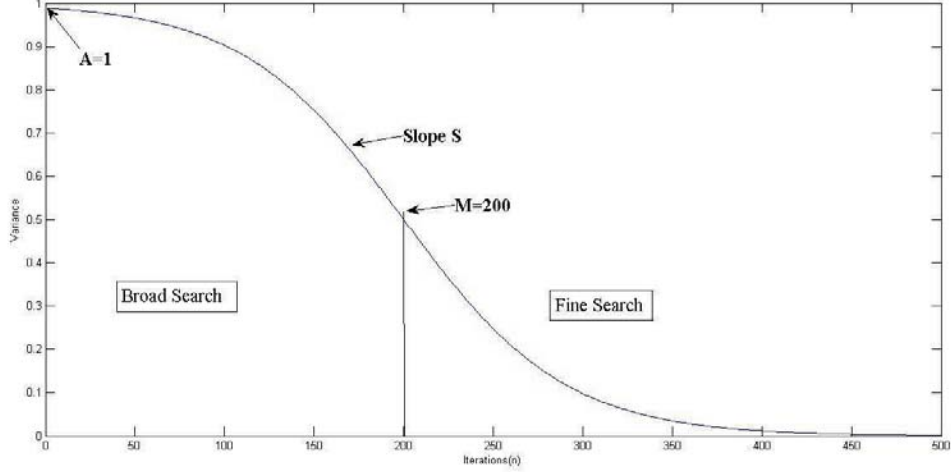
Figure 5.1: Variance Curve for Re-randomization

S will make the transition smooth. Through this variance factor we will divide the search for the optimal solution in two parts. First one is broad search, in which the variance will be higher and particles will re-randomized far from *gbest* to explore more. In second part, which is fine search area, the variance will be assigned smaller values so that particles should lie near new *gbest*, because in the end we want our solution to converge to some optimal value. And these two regions will be separated by a midpoint M. Value of M will decide the duration of these both broad and fine search regions.

Through re-randomization, particles would be given more chance to search for the potential solutions, and there will be improvement in steady state error. As this will decrease the convergence rate, so we used re-randomization along with variance curve, so that the effect of it should decrease as simulation progress.

94

5.3.2 Enhanced social effect with parameter *lbest*

There are many ways to get improvement in the optimization results, and these all methods depend on the way we are performing the search in the swarm and how we are updating the velocity and positions of the particles [85]. If the problem of being trapped in local minimum value is resolved then there will be improvement in steady state error. Eberhart and Kennedy, proposed a version of PSO which use local information for decision making in [99]. In this research they made ring type topology, in which only two particles are included, and they communicate with each other only, not with the whole swarm. They proved that through this technique, the probability that particles will be trapped in some local minimum values is reduced. Although the convergence rate was degraded in these simulation results but still it was proved that with more social effect, steady state error can be reduced.

In this proposed algorithm we want to improve steady state error, so this previously explained idea will be useful for this purpose. Eberhart and Kennedy, in [99], they just use this ring topology of particles for velocity and position updates. Here we will incorporate this technique, which is topology of some specific number of particles, with global best evaluations as well. And we will name this parameter *lbest*. It operates just like *gbest* parameter, except it will divide the particles in to number of sub-groups. And now particles will have to memorize three entities, the best position achieved by any individual particle (pbest), the best position achieved so far among whole particles (*gbest*) and best position achieved by par-

ticles in these sub-groups (*lbest*).

Three methods were under consideration, while incorporating this *lbest* parameter with our algorithm, which are following,

- Use a constant parameter just like acceleration constant, which will be named as c3, will be multiplied with this local best update in velocity update equation

- Use the variance parameter, which is shown in figure 5.1, and explained by equation 5.1. And this will be multiplied with this local best update in velocity update equation

- Use the variance parameter, which is shown in figure 5.2, and explained by equation 5.2. And this will be multiplied with this local best update in velocity update equation

$$variance(n) = \frac{-A}{(1 + e^{\frac{n-M}{S}})} + A \qquad (5.2)$$

While using the first technique of acceleration constant, this enhanced socialized effect will remain effective throughout the process at constant rate. And it has been stated earlier that in the end of the process we want the whole swarm to converge on some optimum value, and if we keep it constant throughout the process, it might will degrade the convergence rate, which is not required.

If we use the third methodology, while using the variance curve shown in figure 5.2, then there will be no effect on the steady state error, it will remain the same like previously used algorithms. Because if fine search will be carried out in the start
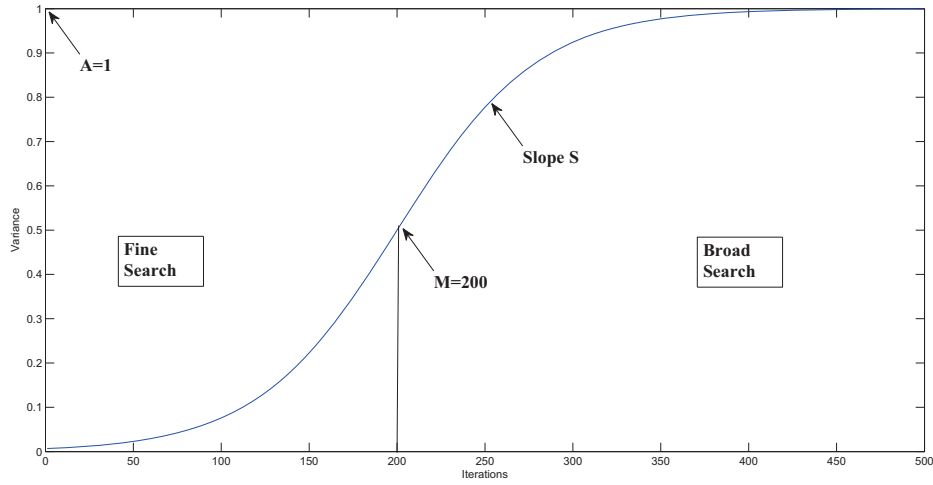
Figure 5.2: Variance Curve

of the iterations, then there will be very minimum effect of enhanced socialized factor on steady state error. Although it will have the good convergence rate but still our goal to achieve better steady state error, with reasonable convergence rate will not be achieved.

Hence the only option left here, which is used in our proposed algorithm, is to use same variance curve which is used for the re-randomization of the particles, shown in figure 5.1, and explained in equation 5.1. While using this variance curve, we are ensuring that in the start of iterations, particles will have all types of interaction with each other, both global and local. In this case there is less chance that particles will get trap in to some local minimum values. As process proceeds, this effect will be minimized in fine search region. All these three techniques have been checked through simulation results and this one gave the best results.

97

5.3.3 Implementation of Proposed Algorithm HPSO

The main purpose of introducing this algorithm is to secure better steady state error with acceptable convergence rate. Hence in this proposed algorithm, we will incorporate all previously stated three techniques. Which are adaptive inertia weight assignment to the particles, re-randomization of particles around new *gbest* and this enhanced social effect.

In the beginning of the iterations, we want our particles to converge swiftly. And through previously explained knowledge, it is known that adaptive inertia weight techniques works well for better convergence rate, and re-randomization of the particles around *gbest*, usually slow down the convergence rate. Therefore in the beginning we will ensure that this re-randomization should not be effective. Once our MSE curve jumped to some value after which there is very small change in error, as compare to change in the start, then we will apply this re-randomization process. And usually it takes almost 20 to 30 iterations to secure a steady error. Hence for this hybrid PSO (HPSO) algorithm, adaptive inertia weight assignment will be effective right from the start of the iterations along with second technique of enhanced socialized effect through variance curve. And after we achieve steady error, re-randomization will also take part in the simulation, and this re-randomization will also be applied through variance curve. In the start due to effect of re-randomization and more socialized effect, our particles will be restrained from local minimum values and after some iterations, when particles will have more clear idea about global optimum value, the effect of these both

techniques will be minimized. At this time our simulation will be in fine search

region where adaptive inertia weight assignment will have more influence.

Hence if these three techniques would be used as explained above, then there is

a very good chance that we can secure improvement in steady state error with

an appropriate convergence rate. Flow chart shown in figure 5.3, explains the

working of this proposed algorithm. It can be seen from the flow chart, as in the
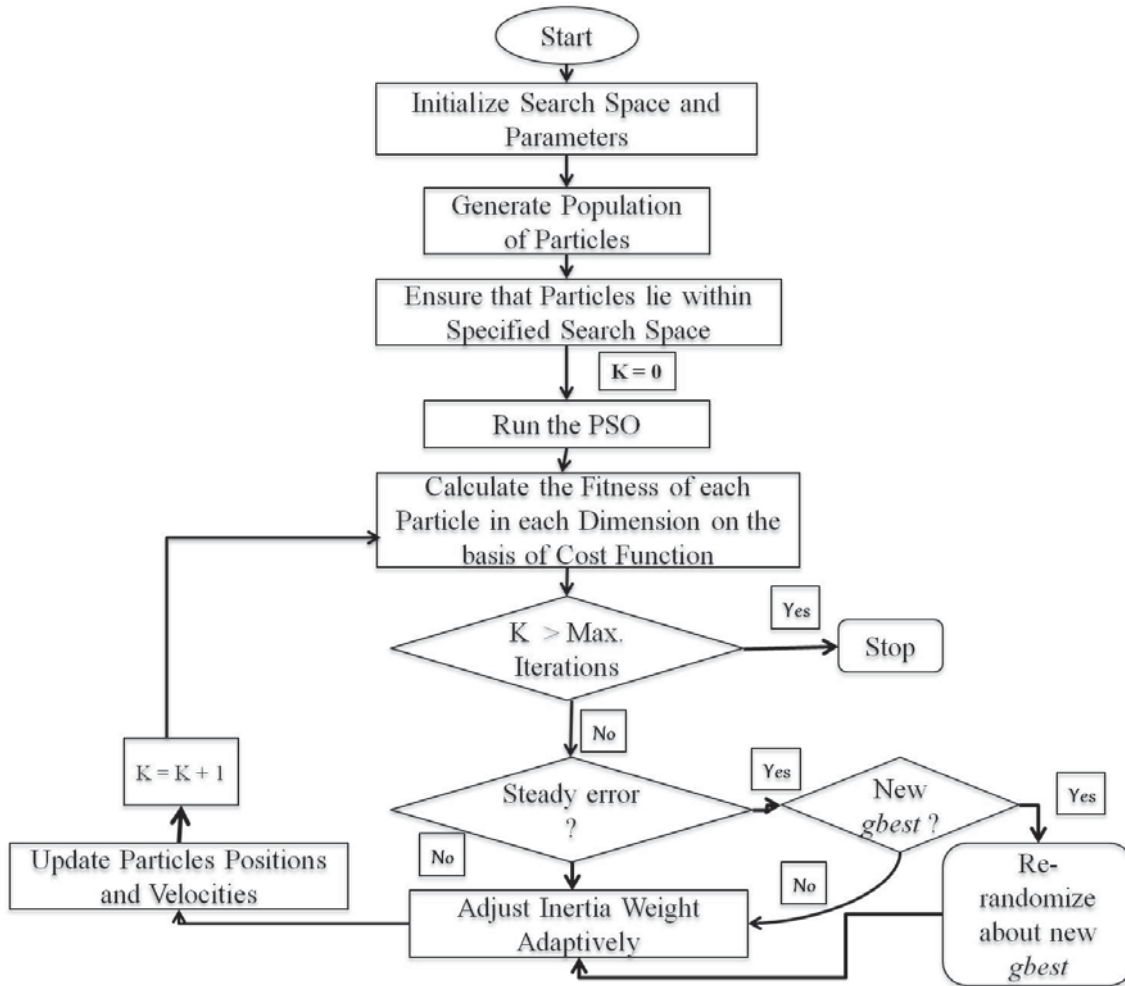
Figure 5.3: Flow Chart of HPSO

start of iteration error will not be steady so re-randomization will be not used,

99

but only adaptive inertia weight assignment and enhanced socialized affect, will be used. As number of iterations increase, error will become steady and then all three techniques will be used.

The velocity and position update equations will be shown in equation 5.3 and 5.4 respectively.

$$v_{id} = w_i(n) * v_{id} + c_1 * rand() * (p_{id} - x_{id}) + c_2 * rand() * (p_{gd} - x_{id})...$$

$$+sqrt(var()) * rand() * (p_{lgd} - x_{id}), \quad 1 \leq d \leq D \quad (5.3)$$

$$x_{id} = x_{id} + v_{id} \quad 1 \leq d \leq D \quad (5.4)$$

5.4 Sensitivity Analysis of HPSO

In order to get stable MSE curves from these algorithms, it should be tuned perfectly with respect to every parameter involved in it. Just like other PSO algorithms, there has to be a complete sensitivity analysis of HPSO with respect to every parameter involve in it. And these parameters are like, number of particles, acceleration constants, window size, midpoint of variance curve, effective value of variance curve, number of taps of the equalizer and maximum allowed velocity to particles. Parameter, slope factor S, will not be included here as it has almost negligible effect on HPSO algorithm and the optimized value of this will be 45. With the help of simulation results, we will find optimized values for all these

parameters, in order to secure optimized results using HPSO in every environment. We will compare these optimized values of the parameters with the previously used PSO algorithms, in order to prove that this HPSO algorithm shows better performance.

5.4.1 Midpoint of Variance Curve

The parameter M, is the midpoint of the variance curve, shown in figure 5.1. This parameter separates the two regions of search, broad search region and fine search region. This parameter will decide that for how many number of iterations, our particles will perform the broad search throughout the search space and for how many iterations particles will perform the fine search.

If we assign higher values to this parameter, then broad search will remain for large number of iterations and particles will have more time to search throughout the region, and there is the chance of getting better steady state error. But this thing will have effects on convergence rate, and it might reduced. On the other hand if we assign lower values to this parameter, then particles might not get the chance of broad search region, and this will effect the steady state error.

Simulation results, shown in figure 5.4, presents the effect of different values of this parameter M on its MSE curves. From simulation results shown in figure 5.4, it is evident that if we assign lower values to M, steady state error is degraded. Like for M=50 and M=100, steady state error is -15.5dB and -17dB respectively. For values of M greater than or equal to 200, results are almost similar. After
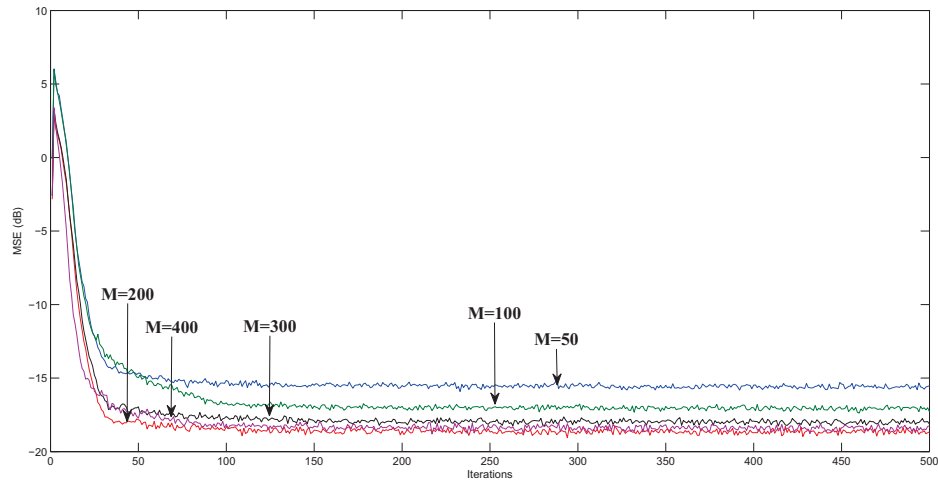
Figure 5.4: Effect of the Midpoint Value of Variance curve, M, on the performance behaviour

M=200, it is not effecting the steady state error. These all were the expected results, that increased values of M will yield improvement in steady state error as compare to lower values of M. The convergence rate did not get effected up to greater extent due to the variations in this parameter M. Hence with the help of this simulation result it can be concluded here that the optimized value for this parameter M will be 200.

5.4.2 Effective Value of Variance Curve

This parameter will decide the effective value of the broad search region, in the variance curve, shown in figure 5.1. Through this parameter the effectiveness of re-randomization and enhanced social effect, will be controlled.

If we assign higher values to this parameter, the particles will be re-randomized

around *gbest*, with more distance, and they will get the chance to explore more in search space. And with higher values of this parameter, there will be more socialized effect among particles and they will get more clear idea about the search space. This thing will help to improve the steady state error and will yield improved results. But with more effect of broad search region, we will suppress the effect of adaptive inertia weight, which will influence the convergence rate. If smaller values will be assigned to this parameter, then effect of better search will be reduced, and particles will not get the chance to search more around the regions of *gbest* and our steady state error will might be degraded. Although this scenario will have the better convergence rate because adaptive inertia weight assignment to the particles will have more effect.

Simulation results, shown in figure 5.5, presents the effect of different values of this parameter A on its MSE curves. Form this simulation result, shown in fig-
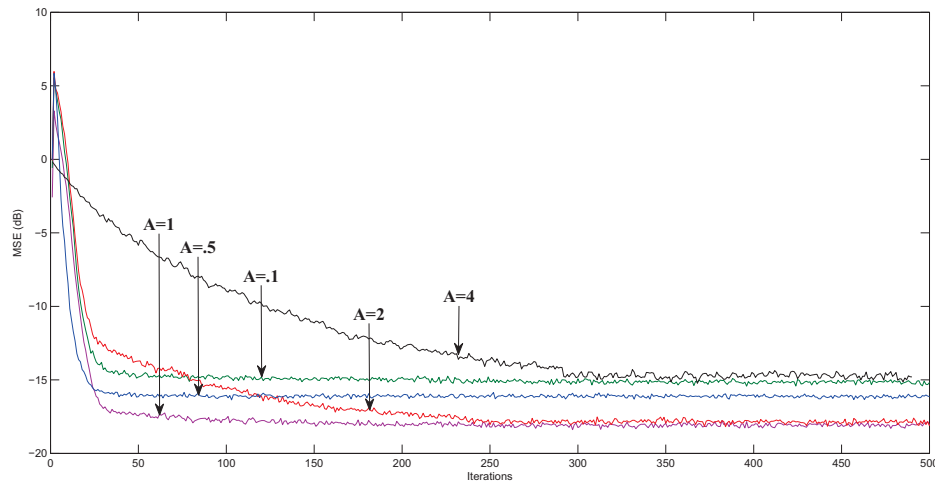

Figure 5.5: Effect of Effective Value of the Variance curve, A, on the performance behaviour

103

ure 5.5, it is evident that higher values of A will yield better steady state error, but convergence rate is reduced. On the contrary lower values of A will yield better convergence rate but we are not getting improvement in steady state error. Simulation result, shows that smaller values of A, like A=0.1 and A=0.5, yield no improvement in steady state error. In this case our algorithm is acting just like the PSO-AW algorithm, because the effect of re-randomization and enhanced socialized effect is almost negligible. That's why the MSE curve in this case is almost similar to MSE curve of PSO-AW, with similar convergence rate. If we assign A=1, then we achieved both better convergence rate and improved steady state error. If we increase the value of this parameter more than 1 then convergence rate is degraded vastly, which is not required. Hence from all this discussion it can be concluded here that the most optimized vale for this parameter will be 1.

5.4.3   Effect of the Number of Particles

Number of particles are like number of potential solution in any search space. As the number of particles increase, the chance of exploring more area of the search space also increases, and the probability of achieving most optimum value, also increases. Therefore with large number of particles we can secure better steady state error, because more particles will search the solution space.

Number of particles also has effect on convergence rate as well. As more particles will be involved in the process of searching the optimum value, so there will be

the possibility that these particles will find the optimum value in short time by making the convergence rate higher. If less number of particles are involved in the process then it will take more time to converge on some optimum value, so it is better to have larger number of particles in search space.

The disadvantage of using large number of particles for search process is more overhead. As process will take more time to generate the results and larger number of computations will be required to get results. Also after some specific number of particles, the increment in convergence speed, and improvement in steady state error will be negligible.

Simulation results shown in figure 5.6, depicts the effect of number of particles on the convergence rate and steady state error. It is evident from simulation results
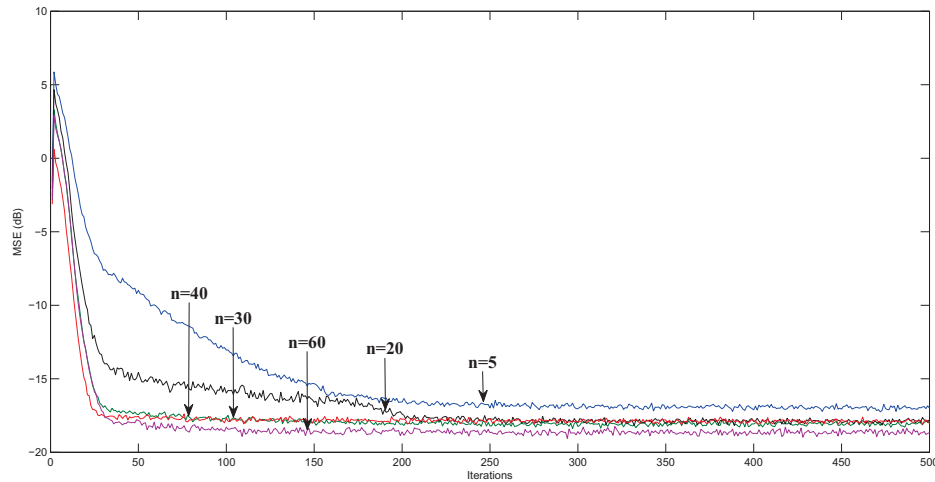

Figure 5.6: Effect of the Number of Particles, n, on the performance behaviour

shown in figure 5.6, that after 30 particles there is not considerable change, with respect to convergence rate and steady state error, in the MSE curves. For smaller

105

number of particles like 5 and 20, the steady state error is degraded, but with the number of particles 30, 40 and 60 we got almost the same steady state error of -18.5dB with same number of iterations taken for convergence. As there is no considerable variation with respect to convergence rate and steady state error after 30 particles so it can be concluded here that n=30 particles will be optimum value for this parameter.

In previously used, PSO algorithms, the optimum value for number of particles was 40 [98]. Here in our proposed algorithm it takes less number of particles to reach even better results which shows the superior performance of this algorithm, over conventionally used PSO algorithms. As this algorithm took less number of particles to generate the desired results so here we achieved the advantage of fast processing as compare to conventional PSO algorithms.

5.4.4 Effect of the Acceleration Constants

The rate, at which particles move in the direction of their local best values, is controlled by acceleration constant c1. And c2 is the acceleration constant, which will control the movement of any particle in the direction of global best value. As stated earlier that these two parameters control both, the steady state error and also the convergence speed of the particles. If we assign smaller values to these parameters then particles will move with slow speed in the search space and explore more solutions. In this way there is quite less chance that they will get trap in to some local value and therefore through this we can achieve better

steady state error. In this case we are making the convergence speed slower and it will take more time by the swarm to converge on some optimum value.

On the other hand if we assign higher values to these parameters, then particles will move quickly and will settle down quickly on some optimum value. This will make the convergence rate higher but there will be the chance of getting trap in to some local minimum which will cause the steady state error to degrade. Therefore the values for these parameters will be application dependent.

Simulation results, shown in figure 5.7, presents the effect of different values of acceleration constants on its MSE curves. From the simulation results shown in
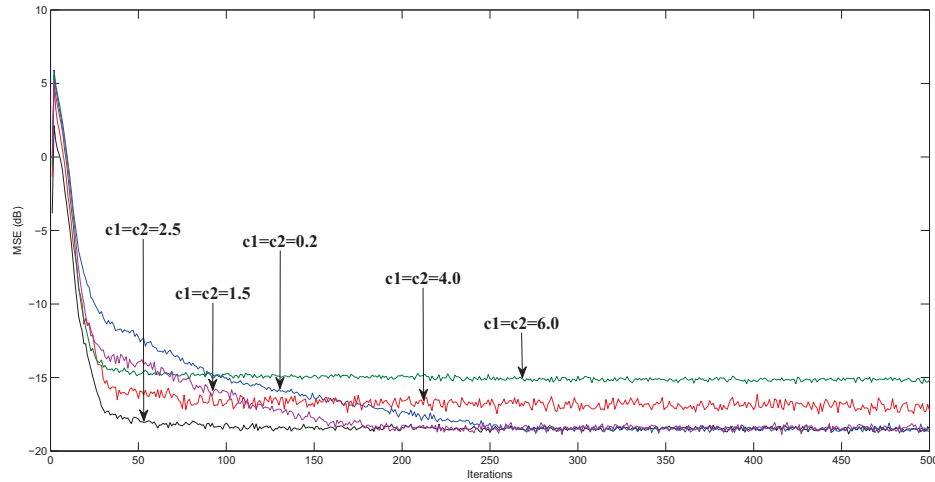
Figure 5.7: Effect of the Acceleration Constants, c1&c2, on the performance behaviour

figure 5.7, it is evident that for smaller values of acceleration constants we achieve better steady state error. For c1=c2=1.5 and c1=c2=0.2, the steady state error is almost -18.5dB. And as we increase in the values of acceleration constants the

steady state error will increase. Like for c1=c2=4 and c1=c2=6, the steady state error is almost -16.5dB and for c1=c2=6 it increases to -15dB. As we increase the value of this parameter, convergence rate is degraded. Hence from these curves shown in figure- 5.7, it is evident that most optimum value for these acceleration constants is c1=c2=2.5.

5.4.5 Effect of the Number of Taps of Equalizer

The number of taps of adaptive equalizer is a parameter which is purely application dependent, and any algorithm modification will have almost negligible effect on it. Here in PSO, for each algorithm, the number of taps of equalizer is equivalent to the dimensions of each particle.

Simulation results, shown in figure 5.8, presents the effect of different values of number of taps on its MSE curves. It is evident from the simulation results shown
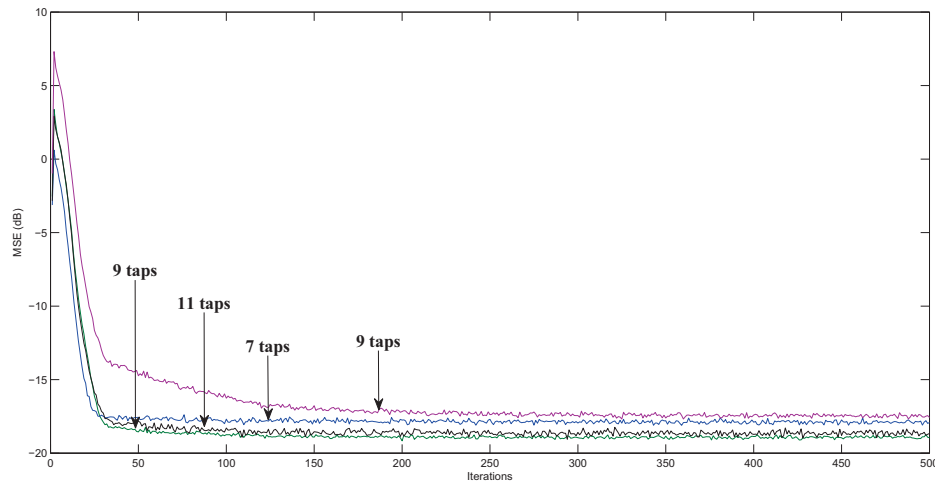

Figure 5.8: Effect of the Number of Taps, d, on the performance behaviour

in figure 5.8, that for this same application of adaptive channel equalization, the different values of number of taps have very slight effect on both areas of concern, steady state error and convergence rate. Hence from these simulation results we can conclude that the optimum value for the number of taps of adaptive equalizer will be 9.

5.4.6 Effect of the Maximum Velocity

This parameter of maximum velocity is actually controls the maximum allowed velocity to the particles, through which these particles move in search space. This parameter is related to maximum allowed position to any particle, means this maximum allowed velocity of any particle is equal to some number times of maximum allowed position to any particle, therefore these two parameters have combined effect.

This maximum allowed velocity is the parameter through which particles move towards their global optimum positions. If particles move towards their global optimum values with higher speed then there will be the chance for these particles to search extensively around the global value and the probability of getting better MSE will be higher. Therefore higher values of this parameter will have productive effects on MSE curves by giving minimum steady state error with better convergence. On other hand if we assign lower values to this parameter then there is chance that these particles might get trap around some local minimum values, and they will keep on searching only around that area. And due to this we

might get higher steady state error and MSE curve can also take higher number of iterations to converge.

Simulation results, shown in figure 5.9, presents the effect of different values of maximum allowed velocity on its MSE curves. From the simulation results shown
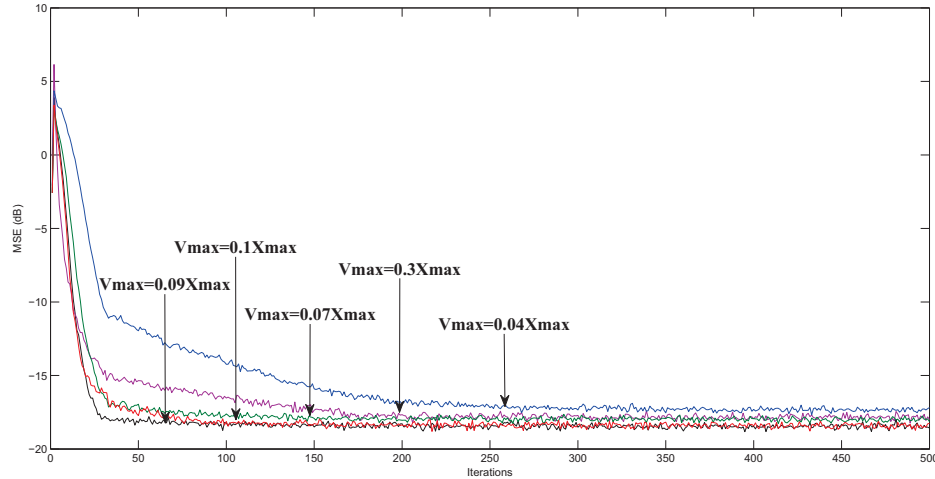
Figure 5.9: Effect of the Maximum Allowed Velocity, Vmax, on the performance behaviour

in figure 5.9, it is evident that larger this parameter, secures minimum steady state error. As minimum value of this parameter, Vmax=0.04Xmax, gives the most degraded steady state error which is almost -17dB. By assigning higher values to this parameter we secured better MSE curves, with minimum steady state error and better convergence rate. By increasing the value of this parameter after, Vmax=0.09Xmax, there is not even noticeable effect on MSE curves. Higher values were also tried in simulation, but those were not shown in this figure, and those values also deliver the similar results, so there is no use of assigning higher

values to this parameter. Hence it can be concluded here that the most optimized value for this parameter will be Vmax=0.09Xmax.

5.4.7   Effect of the Data Window Size

In most of the practical conditions we do not have the whole data at the same time. Most of the time we get the data in form of chunks for equalization, and we have to perform this equalization only on the subset of the data, and from that subset we have to find the behaviour of the incoming data. Therefore to make our simulations more like practical conditions, this parameter will be introduced.

If we assign higher values to this parameter then we are taking more data to observe the statistics of the complete data set, which obviously will yield the better results in sense of steady state error. And if we assign smaller values to this parameter then we have smaller subset of the complete data, to understand the statistics of the complete data. And in this case we might not get better steady state error. This parameter has also effect on convergence rate, but this effect on convergence rate of the MSE curves, is not of great extent.

Simulation results, shown in figure 5.10, depicts the effect of different values of data window size N, on its MSE curves. From the simulation results, shown in figure 5.10, it is evident that for smaller value of this parameter we have degraded steady state error and also slow convergence rate. For N=50, it converges at almost 150 iterations with steady state error of -17.5dB. As we assign higher values to this parameter, convergence rate and steady state error, both improved. For N=

111
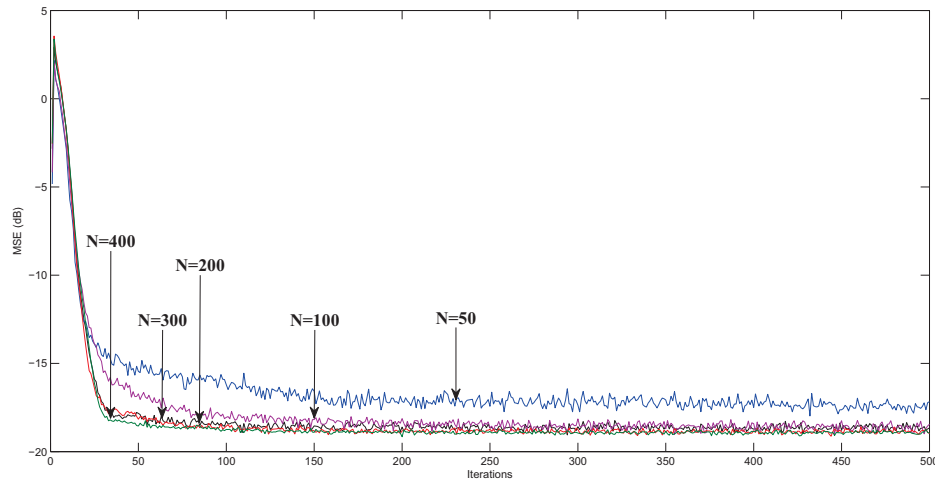
Figure 5.10: Effect of the Data Window Size, N, on the performance behaviour

200, 300, 400, it converges after only 25 iterations with steady error of almost -18.5dB. At N=100 it converges after almost 80 iterations with same steady state error. After N=100, there is no improvement in steady state error. Hence it can be stated here, that this algorithm made the number of computation smaller as compare to all previously used PSO algorithms. As we increase the value of N, it will take more time, as it will use more data, which will increase the delay. Here after N=100, if we keep on increasing the value for this parameter, there will not be improvement in steady state error, only a small improvement in convergence rate. Hence if we are more concerned about convergence rate the optimum value will be N=200, otherwise with slight less convergence rate, N=100 will yield the optimized results with minimum number of computations.

5.5 Simulation Results of Adaptive equalization using HPSO

Here the comparison of the simulation results for adaptive equalization, using different linear channels and non linear system, at different noise levels, will be conducted. We will compare the MSE for this proposed algorithm with the previously used PSO algorithms including PSO-AW and LMS.

5.5.1 Simulation Comparison Using Linear Channels

Same previously used two linear time-invariant channels will be adopted here as well, described by their following transfer functions $H1(z) = 0.2602 + 0.9298z1 + 0.2602z2$, and $H2(z) = 0.408 + 0.816z1 + 0.408z2$. The first channel $H1(z)$ is more stable as compare to second channel $H2(z)$. Following are the details of different parameters which will be used in all these algorithms. For every PSO algorithm the optimal parameters will be, Xmin=-2, Xmax=2, input window size N will be 200, and number of iterations will be 500. For remaining parameters, for PSO-W, Vmax=0.07Xmax, Wmin=0.6 and Wmax=1, c1=c2=1.5, and the number of particles will be 40. For PSO-CCF, Vmax=0.20Xmax, c1=c2=4, k=5, Vmax=0.20Xmax, and the number of particles will be 40. For PSO-VCF, c1 = c2 = 4, kmin = 4, kmax = 6, and vmax = 0.20xmax. For PSO-AW, S=45, Vmax=0.1Xmax, Wmin=0.6 and Wmax=1, c1=c2=2 and the number of particles will be 30. For this proposed algorithm HPSO, S=45, Vmax=0.09Xmax, Wmin=0.6 and Wmax=1, c1=c2=2.5, M=200 and A=1 and the number of par-

ticles will be 30. Values of these parameters are secured from sensitivity analysis done in previous section. For LMS the step size will be 0.025. And these all results will be averaged over 25 runs. The SNR for all these algorithms will be 20dB. The number of taps for the adaptive equalizer will be 9. As it was stated earlier while explaining the functionality of the equalizer that there will be delay of processing after signal pass through the equalizer and in order to compare it properly with the original signal we have to introduce a delay factor in the original signal, and the value of this delay, D will be 11.

Figure 5.11and 5.12, shows the simulation comparison of all these algorithms, using first and second linear time variant channels respectively. From these two
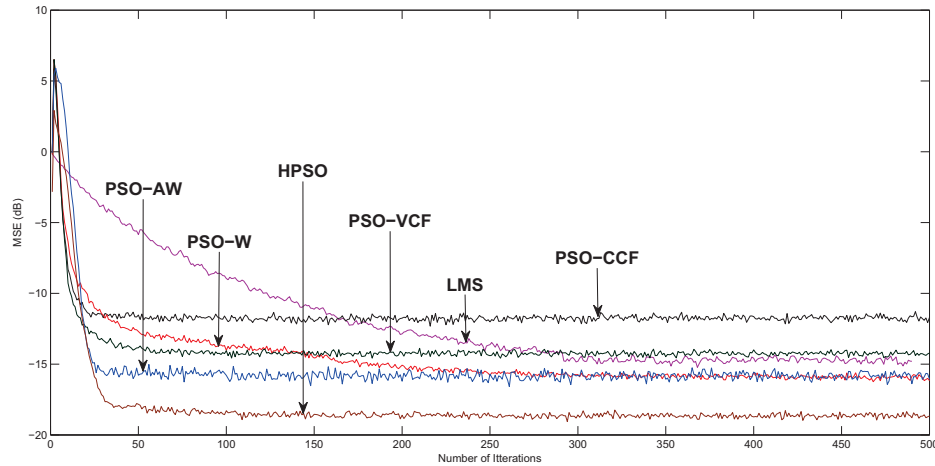
Figure 5.11: MSE curves for PSO-W, PSO-CCF, PSO-VCF, PSO-AW, LMS and HPSO using H1(z)

figures 5.11and 5.12, we can conclude that HPSO, exhibits better performance as compare to all previously used algorithms with respect to steady state error. While using PSO-W and PSO-VCF, we achieved better steady state error and

114
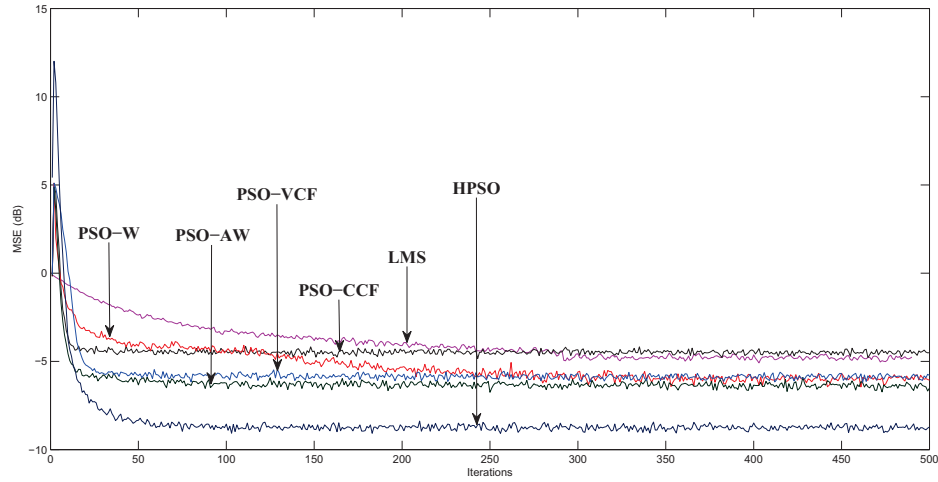
Figure 5.12: MSE curves for PSO-W, PSO-CCF, PSO-VCF, PSO-AW, LMS and HPSO using H2(z)

while using PSO-CCF we achieved higher convergence rate but the steady state error is highly degraded. With PSO-AW we achieved both better convergence rate and improved steady state error simultaneously, but still improvement was required in steady state error and it has been achieved through HPSO. In case of HPSO, although convergence rate is not as swift as of PSO-CCF but the steady state error is highly improved as compare to all algorithms. Hence with slightly less convergence rate, HPSO secured the minimum steady state error as compare to all algorithms, for both channels.

In order to check the proper functionality of this proposed algorithm, we plotted it at different SNR values, as SNR increase the steady state error should be improved. The simulation of HPSO at different SNR values, while using H1(z), is shown in figure 5.13. From figure 5.13, it can be inferred that, steady state

115
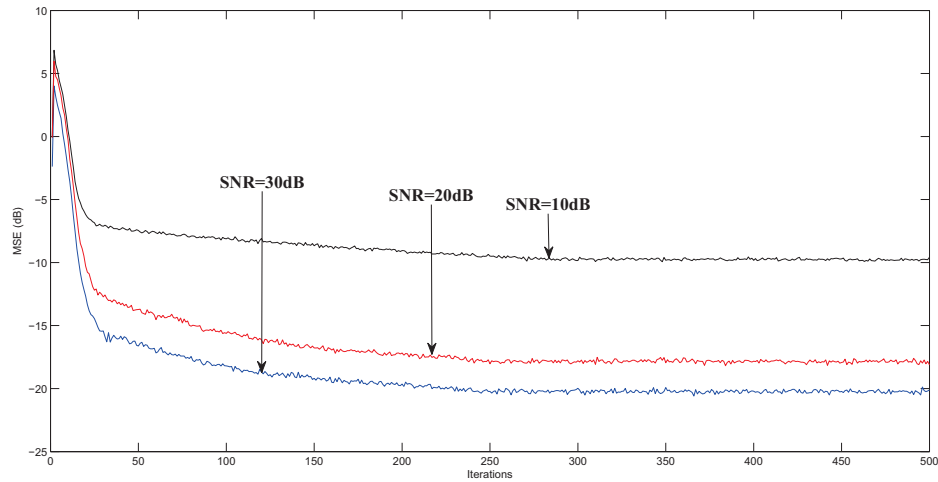
Figure 5.13: MSE curves for HPSO at different SNR values

error improves as SNR increase which shows that this implemented algorithm is working properly without error.

5.5.2 Simulation Comparison Using Non Linear Systems

The main reason for using PSO algorithms for any application was this, that these algorithms perform better for non linear systems. Therefore in order to strengthen this statement, simulation comparison has been shown in figure 5.14, of all previously used algorithms with HPSO. And the non linear system which is used for simulation is shown in figure 3.4, in which b1=1, b2=0.1, b3=0.05, and the channel used is $H1(z) = 0.2602+0.9298z1 +0.2602z2$.

It is evident from the results, shown in figure 5.14, that all PSO algorithms shown better performance as compare to LMS, which shows that PSO has better functionality in nonlinear systems. There is improvement in steady state error, in

116
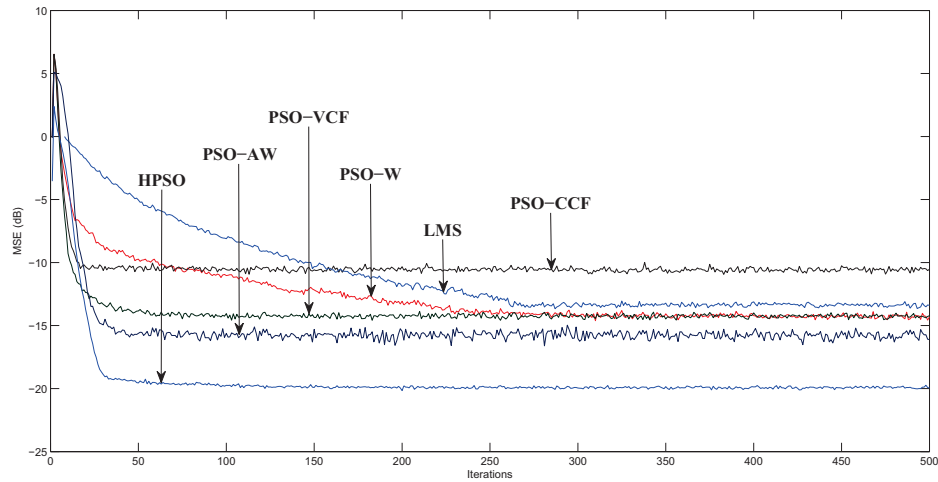
Figure 5.14: MSE curves for PSO-W, PSO-CCF, PSO-VCF, PSO-AW, LMS and HPSO using nonlinear system

this simulation result as well. This improvement is of 4 dB with respect to PSO-AW, 6 dB with respect to PSO-W and PSO-VCF, and almost 9 dB improvement with respect to LMS, which is a great improvement. Here although convergence is not good as compare to PSO-CCF but still the improvement in steady state error is remarkable.

If we compare the statistics of simulation results achieved from non linear systems with that of linear systems, the improvement in steady state error for non linear system is more. For non linear system these all PSO algorithms have better steady state error as compare to LMS, and among all these PSO algorithms, HPSO have the minimum steady state error of -20.5dB approximately. This is the minimum steady state error which we have achieved so far in our whole simulations.

117

5.6 BER Analysis

The comparison for BER between our proposed algorithm HPSO, PSO-AW and LMS will be done in this section. As it has been stated earlier that, among all previously used conventional algorithms like, RLS and Steepest Descent etc, LMS is the most commonly used algorithm. Hence to make the comparison, this LMS algorithm was selected.

We will compare, with respect to BER, while using both linear time invariant channel and nonlinear system.

5.6.1 Comparison of BER using LTI Channel

We will use the same linear time invariant channel which has been used earlier for the comparison of MSE curves. The transfer function of which is H1(z) = 0.2602+0.9298z1 +0.2602z2.

Simulation results, shown in figure 5.15, depicts the effect of SNR on BER curve. It is evident from the results shown in figure 5.15, that as SNR increase BER of HPSO decreases. The BER while using LMS at SNR=15dB; this same BER is achieved at lower SNR=12.5dB while using PSO-AW and with HPSO this same BER achieved even at lesser SNR value of 12dB. Means if we use HPSO and increase the SNR up to same level of 15dB, then BER will be less as compare to LMS at that same SNR value. Hence it can be concluded here that while using HPSO, we can have less BER at same SNR value as compare to LMS and PSO-AW as well.
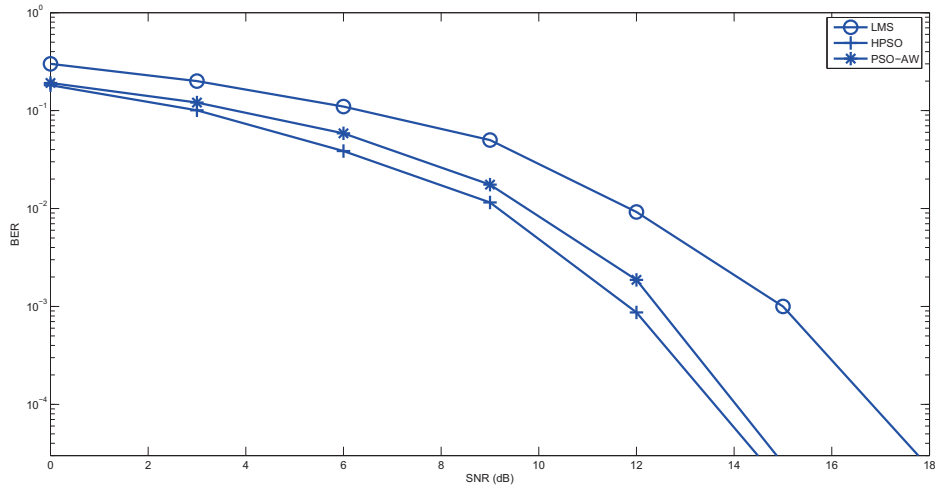
Figure 5.15: BER performance of LMS, PSO-AW and HPSO while using LTI channel H1(z)

5.6.2 Comparison of BER using Nonlinear System

In previous subsection comparison was taken for linear system, now we will compare the BER performance of these algorithms while using same nonlinear system, which is shown in figure 3.4, in which values of the coefficients will be b1=1, b2=0.1, b3=0.05, and the channel used is $H1(z) = 0.2602+0.9298z1 +0.2602z2$. Simulation results, shown in figure 5.16, presents the effect of SNR on BER curve. From the simulation shown in figure 5.16, it is evident that while using HPSO we got better BER at even less SNR as compare to LMS and PSO-AW. Like the same value of BER was achieved, at SNR=14dB for HPSO, for PSO-AW this value of BER was achieved at 14.5dB and for LMS we achieved this value of BER at SNR=19dB. It can be concluded here that for also nonlinear systems, we get better BER while using HPSO as compare to LMS and PSO-AW. From these simulation
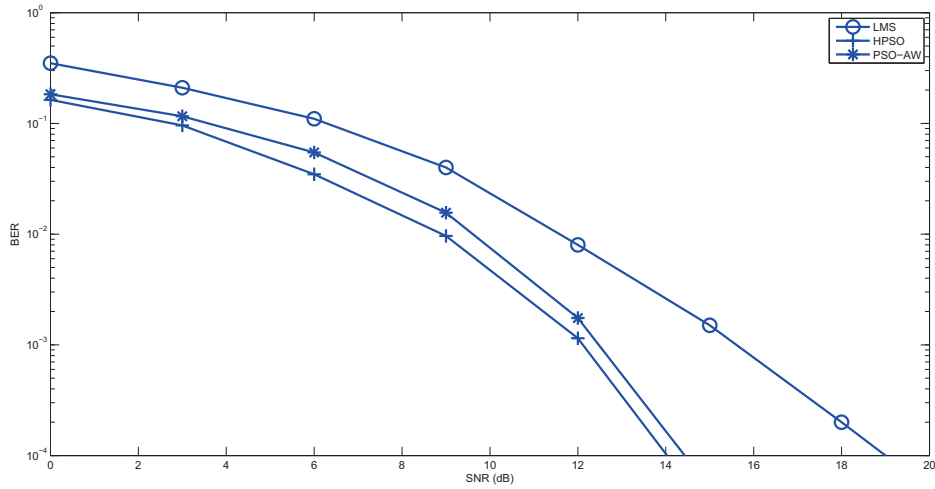
Figure 5.16: BER performance of LMS, PSO-AW and HPSO while using Nonlinear System

results, it is also evident that for nonlinear system, we have more improvement in BER as compare to linear system. Simulation results shown in figure 5.15, for linear system, HPSO showed improvement of 3.5dB as compare to LMS. Here in figure 5.16, for nonlinear system, HPSO has more than 5dB improvement as compare to LMS. Hence these results again proved that PSO algorithms showed better performance in nonlinear systems as compare to conventionally used algorithms.

5.7   Conclusion

In this chapter we described the main reasons to introduce HPSO algorithm, then we completely explained the working of this algorithm with the help of simulation results. Simulation based comparison with previously used algorithms and BER analysis, while using linear and nonlinear systems, also presented in this chapter.

Sensitivity analysis of this proposed algorithm has been presented as well. The main reason behind simulating this algorithm is following that, while using PSO-AW we achieved better convergence rate and nominal steady state error but there was still need to improve the steady state error. Therefore to improve the steady state error, we introduced such techniques and incorporated them in such a way that it will secure similar convergence rate but with improved steady state error. Although convergence rate of HPSO was less as compare to PSO-CCF, sometimes, but we got the large improvement in steady state error. Hence while using HPSO, with slight less convergence rate we achieved almost 5dB improvement in steady state error using linear channel and improvement of almost 9dB using non linear channel, as compare to LMS.

CHAPTER 6

COMPUTATIONALLY EFFECTIVE ALGORITHMS WITH CONCLUSION AND RECOMMENDATIONS

6.1 Introduction

In this last chapter we will introduce two new methodologies, which are used to reduce the number of computations. These methodologies are Local Search (LS) and Train and Verify (TV). Simulation based comparison, of these both techniques, with base case of PSO (PSO-W) will be presented. We will incorporate these methods with our proposed algorithm HPSO, to observe the effect on number of computations. A tabular comparison will also be presented of all the PSO

algorithms used so far in this research to check the performance of these algorithms, with respect to minimum number of computations. BER analysis will also be done for these both new techniques, in both linear and nonlinear systems. In the end of this chapter conclusion will be made, based on this whole research and future recommendation will be suggested as well.

6.2 Techniques to Reduce the Number of Computations

In this section, we will introduce two new techniques to reduce the number of computations and processing delay. Comparison of these both techniques with base case of PSO (PSO-W) will be done. Both types of systems, linear and nonlinear, will be used for the comparison. In order to assert that these two techniques have minimum number of PSO operations, tabular comparison will be done as well.

In order to calculate the performance, with respect to number of computations, of each algorithm, we will use the following formula, stated in equation 6.1,

$$N_p = n(N - d) * m_c \tag{6.1}$$

Where N_p is the number of PSO operations, and m_c is the number of iterations required for convergence. And n is the number of particles, N is the data window

length and d is the number of equalizer's taps weights to be estimated. From equation 6.1, it can be seen that number of PSO operations mainly depend on n, number of particles, and N, data window size. Hence in order to reduce number of PSO operations, we have to assign smaller values to these parameters. On other hand if we assign lower values to these parameters, then there will be degradation in performance with respect to both convergence rate and steady state error. And this thing will increase m_c which in turn will become the reason to increase the number of PSO operations. Therefore we have to consider all these aspects while reducing the number of PSO operations.

6.2.1 Local Search (LS)

In base case of PSO or PSO using linearly decreasing inertia weight, we assigned the same values to both acceleration constants, c1 and c2. As first acceleration constant c1, assign weight to local search and c2 assign weight to global search. In base PSO case, equal weights were given to the local and global search. Here as the name of this new technique suggests that we will assign more weight to local search. Although it will reduce the convergence rate, but it will have less number of PSO operations, shown in table-7.1.And the reason for this less number of PSO operations is that we assigned smaller values to n, number of particles, and N, data window size. Here c1 will be assigned higher weight 5.5, and c2 will be assigned only 0.5.

6.2.2   Train and Verify (TV)

Following are the two reasons which motivate this new methodology. In all pre-viously used algorithms we have been using fixed data window N, and due to this our problem might become deterministic and if the final tap weights were tested with the complete data set, the MSE would be larger.

As the main purpose here is to reduce the number of PSO operations, and while using LS, the convergence rate reduced and steady state error also increased. Therefore to address these issues we use this new methodology train and verify. In this method a fixed data window is used for Wver iterations. At the end of Wver iterations, a new data window of length $N_{ver} > N$ is used to test pbest and gbest obtained so far. These will be updated and then, a new fixed data window is used for another Wver iterations, and so on. As in this methodology new parameters were introduced, so following formula shown in equation- 6.2, will be used to compute number of PSO operations for this technique.

$$N_p = n(N - d) * m_c + \frac{(n(N_{ver} - d) * m_c)}{W_{ver}} \tag{6.2}$$

6.2.3   Simulation Based Comparison using Linear System

In this section we compare these two techniques, LS and TV, with PSO-W, which is also base case of PSO. We will be using the same previously used lin-ear time-invariant channel, described by their following transfer function *H1(z)* = 0.2602+0.9298z1 +0.2602z2. Following are the details of different parameters

which will be used in all these algorithms. For each algorithm the optimal parameters will be, Xmin=-2, Xmax=2, Vmax=0.07Xmax, Wmin=0.6 and Wmax=1, and number of iterations will be 500. For remaining parameters, for PSO-W, c1=c2=1.5, the number of particles will be 40 and N=200. And for LS, c1=5.5 and c2=0.5, n=20 and N=100. For HPSO-TV, c1=c2=4.0, n=10 and N=50. We found the values of these parameters from the sensitivity analysis done in previous chapters, and these all results will be averaged over 25 runs. The SNR for all these algorithms will be 20dB. The number of taps for the adaptive equalizer will be 9. As it was stated earlier while explaining the functionality of the equalizer that there will be delay of processing after signal pass through the equalizer and in order to compare it properly with the original signal we have to introduce a delay factor in the original signal, and the value of this delay, D will be 11. Simulation results are presented in figure 6.1.
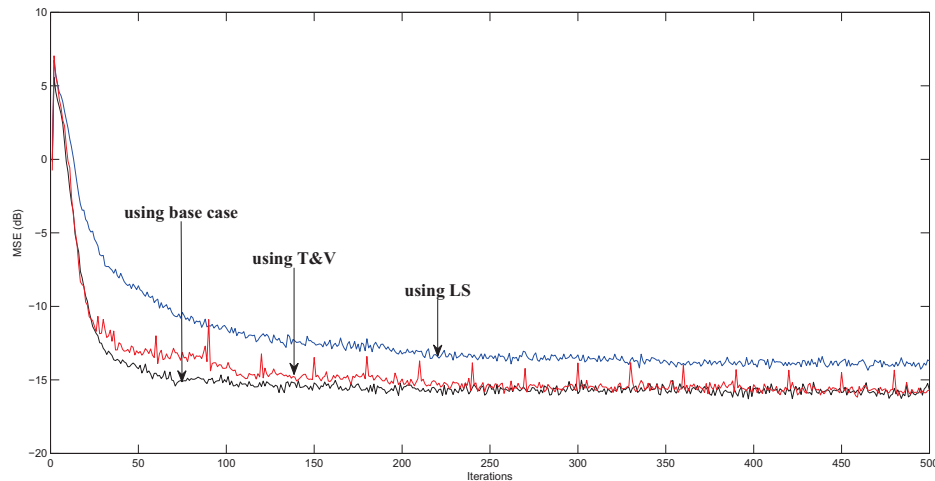

Figure 6.1: MSE curves for PSO using Base Case(PSO-W), T&V and LS with H1(z)

Figure 6.1, shows the simulation comparison of above stated two new techniques with the PSO using base case, PSO-W, with same previously stated LTI channel H1(z). It is evident that LS bears the slowest convergence rate and steady state error is also higher. For TV, although we got the jumps in MSE curve, but still its convergence rate is improved and it also secured the same steady state error like PSO-W. The comparison with respect to number of PSO operations will be described in next section, in table- 6.1. From this table we can conclude that while using TV, we secured the minimum number of PSO operations. LS also secured less number of PSO operations with respect to PSO-W, but the convergence rate and steady state error is degraded. The main advantage which has been secured here is that the process of adaptive equalization completed with significantly less processing delay. The reason of this reduced processing delay, while using LS and TV, is due to less number of particles and reduced data window size.

6.2.4 Simulation Based Comparison using Non-Linear System

In this section comparison will be made using nonlinear system and same LTI chaneel, $H1(z) = 0.2602+0.9298z1 +0.2602z2$, will be used. And the non linear system which is used for simulation is shown in figure 3.4. Following are the details of different parameters which will be used in all these algorithms. For each algorithm the optimal parameters will be, Xmin=-2, Xmax=2, Vmax=0.07Xmax,

Wmin=0.6 and Wmax=1, and number of iterations will be 500. For remaining parameters, for PSO-W, c1=c2=1.5, the number of particles will be 40 and N=200. And for LS, c1=5.5 and c2=0.5, n=20 and N=100. For TV, c1=c2=4.0, n=10 and N=50. We found the values of these parameters from the sensitivity analysis done in previous chapters, and these all results will be averaged over 25 runs. The SNR for all these algorithms will be 20dB. The number of taps for the adaptive equalizer will be 9. As it was stated earlier while explaining the functionality of the equalizer that there will be delay of processing after signal pass through the equalizer and in order to compare it properly with the original signal we have to introduce a delay factor in the original signal, and the value of this delay, D will be 11.

Simulation results are presented in figure 6.2.

Figure 6.2: MSE curves for PSO using Base Case(PSO-W), T&V and LS with Nonlinear System

Figure 6.2, shows the simulation comparison of above stated two new techniques with the PSO using base case, PSO-W, while using nonlinear system. It is evident that LS bears the slowest convergence rate and steady state error is also higher. For TV, although we got the jumps in MSE curve, but still its convergence rate is improved and it also secured the same steady state error like PSO-W. The comparison with respect to number of PSO operations will be described in next section, in table- 6.2. From this table we can conclude that while using TV, we secured the minimum number of PSO operations, also in nonlinear system. For nonlinear system the reduction in number of PSO operations is more as compare to linear system. The same advantage has been secured here as well, while using nonlinear system, that the process of adaptive equalization completed with significantly less processing delay. And this happened due to same reason of using less number of particles and reduced data window size.

6.3   Proposed Hybrid Algorithm with LS and T&V

Here we will incorporate our proposed HPSO algorithm with these two methodologies LS and T&V. Hybrid with LS will becomes HPSO-LS and here will simply assign higher weight to c1 as compare to c2, rest of the algorithm will remain same. And for Hybrid with T&V becomes HPSO-TV and here for hybrid algorithm we will use re-randomization with variance curve shown figure 5.2.

And the reason for using this variance curve is that in re-randomization we are already modifying the values of pbest and in T&V we have to update pbest and gbest after some Wver iterations as well. Therefore if variance curve shown in figure 5.1 will be used, which has higher influence of re-randomization in beginning, then these both will keep on varying the values of pbest due to which there will be the chance that we get unstable MSE curves. That's why we will use the other variance curve in which re-randomization have almost negligible effect in the beginning and higher influence at the end of iterations, and at the end our particles would have settled and it will not create instability in the MSE curves. Here also the numbers of sudden jumps are reduced.

6.3.1   Simulation Based Comparison using Linear System

Here same previously stated LTI channel has been used, $H1(z) = 0.2602+0.9298z1$ $+0.2602z2$. Following are the details of different parameters which will be used in all these algorithms. For each algorithm the optimal parameters will be, Xmin=-2, Xmax=2, S=45, Vmax=0.09Xmax, Wmin=0.6 and Wmax=1, M=200 and A=1 and number of iterations will be 500. For remaining parameters, for HPSO c1=c2=2.5, the number of particles will be 30 and N=200. And for HPSO-LS, c1=5.5 and c2=0.5, n=20 and N=100. For TV, c1=c2=4.0, n=10 and N=50. Values of these parameters are secured from sensitivity analysis done in previous chapter. And these all results will be averaged over 25 runs. The

SNR for all these algorithms will be 20dB. The number of taps for the adaptive

equalizer will be 9. As it was stated earlier while explaining the functionality of

the equalizer that there will be delay of processing after signal pass through the

equalizer and in order to compare it properly with the original signal we have to

introduce a delay factor in the original signal, and the value of this delay, D will
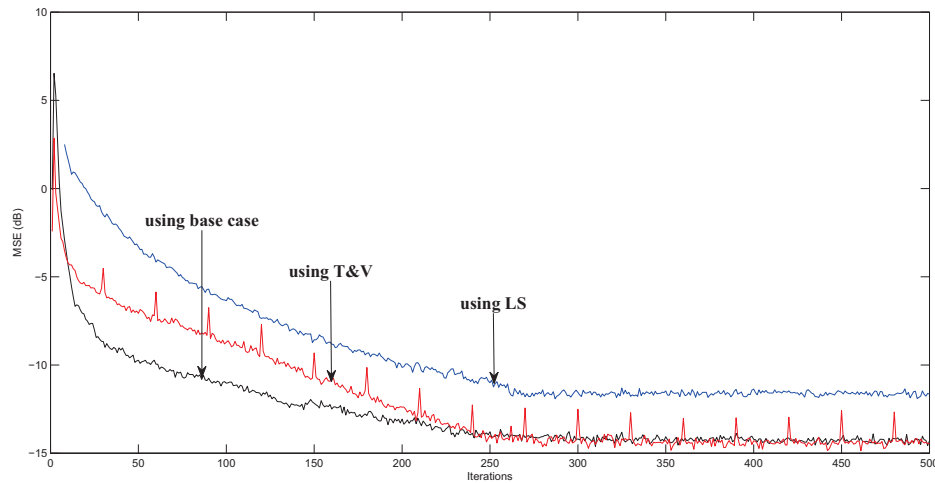
be 11.

The simulation comparison is shown figure 6.3 and the comparison of all these

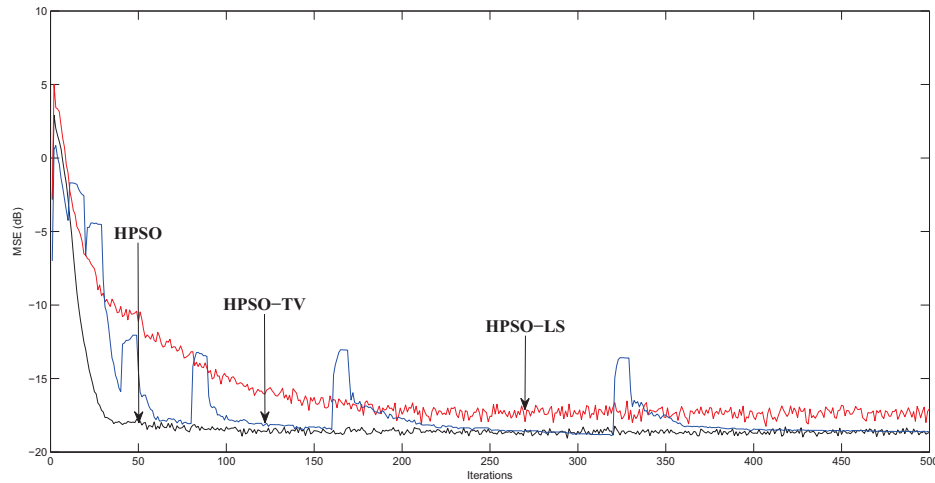algorithms with respect to number of operations is shown in table 6.1.

Figure 6.3: MSE curves for HPSO, HPSO-LS and HPSO-T&V using H1(z)

From the simulation comparison shown in figure 6.3 and table 6.1, it is clear

that although the convergence rate of HPSO-LS and HPSO-TV is less than

HPSO but we have large reduction in number of PSO operations. For HPSO-LS,

131

Table 6.1: Comparison of Parameters and Number of PSO Operation of PSO Algorithms for Linear System

	PSO-W(base case)	LS	TV	HPSO-LS	HPSO-TV	HPSO
N	200	100	50	100	50	100
n	40	20	10	20	10	30
C1	1.5	5.5	4	5.5	4	2.5
C2	1.5	0.5	4	0.5	4	2.5
MSE	-16	-14	-16	-17	-19	-19
mc	90	200	180	150	90	30
Wver	–	–	20	–	20	–
Nver	–	–	200	–	200	–
Np	687600	364000	75825	61500	45495	81900
%age Reduction	–	47%	88.5%	91%	93%	88%

it is converging after 150 iterations and error is relatively high as compare to HPSO. For HPSO-TV, it is converging after 90 iterations but the error is similar to HPSO. For HPSO-TV only the convergence rate is less as compare to HPSO but the error is same.

In table 6.1, percentage reduction in PSO operations of all algorithms with respect to PSO using base case or PSO-W, is shown. We can conclude from this table that if we are more concerned about the less number of operations then HPSO-TV would be the most appropriate algorithm to used, as it showed improvement of almost 93% as compare to PSO-W. If we are concerned about steady state error, then both HPSO and HPSO-TV showed same results but the convergence rate of HPSO-TV is slower as compare to HPSO. Hence in this table the details for each algorithm has been presented, now for selecting the appropriate algorithm will depend on user requirement.

132

6.3.2 Simulation Based Comparison using Non-Linear System

Here same previously stated nonlinear system and same LTI chaneel, $H1(z) = 0.2602+0.9298z1 +0.2602z2$, will be used. And the non linear system which is used for simulation is shown in figure 3.4.. Following are the details of different parameters which will be used in all these algorithms. For each algorithm the optimal parameters will be, Xmin=-2, Xmax=2, S=45, Vmax=0.09Xmax, Wmin=0.6 and Wmax=1, M=200 and A=1 and number of iterations will be 500. For remaining parameters, for HPSO c1=c2=2.5, the number of particles will be 30 and N=200. And for HPSO-LS, c1=5.5 and c2=0.5, n=20 and N=100. For TV, c1=c2=4.0, n=10 and N=50. Values of these parameters are secured from sensitivity analysis done in previous chapter. And these all results will be averaged over 25 runs. The SNR for all these algorithms will be 20dB. The number of taps for the adaptive equalizer will be 9. As it was stated earlier while explaining the functionality of the equalizer that there will be delay of processing after signal pass through the equalizer and in order to compare it properly with the original signal we have to introduce a delay factor in the original signal, and the value of this delay, D will be 11.

The simulation comparison is shown figure 6.4 and the comparison of all these algorithms with respect to number of operations is shown in table 6.2.
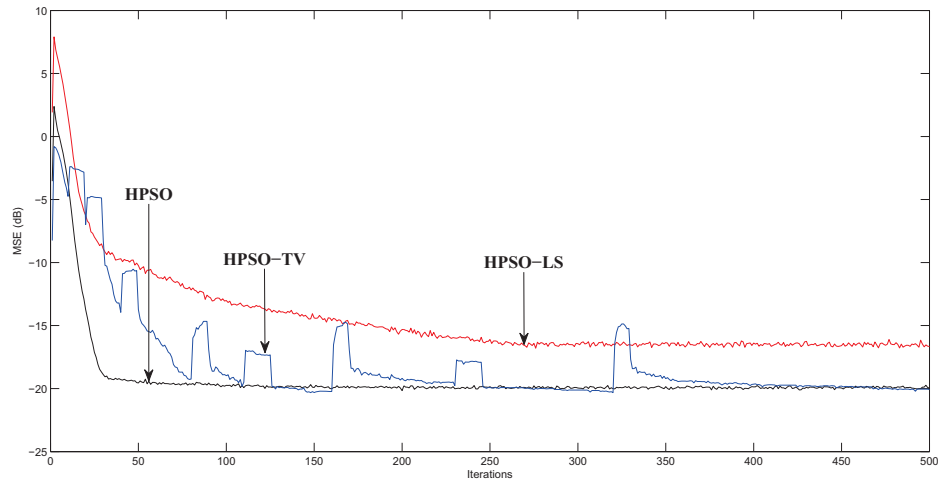
Figure 6.4: MSE curves for HPSO, HPSO-LS and HPSO-T&V using Nonlinear System

From the simulation comparison shown in figure 6.4 and table 6.2, it is clear that we achieved almost similar improvements even using nonlinear system as well. Again a large reduction in number of PSO operations has been achieved for nonlinear system.

In table 6.2, percentage reduction in PSO operations of all algorithms with respect to PSO using base case or PSO-W, is shown. We can conclude from this table that if we are more concerned about the less number of operations then HPSO-TV would the most appropriate algorithm to use, as it showed improvement of almost 96% as compare to PSO-W, for nonlinear system. For nonlinear system, same conclusion can be made that if we are concerned about steady state error, then both HPSO and HPSO-TV showed same results but

Table 6.2: Comparison of Parameters and Number of PSO Operation of PSO Algorithms for Nonlinear System

	PSO-W(base case)	LS	TV	HPSO-LS	HPSO-TV	HPSO
N	200	100	50	100	50	100
n	40	20	10	20	10	30
C1	1.5	5.5	4	5.5	4	2.5
C2	1.5	0.5	4	0.5	4	2.5
MSE	-15	-12	-14.5	-17	-20	-20
mc	220	290	260	230	120	40
Wver	–	–	20	–	20	–
Nver	–	–	200	–	200	–
Np	1680800	527800	131430	94300	60660	109200
%age Reduction	–	68%	92%	94%	96%	93%

the convergence rate of HPSO-TV is slower as compare to HPSO. Hence in this table the details for each algorithm has been presented, now for selecting the appropriate algorithm will depend on user requirement.

6.4 BER Analysis

Here we will perform the comparison for BER among all PSO algorithms and LMS. These PSO algorithms will be HPSO, PSO-AW, HPSO-LS and HPSO-TV. All previously used conventional algorithms like, RLS and Steepest Descent etc, LMS is the most commonly used algorithm. Therefore to make the comparison, this LMS algorithm was selected.

We will compare, with respect to BER, while using both linear time invariant channel and nonlinear system.

6.4.1 Comparison of BER using LTI Channel

We will use the same linear time invariant channel which has been used earlier, which is H1(z) = 0.2602+0.9298z1 +0.2602z2. Usually the analysis of BER curve is considered to be like this, that as we increase SNR, BER should decrease, while making the water fall curve.

Simulation results, shown in figure 6.5, present the effect of SNR on BER curve.

It is evident from the results shown in figure 6.5, that as SNR increase BER for
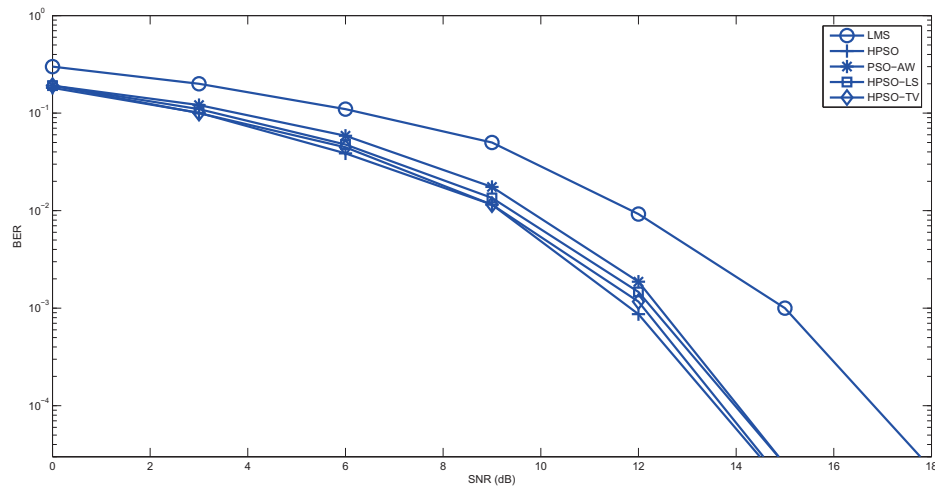
Figure 6.5: BER performance of LMS, PSO-AW, HPSO, HPSO-LS and HPSO-TV while using LTI channel H1(z)

both the algorithms, HPSO-LS and HPSO-TV, decreases. Comparison among these all algorithms reveals that, the BER performance of all PSO algorithms which have been presented so far, is better than LMS. If we compare among PSO algorithms only, BER of these both algorithms, HPSO-LS and HPSO-TV, is better than PSO-AW. Although these both algorithms have slight degraded

136

performance as compare to HPSO, with respect to BER, but it is adequate as these algorithms, HPSO-LS and HPSO-TV, secured minimum number of PSO operations. Comparison between HPSO-LS and HPSO-TV, with respect to BER, reveals that HPSO-TV have better performance, but due to sudden jumps in it, sometimes there might be the problem of stability in it. Hence it can be concluded that while using HPSO-LS and HPSO-TV, we can secure almost similar BER like PSO-AW and HPSO, with quite less number of PSO operations.

6.4.2   Comparison of BER using Nonlinear System

In previous part comparison was taken for linear system, now we will compare the BER performance of these all algorithms while using nonlinear system. The same nonlinear system will be used here, which is shown in figure 3.4, in which values of the coefficients will be b1=1, b2=0.1, b3=0.05, and the channel used is $H1(z) = 0.2602+0.9298z1 +0.2602z2$.

Simulation results, shown in figure 6.6, present the effect of SNR on BER curve. Again from the simulation shown in figure 6.6, it is evident that even while using nonlinear system, all PSO algorithms performed better than LMS, with respect to BER. For nonlinear system we achieved almost similar results as we achieved for linear system.HPSO-TV have better BER performance as compare HPSO-LS, similar to linear system. Similarly these both algorithms have slight degraded performance as compare to HPSO, with respect to BER, but it is acceptable, here
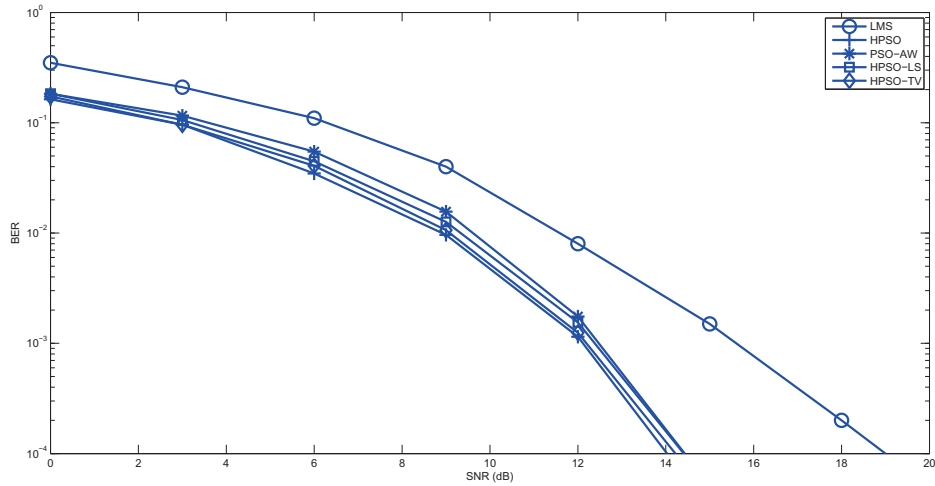
Figure 6.6: BER performance of LMS, PSO-AW, HPSO, HPSO-LS and HPSO-TV while using Nonlinear System

as well, as these algorithms, HPSO-LS and HPSO-TV, secured minimum number of PSO operations. Hence, for nonlinear system as well, it can be concluded that while using HPSO-LS and HPSO-TV, we can secure almost similar BER like PSO-AW and HPSO, with quite less number of PSO operations.

6.5 Thesis Conclusion and Future Recommendations

In this thesis we implemented a new algorithm, PSO-AW, and proposed a new algorithm HPSO for adaptive equalization. And we implemented also two new techniques to reduce the number of operations and both these techniques were also

been used incorporation with our proposed algorithm HPSO. The main reason for implementing PSO-AW was to improve the convergence rate with nominal steady state error. And through our proposed algorithm HPSO we secured the minimum steady state error. If we compare these both algorithms with LMS, for steady state error, then PSO-AW secured almost similar Steady state error as compare to PSO-W but with better convergence rate. With HPSO we secured the minimum steady state error as compare to all previously used algorithms, for adaptive equalization, including LMS. When we incorporate HPSO with two new techniques, LS and TV, we secured almost similar results with quite less number of PSO operations. Although the convergence rate for HPSO-LS is less, still the improvement of reduction in processing delay is remarkable. All algorithms which have been presented in this research, PSO-AW, HPSO, HPSO-LS and HPSO-TV, exhibited better performance with respect to BER as compare to LMS. These all simulations have been done in both, linear and nonlinear, environments. Another fact revealed from all simulation results that improvements for nonlinear systems were better as compare to linear systems, with respect to convergence rate, steady state error and BER. Hence it can be stated that these all algorithms, presented in this report, served well for the purposes.

All the contributions of this thesis report, and conclusion, has been stated. If we observe the results incisively then, there are still problems which can be addressed in future. Like in HPSO we achieved the minimum steady state error as compare to all previously used algorithms but still the convergence rate of HPSO is less

than PSO-CCF. In HPSO-TV we achieved the minimum number of operations but the convergence rate was slower as compare to HPSO, different techniques can be used to improve this convergence rate. And still we are facing the sudden jumps in MSE curves of HPSO-TV, research can be done to address this issue.

REFERENCES

[1] S. U. H. Qureshi, "Adaptive equalization," in *Poc. IEEE*, 1985.

[2] J. Treichler, M. Larimore, and J. Harp, "Practical blind demodulators for high-order qam signals," in *Proceedings of the IEEE special issue on Blind System Identification and Estimation*, 1998.

[3] R. W. Lucky, J. Salz, and E. J. Weldon, "Principles of data communication," in *New York: McCraw-Hill*, 1968.

[4] E. Biglieri, A. Cersho, R. D. Citlin, and T. L. Lim, "Adaptive cancellation of nonlinear inter symbol interference for voice band data transmission," in *IEEE 1. Selected Areas Commun.*, 1984.

[5] J. C. A. Siller, "Multipath propagation," in *IEEE Commun. Mag.*, 1984.

[6] P. Monsen, "Mmse equalization of interference on fading diversity channels," in *IEEE Trans. Commun*, 1984.

[7] T. Murase, K. Morita, and S. Komaki, "200 mb/s 16-qam digital radio system with new countermeasure techniques for multipath fading," in *IEEE Int. Conf. Commun*, 1981.

[8] B. Widrow and S. Sterns, "Adaptive signal processing," in *Prentice Hall, New York*, 1985.

[9] H. Nyquist, "Certain topics in telegraph transmission theory," in *Trans. IEEE*, 1928.

[10] B. Widrow and J. M. E. Hoff, "Adaptive switching circuits," in *IRE WESCON Conv. Rec*, 1960.

[11] R. W. Lucky, "Automatic equalization for digital communication," in *Bell Syst. Tech. I*, 1965.

[12] ——, "Techniques for adaptive equalization of digital communication systems," in *Bell Syst. Tech J*, 1966.

[13] D. A. George, "Matched filters for interfering signals," in *IEEE Trans. Inform. Theory (Corresp)*, 1965.

[14] W. S. J, "The joint optimization of transmitted signal and receiving filter for data transmission systems," in *Bell System. Tech. J*, 1965.

[15] D. W. Tufts, "Nyquists problem-the joint optimization of transmitter and receiver in pulse amplitude modulation," in *Proc. IEEE*, 1965.

[16] A. Cersho, "Adaptive equalization of highly dispersive channels," in *Bell Syst. Teh.J*, 1969.

[17] R. W. Lucky, J. Salz, and J. E. J. Weldon, "Principles of data communication," in *McCraw-Hill*, 1968.

[18] J. G. Proakis and H. M. J, "An adaptive receiver for digital signalling through channels with inter symbol interference," in *IEEE Trans. Inform. Theory*, 1969.

[19] K. Abend and B. D. Fritchman, "Statistical detection for communication channels with inter symbol interference," in *Proc. IEEE*, 1970.

[20] R. Lucky, "A survey of the communication theory literature: 1968-1973," in *IEEE Trans. Inform. Theory*, 1973.

[21] B. Sklar, "Digital communications," in *McCraw-Hill*, 1983.

[22] W. U. Lee and F. S. Hill, "A maximum likelihood sequence estimator with decision feedback equalization," in *IEEE Trans Communication. Techno J.*, 1977.

[23] L. R. MacKechnie, "Maximum likelihood receivers for channels having memory," in *Ph.D. dissertation, Dep. Elec. Eng., Univ. of Notre Dame, Notre Dame*, 1973.

[24] J. F. R. Magee and J. C. Proakis, "Adaptive maximum-likelihood sequence estimation for digital signaling in the presence of inter symbol interference," in *IEEE Trans. Inform. Theory (Corresp.)*, 1973.

[25] S. U. H. Qureshi, "New approaches in adaptive reception of digital signals in the presence of inter symbol interference," in *Ph.D Dissertation, Univ. of Toronto, Toronto, Ont., Canada*, 1976.

[26] S. U. H. Qureshi and E. E. Newhall, "An adaptive receiver for data transmission over time dispersive channels," in *IEEE Trans. Inform. Theory*, 1973.

[27] G. Ungerboeck, "Adaptive maximum likelihood receiver for carrier modulated data-transmission systems," in *Cambridge University Press*, 1974.

[28] J. C. D. Forney, "Maximum-likelihood sequence estimation of digital sequences in the presence of inter symbol interference," in *IEEE Trans. Inform. Theory*, 1972.

[29] J. Forney, G.D., "The viterbi algorithm," in *Proc.IEEE*, 1973.

[30] M. E. Austin, "Decision-feedback equalization for digital communication over dispersive channels," in *MIT Lincoln Lab., Lexington, MA, Tech*, 1967.

[31] C. A. Belfiore and J. J. H. Park, "Decision feedback equalization," in *Proc.IEEE*, 1979.

[32] J. E. M. D. L. Duttweiler and D. C. Messerschmitt, "An upper bound on the error probability in decision-feedback equalization," in *IEEE Trans. Inform. Theory*, 1974.

[33] R. R. B. D. A. George and J. R. Storey, "An adaptive decision feedback equalizer," in *IEEE Trans. Commun. Techno J*, 1971.

[34] P. Monsen, "Feedback equalization for fading dispersive channels," in *IEEE Trans. Inform. Theory*, 1971.

[35] R. Price, "Nonlinearly feedback-equalized pam vs. capacity for noisy filter channels," in *Cambridge University Press*, 1972.

[36] J. Salz, "Optimum mean-square decision feedback equalization," in *Bell Syst. Tech. I*, 1973.

[37] D. D. Falconer and C. J. Foschini, "Theory of minimum mean-square-error qam system employing decision feedback equalization," in *Bell Syst. Tech J*, 1973.

[38] D. D. Falconer, "Jointly adaptive equalization and carrier recovery in two-dimensional digital communication systems," in *Bell Syst. Tech. I*, 1976.

[39] H. Kobayashi, "Simultaneous adaptive estimation and decision algorithm for carrier modulated data transmission systems," in *IEEE Trans. Commun Technol*, 1971.

[40] J. G. Proakis and H. M. J, "An adaptive receiver for digital signalling through channels with inter symbol interference," in *IEEE Trans. Inform. Theory*, 1969.

[41] D. M. Brady, "Matrix analysis," in *An adaptive coherent diversity receiver for data transmission through dispersive media*, 1970.

[42] R. W. Lucky, "filtering with the transversal equalizer," in *Proc. 7th Ann. Allerton Conf. on Circuits and System Theory*, 1969.

[43] P. Monsen, "Theoretical and measured performance of a dfe modem on a fading multipath channel," in *IEEE Trans. Commun*, 1977.

[44] R. Gitlin and S. Weinstein, "Fractionally spaced equalization: An improved digital transversal equalizer," in *Bell Syst. Tech*, 1981.

[45] L. Cuidoux, "Egaliseur autoadaptif a double echantillonnage," in *LOnde flectrique*, 1975.

[46] S. U. H. Qureshi and J. G. D. Forney, "Performance and properties of a t/2 equalizer," in *Nat. Telecomm. Conf. Rec*, 1977.

[47] G. Ungerboeck, "Fractional tap-spacing equalizer and consequences for clock recovery in data modems," in *IEEE Trans. Commun*, 1976.

[48] A. Cersho and T. L. Lim, "Adaptive cancellation of inter symbo1 interference for data transmission," in *Bell Syst. Tech. I*, 1981.

[49] M. S. Mueller and J. Salz, "A unified theory of data-aided equalization," in *Bell Spt. Tech. J*, 1981.

[50] R. D. Citlin and S. B. Weinstein, "On the required tap-weight precision for digitally-implemented adaptive mean-squared equalizers," in *Bell Syst. Tech. I*, 1979.

[51] J. Mazo, "On the independence theory of equalizer convergence,," in *Bell Syst. Tech. J*, 1979.

[52] C. Ungerboeck, "Theory on the speed of convergence in adaptive equalizers for digital communication," in *ISM /. Res. Devel., vol. 16, pp. 546-555*, 1972.

[53] R. W. Chang, "A new equalizer structure for fast start-up digital communication," in *Bell Syst. Tech. I., vol. 50, pp. 1969-20 14*, 1971.

[54] K. H. Mueller, "A new, fast-converging mean-square algorithm for adaptive equalizers with partial response signalling," in *Syst. Tech. J, vol. 54, pp. 143-153*, 1975.

[55] S. U. H. Qureshi, "A 9600 bit/s modem for multipoint communication systems," in *Nat. Telecomm. Conf. Rec*, 1981.

[56] A. Milewski, "Periodic sequences with optimal properties for channel estimation and fast start-up equalization," in *ISM J. Res. Develop., vol. 27, pp. 426-431*, 1983.

[57] K. H. Mueller and A. Spaulding, "Cyclic equalization-a new rapidly converging equalization technique for synchronous data communication," in *Bell Syst. Tech. J., vol. 54, pp. 369-406*, 1975.

[58] D. N. Godard, "Channel equalization using a kalman filter for fast data transmission," in *ISM /. Res. Develop., vol. 18, pp.267-273*, 1974.

[59] M. L. Honig and D. G, "Messerschmitt, adaptive filters; structures, algorithms, and applications," in *Boston, MA: Kluwer Academic Pub*, 1984.

[60] T. K. M. Morf and L. Ljung, "Fast algorithms for recursive identification," in *Proc. 7976 IEEE Conf. Decision Contr*, 1976.

[61] J. M. Cioffi and T. Kailath, "An efficient exact-least-squares fractionally spaced equalizer using inter symbol interpolation," in *IEEE J, Selected Areas Commun., vol. SAC-2, pp. 743-756*, 1984.

[62] F. Ling and J. G. Proakis, "A generalized multichannel least squares lattice algorithm based on sequential processing stages," in *IEEE Trans. Acoust., Speech, Signal Process., vol. ASSP-32, pp. 381-389*, 1984.

[63] E. Eleftheriou and D. D. Falconer, "Restart methods for stabilizing frls adaptive equalizers in digital hf transmission," in *Proc. IEEE Globecorn (Atlanta, CA), pp. 1558-1562*, 1984.

[64] J. Kennedy and R. Eberhart, "Swarm intelligence," in *Academic Press*, 2001.

[65] R. E. J. Kennedy, "Particle swarm optimization," in *Proc. IEEE Intl. Conf. Neural Networks*, 1995.

[66] M. N. V. K. E. Parsopoulos, "On the computation of all global minimisers through particle swarm optimization," in *IEEE Transactions on Evolutionary Computation, vol.8, no.3, pp. 211- 224*, 2004.

[67] S. YH and E. RC, "Empirical study of particle swarm optimization," in *Proceedings of the IEEE International Conference on Evolutionary Computation, pages 1945-1950*, 1999.

[68] C. M, "The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization," in *Proceedings of the 1999 Congress on Evolutionary Computation, pages 1951-1957*, 1999.

[69] R. T. Lvbjerg M and K. T, "Hybrid particle swarm optimiser with breeding and subpopulations," in *Proceedings of the Third Genetic and Evolutionary Computation Conference, pages 469-476*, 2001.

[70] Z. C. L. H. L. W. Shi XH, Lu YH and L. YC, "Hybrid evolutionary algorithms based on pso and ga," in *The 2003 Congress on Evolutionary Computation*, 1996.

[71] S. YH and Eberhart, "Rc. parameter selection in particle swarm optimization," in *Evolutionary Programming VII, Springer, Lecture Notes in Computer Science*, 1998.

[72] E. R and S. Y, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computing*, 2000.

[73] C. A and D. G., "An off-the-shelf pso. proceedings of the particle swarm," in *Optimization Workshop*, 2001.

[74] A. PJ, "Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences," in *Lecture Notes in Computer Sciences*, 1998.

[75] E.-G. A, E.-H. M, S. A, and K. A, "Enhancing the pso via proper parameters selection," in *Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering*, 2002.

[76] K. J, "Small worlds and mega-minds: Effects of neighbourhood topology on particle swarm performance," in *Proceedings of the 1999 IEEE Congress of Evolutionary Computation*, 1999.

[77] van den Bergh F and E. AP, "Cooperative learning in neural networks using particle swarm optimizers," in *Annual Research Conference of South African Institute of Computer Scientists and Information Technologists*, 2001.

[78] ——, "A cooperative approach to particle swarm optimization," in *IEEE Transactions on Evolutionary Computing*, 2004.

[79] ——, "Effects of swarm size on cooperative particle swarm optimisers," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2001.

[80] P. KE and V. MN, "Particle swarm optimization method in multi objective problems," in *Proceedings of the ACM 2002 Symposium on Applied Computing (SAC 2002)*, 2002.

[81] S. XH, L. YH, Z. CG, L. HP, L. WZ, and L. YC, "Hybrid evolutionary algorithms based on pso and ga," in *The 2003 Congress on Evolutionary Computation*, 2003.

[82] K. A. Katare S and W. D, "A hybrid swarm optimizer for efficient parameter estimation," in *Congress on Evolutionary Computation*, 2004.

[83] W. H, L. Y, C. L, and S. Z, "A hybrid particle swarm algorithm with cauchy mutation," in *Proceedings of the 2007 IEEE Swarm Intelligence Symposium*, 2007.

[84] W. J and W. Y, "A dynamical particle swarm algorithm with dimension mutation," in *International Conference on Computational Intelligence and Security*, 2006.

[85] X. J and X. Z, "An extended particle swarm optimizer," in *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium*, 2005.

[86] E. R and S. Y, "Particle swarm optimization: developments, applications and resources," in *Proceedings of the 2001 Congress on Evolutionary Computation*, 2001.

[87] D. J. Krusienski and W. K. Jenkins, "A modified particle swarm optimization algorithm for adaptive filtering," in *IEEE Intl Symposium on Circuits and Systems*, 2006.

[88] K. J., "Some issues and practices for the particle swarms," in *Proceedings of the 2007 IEEE Swarm Intelligence Symposium*, 2007.

[89] C. W. Reynolds, "Flocks, herds and schools: a distributed behavioural model," in *Computer Graphics*, 1987.

[90] F. Heppner and U. Grenander, "A stochastic nonlinear model for coordinated bird flocks," in *In S . Krasner, Ed., The Ubiquity of Chaos. AAAS Publications, Washington, DC*, 1990.

[91] E. Wilson, "Sociobiology: The new synthesis," in *Belknap Press, Cambridge, hlA*, 1975.

[92] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE International Conf. on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ,*, 1995.

[93] E. R. C. Kennedy J and S. Y. H, "Swarm intelligence [m]," in *San Francisco, CA: Morgan Kaufmann*, 2001.

[94] M. M. Millonas, "Swarms, phase transitions, and collective intelligence," in *In C. G. Langton, Ed., Artijicial Life III. Addison Wesley, Reading, MA*, 1994.

[95] W. T. Reeves, "Particle systems - a technique for modeling a class of fuzzy bjects," in *ACM Transactions on Graphics, 2(2):91-108*, 1983.

[96] Y. Shi and R. C. Eberhart, "Comparison between genetic algorithms and particle swarm optimization," in *Evolutionary Programming VII, vol. 1447, Proc. 7th Int. Conf. Evolutionary Programming.*

[97] R. E. X. Hu, Y. Shi, "Recent advances in particle swarm," in *Proc. IEEE Congr. Evol. Comput, vol. 1.*

[98] A. T. Al-Awami, A. Zerguine, L. Cheded, A. Zidori, and W. Saif, "'a new modified particle swarm optimization algorithm for adaptive equalization," in *Digital Signal Processing*, 2011.

[99] K. J. EBERHART R C, "A new optimizer using particle swarm theory," in *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, Oct.

Vitae

- Name: Adil Humayun Khan

- Nationality: Pakistani

- Date of Birth: 12 February 1985

- Email: *aadilhk@gmail.com*

- Permenant Address: House-6, Street-26, Gunj Mughalpura, Lahore, Pakistan

- Phone Number: +966 599929553