On Applications of New Soft and Evolutionary Computing Techniques to Direct and Inverse Modeling Problems

Thesis submitted in partial fulfillment of the requirements for the award of the Doctor of Philosophy

by

Babita Majhi

Roll No. 50609003, Ph. D.

Under the guidance of

Prof. (Dr.) G. Panda, FNAE, FNASc.

Electronics & Communication Engineering

National Institute of Technology

Rourkela - 769008

CERTIFICATE

This is to certify that the thesis entitled "**On Applications of New Soft and Evolutionary Computing Techniques to Direct and Inverse Modeling Problems**" by Ms. Babita Majhi, submitted to the National Institute of Technology, Rourkela for the degree of Doctor of Philosophy, is a record of bonafide research work carried out by her in the department of Electronics and Communication Engineering under my supervision. I believe that the thesis fulfills part of the requirements for the award of degree of Doctor of Philosophy. The results embodied in the thesis have not been submitted for award of any other degree.

Dr. Ganapati Panda, FNAE, FNASc.

Professor
Department of ECE
National Institute of Technology
Rourkela – 769008
India

ACKNOWLEDGEMENT

I am indebted to many people who contributed through their support, knowledge and friendship, to this work and the years at NIT Rourkela.

I am grateful to my supervisor, Prof. G. Panda, who gave me the opportunity to realize this work in the laboratory. He encouraged, supported and motivated me with much kindness throughout the work. I always had the freedom to follow my own ideas, which I am very grateful for. I really admire him for patience and staying power to carefully read the whole thesis. It is his help for which I stand where I am.

I am also grateful to NIT Rourkela for providing me adequate infrastructure to carry out the present investigations.

I am thankful to Prof. K. K. Mahapatra, Prof. S. K. Patra, Prof. S. Meher of Electronics and Communication Engg. department and Prof. U. K. Mohanty of Metallurgical and Materials Engg. department for extending their valuable suggestions and help whenever I approached.

My special thanks to Dr. D. P. Acharya and Ajit Kumar Sahoo for their constant inspiration and encouragement during my research.

My hearty thanks to Sitanshu, Jagganath, Trilochan, Upendra, Sudhansu, Pyari, Nithin, Vikas, Pawan, Sasmita and Piter for their help, cooperation and encouragement.

I acknowledge all staff, research scholars and juniors of ECE department, NIT Rourkela for helping me.

I render my respect to all my family members for giving me mental support and inspiration for carrying out my research work.

(Babita Majhi)
Roll. No. 50609003, Ph. D.

ABSTRACT

Adaptive direct modeling or system identification and adaptive inverse modeling or channel equalization find extensive applications in telecommunication, control system, instrumentation, power system engineering and geophysics. If the plants or systems are nonlinear, dynamic , Hammerstein and multiple-input and multiple-output (MIMO) types, the identification task becomes very difficult.

Further, the existing conventional methods like the least mean square (LMS) and recursive least square (RLS) algorithms do not provide satisfactory training to develop accurate direct and inverse models. Very often these (LMS and RLS) derivative based algorithms do not lead to optimal solutions in pole-zero and Hammerstein type system identification problem as they have tendency to be trapped by local minima.

In many practical situations the output data are contaminated with impulsive type outliers in addition to measurement noise. The density of the outliers may be up to 50%, which means that about 50% of the available data are affected by outliers. The strength of these outliers may be two to five times the maximum amplitude of the signal. Under such adverse conditions the available learning algorithms are not effective in imparting satisfactory training to update the weights of the adaptive models. As a result the resultant direct and inverse models become inaccurate and improper.

Hence there are three important issues which need attention to be resolved. These are :

(i) Development of accurate direct and inverse models of complex plants using some novel architecture and new learning techniques.

(ii) Development of new training rules which alleviates local minima problem during training and thus help in generating improved adaptive models.

(iii) Development of robust training strategy which is less sensitive to outliers in training and thus to create identification and equalization models which are robust against outliers.

These issues are addressed in this thesis and corresponding contribution are outlined in seven Chapters. In addition, one Chapter on introduction, another on required architectures and

algorithms and last Chapter on conclusion and scope for further research work are embodied in the thesis.

A new cascaded low complexity functional link artificial neural network (FLANN) structure is proposed and the corresponding learning algorithm is derived and used to identify nonlinear dynamic plants. In terms of identification performance this model is shown to outperform the multilayer perceptron and FLANN model. A novel method of identification of IIR plants is proposed using comprehensive learning particle swarm optimization (CLPSO) algorithm. It is shown that the new approach is more accurate in identification and takes less CPU time compared to those obtained by existing recursive LMS (RLMS), genetic algorithm (GA) and PSO based approaches. The bacterial foraging optimization (BFO) and PSO are used to develop efficient learning algorithms to train models to identify nonlinear dynamic and MIMO plants. The new scheme takes less computational effort, more accurate and consumes less input samples for training. Robust identification and equalization of complex plants have been carried out using outliers in training sets through minimization of robust norms using PSO and BFO based methods. This method yields robust performance both in equalization and identification tasks. Identification of Hammerstein plants has been achieved successfully using PSO, new clonal PSO (CPSO) and immunized PSO (IPSO) algorithms. Finally the thesis proposes a distributed approach to identification of plants by developing two distributed learning algorithms : incremental PSO and diffusion PSO. It is shown that the new approach is more efficient in terms of accuracy and training time compared to centralized PSO based approach. In addition a robust distributed approach for identification is proposed and its performance has been evaluated.

In essence the thesis proposed many new and efficient algorithms and structure for identification and equalization task such as distributed algorithms, robust algorithms, algorithms for ploe-zero identification and Hammerstein models. All these new methods are shown to be better in terms of performance, speed of computation or accuracy of results.

Contents

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

ADSL	Adaptive digital subscriber loop
AIS	Artificial immune system
AWGN	Additive white Gaussian noise
BER	Bit error ratio
BFO	Bacterial foraging optimization
BIBO	Bounded input bounded output
BP	Back propagation
CF	Cost function
CFLANN	Cascaded functional link artificial neural network
CLPSO	Comprehensive learning particle swarm optimization
CNN	Chebyshev neural network
CPSO	Clonal particle swarm optimization
DPSO	Diffusion particle swarm optimization
DSP	Digital signal processing
EVR	Eigen value ratio
FE	Functional expansion
FIR	Finite impulse response
FLANN	Functional link artificial neural network
FPGA	Field programmable gate array
GA	Genetic Algorithm
HVAC	Heating, ventilating and air conditioning
IIR	Infinite impulse response
ILMS	IIR LMS
INPSO	Incremental particle swarm optimization
IPSO	Immunized particle swarm optimization
ISI	Inter Symbol Interference
LMS	Least mean square
MGS	Mackey glass system
MIMO	Multiple input multiple output
MLANN	Multilayer artificial neural network
MLP	Multilayer perceptron
MLSE	Mean log squared error
MMSE	Minimum mean square error
MSE	Mean square error
NMSE	Normalized mean square error
PPN	Polynomial perceptron network
PSO	Particle swarm optimization
RBF	Radial basis function
RCF	Robust cost function
RLMS	Recursive least mean square
RLS	Recursive least square
RNN	Recurrent neural network
SI	Swarm intelligence
SISO	Single input single output
SSE	Sum squared error

SSE	Sum squared errors
VLSI	Very large scale integrated
WNN	Wavelet neural network

Chapter

1

Introduction

1.1 Background

OUT of many applications of adaptive filtering, direct modeling and inverse modeling are very important. The direct modeling or system identification finds applications in control system engineering including robotics [1.1], intelligent sensor design [1.2], process control [1.3], power system engineering [1.4], image and speech processing [1.4], geophysics [1.5], acoustic noise and vibration control [1.6] and biomedical engineering [1.7]. Similarly inverse modeling technique is used in digital data reconstruction [1.8], channel equalization in digital communication [1.9], digital magnetic data recording [1.10], intelligent sensor [1.2], deconvolution of seismic data [1.11]. The direct modeling mainly refers to adaptive identification of unknown plants. Simple static linear plants are easily identified through parameter estimation using conventional derivative based least mean square (LMS) type algorithms [1.12]. But most of the practical plants are dynamic, nonlinear and combination

of these two characteristics. In many applications Hammerstein and MIMO plants need identification. In addition the output of the plant is associated with measurement or additive white Gaussian noise(AWGN). Identification of such complex plants is a difficult task and poses many challenging problems. Similarly inverse modeling of telecommunication and magnetic medium channels is also important for reducing the effect of inter symbol interference (ISI) and achieving faithful reconstruction of original data. Similarly adaptive inverse modeling of sensors is required to extend their linearities for direct digital readout and enhancement of dynamic range. If the channel or the sensor characteristic is modeled as a nonlinear filter with large eigen-value ratio (EVR) together with AWGN building up of an accurate inverse model is also a difficult and challenging task. These two important and complex issues are addressed in the thesis and attempts have been made to provide improved efficient and alternate promising solutions.

The conventional LMS and recursive least square (RLS) [1.13] techniques work well for identification of static plants but when the plants are of dynamic type, the existing forward-backward LMS [1.14] and the RLS algorithms very often lead to non optimal solution due to premature convergence of weights to local minima [1.15]. This is a major drawback of the use of existing derivative based techniques. To alleviate this burning issue this thesis suggests the use of derivative free optimization techniques in place of conventional techniques.

In recent past population based optimization techniques have been reported which fall under the category of evolutionary computing [1.16] or computational intelligence [1.17]. These are also called bio-inspired techniques which include genetic algorithm (GA) and its variants [1.18], particle swarm optimization (PSO) and its variants [1.19], bacterial foraging optimization (BFO) and its variants [1.20] and artificial immune system (AIS) and its variants [1.21]. These techniques are suitably employed to obtain efficient iterative learning algorithms for developing adaptive direct and inverse models of complex plants and channels.

Development of direct and inverse adaptive models essentially consists of two components. The first component is an adaptive network which may be linear

or nonlinear in nature. Use of a nonlinear network is preferable when nonlinear plants or channels are to be identified or equalized. The linear networks used in the thesis are adaptive linear combiner or all-zero or FIR structure [1.7] and pole-zero or IIR structure[1.7]. Under nonlinear category low complexity single layer function link artificial neural network (FLANN) [1.22] and multilayer perceptron network (MLP) [1.23] are used. The second component is the training or learning algorithm used to train the parameters of the model. As stated earlier the structures used are trained by bio-inspired techniques such as GA, PSO and modified PSOs, BFO and modified BFOs. Depending upon the complexity and nature of the plants to be identified proper combination of network of the model and corresponding bio-inspired learning rule is selected so that the combination yields the best possible performance in direct and inverse modeling tasks. This requires the knowledge of prior experience and simulation results. One of the objectives of the present investigation is to choose models with appropriate combination of structure and algorithm so that it provides best possible performance of direct and inverse models. The bio-inspired optimization tools can not directly be applied to develop direct and inverse models of plants as those are not aimed to be used for training of parameters of models. Therefore another motivation of investigation is to formulate the direct and inverse modeling problems as optimization problems and then to introduce bio-inspired techniques suitably to effectively optimize the cost function of the models. In conventional identification and equalization problems, the mean square error at the output is considered as the cost function to be minimized by using bio-inspired techniques.

In many practical situations the training signal available is highly corrupted by outliers and may be as high as 50%. Under such constraints the training of the models gets severely affected if the squared error is used as the cost function for minimization. This is because this conventional cost function is not robust against outliers [1.24]. In statistics few cost functions have been defined which are robust in nature and are not affected by outliers. These are Wilcoxon norm, $\sigma(1-\exp(-e^2/2\sigma))$ and $\log(1+\frac{e^2}{2})$, where σ is a parameter to be adjusted during

training and e^2 is mean square error. In this thesis robust identification, equalization and time series prediction schemes by minimization of the robust norm using bio-inspired techniques have been proposed. This is a novel contribution in this thesis.

In recent years distributed signal processing has played an important role in sensor networks in which individual node collects local information but the objective is to compute the global solution. Some representative problems are parameter estimation using locally measured data and nonlinear identification using the local data in a cooperative manner. Attempts have been made to solve this interesting problem by using an approach based on newly introduced distributed PSO. In this work two distributed versions: incremental and diffusion type PSO techniques have been proposed and then used for robust identification of linear and nonlinear plants.

1.2 Motivation

In summary the main motivations of the research work carried in the present thesis are the following :

(i) To formulate the direct and inverse modeling problems as error square optimization problems

(ii) To introduce bio-inspired optimization tools such as PSO and BFO and their variants to efficiently minimize the squared error cost function of the models. In other words to develop alternate identification scheme.

(iii) To achieve improved identification (direct modeling) of complex nonlinear all-zero, pole-zero, Hammerstein and MIMO plants and channel equalization (inverse modeling) of nonlinear noisy digital channels by introducing new and improved identification algorithms.

(iv) To devise new bio-inspired training strategy for robust identification of complex plants and robust equalization of complex channels.

(v) To suggest distributed incremental and diffusion type PSO algorithms and use them for identification of linear and nonlinear plants using local data of each sensor node.

(vi) To introduce distributed robust algorithms for identification of nonlinear plants.

1.3 Major contribution of the thesis

The following novel contributions have been made in the thesis :

A low complexity functional link artificial neural network based nonlinear dynamic system identifier has been developed and its learning algorithm has been derived. Improved identification performance has been demonstrated through simulation study.

A comprehensive learning particle swarm optimization technique has been used to effectively identify IIR plants. Further, extensive simulation study has been made on the use of the proposed method to effectively identify higher order plants with lower order models. The new approach has been shown to overcome local minima problem in a multimodal situation.

The bacterial foraging optimization and particle swarm optimization have been used as learning tools in developing new models for identification of dynamic systems. Robust identification and prediction task has also been carried using PSO. Similarly a new approach to develop robust inverse model in presence of outliers has been successfully implemented using BFO.

Identification of complex Hammerstein plants using two new PSO algorithms : clonal PSO and immunized PSO have been proposed. It is shown that the immunized PSO model outperforms its counterpart in all counts.

Distributed incremental and diffusion type PSO algorithms have been suggested for identification of nonlinear plants with outliers in the training signal under a sensor network frame work. The results are observed to be superior to conventional method of identification.

1.4 Chapter wise contribution

The research work undertaken is embodied in 10 Chapters.

1. Introduction
2. Selected adaptive architectures and bio-inspired techniques, principles and algorithms
3. Development of a new cascaded functional link artificial neural network(CFLANN) for nonlinear dynamic system identification
4. Identification of IIR plants using comprehensive learning particle swarm optimization
5. Dynamic systems identification using PSO and BFO based learning algorithms.
6. Robust identification and prediction using particle swarm optimization technique
7. Robust adaptive inverse modeling using bacterial foraging optimization technique and applications
8. Identification of Hammerstein plants using Clonal PSO and Immunized PSO algorithms
9. Development of distributed PSO algorithms for robust nonlinear system identification
10. Conclusion and scope for further work

Out of 10 Chapters, the research contribution is contained in Chapters 3 to 9. A brief outline of chapter wise contribution is presented in sequel. **Chapter 1** outlines the introduction to the problem, the motivation of the research work and a condensed version of chapter wise contribution made in the thesis. Finally **Chapter 10** deals with the overall conclusion of the investigation and scope for further research work.

A brief outline of each of the linear and nonlinear networks used for identification and equalization purpose is presented in **Chapter 2.** This

includes all-zero and pole-zero adaptive filters under linear category and MLP, FLANN, CNN under nonlinear category. This Chapter also reviews the existing derivative based algorithms such as the LMS, recursive LMS (RLMS), FLANN, BP and evolutionary computing optimization algorithms like the GA, PSO, BFO and AIS. Various combinations of the structure and the learning algorithms are suitably used to obtain novel adaptive models for identification and equalization of complex plants and channels respectively.

In **Chapter 3**, a new cascaded FLANN structure is proposed and the corresponding learning algorithm is derived and used to identify nonlinear dynamic plants. Four different identification models have been suggested. Results of identification through simulation demonstrate that the new models outperform the conventional MLP and FLANN based models in terms of computational load and accuracy.

Identification of IIR plants or pole-zero systems finds extensive applications in echo cancellation, channel estimation, process control, array processing and speech recognition. In **Chapter 4** a new adaptive IIR algorithm using comprehensive learning particle swarm optimization (CLPSO) is proposed which avoids potential local minima problem and provides accurate estimate of pole-zero coefficients of IIR plants. Simulation study of identification of some benchmark IIR plants reveals that the proposed method outperforms the existing recursive LMS (RLMS), GA and PSO based methods in terms of mean square error(MSE), execution time and product of population size and number of input samples used during training.

Chapter 5 deals with the development of PSO and BFO based schemes to identify nonlinear dynamic single-input-single-output (SISO) and multiple-input-multiple-output (MIMO) systems. The BFO and PSO based training of the weights of the FLANN identification model have been newly introduced. In both cases it is observed that the new training schemes in complex identification task work better in terms of speed of computation, accuracy and number of input samples used for training. However, both the new schemes offer almost similar identification performance.

The problem of robust identification and prediction is studied in depth in **Chapter 6**. Development of such identification scheme is required when either the training or input signal samples are contaminated with outliers. The existing squared error cost function based learning schemes fail to offer satisfactory identification performance. Therefore the use of new robust norms which are insensitive to outliers have been suggested as the cost function. Such cost functions are minimized by PSO method to develop robust identification of nonlinear dynamic systems and prediction models of complex time series. Simulation results show that the Wilcoxon norm produces the best robust models for identification and prediction compared to that produced by other norms used in the investigation.

Inverse modeling plays an important role in channel equalization, sensor linearization and deconvolution operation in geophysics applications. In practice it is difficult to develop an inverse model using squared error norm when the training signal contains outliers. Therefore investigation has been made in **Chapter 7** to identify robust norms of errors and use them to develop robust inverse models. To obtain such models the robust norms are minimized using BFO scheme. The robustness of the new inverse models is evaluated through simulation study using some benchmark channels and different percentage of outliers in the training signal. The results indicate that the use of squared error provides the least robust inverse models where as the Wilcoxon norm generates most robust models.

Hammerstein plant contains all features of complex plants because it contains a static nonlinear part, a dynamic linear system and an additive coloured noise. Identification of such complex plants really poses difficulty. In **Chapter 8**, attempt has been made to identify such plants using two new PSO algorithms : clonal PSO (CPSO) and immunized PSO (IPSO). In this Chapter two new variants of PSO algorithm have been proposed and then used them in training the model parameters. The potentiality of the proposed method is demonstrated through simulation of benchmark Hammerstein plants. The potentiality of identification is evaluated by verifying three features : the response matching at the output of static nonlinear part, comparison of

estimated parameters of linear dynamic part with the corresponding true values and comparison of sum of squared errors (SSE) between true and overall estimated responses. Comparing all these test results, it is observed that the IPSO model outperforms its counterpart in all counts.

In application like sensor networks linear and nonlinear identifications are required using local data of each sensor. To achieve this objective distributed signal processing algorithm for training is required. In **Chapter 9** two such distributed algorithms known as incremental and diffusion PSO (INPSO and DPSO) algorithms have been proposed to identify linear and nonlinear plants. Further the squared and Wilcoxon norm of errors are minimized in a cooperative manner using distributed PSO algorithms. Simulation results demonstrate that both the distributed algorithms provide excellent identification performance when conventional squared error norm is used. When outliers are present in the training samples both the Wilcoxon norm based distributed algorithms provide superior performance compared to the conventional norm based training.

The overall conclusion of the total investigation is listed in **Chapter 10**. This Chapter also contains the details of further research work that can be carried out in the same or the related field.

References

[1.1] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks", IEEE Trans. on Neural Networks, vol. 1, pp. 4-26, January 1990.

[1.2] J. C. Patra, A. C. Kot and G. Panda, "An intelligent pressure sensor using neural networks", IEEE Trans. on Instrumentation and Measurement, vol. 49, issue 4, pp. 829-834, Aug. 2000.

[1.3] M. Pachter and O. R. Reynolds, "Identification of a discrete time dynamical system", IEEE Trans. Aerospace Electronic System, vol. 36, issue 1, pp. 212-225, 2000.

[1.4] G. B. Giannakis and E. Serpedin, "A bibliography on nonlinear system identification", Signal Processing, vol. 83, no. 3, pp. 533-580, 2001.

[1.5] E. A. Robinson and T. Durrani, *Geophysical Signal Processing*, Prentice-Hall, Englewood Cliffs, NJ, 1986.

[1.6] D. P. Das and G. Panda, "Active mitigation of nonlinear noise processes using a novel filtered-s LMS algorithm", IEEE Trans. on Speech and Audio Processing, vol. 12, issue 3, pp. 313-322, May 2004.

[1.7] B. Widrow and S.D. Sterns, "*Adaptive Signal Processing*" Prentice-Hall, Inc. Engle-wood Cliffs, New Jersey, 1985.

[1.8] G. J. Gibson, S. Siu and C. F. N. Cowan, "The application of nonlinear structures to the reconstruction of binary signals", IEEE Trans. signal processing, vol. 39, no. 8, pp. 1877-1884, Aug. 1991.

[1.9] R. W. Lucky, Techniques for adaptive equalization of digital communication systems, Bell Sys.Tech. J., 45, 255-286, Feb. 1966.

[1.10] H. Sun, G. Mathew and B. Farhang-Boroujeny, "Detection techniques for high density magnetic recording", IEEE Trans. on Magnetics, vol. 41, no. 3, pp. 1193-1199, March 2005.

[1.11] L. J. Griffiths, F. R. Smolka and L. D. Trenbly, "Adaptive deconvolution : a new technique for processing time varying seismic data", Geohysics, June 1977.

[1.12] B. Widrow, J. M. McCool, M. G. Larimore and C. R. Johnson, Jr., :Stationary and nonstationary learning characteristics of the LMS adaptive filter", Proc. IEEE, vol. 64, no. 8, pp. 1151-1162, Aug., 1976.

[1.13] B. Friedlander and M. Morf, "Least-squares algorithms for adaptive linear phase filtering", IEEE Trans., vol. ASSP-30, no. 3, pp. 381-390, June 1982.

[1.14] S. A. White, "An adaptive recursive digital filter", Proc. 9th Asilomar Conf. Circuits Syst. Comput., p. 21, Nov. 1975.

[1.15] John J. Shynk, "Adaptive IIR filtering", IEEE ASSP Magazine, April 1989, pp. 4-21.

[1.16] A. E. Eiben and J. E. Smith, "*Introduction to Evolutionary Computing*", Springer, 2003, ISBN 3-540-40184-9.

[1.17] Andries Engelbrecht, "*Computational Intelligence : An introduction*", Wiley & Sons, ISBN 0-470-84870-7.

[1.18] D.E.Goldberg, "*Genetic algorithms in search, optimization and machine learning*", Addition-Wesley,1989.

[1.19] J. Kennedy, R. C. Eberhart and Y. Shi, "*Swarm intelligence*", San Francisco: Morgan Kaufmann Publishers, 2001.

[1.20] K. M. Passino, "Biomimicry of Bacterial Foraging for distributed optimization and control", IEEE control system magazine, vol 22, issue 3, pp. 52-67, June 2002.

[1.21] D. Dasgupta, *Artificial Immune Systems and their Applications*, Springer-Verlag, 1999.

[1.22] Y.H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison Wesley, Reading, Massachusetts, 1989.

[1.23] S. Haykin, "*Neural Networks: A comprehensive foundation*" 2nd Edition, Pearson Education Asia, 2002.

[1.24] Jer-Guang Hsieh, Yih-Lon Lin and Jyh-Horng Jeng, "Preliminary study on Wilcoxon learning machines", IEEE Trans. on neural networks, vol. 19, no. 2, pp. 201-211, Feb. 2008.

Chapter

2

Selected Adaptive Architectures and Bio-Inspired Techniques, Principles and Algorithms

2.1 Introduction

THE main motive of the research work carried out in this thesis is to develop elegant and efficient adaptive identification schemes for complex nonlinear and dynamic plants, adaptive inverse models of nonlinear plants, equalization of complex channels and prediction of nonlinear time series. All these adaptive models inherently need suitable adaptive structures and appropriate learning rules to train the parameters of these models. In the present investigation, I briefly outline some selected adaptive architectures such as adaptive linear combiner, adaptive pole-zero filters, functional link artificial neural network and multilayer artificial neural network. In addition we present some recently developed population based bio-inspired derivative free techniques such as particle swarm optimization and its

variants, bacterial foraging optimization and its variants and artificial immune system and its variants for training the parameters or coefficients of the adaptive structures. An adaptive linear combiner or filter is feed forward in structure. These are [2.1, 2.2] often realized either as a set of program instructions running on an arithmetical processing device such as a microprocessor or DSP chip, or as a set of logic operations implemented in a field-programmable gate array (FPGA) or in a semi-custom or custom Very large scale integrated (VLSI) circuit. An adaptive linear combiner is characterized by

1. the input signal sampled employed,
2. the structure that defines how the output signal of the combiner is computed from its input samples,
3. the parameters of the structure which are iteratively changed based on some learning rule and
4. the adaptive algorithms that guide how the parameters are to be adjusted iteratively until the predefined objective is fulfilled.

By choosing a particular adaptive filter structure, one specifies the number and type of parameters that need adjustments. The adaptive algorithms used to update these parameters tend to minimize the cost function of the model. The cost function normally are mean squared error of the model which is not robust to outliers in the training or desired signal samples. The thesis also introduces minimization of some robust cost functions of the error for updating the model parameters. Essentially the development of adaptive models for identification, equalization, prediction and function approximation is viewed as a minimization problem of some suitable cost functions. The main contribution of the thesis is to solve these complex optimization problems using bio-inspired based learning rules.

In following section, the general adaptive filtering problem is presented and the mathematical notation for representing the form and operation of the adaptive filter is introduced.

2.2 The adaptive filtering problem

Figure 2.1 shows a block diagram of an adaptive FIR filter or an adaptive linear combiner in which a sample from a digital input signal x_k is fed into an adaptive filter, that computes a corresponding output signal sample y_k at time k. The output signal is compared to a second signal d_k, called the desired signal.

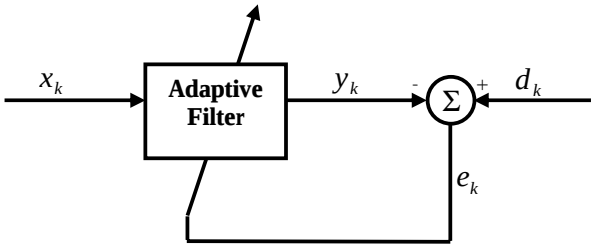

Fig. 2.1 The general adaptive filtering problem

The difference signal given by

$$e_k = d_k - y_k \tag{2.1}$$

is known as the error signal. The error signal is used to adapt the parameters of the filter from time k to time $(k+1)$ in a well-defined manner. This process of adaptation is represented by an oblique arrow. As the time index k is incremented the output of the adaptive filter matches a better and better to the desired signal following an adaptation process such that the magnitude of e_k decreases over time.

In the adaptive filtering framework, adaptation refers to the mechanism by which the parameters of the system are changed from time index k to time index $(k+1)$. The number and types of parameters within this system depend on the computational structure chosen for the system. Different filter structures that have been used for model development are presented below.

2. 2. 1 Adaptive FIR filter

The general architecture of an FIR adaptive filter or a adaptive linear combiner [2.1] is depicted in Fig. 2.2. Let X is N^{th} input pattern having one unit delay in each instant. This process is also called as adaptive linear combiner [2.1-2.2]. Let $X_n = [x(n) \quad x(n-1)..... x(n-M+1)]$ form of the M -by-1 tap input vector and $M-1$ is the number of delay elements. The tap weights $W_n = [w_0(n) \quad w_1(n) ..... w_{M-1}(n)]^T$ form the elements of the M -by-1 tap weight vector. The output is represented as,

$$y(n) = \sum_{m=0}^{M-1} w_m(n) x(n-m) \tag{2.2}$$

The output can be represented in vector notation as

$$y(n) = X_n^T W_n = W_n^T X_n \tag{2.3}$$

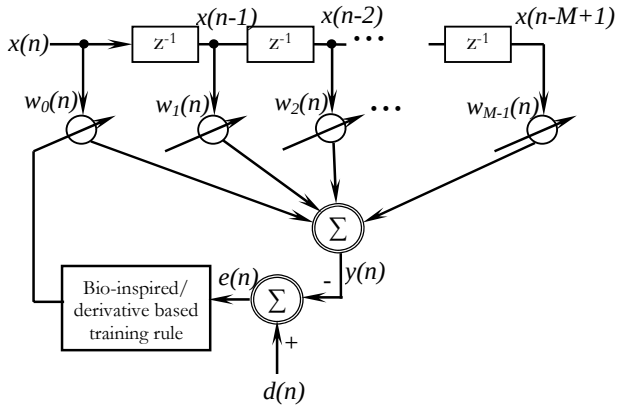

Fig. 2.2 Adaptive filter using Bio-inspired/Derivative based algorithms

2. 2. 2 Adaptive IIR filter

The structure of a direct-form adaptive IIR filter [2.1] is shown in Fig. 2.3. In this case, the output of the system is given by

$$y(n) = \sum_{m=1}^{N-1} a_m(n) y(n-m) + \sum_{m=0}^{M-1} b_m(n) x(n-m) \tag{2.4}$$

The terms $a_m(n)$ and $b_m(n)$ represent the feed forward and feed back coefficients of the filter respectively. In matrix form, $y(n)$ may be written as

$$y(n) = W_n^T S_n \tag{2.5}$$

Where the combined weight vector is

$$W_n = [b_0(n)\ b_1(n) \ldots\ldots\ldots\ldots b_{M-1}(n)\ \ a_1(n)\ \ a_2(n) \ldots\ldots\ldots\ldots a_{N-1}(n)]^T \tag{2.6}$$

And the combined input and output signal vector is

$$S_n = [x(n)\ \ x(n-1) \ldots\ldots\ldots x(n-M+1)\ \ y(n-1)\ \ y(n-2) \ldots\ldots\ldots y(n-N+1)]^T \tag{2.7}$$

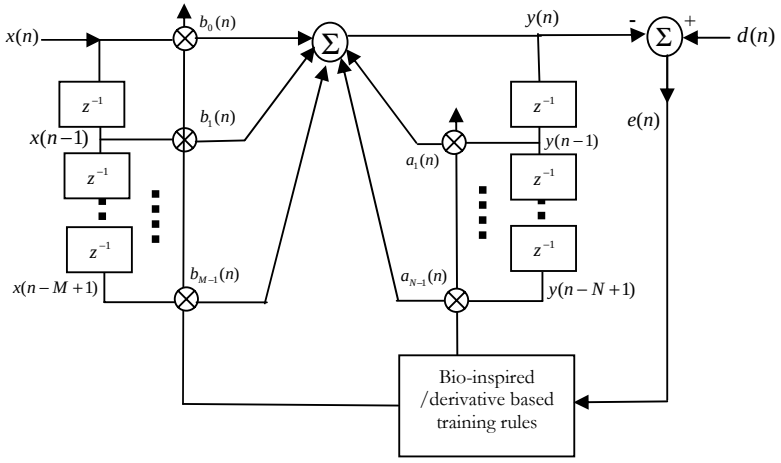

Fig. 2.3 Structure of an adaptive IIR filter

The weight update operation of adaptive IIR filter is carried out using either conventional derivative based or derivative free learning algorithms. In addition to the linear structures nonlinear structure can be used for which the principle of superposition does not hold when the parameter values are fixed. Such systems are useful when the relationship between $d(n)$ and $x(n)$ is not linear in nature. This class of nonlinear structure consists of artificial neural network (ANN), functional link artificial neural network (FLANN) and radial basis function (RBF) network. These networks inherently contain distributed nonlinear elements in each path like the sigmoid function in ANN, sine / cosine terms in FLANN and Gaussian function in RBF network. In the next section details of these nonlinear structures are dealt.

2.3 Artificial neural network (ANN)

Artificial neural network (ANN) takes its name from the network of nerve cells in the brain. Recently, ANN has proved to be an important technique for classification and optimization problems [2.3-2.5]. McCulloch and Pitts have developed the neural networks for different computing machines. There are extensive applications of various types of ANN in the field of communication, control, instrumentation and forecasting. The ANN is capable of performing nonlinear mapping between the input and output space due to its large parallel interconnection between different layers and the nonlinear processing characteristics. An artificial neuron basically consists of a computing element that performs the weighted sum of the input signal and the connecting weight. The sum is added with the bias or threshold and the resultant signal is then passed through a nonlinear function of sigmoid or hyperbolic tangent type. Each neuron is associated with three parameters whose learning can be adjusted; these are the connecting weights, the bias and the slope of the nonlinear function. For the structural point of view, a neural network (NN) may be single layer or it may be multilayer. In multilayer structure, there is one or many artificial neurons in each layer and for a practical case there may be a number of layers. Each neuron of the one layer is connected to each neuron of the next layer. The functional-link ANN is another type of single layer NN. In this type of network the input data is allowed to pass through a functional expansion block where the input data are nonlinearly mapped to more number of points. This is achieved by using trigonometric functions, tensor products or power terms of the input. The output of the functional expansion is then passed through a single neuron.

Twp types of NNs used in this thesis are discussed next.

2. 3. 1 Single neuron structure

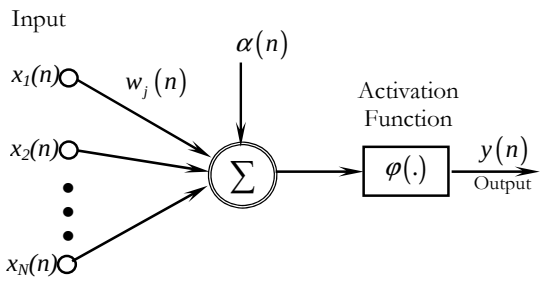

Fig. 2.4 Structure of a single neuron

The basic structure of an artificial neuron is presented in Fig. 2.4. The operation in a neuron involves the computation of the weighted sum of inputs and threshold [2.3-2.5]. The resultant signal is then passed through a nonlinear activation function. This is also called as a perceptron, which is built around a nonlinear neuron. The output of the neuron may be represented as,

$$y(n) = \varphi\left(\sum_{j=1}^{N} w_j(n)\, x_j(n) + \alpha(n)\right) \tag{2.8}$$

where $\alpha(n)$ is the threshold to the neurons at the first layer, $w_j(n)$ is the weight associated with the j^{th} input, N is the no. of inputs to the neuron and $\varphi(.)$ is the nonlinear activation function. Different types of nonlinear function are shown in Fig. 2.5.

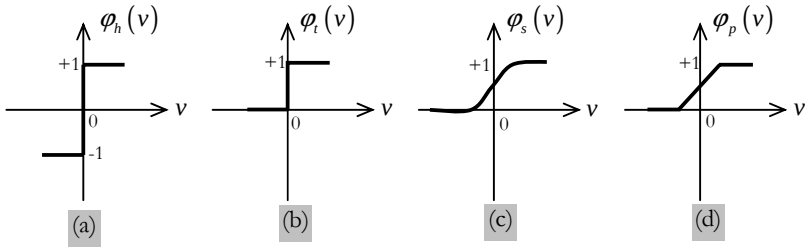

Fig. 2.5 Different types of nonlinear activation function,
(a) Signum function or hard limiter,
(b) Threshold function,
(c) Sigmoid function,
(d) Piecewise Linear

Signum Function: For this type of activation function, we have

$$\varphi(v) = \begin{cases} 1 & if\ v > 0 \\ 0 & if\ v = 0 \\ -1 & if\ v < 0 \end{cases} \tag{2.9}$$

Threshold Function: This function is represented as,

$$\varphi(v) = \begin{cases} 1 & if\ v \geq 0 \\ 0 & if\ v < 0 \end{cases} \tag{2.10}$$

Sigmoid Function: This function is S-shaped and is the most common form of the activation function used in artificial neural network. It is a function that exhibits a graceful balance between linear and nonlinear behaviour.

$$\varphi(v) = \frac{1}{1+e^{-av}} \tag{2.11}$$

where v is the input to the sigmoid function and a is the slope of the sigmoid function. For the steady convergence a proper choice of a is required.

Piecewise-Linear Function: This function is

$$\varphi(v) = \begin{cases} 1, & v \geq +\frac{1}{2} \\ v, & +\frac{1}{2} > v > -\frac{1}{2} \\ 0, & v \leq +\frac{1}{2} \end{cases} \tag{2.12}$$

where the amplification factor inside the linear region of operation is assumed to be unity. Out of these nonlinear functions the sigmoid activation function is extensively used in ANN.

2. 3. 2 Multilayer perceptron (MLP)

In the multilayer neural network or multilayer perceptron (MLP), the input signal propagates through the network in a forward direction, on a layer-by-layer basis. This network has been applied successfully to solve some difficult and diverse problems by training in a supervised manner with a highly popular algorithm known as the error back-propagation algorithm [2.3, 2.4]. The scheme of MLP using four layers is shown in Fig. 2.6. $x_i(n)$ represent the input to the network, f_j and f_k represent the output of the two hidden layers and $y_l(n)$ represents the output of the final layer of the neural network. The connecting weights between the input to the first hidden layer, first to second hidden layer and the second hidden layer to the output layers are represented by w_{ij}, w_{jk} and w_{kl} respectively.

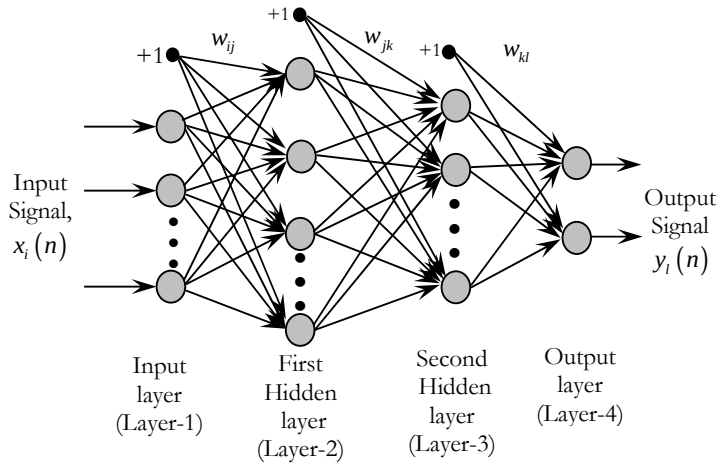

Fig. 2.6 MLP Structure

If P_1 is the number of neurons in the first hidden layer, each element of the output vector of first hidden layer may be calculated as,

$$f_j = \varphi_j \left(\sum_{i=1}^{N} w_{ij} x_i(n) + \alpha_j \right), \quad i = 1,2,3,...N, \; j = 1,2,3,...P_1 \tag{2.13}$$

where α_j is the threshold to the neurons of the first hidden layer, N is the number of inputs and $\varphi(.)$ is the nonlinear activation function of the neurons of the first hidden layer which is defined in (2.11). The time index n has been dropped to make the equations simpler. Let P_2 be the number of neurons in the second hidden layer. The output of this layer is represented as, f_k and may be written as

$$f_k = \varphi_k \left(\sum_{j=1}^{P_1} w_{jk} f_j + \alpha_k \right), \quad k=1, 2, 3, \ldots, P_2 \tag{2.14}$$

where, α_k is the threshold to the neurons of the second hidden layer. The output of the final output layer can be calculated as

$$y_l(n) = \varphi_l \left(\sum_{k=1}^{P_2} w_{kl} f_k + \alpha_l \right), \quad l=1, 2, 3, \ldots, P_3 \tag{2.15}$$

where, α_l is the threshold to the neuron of the final layer and P_3 is the number of neurons in the output layer. The output of the MLP may be expressed as

$$y_l(n) = \varphi_n \left(\sum_{k=1}^{P_2} w_{kl} \varphi_k \left(\sum_{j=1}^{P_1} w_{jk} \varphi_j \left(\sum_{i=1}^{N} w_{ij} x_i(n) + \alpha_j \right) + \alpha_k \right) + \alpha_l \right) \tag{2.16}$$

The details of BP algorithm used to train the weights of various layers of the ANN are discussed in 2. 4. 1. 3.

2. 3. 3 Functional link artificial neural network (FLANN)

Pao originally proposed FLANN which is a novel single layer ANN structure capable of forming arbitrarily complex decision regions by generating nonlinear decision boundaries [2.6, 2.7]. In this structure, the initial representation of a pattern is enhanced by using nonlinear function and thus the pattern dimension space is increased. The functional link acts on an element of a pattern or entire pattern itself by generating a set of linearly independent function and then evaluates these functions with the pattern as the argument. Hence separation of the patterns becomes possible in the enhanced space. The use of FLANN not only increases the learning rate but also has less computational complexity [2.9]. Pao *et al* [2.8] have investigated the learning and generalization characteristics of a random vector FLANN and compared with those attainable with MLP structure trained with back propagation algorithm by taking few functional approximation problems. A FLANN structure with two inputs is shown in Fig. 2.7.

Let $\mathbf{X}$ is the input vector of size $N \times 1$ which represents N number of elements; the n^{th} element is given by:

$$\mathbf{X}(n) = x_n, 1 \le n \le N \tag{2.17}$$

Each element undergoes nonlinear expansion to form M elements such that the resultant matrix has the dimension of $N \times M$.

The functional expansion of the element x_n by power series expansion is carried out using the equation given in (2.18)

$$s_i = \begin{matrix} x_n \text{ for } i = 1 \\ x_n^l \text{ for } i = 2,3,4,\ldots,M \end{matrix} \tag{2.18}$$

where $l = 1,2,\cdots,M$.

For trigonometric expansion, the expanded elements are

$$s_i = \begin{matrix} x_n & \text{for } i = 1 \\ \sin\left(l\pi x_n\right) & \text{for } i = 2,4,\ldots,M \\ \cos\left(l\pi x_n\right) & \text{for } i = 3,5,\ldots,M+1 \end{matrix} \tag{2.19}$$

where $l = 1,2,\cdots,M/2$.

For Chebyshev expansion the terms are given by

$T_0(x_n) = 1$ for $n = 0$

$T_1(x_n) = x_n$ for $n = 1$

$T_2(x_n) = 2x_n^2 - 1$ for $n = 2$

$$T_{n+1}(x_n) = 2x_n T_n(x_n) - T_{n-1}(x_n) \text{ for } n > 2 \tag{2.20}$$

In matrix notation the expanded elements of the input vector is denoted by **S** of size $N\times(M+1)$.

The bias input is unity. So an extra unity value is padded with the **S** matrix and the dimension of the **S** matrix becomes $N\times Q$, where $Q = \left(M + 2\right)$.

Let the weight vector is represented as **W** having Q elements. The output y is given as

$$y = \sum_{i=1}^{Q} s_i w_i \tag{2.21}$$

In matrix notation the output is written as

$$\mathbf{Y} = \mathbf{S} \cdot \mathbf{W}^{\mathrm{T}} \tag{2.22}$$

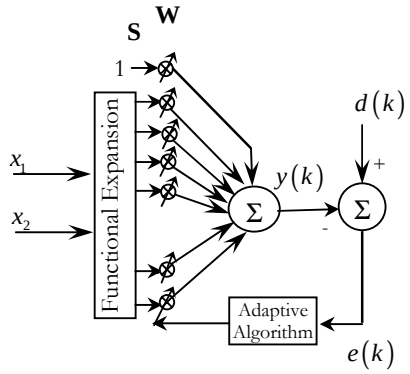

Fig. 2.7 Structure of the FLANN model

2. 4 Learning algorithms

There are many learning algorithms which are employed to train various adaptive models. The performance of these models depends on rate of convergence, training time, computational complexity involved and minimum mean square error achieved after training. The learning algorithms may be broadly classified into two categories (a) derivative based (b) derivative free. The derivative based algorithms include least means square(LMS), IIR LMS (ILMS), back propagation(BP) and FLANN-LMS. Under the derivative free algorithms, genetic algorithm(GA), particle swarm optimization(PSO), bacterial foraging optimization(BFO) and artificial immune system(AIS) have been employed. In this section the details of these algorithms are outlined in sequel.

2.4.1 Derivative based algorithms

These algorithms are gradient search in nature and have been derived by taking derivative of the squared error as the cost function. During the process of training these algorithms tend to drive the weights of the model to local minima. This leads to premature termination of the weights. As

a result the mean square error does not attain the least possible value and hence the accuracy of prediction becomes inferior. However these learning algorithms are simple to implement and can be expressed in close form equations. A brief description of each of them is presented below.

2.4.1.1 LMS algorithm for adaptive FIR filters

In an adaptive FIR filter, at any k th instant the error signal, e_k is computed as

$$e_k = d_k - y_k \tag{2.23}$$

where

d_k = the desired or training signal at k th instant

y_k = the output of the filter at k th instant

The weights associated with the filter are then updated using the LMS algorithm [2.1]. The weight updation equation for n^{th} instant is given by

$$w_k(n+1) = w_k(n) + \Delta w_k(n) \tag{2.24}$$

where $\Delta w_k(n)$ is the change of k th weight at n th iteration.

The change in weight of each path in each iteration is obtained by minimizing the mean squared error [2.1]. Using this value the weight update equation is given as

$$w_k(n+1) = w_k(n) + 2 \cdot \eta \cdot e_k(n) \cdot \mathbf{X}_k^{\mathrm{T}} \tag{2.25}$$

where η is the learning rate parameter ($0 \leq \eta \leq 1$). This procedure is repeated till the mean square error (MSE) of the network approaches a minimum value. The MSE at the time index k may be defined as, $\xi = E\left[e_k^2\right]$, where $E[.]$ is the expectation value or average of the signal.

2.4.1.2 Adaptive IIR LMS (ILMS) algorithm

Referring to Fig. 2.3, the error term is given by

$$e_k = d_k - y_k \tag{2.26}$$

where y_k is obtained from (2.4) and (2.5). The ILMS update rule is given as

$$W_{k+1} = W_k - M\ \hat{\nabla}_k \tag{2.27}$$

where $\hat{\nabla}_k$ = estimate of the gradient at k th instant and is given by

$$\hat{\nabla}_k = -2e_k[\alpha_{0k} \ldots\ldots\ldots \alpha_{Nk} \quad \beta_{1k} \ldots\ldots\ldots \beta_{Nk}]^T \tag{2.28}$$

M is a diagonal matrix of $N+1$ learning parameters for zero coefficients and N parameters for pole coefficients and is represented as

$$M = diag[\mu \ldots\ldots\ldots\ldots \mu \quad \eta \ldots\ldots\ldots\ldots \eta] \tag{2.29}$$

The variables α_{nk} and β_{nk} are given by

$$\alpha_{nk} = \frac{\partial y_k}{\partial a_n} = x_{k-n} + \sum_{l=1}^{L} b_l\ \alpha_{n,\,k-l}; \qquad 0 \le n \le L \tag{2.30}$$

$$\beta_{nk} = \frac{\partial y_k}{\partial b_n} = y_{k-n} + \sum_{l=1}^{L} b_l\ \beta_{n,\,k-l}; \qquad 1 \le n \le L \tag{2.31}$$

Equations (2.26) to (2.31) represent the key equations of ILMS algorithm.

2.4.1.3 Back propagation (BP) algorithm

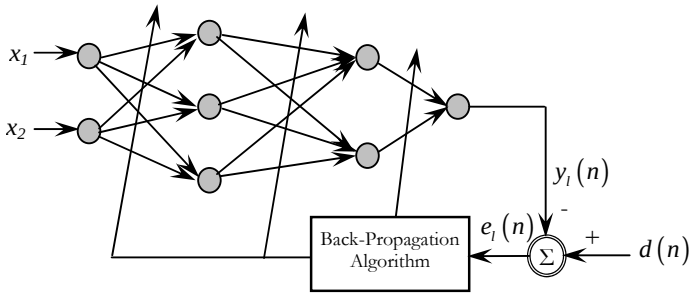

Fig. 2.8 Neural network using BP algorithm

The generalized structure of MLP is shown in Fig. 2.6. To derive the BP algorithm a simplified neural network with two inputs and 2-3-2-1 neurons (2, 3, 2 and 1 denote the number of neurons in the input layer, the first hidden layer, the second hidden layer and the output layer respectively) is depicted in Fig. 2.8. The parameters of the neural network can be updated in both sequential and batch mode of operation. In BP algorithm, the weights and the thresholds are initialized as very small random values. The intermediate and the final outputs of the MLP are calculated by using (2.13), (2.14), and (2.15) respectively.

The final output $y_l(n)$ at the output of neuron l, is compared with the desired output $d(n)$ and the resulting error signal $e_l(n)$ is obtained as

$$e_l(n) = d(n) - y_l(n) \tag{2.32}$$

The instantaneous value of the total error energy is obtained by summing all errors squared over all neurons in the output layer, that is

$$\xi(n) = \frac{1}{2}\sum_{l=1}^{P_3} e_l^2(n) \tag{2.33}$$

where P_3 is the no. of neurons in the output layer.

This error signal is used to update the weights and thresholds of the hidden layers as well as the output layer. The reflected error components at each of the hidden layers is computed using the errors of the last layer and the connecting weights between the hidden and the last layer and error obtained at this stage is used to update the weights between the input and the hidden layer. The thresholds are also updated in a similar manner as that of the corresponding connecting weights. The weights and the thresholds are updated in an iterative method until the error signal becomes minimum.

The weights are updated according to the following equations

$$w_{kl}(n+1) = w_{kl}(n) + \Delta w_{kl}(n) \tag{2.34}$$

$$w_{jk}(n+1) = w_{jk}(n) + \Delta w_{jk}(n) \tag{2.35}$$

$$w_{ij}(n+1) = w_{ij}(n) + \Delta w_{ij}(n) \tag{2.36}$$

where, $\Delta w_{kl}(n), \Delta w_{jk}(n)$ and $\Delta w_{ij}(n)$ are the change in weights of the second hidden layer-to-output layer, first hidden layer-to-second hidden layer and input layer-to-first hidden layer respectively. This can be computed as

$$\begin{aligned} \Delta w_{kl}(n) &= -2\mu \frac{d\xi(n)}{dw_{kl}(n)} = 2\mu e(n) \frac{dy_l(n)}{dw_{kl}(n)} \\ &= 2\mu e(n) \varphi_l' \left(\sum_{k=1}^{P_2} w_{kl} f_k + \alpha_l \right) f_k \end{aligned} \tag{2.37}$$

where, μ is the convergence coefficient ($0 \le \mu \le 1$). Similarly $\Delta w_{jk}(n)$ and $\Delta w_{ij}(n)$ can also be computed as

The thresholds of each layer can be updated in a similar manner using equations

$$\alpha_l(n+1) = \alpha_l(n) + \Delta\alpha_l(n) \tag{2.38}$$

$$\alpha_k(n+1) = \alpha_k(n) + \Delta\alpha_k(n) \tag{2.39}$$

$$\alpha_j(n+1) = \alpha_j(n) + \Delta\alpha_j(n) \tag{2.40}$$

where, $\Delta\alpha_l(n), \Delta\alpha_k(n)$ and $\Delta\alpha_j(n)$ are the change in thresholds of the output, hidden and input layer respectively. The change in threshold is represented as,

$$\begin{aligned} \Delta\alpha_l(n) &= -2\mu \frac{d\xi(n)}{d\alpha_l(n)} = 2\mu e(n) \frac{dy_l(n)}{d\alpha_l(n)} \\ &= 2\mu e(n) \varphi_l' \left(\sum_{k=1}^{P_2} w_{kl} f_k + \alpha_l \right) \end{aligned} \tag{2.41}$$

2.4.1.4 The FLANN algorithm

Referring the structure of the FLANN of Fig. 2.7, the error signal $e(k)$ at k^{th} iteration can be computed as

$$e(k) = d(k) - y(k) \tag{2.42}$$

Let $\xi(k)$ denotes the cost function at iteration k and is given by

$$\xi(k) = \frac{1}{2}\sum_{j=1}^{P} e_j^2(k) \tag{2.43}$$

where P is the number of nodes at the output layer.

The weight vector can be updated by least mean square (LMS) algorithm, as

$$w(k+1) = w(k) - \frac{\mu}{2}\hat{\nabla}(k) \tag{2.44}$$

where $\hat{\nabla}(k)$ is an instantaneous estimate of the gradient of ξ with respect to the weight vector $w(k)$. This gradient is computed as

$$\begin{aligned}\hat{\nabla}(k) &= \frac{\partial \xi}{\partial w} = -2e(k)\frac{\partial y(k)}{\partial w} = -2e(k)\frac{\partial [w(k)s(k)]}{\partial w} \\ &= -2e(k)s(k)\end{aligned} \tag{2.45}$$

Substituting the values of $\hat{\nabla}(k)$ in (2.44) we get

$$w(k+1) = w(k) + \mu e(k)s(k) \tag{2.46}$$

where μ denotes the step-size $(0 \le \mu \le 1)$, which controls the convergence speed of the algorithm.

2.5 Derivative free algorithms / Evolutionary computing based algorithms

2.5.1 Genetic algorithm (GA)

Genetic algorithms are a class of evolutionary computing techniques, which is a rapidly growing area of artificial intelligence. Genetic algorithms are inspired by Darwin's theory of evolution. Simply said, problems are solved by an evolutionary process resulting in a best (fittest) solution (survivor) - in other words, the solution is evolved.

Evolutionary computing was introduced in the 1960s by Rechenberg in his work "*Evolution strategies*" (*Evolutionsstrategie* in original). His idea was then developed by other researchers. Genetic Algorithms (GAs) were invented by John Holland and developed by him and his students and colleagues [2.10]. This led to Holland's book "*Adaption in Natural and Artificial Systems*" published in 1975.

The algorithm begins with a set of solutions (represented by chromosomes) called population. Solutions from one population are taken and used to form a new population. This is motivated by a hope, that the new population will be better than the old one. Solutions which are then selected to form new solutions (offspring) are selected according to their fitness - the more suitable they are, the more chances they have to reproduce.

This is repeated until some condition (for example number of populations or improvement of the best solution) is satisfied.

2.5.1.1 Outline of the basic genetic algorithm

1. **[Start]** Generate random population of n chromosomes (suitable solutions for the problem)
2. **[Fitness]** Evaluate the fitness $f(x)$ of each chromosome x in the population
3. **[New population]** Create a new population by repeating following steps until the new population is complete
    a **[Selection]** Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)

b **[Crossover]** With a crossover probability cross over the parents to form new offspring (children). If no crossover was performed, offspring is the exact copy of parents.

c **[Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome).

d **[Accepting]** Place new offspring in the new population

4. **[Replace]** Use new generated population for a further run of the algorithm
5. **[Test]** If the end condition is satisfied, **stop**, and return the best solution in current population
6. **[Loop]** Go to step **2**

The outline of the Basic GA provided above is very general. There are many parameters and settings that can be implemented differently in various problems.

Elitism is often used as a method of selection. Which means, that at least one of a generation's best solution is copied without changes to a new population, so the best solution can survive to the succeeding generation.

2.5.1.2 Operators of GA

Overview

The crossover and mutation are the most important parts of the genetic algorithm. The performance is influenced mainly by these two operators.

Encoding of a Chromosome

A chromosome should in some way contain information about solution that it represents. The most commonly used way of encoding is a binary string. A chromosome then could look like this:

Chromosome 1	1101100100110110
Chromosome 2	1101111000011110

Fig. 2.9 Chromosome

Each chromosome is represented by a binary string. Each bit in the string can represent some characteristics of the solution. There are many other ways of encoding. The encoding depends mainly on the problem to be solved. For example, one can encode directly integer or real numbers; sometimes it is useful to encode some permutations and so on.

Crossover

Crossover operates on selected genes from parent chromosomes and creates new offspring. The simplest way how to do that is to choose randomly some crossover point and copy everything before this point from the first parent and then copy everything after the crossover point from the other parent.

Crossover is illustrated in the following Fig. 2.10 (| is the crossover point)

Chromosome 1	11011 | 00100110110
Chromosome 2	11011 | 11000011110
Offspring 1	11011 | 11000011110
Offspring 2	11011 | 00100110110

Fig. 2.10 Crossover

There are other ways how to make crossover, for example we can choose more crossover points.

Mutation

Mutation is intended to prevent falling of all solutions in the population into a local optimum of the solved problem. Mutation operation randomly changes the offspring resulted from crossover. In case of binary encoding we can switch a few randomly chosen bits from 1 to 0 or from 0 to 1. Mutation can be then illustrated as follows (Fig. 2.11)

Original offspring 1	1101111000011110
Original offspring 2	1101100100110110
Mutated offspring 1	1100111000011110
Mutated offspring 2	1101101100110110

Fig. 2.11 Mutation

The technique of mutation (as well as crossover) depends mainly on the encoding of chromosomes. For example when we are encoding by permutations, mutation could be performed as an exchange of two genes.

2.5.1.3 Parameters of GA

Crossover and Mutation Probability

There are two basic parameters of GA - crossover probability and mutation probability.

Crossover probability: It indicates how often crossover will be performed. If there is no crossover, offspring are exact copies of parents. If there is crossover, offspring are made from parts of both parent's chromosome. If crossover probability is **100%**, then all offspring are made by crossover. If it is **0%**, whole new generation is made from exact copies of chromosomes from old population (but this does not mean that the new generation is the same!). Crossover is made in hope that new chromosomes will contain good parts of old chromosomes and therefore the new chromosomes will be better. However, it is good to leave some part of old population survives to next generation.

Mutation probability: This signifies how often parts of chromosome will be mutated. If there is no mutation, offspring are generated immediately after crossover (or directly copied) without any change. If mutation is performed, one or more parts of a chromosome are changed. If mutation probability is **100%**, whole chromosome is changed, if it is **0%**, nothing is changed. Mutation generally prevents the GA from falling into local extremes.

Mutation should not occur very often, because then GA will in fact change to **random search**.

Other Parameters

There are also some other parameters of GA. One another particularly important parameter is population size.

Population size: It signifies how many chromosomes are present in population (in one generation). If there are too few chromosomes, then GA has few possibilities to perform crossover and only a small part of search space is explored. On the other hand, if there are too many chromosomes, then GA slows down.

2.5.1.4 Selection

Introduction

The chromosomes are selected from the population to be parents for crossover. The problem is how to select these chromosomes. According to Darwin's theory of evolution, the best ones survive to create new offspring. There are many methods in selecting the best chromosomes. Examples are roulette wheel selection, Boltzman selection, tournament selection, rank selection, steady state selection and some others. In this thesis we have used the tournament selection as it performs better than the others.

Tournament Selection

A selection strategy in GA is simply a process that favours the selection of better individuals in the population for the mating pool. There are two important issues in the evolution process of genetic search, population diversity and selective pressure. Population diversity means that the genes from the already discovered good individuals are exploited while promising the new areas of the search space continue to be explored. Selective pressure is the degree to which the better individuals are favoured. The tournament selection strategy provides selective pressure by holding a tournament competition among individuals [2.11].

2.5.1.5 GA for Function optimization

To understand the use of GA, minimization of a multimodal function given in (2.47) is carried out through simulation study.

```
f(k)=...    +5*exp(-0.1*((x-15)^2+(y-20)^2))...
            -2*exp(-0.08*((x-20)^2+(y-15)^2))...
            +3*exp(-0.08*((x-25)^2+(y-10)^2))...
            +2*exp(-0.1*((x-10)^2+(y-10)^2))...
            -2*exp(-0.5*((x-5)^2+(y-10)^2))...
            -4*exp(-0.1*((x-15)^2+(y-5)^2))...
            -2*exp(-0.5*((x-8)^2+(y-25)^2))...
            -2*exp(-0.5*((x-21)^2+(y-25)^2))...
            +2*exp(-0.5*((x-25)^2+(y-16)^2))...
            +2*exp(-0.5*((x-5)^2+(y-14)^2));                    (2.47)
```

The function has global minimum point at [15, 5]. Single point crossover was applied and best 20 individuals having higher fitness values were selected for next generation. Mutation and crossover rate were taken as 0.25 and 0.8 respectively and population size was set at 20. Each parameter was represented by eight bits.

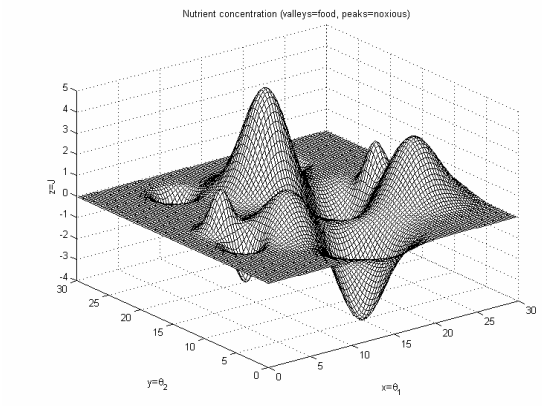

Fig. 2.12 Multimodal function of (2.47)

The fitness value settles to the global minimum value very soon. Results given by GA was [15.1373 5.0980] and $g_{\min}$ = -3.9757.

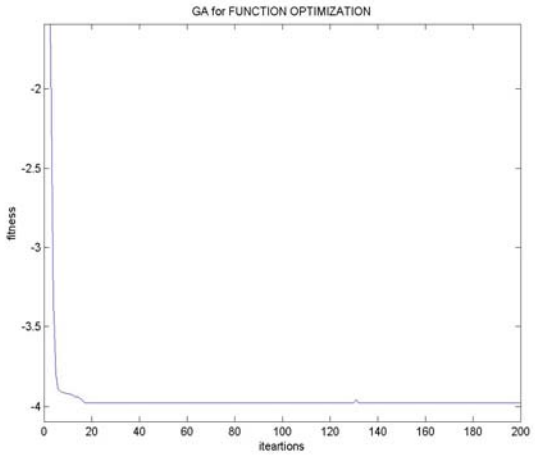

Fig. 2.13 Fitness curve of the function vs iteration

At iteration 1 the result is

Table 2.1

Initial Generation C1

W	Fitness
7.1373 0.3922	-0.0009
-16.2353 -16.3922	0.0000
20.0000 -7.7647	-0.0000
-16.0784 -5.4118	0.0000
7.9216 3.8431	0.0060
9.1765 -10.1176	-0.0000
-13.8824 -8.8627	0.0000
-8.8627 0.5490	0.0000
1.9608 12.4706	0.0069
-15.4510 16.5490	0.0000
2.5882 -14.3529	-0.0000
15.4510 - 16.3922	-0.0000
4.4706 -19.8431	-0.0000
18.5882 -11.2157	-0.0000
-19.6863 -7.6078	0.0000
-10.5882 7.4510	0.0000
13.7255 16.8627	1.5280
15.1373 5.4118	-3.9079
-11.8431 -10.1176	0.0000
19.3725 12.1569	-0.8527

After 400 iterations, all W and fitness values become equal which provides the desired result.

Table 2.2

C1 population after 400th iteration

W	Fitness
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757
15.1373 5.0980	-3.9757

2.5.2 Particle swarm optimization (PSO)

2.5.2.1 Basic method

Natural creatures sometimes behave as a swarm. One of the main stream of artificial life researches is to examine how natural creatures behave as a swarm and reconfigure the swarm models inside a computer. Swarm behavior can be modeled with a few simple rules. School of fishes and swarm of birds can be modeled with such simple models.

In 2001 Kennedy and Eberhart developed a PSO [2.12] concept through simulation of bird flocking in two-dimensional space. The position of each agent is represented by XY axes position and also the velocity is expressed by v_x (the velocity of X axis) and v_y (the velocity of Y axis). Modification of the agent position is realized by the position and velocity information. Bird flocking optimizes a certain objective function. Each agent knows its best value so far (pbest) and its XY position. This information is analogy of personal experiences of each agent. Moreover, each agent knows the best value so far in the group

(gbest) among pbests. This information is analogy of knowledge of how the other agents around them have performed. Namely, each agent tries to modify its position using the following informations

1. the current positions (x, y)
2. the current velocities (v_x, v_y)
3. to go to the center of the swarm
4. the distance between the current position and pbest
5. the distance between the current position and gbest

2.5.2.2 Particle swarm optimization algorithm

This modification can be represented by the concept of velocity. Velocity of each agent can be modified by the following equation [2.12] :

$$V_i^{k+1} = wV_i^k + c_1 \times rand_1 \times (pbest_i - s_i^k) + c_2 \times rand_2 \times (gbest_i - s_i^k) \tag{2.48}$$

where

V_i^k : velocity of agent i at iteration k

w : weighting function,

c_j : weighting factor,

$rand$: random number between 0 and 1,

s_i^k : current position of agent i at iteration k

$pbest_i$: personal best of agent i,

$gbest$: global best of the group.

The following weighting function [2.12] is usually utilized in (2.48)

$$w = w_{\max} - \frac{(w_{\max} - w_{\min})}{iter_{\max}} \times iter \tag{2.49}$$

where

$w_{\max}$: initial weight,

$w_{\min}$: final weight,

$iter_{\max}$: maximum iteration number,

$iter$: current iteration number.

Using the above equation, a certain velocity, which gradually gets close to pbest and gbest can be calculated. The current position (searching point in the solution space) can be modified by the following equation [2.12] :

$$S_i^{k+1} = S_i^k + V_i^{k+1} \tag{2.50}$$

The general flow chart of PSO is shown in Fig. 2.14 and the step wise procedure is detailed below

1. **Step. 1** Generation of initial condition of each agent

Initial searching points (s_i^0) and velocities (v_i^0) of each agent are usually generated randomly within the allowable range. The current searching point is set to pbest for each agent. The best-evaluated value of pbest is set to gbest and the agent number with the best value is stored.

2. **Step. 2** Evaluation of searching point of each agent

The objective function value is calculated for each agent. If the value is better than the current pbest of the agent, the pbest value is replaced by the current value. If the best value of pbest is better than the current gbest, gbest is replaced by the best value and the agent number with the best value is stored.

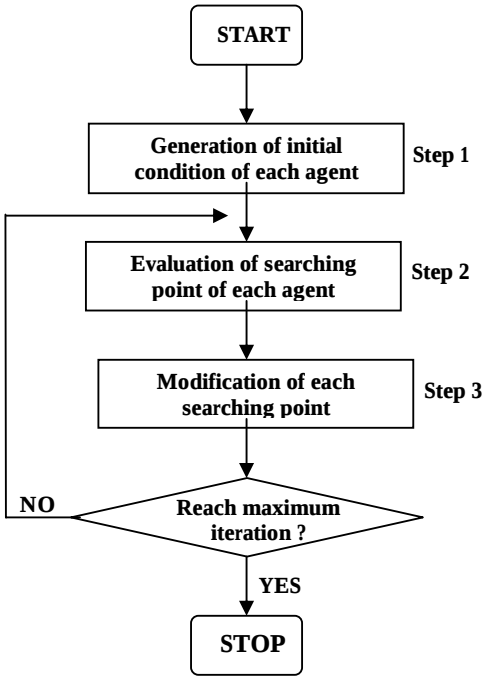

Fig. 2.14 General flow chart of PSO

3. **Step. 3** Modification of each searching point

The current search point of each agent is changed using (2.48), (2.49) and (2.50).

4. **Step. 4** Checking the exit condition

If the current iteration number reaches the predetermined maximum iteration number, then exit. Otherwise, go to step 2.

The features of the searching procedure of PSO can be summarized as follows:

1. As shown in (2.48), (2.49) and (2.50), PSO can essentially handle continuous optimization problem.

2. PSO utilizes several search points like genetic algorithm and the search points gradually get close to the optimal point using their pbest and the gbest information.

3. The first term of the right-hand side (RHS) of (2.48) is corresponding to diversification in the search procedure. The second and third terms of that are corresponding to intensification in the search procedure. This method has a well-balanced mechanism to utilize diversification and intensification in the search procedure efficiently.

4. The PSO can handle continuous optimization problems with continuous state variables in a n-dimension solution space.

Feature (3) can further be explained as follows. The RHS of (2.48) consists of three terms. The first term is the previous velocity of the agent. The second and third terms are utilized to change the velocity of the agent. Without the second and third terms, the agent will keep on "flying" in the same direction until it hits the boundary. Namely, it tries to explore new areas and, therefore, the first term is corresponding to diversification in the search procedure. On the other hand, without the first term, the velocity of the "flying" agent is only determined by using its current position and its best positions in history. Namely, the agents will try to converge to their pbests and/or gbest and, therefore, the terms are corresponding to intensification in the search procedure.

2.5.3 Bacterial foraging optimization (BFO)

2.5.3.1 Introduction

Natural selection tends to eliminate animals with poor "foraging strategies" (methods for locating, handling, and ingesting food) and favor the propagation of genes of those animals that have successful foraging strategies since they are more likely to enjoy reproductive success (they obtain enough food to enable them to reproduce). After

many generations, poor foraging strategies are either eliminated or shaped into good ones (redesigned). Logically, such evolutionary principles have led scientists in the field of "foraging theory" to hypothesize that it is appropriate to model the activity of foraging as an optimization process: A foraging animal takes actions to maximize the energy obtained per unit time spent foraging, in the face of constraints presented by its own physiology and environment

2.5.3.2 Bacterial foraging

Bacteria have the tendency to gather to the nutrient-rich areas by an activity called chemotaxis. It is known that bacteria swim by rotating whip like flagella driven by a reversible motor embedded in the cell wall. E. coli has 8-10 flagella placed randomly on a cell body. When all flagella rotate counterclockwise, they form a compact, helically propelling the cell along a helical trajectory, which is called run. When the flagella rotate clockwise, they pull on the bacterium in different directions, which causes the bacteria to tumble. The four steps involved in bacterial foraging are briefly outlined next.

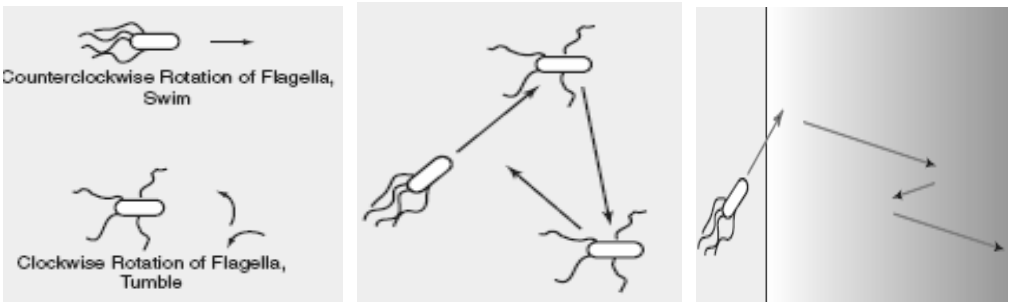

Fig. 2.15 Swimming, Tumbling and Chemotactic behavior of Ecoli

(1)**Chemotaxis** : An E. coli bacterium can move in two different ways; it can run (swim for a period of time) or it can tumble, and alternate between these two modes of operation in the entire lifetime. In the BFO, a unit walk with random direction represnts a tumble and a unit walk with the same direction in the last step indicates a run. After one step move, the position of the i th bacterium can be presented [2.13] as

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\phi(j) \tag{2.51}$$

where $\theta^i(j,k,l)$ represents the i th bacterium at j th chemotactic, k th reproductive and l th elimination and dispersal step. $C(i)$ is the length of unit walk in the random direction. It is assumed to be constant and $\phi(j)$ is the direction angle of the j th step. During run operation, $\phi(j)$ is same as $\phi(j-1)$, otherwise, $\phi(j)$ is a random angle directed within a range of [0,2π].

If the cost at $\theta^i(j+1,k,l)$ is better than the cost at $\theta^i(j,k,l)$ then the bactrium takes another step of size $C(i)$ in the same direction otherwise it tumbles. This swim process is as long as it continue to reduce the cost function, but only to a maximum number of steps, N_s.

(2) **Swarming** : The bacteria in times of stresses release attractants to signal bacteria to swarm together. It however also releases a repellant to signal others to be at a minimum distance from it. Thus all of them will have a cell to cell attacrtion via attractant and cell to cell repulsion via repellant. The cell to cell signalling in *E. coli* swarm may be represented [2.13] by the function

$$J_{cc}(\theta,P(j,k,l)) = \sum_{i=1}^{S} J_{cc}(\theta,\theta^i(j,k,l)) = \sum_{i=1}^{S}[-d_a \exp(-w_a \sum_{m=1}^{p}(\theta_m - \theta_m^i)^2)] + \sum_{i=1}^{S} h_r \exp(-w_r \sum_{m=1}^{p}(\theta_m - \theta_m^i)^2)] \quad (2.52)$$

where $J_{cc}(\theta,P(j,k,l))$ represents the objective function value to be added to the actual objective function, S is the total number of bacteria, p is the number of variables to be optimized, $\theta = [\theta_1,\theta_2,.....\theta_p]^T$ is a point in the p-dimensional serach domain, d_a = depth of the attractant, w_a = width of the attractant, h_r = height of the repellant and w_r = width of the repellant.

(3) **Reproduction** : A reproduction step is taken after N_c chemotactic steps. Let $S_r = S_b/2$ be the number of population members who have had sufficient nutrients so that they reproduce i. e. split into two. For reproduction, the population is stored in order of ascending fitness function. The S_r least haelthy bacteria die and the other S_r bacteria

each split into two identical ones which occupy the same positions in the environment. This method keeps the population size constant.

(4) **Elimination and Dispersal** : Since bacteria may be stuck around the initial positions or local optima, it is possible for the diversity of BFA to change either gradually or suddenly to eliminate the possibility of being trapped into local minima. The dispersion vent happens after a certain number of reproduction process. A bacterium is chosen, according to a preset probability p_{ed}, to be dispersed and moved to another position within the environment. These events may prevent the local minima trapping effectively, but unexpectedly disturb the optimization process. The mathematical treatment of this new concept is presented in[2.13].

2.5.4 Artificial immune system (AIS)

The human immune system is a very complex system formed by a large number of cells, molecules and diverse mechanisms. The main function of immunity is to protect our bodies from the invasion of external microorganisms. The cells and molecules responsible for immunity constitute biological immune system (BIS). The AIS is developed by following the principles of BIS. Bersini first used immune algorithms to solve practical problems. The books [2.14, 2.15] provide excellent materials about the various principles and algorithms of AIS. According to BIS theory our body immunity is composed of two defense lines: innate and adaptive immunity. Innate immunity is nonspecific which means that it is independent of the foreign antigen. The adaptive immunity has memory and learning capabilities and it is antigen dependent, meaning that each different type of antigen provokes a different immune response. The main components of the adaptive immunity are the cells called B lymphocytes or simply B cells. When B lymphocytes are stimulated by a specific antigen, they produce a large number of molecules called antibodies, which play a major role in the adaptive immune response.

The clonal selection principle of AIS describes how the immune cells eliminate a foreign antigen and is simple but efficient approximation algorithm for achieving optimum solution. The steps involved in the clonal selection algorithm are

Step 1 : Initialize a number of antibodies(immune cells) which represent initial population size.

Step 2 : When an antigen or pathogen invades the organism; a number of antibodies that recognize these antigens survives. In Fig.2.16, only the antibody C is able to recognize the antigen as its structure fits to a portion of the pathogen. So fitness of antibody C is higher than others.

Step 3 : The immune cells recognize antigens undergo cellular reproduction. During reproduction the somatic cells reproduce in an asexual form, i.e. there is no crossover of genetic material during cell mitosis. The new cells are copies (clones) of their parents as shown for antibody C.

Step 4 : A portion of cloned cells undergo a mutation mechanism which is known as somatic hypermutation as described in [2.15] .

Step 5 : The affinity of every cell with each other is a measure of similarity between them. It is calculated by the distance between the two cells. The antibodies present in a memory response have on average a higher affinity than those of early primary response. This phenomenon is referred to as maturation of immune response. During the mutation process the fitness as well as the affinity of the antibodies gets changed. In each iteration after cloning and mutation those antibodies which have higher fitness and higher affinity are allowed to enter the pool of efficient cells. Those cells with low affinity or self-reactive receptors must be efficiently eliminated.

Step6: At each iteration among the efficient immune cells some become effecter cells (Plasma Cell), while others are maintained as memory cells. The effecter cells secrete antibodies and memory cells having longer span of life so as to act faster or more effectively in future when the organism is exposed to same or similar pathogen.

Step7: The process continues till the termination condition is satisfied else steps 2 to 7 are repeated.

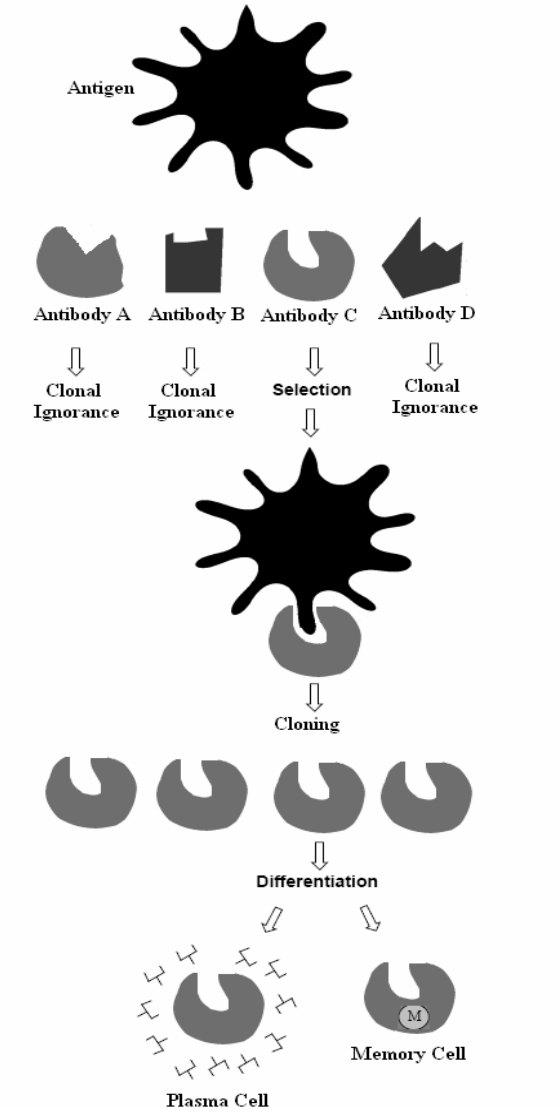

Fig. 2.16 The Clonal Selection Principle

References

[2.1] B. Widrow and S.D. Sterns, "*Adaptive Signal Processing*" Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1985.

[2.2] S. Haykin, "*Adaptive Filter Theory*", 4th edition, Pearson Education Asia, 2002.

[2.3] S. Haykin, "*Neural Networks: A comprehensive foundation*" 2nd Edition, Pearson Education Asia, 2002.

[2.4] E. J. Dayhoff, "*Neural Network Architecture – An Introduction*" Van Norstand Reilold, New York, 1990.

[2.5] N. K. Bose and P. Liang., "*Neural Network Fundamentals with Graphs, Algorithms, Applications*", TMH Publishing Company Ltd, 1998.

[2.6] Richard O. Duda, Peter E. Hart and David G. Stork, "*Pattern Classification*", 2nd edition, John Wiley & Sons, INC.2001.

[2.7] Y.H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison Wesley, Reading, Massachusetts, 1989.

[2.8] Y. H. Pao, G. H. Park and D. J. Sobjic., "Learning and Generalization Characteristics of the Random Vector Function", Neuro Computation, 6: 163-180, 1994.

[2.9] J. C. Patra and R. N. Pal, "A Functional Link Artificial Neural Network for Adaptive Channel Equalization", Signal Processing, vol.43, no.2, pp.181-195, May 1995.

[2.10] D.E.Goldberg, "*Genetic algorithms in search, optimization and machine learning*", Addition-Wesley,1989.

[2.11] D. E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in GA", *Foundations of genetic Algorithms*, I, pp. 53-69, 1991.

[2.12] J. Kennedy, R. C. Eberhart and Y. Shi, "*Swarm intelligence*", San Francisco: Morgan Kaufmann Publishers, 2001.

[2.13] K. M. Passino, "Biomimicry of Bacterial Foraging for distributed optimization and control", IEEE control system magazine, vol 22, issue 3, pp. 52-67, June 2002.

[2.14] D.Dasgupta, *Artificial Immune Systems and their Applications*, Springer-Verlag, 1999.

[2.15] L N de Castro and F. J. Von Zuben , "Learning and Optimization using Clonal Selection Principle," IEEE Trans on Evolutionary Computation, Special issue on Artificial Immune Systems, vol. 6, issue 3, pp.239-251, 2002.