CONTENTS

Abstract

The back propagation (BP) algorithm is perhaps the most widely used supervised training algorithm for multilayered feed forward neural network.

The main draw back of BP is trapping in local minimum. Theoretically, the larger the learning rate, the faster the training process goes. But practically, the learning rate may have to be set to a small value in order to prevent the training process from being dropped to a local minimum, resulting in an oscillating response .But this small value of learning rate will slow down the learning process.

Different methods was proposed to prevent the draw back of BP, one of them is the optical back propagation (OBP), which is designed to overcome some of the problems of BP. The OBP tried to escape from local minimum with a faster convergence through learning process.

In this project, ANN based on BP and OBP will be test for different case studies. A comparison will be made to investigate the effectiveness of the improved algorithm.

CHAPTER ONE

INTRODUCTION

CHAPTER ONE

I NTRODUCTI ON

1.1 Introduction

Neural networks made a great come back in 1990's and are new generally accepted as a major tool in the development of intelligent systems. Their origins go way back to the 1940's, when the first mathematical model of biological neurons was published by Mc Culloch and Pitts. Unfortunately, there was a period of about 20 years or so when research in neural network effectively stopped

In the mid of 1980's, there was a resurgence of interest in neural network, largely prompted by the publication of Rome Lhart and Mc Clelland's book, "parallel distributed processors ". Suddenly, it seemed that everyone was interested in and talking about neural network again. During these years, new eminent–names such as Windrow, Kohonen and Grossbery has continued working on neural network and developed their own versions. Problems such as the Exclusive OR, which had originally contributed to the admire of neural network in 1960's had been overcome using new learning techniques such as back propagation, the result has been that researchers in the subject now have to familiarize themselves with a wide variety of network, all with difference in architecture, learning strategies and weight updating method. [1]

Neural network are developed by morphologically and computational simulating a human brain, the precise operation details of artificial neural network are quite different from human brains; they are similar in three aspects. First, a neural consists of a very large number of simple processing elements (the neuron). Second, each neuron is connected to a large number of other neurons. Third, the functionality of the network is determined by modifying the strength of connections during learning phase. [2]

1.2 Fundamental concepts

Much is still UN known about how the brain trains itself to process information. In human brain, a typical neuron collects signal from others through a host of fine structures called dendrites.

The neuron sends out spikes of electrical activity through long, thin stand known as an axon, which splits into thousand of branches. At the end of each branch a structure called synapse converts the activity of the axon into electrical effects that inhibits or excites activity in connected neurons. When a neuron receives excitory input, it sends a spike of electrical activity down its axon. Learning occurs by changing the effectiveness of the synapses so that the influence of one neuron on another changes.

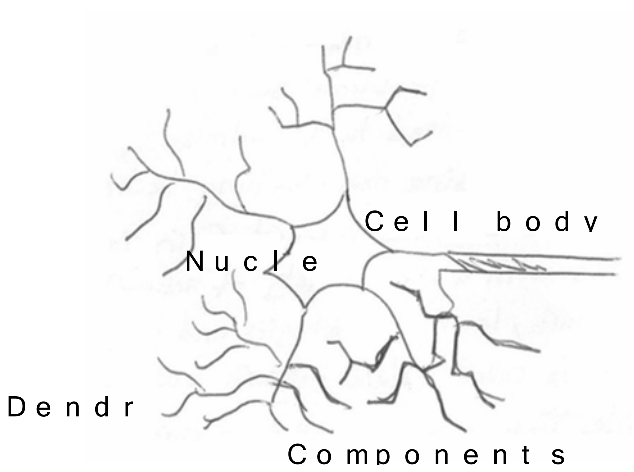


Fig (1.1) components of neuron [3]

1.3 Basic types of Neural Network

Artificial neural network can be divided in terms of their structure:

Feed forward network: This type of network can be connected in cascade to create a multilayer network. In such network, the output of a layer is the input to the following layer. Even though thefeedforward network has no explicit feedback connection, when the input is mapped into the output. The output values are often compared with the desired output (value), and an error signal can be employed for adapting network weights, as shown in fig (1.2a). [4]

Feedback network: - Feedback network can be obtained from the feed forward network by connecting the neuron's output to their inputs. Fig (1.2b) shows, the interconnection scheme. The network is also called recurrent since its response at (k+1) the instant depends on the entire history of network starting at k=0

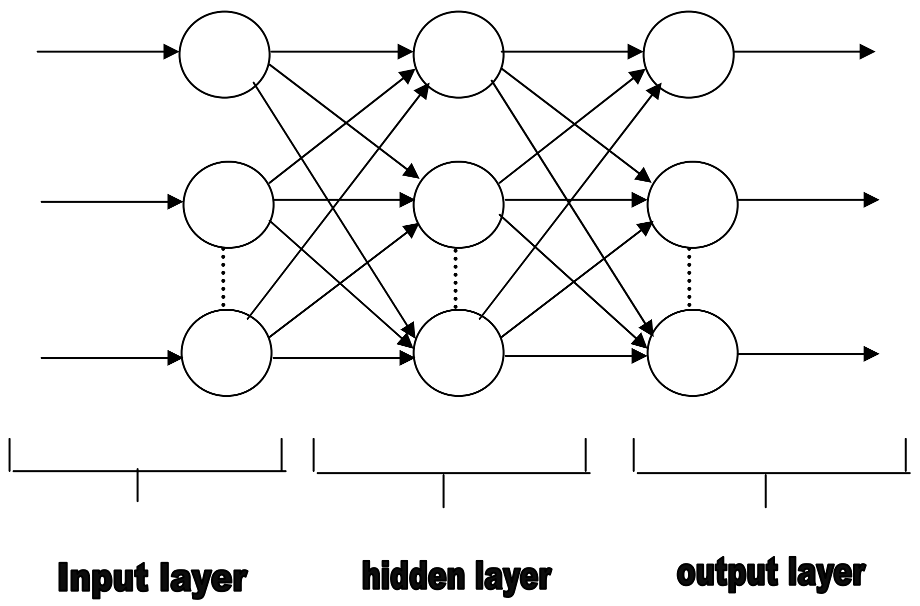


Fig (1.2a) interconnection scheme of feed forward N. N. [5]

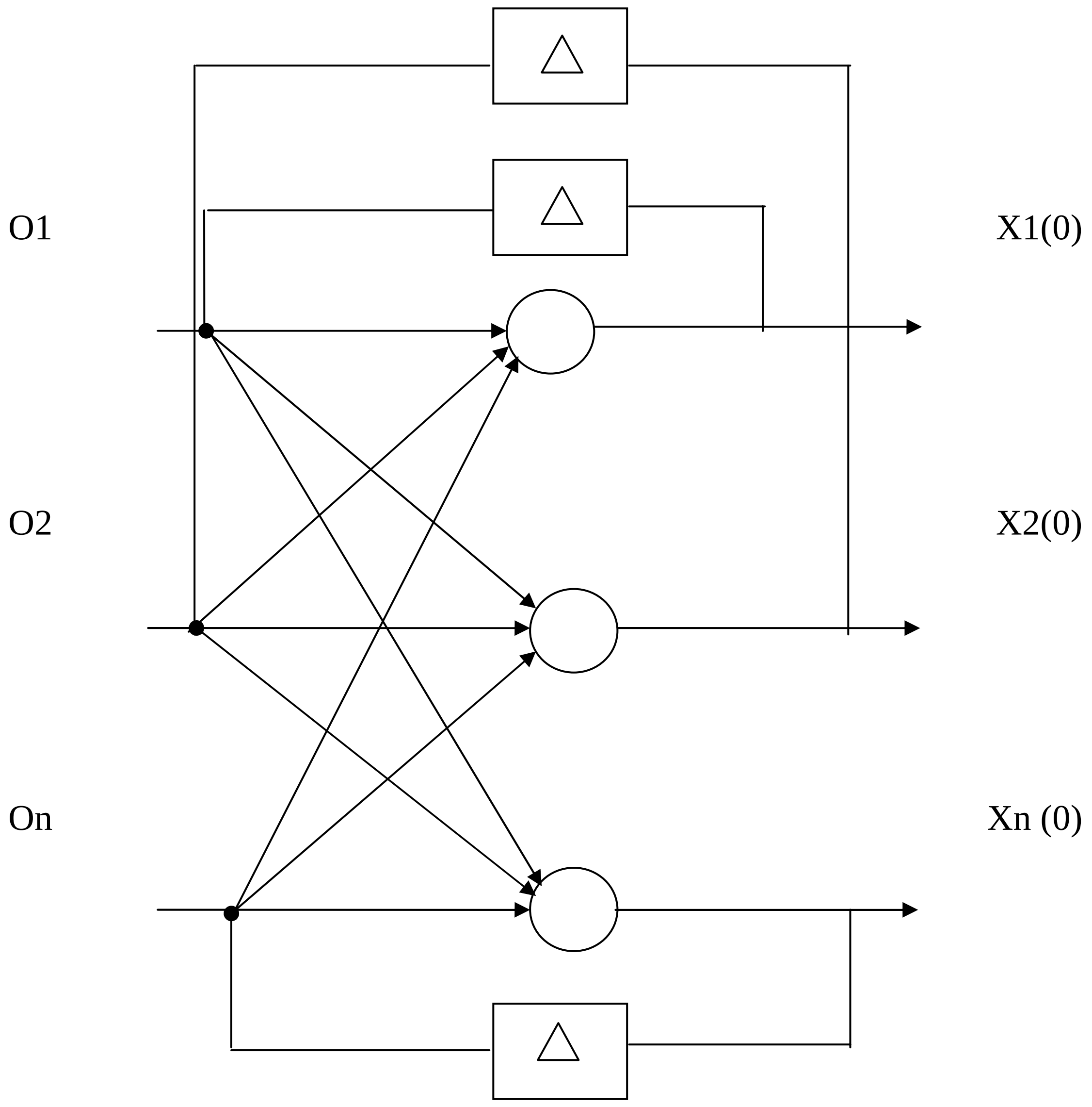


Fig (1.2b) interconnection scheme of recurrent N. N. [5]

1.4 Basic of Artificial Neural Network (ANN)

ANN can be defined as a computing system made up of number of simple, highly interconnected processing elements which process information by its dynamic state in response to external input.

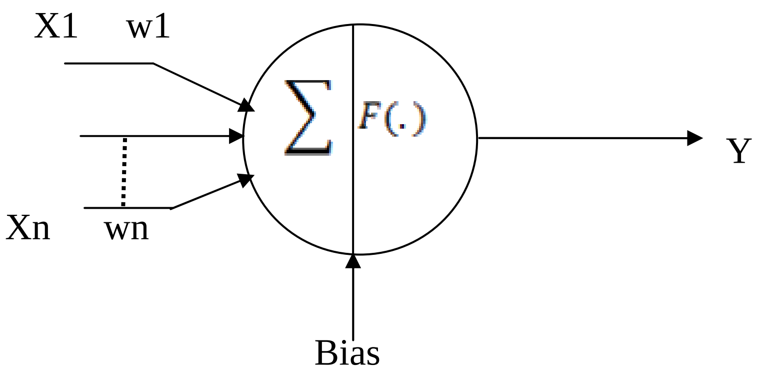


Fig. (1.3) Artificial neuron

Where

X1………Xn: Input vector

W1……..Wn: Weights value

Y Output

So.

Y = f [($\sum_{i=0}^{n}$ wi xi) + bais)]

In a single layer neural network, it adapts by changing the weights by an amount proportional to the difference between the desired output and the actual output. As in the following equation:

$$W\ new = W\ old + ʒ(Yd - Yi) * Xi$$

Where

ʒ : is the learning rate

Yd: is the desired output

Yi : is the actual input

The above equation is called the proton learning rule and goes back to the early of 1960.Also, known as back–propagated delta network (PB), which is developed from the single layer network but with extra hidden layers.

Fig. (1.4) shows the structures of single and multi layer NN

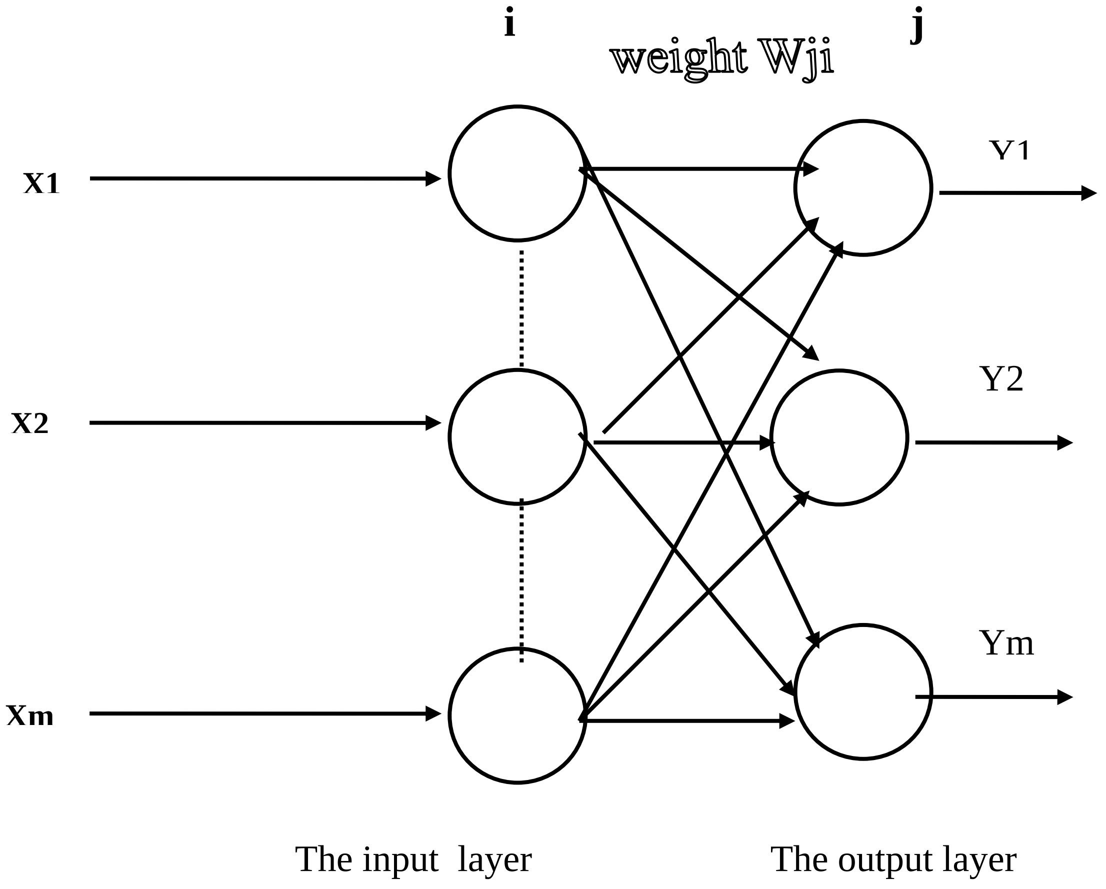


Fig.(1.4 a) Single layer Neural Network Structure .[4]

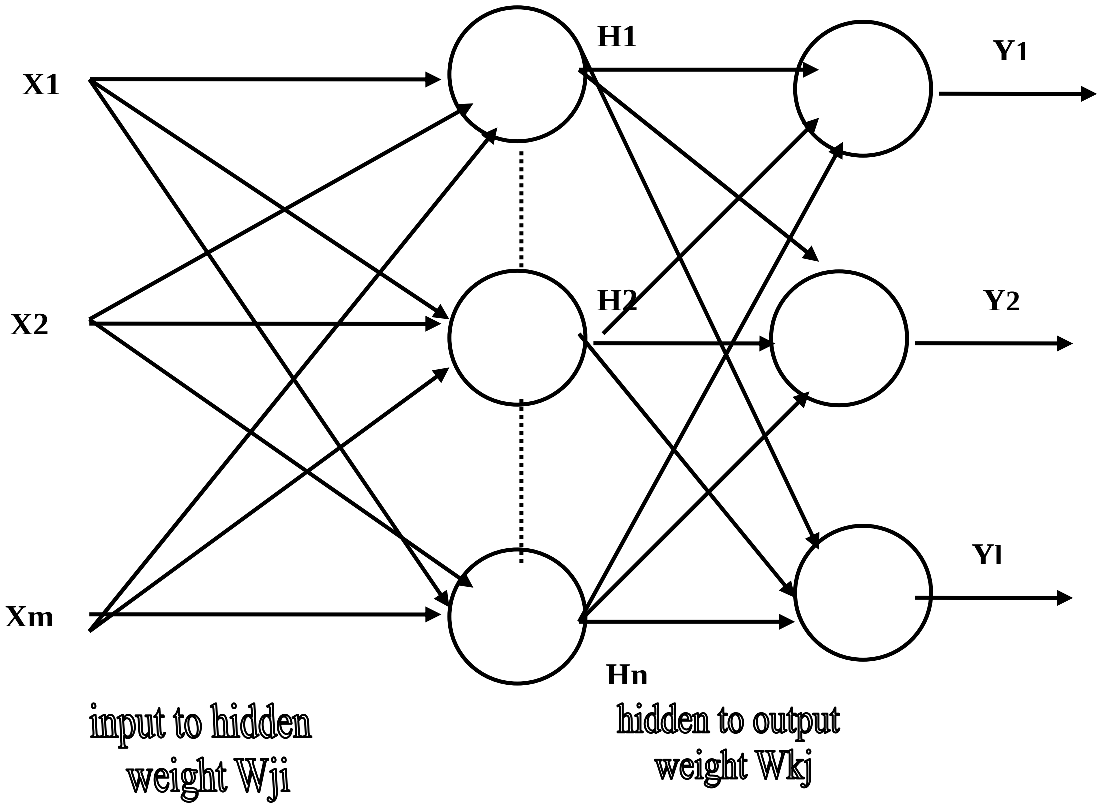


Fig.(1.4 b) Multilayer Neural Network Structure .[4]

1.5 Applications

Neural network are widely applied successfully in applications such as:

● ***Control***

The use of neural network in control systems can be seen as a neural step in the evaluation of control methodology to meet the needs of rapidly advancing technology and competitive market.

Basically, there are three major needs: the need to deal with increasingly complex systems, and the need to accomplish increasingly demanding design requirements, and the need to attain these requirements with less precise advanced knowledge of the plant and its environment. The increasing emphasis on these needs has led to more general concepts of control, one that has the ability to compare hand and learn about plants, disturbances, environment and operating condition. These abilities are necessary when it is desired to design a system with acceptable performance characteristics over a very wide range of uncertainty. And for this purpose, neural networks are used as a controller to overcome the difficulties of designing complex control system. [3]

● ***Pattern recognition***

Pattern recognition are involved the recognition of something that can not be entirely described or predicted. Neural network are good at learning perspective types of task such as recognition of complex patterns .Increasingly, researchers are publishing accounts of successful applications of neural network in the area of image processing, speech recognition, face recognition, hand written .etc. [1]

CHAPTER

TWO

NEURAL NETWORK

CHAPTER TWO

Neural Network and Error Back Propagation Algorithm

2.1 Feedforward Structures:

The feed forward network is composed of a hierarchy of processing unit, recognized in a series of two or more exclusive sets of neurons or layers. The first or input layer serves as a holding site for the inputs applied to the network. The last or output layer is the point at which the overall mapping of the network input is available. Between these two extremes lie zero or more layers of hidden units; it is in these internal layers that additional computing takes place.

Links, or weights, connect each unit in one layer only to those in the next higher layer , in that the output of a unit, scaled by the value of a connecting weight , is feed forward to provide a portion of the activation for the units in the next higher layer Fig.(2.1) illustrates the typical multilayer Neural Network that has in general ,m – input nodes X1 to Xm , n – hidden nodes H1 to Hn , P – output nodes Y1 to Yp , a set of connections between input and hidden layer referred to as the input to hidden weight and finally, a set of connections between hidden and output layer referred to as the hidden to output weights . [6]

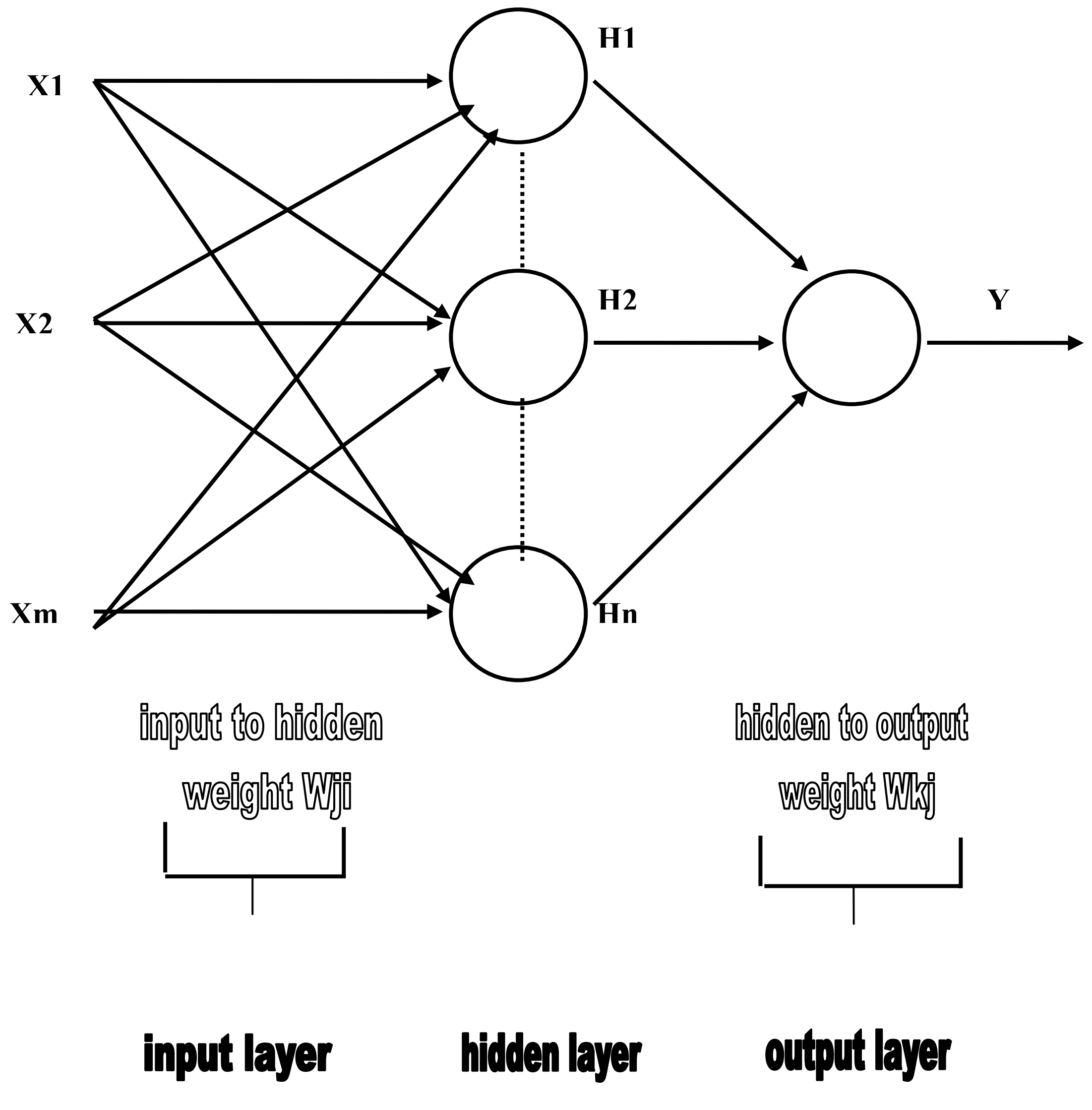


Fig.(2.1) Multilayer Neural Network Structure .

Node

Each node in the hidden and output layer can be represented in fig. (2.2) which show the block diagram of artificial neuron. Input signals are multiplied by the weights. The weighted input is then summed together to produce sum ($\sum XiWi$). This sum is then transformed by a linear or non – linear activation function to produce the output

.

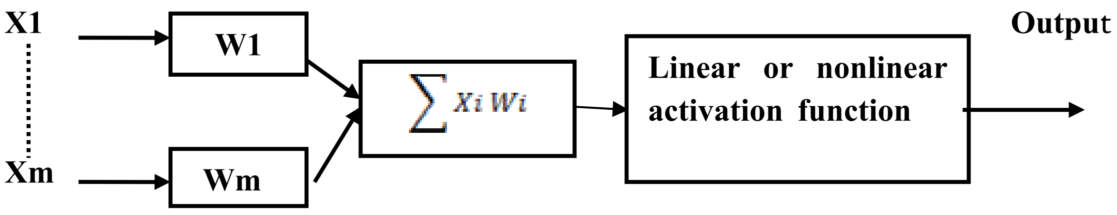


Fig. (2.2) Representation of a node

Number of network weights [7]

The number of network weights is simple function of the number of nodes and layer comprising the network

*Number of network weights = (ni +no) * nh + (nl-1) * nh^2*

Where :

- **ni = Number if input nodes**
- **no = Number of output nodes**
- **nl = number of hidden layer**
- **nh = number of hidden nodes**

2.2 The activation function: - [6]

As was maintioned previously, the computed net, which is the sum of the product produce by multiplying the weights and the input signals that lead to that node.This calculated net should be applied to a transformation function, which is called the activation function. The type of this function may be linear, linear with threshold (saturation), step (hard limiter), sigmoid (s – shaped) and hyperbolic.

- ***Linear function***

As shown in fig. (2.3) the linear function is more accurate and has higher speed than non linear function for linear plants.

$F(x) = k.x$ …………… (2.1

Where :

X : input

F(x) : output

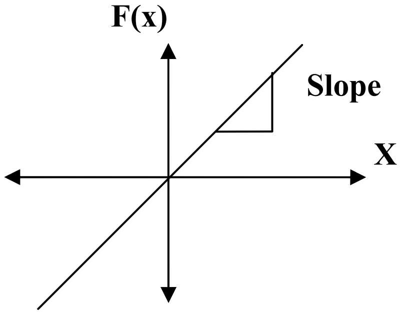


Fig. (2.3) Linear function

- ***<u>Step function</u>***

As shown in fig. (2.4), the output is limited into two possible values.

$$F(x) = \begin{cases} 1 & for\ x \geq 0 \\ -1 & for\ x < 0 \end{cases}$$

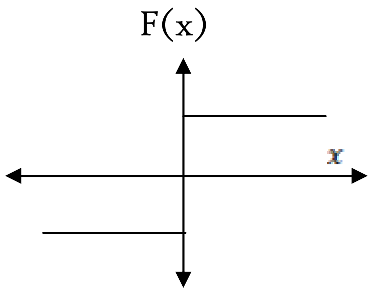


Fig. (2.4) Step function

- ***<u>Sigmoid function</u>***

The logistic function has a rich history of application as a cumulative distribution function in demographic studies. The particular functional form that is often used for sigmoid activation function is:

$$F(net) = F(x) = \frac{1}{1+e^{-x}} \qquad \dots\dots\dots\dots\dots\dots\dots\dots.. (2.3)$$

Which yields $F(x) \in [0,1]$. This function is shown in fig. (2.5).

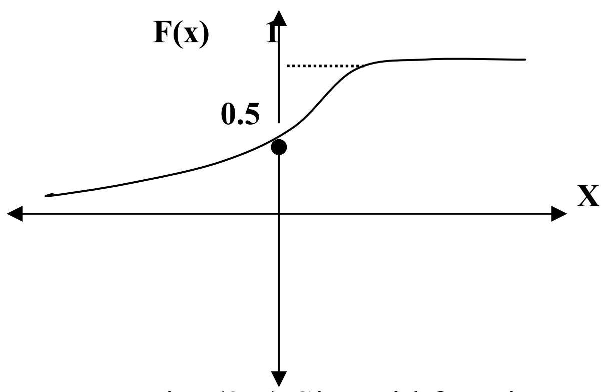


Fig. (2.5) Sigmoid function

- ***Hyperbolic Tangent Function***

It is a continuous non–linear function with an output range from (-1) to (1), as shown in fig. (2.6)

$$F(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \quad \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots. (2.4)$$

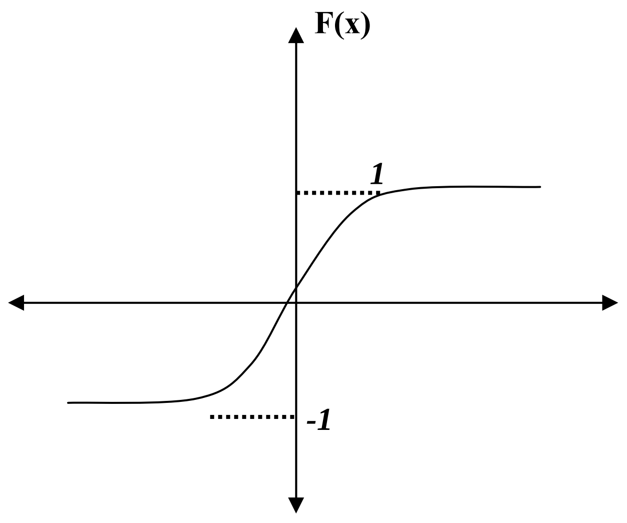


Fig. (2.6) Hyperbolic function.

- ***Linear Threshold Function***

As shown in fig. (2.7), the output is linear over some range (-L to L), outside range is dipped to a constant value (-H to H).

$$\mathbf{F}(x) = \begin{Bmatrix} H & x \geq L \\ k.x & -L < x < L \\ -H & x < -L \end{Bmatrix} \dots\dots\dots\dots\dots\dots \ (2.5)$$

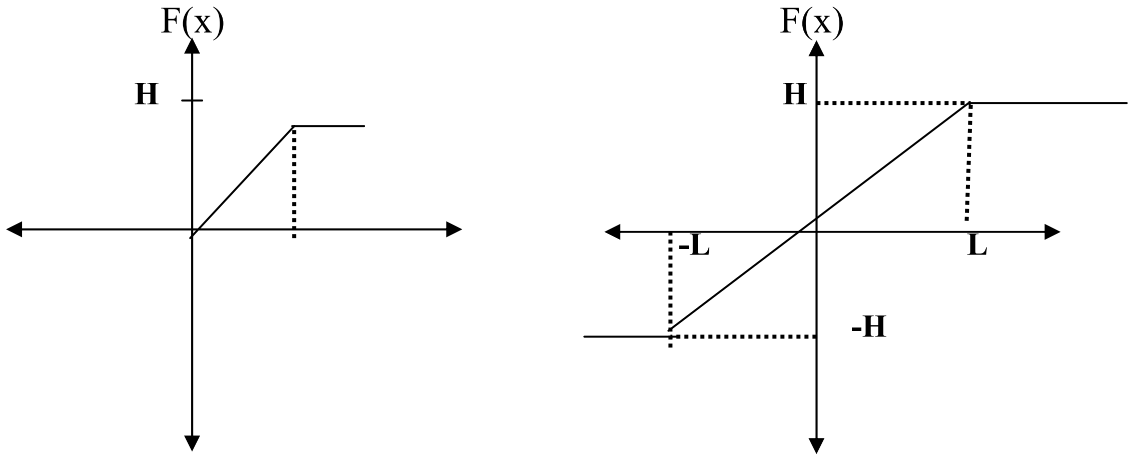


Fig. (2.7) Linear threshold function.

2.3 Computing the weight change

Back Propagation Algorithm [3]

The back propagation BP learns a predefined set of output example pairs by using a two-phase propagate adapts cycle. As seen in fig. (2.1), after the input pattern has been applied as a stimulus to first layer of network units, it is propagated through each upper layer until an output is generated. This output pattern is then compared to the desired output, and an error signal is computed for each output unit. The signals are then transmitted backward from the output layer to each unit in the intermediate layer that contributes directly to the output. How ever, each unit in the intermediate layer receives only a portion of the total error signal, based roughly on the relative contribution the unit made to the original output. This process repeats, layer by layer, until each unit in the network has received an error signal that describes its relative contribution to the total error. Referring to fig. (2.1), the output Yk is determined by

$$Y(k) = F(net(k)) \quad \ldots\ldots\ldots\ldots\ldots (2.6)$$

$$net(k) = \sum Wkj\ Hi \quad \ldots\ldots\ldots\ldots (2.7)$$

The output for hidden layer:

$$Hj = F(net(j)) \ldots\ldots\ldots\ldots\ldots\ldots (2.8)$$

$$net(j) = \sum Wji\ Xi \quad \ldots\ldots\ldots\ldots\ldots (2.9)$$

<u>Main Step of Back Propagation</u>

The Back Propagation algorithm can be summarized as in the following steps , using unipolar sigmoid activation function

<u>step 1</u>**:** Set initial values of Wkj**,** Wji**,** Ʒ

<u>step 2</u> **:** Apply the input to the neural network , specify the corresponding desired output Ꚍk , and calculate Hj , Yk and δk by the formula

δK = Yk* (1 –Yk) * (Ꚍk – Yk)

<u>step 3</u> **:** Change the connection weight by

ΔWkj = Ʒ * δk * Hj

<u>Step 4</u>**:** Calculate **δj** by

δj = Hj (1 – Hj) * **∑ δk Wkj**

Step 5**:** Change the connection weights by

$\Delta Wji = Ʒ * \delta j * Xi$

Step 6**:** $t \longrightarrow t+1$ and go to step 2

The details of the derivative of back propagation algorithm are described in Appendix A .

The learning rate (Ʒ) is a factor that determines the size of the step that the network takes in order to minimize the magnitude of training error.

CHAPTER THREE

OPTICAL BACK PROPAGATION

CHAPTER THREE

Optical Back Propagation Algorithm

3.1 Introduction

In almost every field of engineering, researchers began to actively engage in the application of neural networks with the intention of finding better solutions of conventional methods.

An important aspect in the design of neural network is its adaptation algorithm, which is used to change the weighting elements of the network such that it minimizes R.M.S error between the actual output and the desired output.

There are many successful applications of back propagation for training multi layer network. But, it has many short coming. Learning often takes long time to converge, and it may fall into local minima. There has been much research proposed to improve this algorithm; some of them was based on using small learning rate, which slow down the learning process or was based on the adaptive learning parameters.

One the proposed algorithm was the ***Optical back propagation*** which is designed to overcome some of the problems associated with the standard back propagation algorithm. This algorithm has the ability to escape from local minima with high speed of convergence during training period.

3.2 *Optical Back propagation (OBP) [8]*

Optical back propagation algorithm ***OBP*** is used for training a multilayer network with a very small learning rate for any network size that uses back propagation algorithm through an optical time (seen time). The convergence speed of the learning process can be improved through adjusting the error, which will be transmitted backward from the output to each unit in the intermediate layer.

The error at a single output unit in adjusted OBP will be as:

New $\delta^{o} = (1 + e^{(Yd-O)^2})$ if (Yd-O) >= zero

New $\delta^{o} = -(1 + e^{(Yd-O)^2})$ if (Yd-O) < zero

Where new δ^{o} is considered as the new proposed in the OBP algorithm. This new δ^{o} will minimize the errors of each output unit more quickly than the old δ^{o}, and the weights on certain units change large from their starting values.

3.3 *The steps of Optical Back propagation (OBP) [8]*

Assume there are ***m*** input units, ***n*** hidden units, and ***p*** output units

Step 1: Apply the input vector Xp= (Xp1, Xp2, Xpn) to the input unit**.**

Step 2**:** Calculate the net- input values to the hidden layer units:

$$net^{h}pj = (\sum_{i=1}^{N} w^{h}ji * X_{pi})$$

Step 3: Calculate the output from the hidden layer

$$i_{pj} = f^{h}j(net^{h}pj)$$

Step 4**:** Move to the output layer. Calculate the net values for each unit

$$net^{o}pk = (\sum_{j=1}^{L} w^{o}kj * i_{pj})$$

Step 5**:** Calculate the output

$$o_{pk} = f^{o}j(net^{o}pk)$$

Step 6: Calculate the new error term for the output

New $\delta^{o} = (1 + e^{(Yd-O)^{2}})$ if (Yd-O) >= zero

New $\delta^{o} = -(1 + e^{(Yd-O)^{2}})$ if (Yd-O) < zero

Step 7**:** Calculate the error term for the hidden layer

$$\delta^{h}pj = f^{h'}(net^{h}pj) * \sum_{k=1}^{M} new\delta^{o} * w_{kj})$$

Notice that the error terms on the hidden units are calculated before the connection weights to the output layer units have been updated.

Step 8**:** Update weights on the output layer

$$w^{o}kj(t+1) = w^{o}kj(t) + \eta * new\delta^{o}pk * i_{pj}$$

Step 9: Update weights on the hidden layer

$$w^{h}ji(t+1) = w^{h}ji(t) + \eta * \delta^{h}pj * X_{i}$$

3.4 *Comparative Study*

The optical back propagation will be tested using different examples. These examples will train the network using OBP, and they train using standard BP, and a comparison will be made between the two algorithms.

Example 1: Consider the neural network with 4 units for input layer, 3 units for hidden layer, and 3 units for output layer. The initial weights are selected between -.5 to .5 and the learning rate equals to 0.01.

If ***yd1=0.9 and yd2=0.8 and yd3=0.7***

Discussion

The network is tested using the two algorithms with different learning rates. Table (3.1); shows the surprising results in which one can notice that OBP is much better and faster than the standard BP. The main draw back of BP is happing in local minima, one of the suggested methods is to decrease the learning rate but that leads to an increase in learning time. The simulation results presented in fig. (3.1 a, b), fig. (3.2 a, b) and in fig. (3.3 a, b) show that the final weights that produced from OBP is very close to that weights when BP is used with small learning rate. But when learning rate increase the match between the two groups of weights are decreased. So, OBP helps the NN to escape from the local minimum when learning rate is very small.

Learning rate	***OBP Epochs***	***BP Epochs***	***Actual value of y1***	***Actual value of y2***	***Actual value of y3***
0.01	***853***	***39542***	***0.898***	***0.8***	***0.7***
.05	***433***	***7905***	***0.898***	***0.8***	***0.7***
0.15	***250***	***2634***	***0.898***	***0.8***	***0.7***
0.2	***150***	***2095***	***0.899***	***0.8***	***0.8***

Table (3.1) Training processes using different learning rate

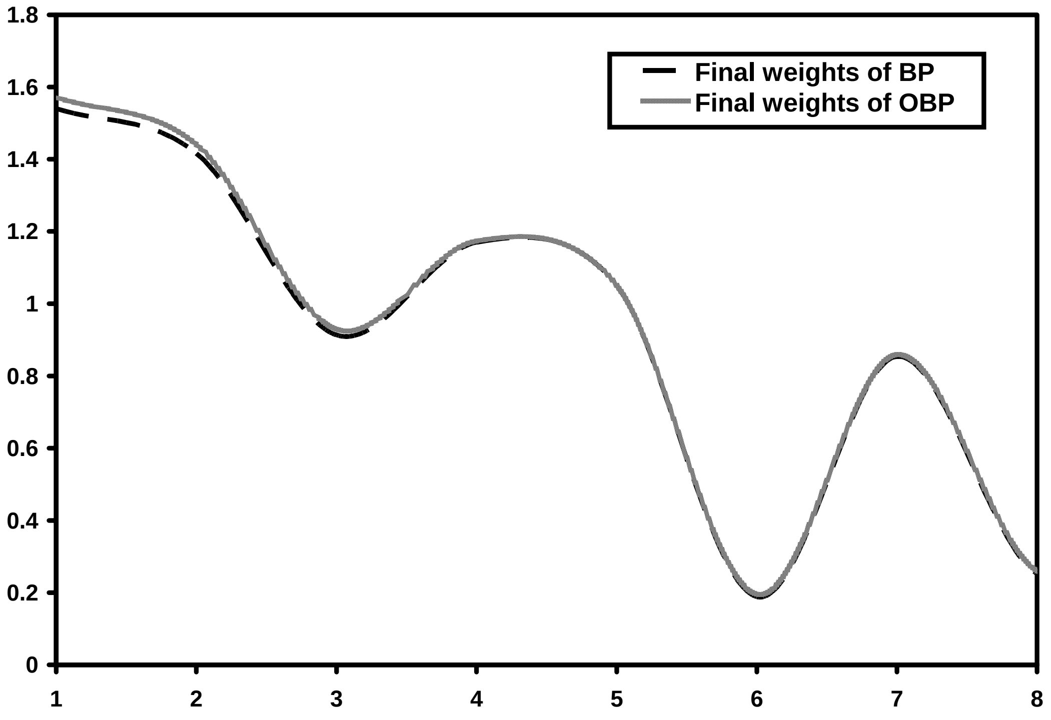


Fig. (3.1) Final weights from input to hidden and from hidden to output, with learning rate 0.01

Fig. (3.1 a) Final weights of hidden to output layer, BP with 39542 Epochs and an OBP

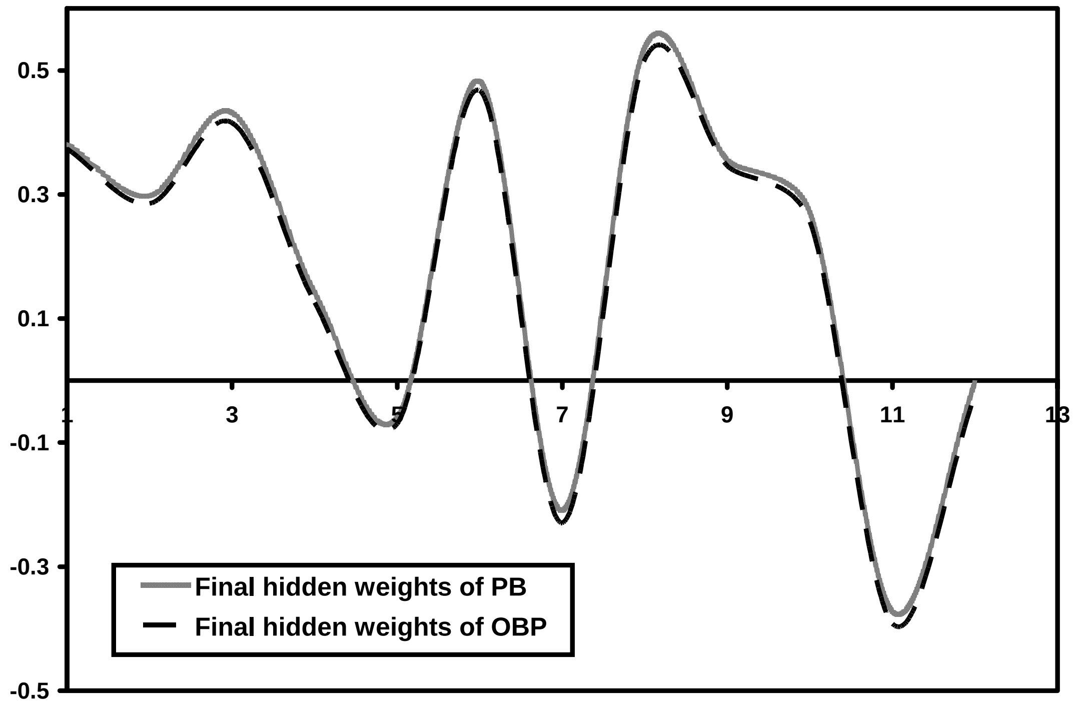


Fig. (3.1 b) Final weights of input to hidden layer, BP with 39542 Epochs and an OBP with 853 Epochs

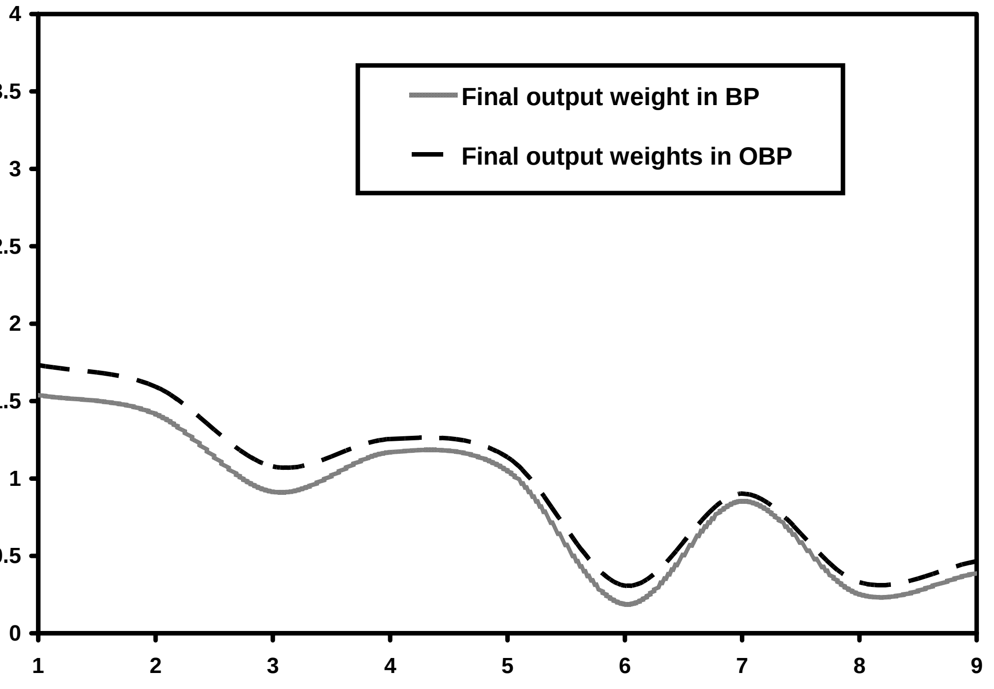


Fig. (3.2) Final weights from input to hidden and from hidden to output, with learning rate 0.05

Fig. (3.2 a) Final weights of hidden to output layer, BP with7905 Epochs and an OBP with 433 Epochs

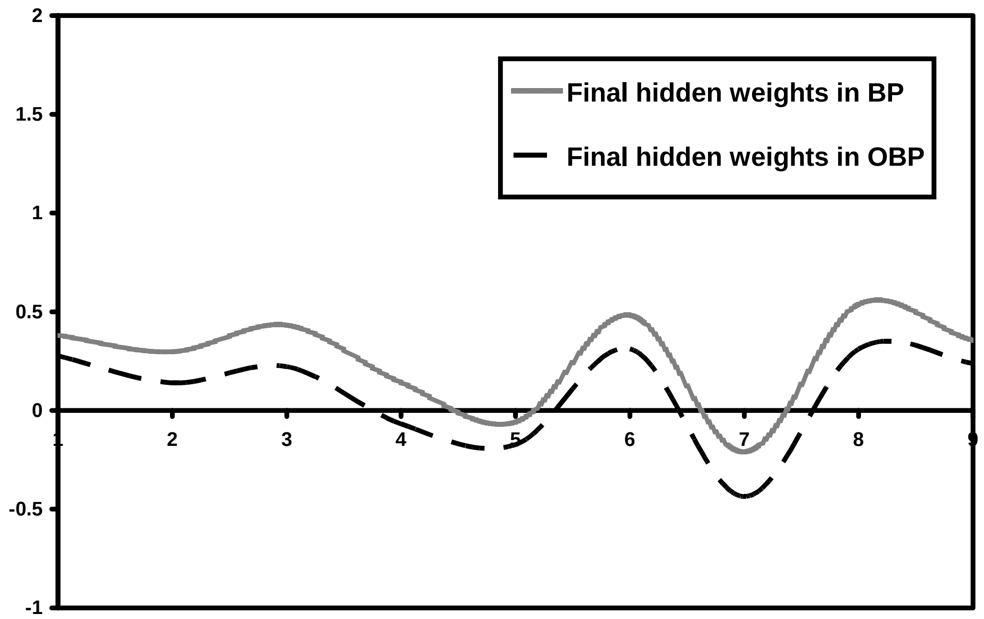


Fig. (3.2 b) Final weights of input to hidden layer

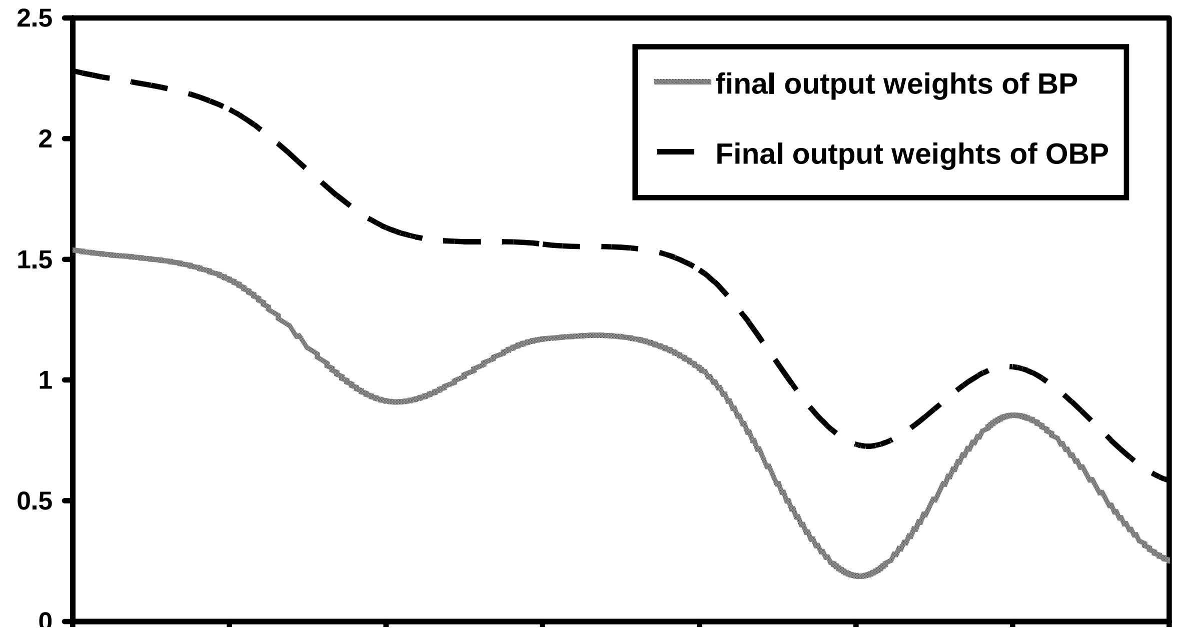


Fig. (3.3) Final weights from input to hidden and from hidden to output, with learning rate 0.1

Fig. (3.3 a) Final weights of hidden to output layer, BP with 2634 Epochs and an OBP with 250 Epochs

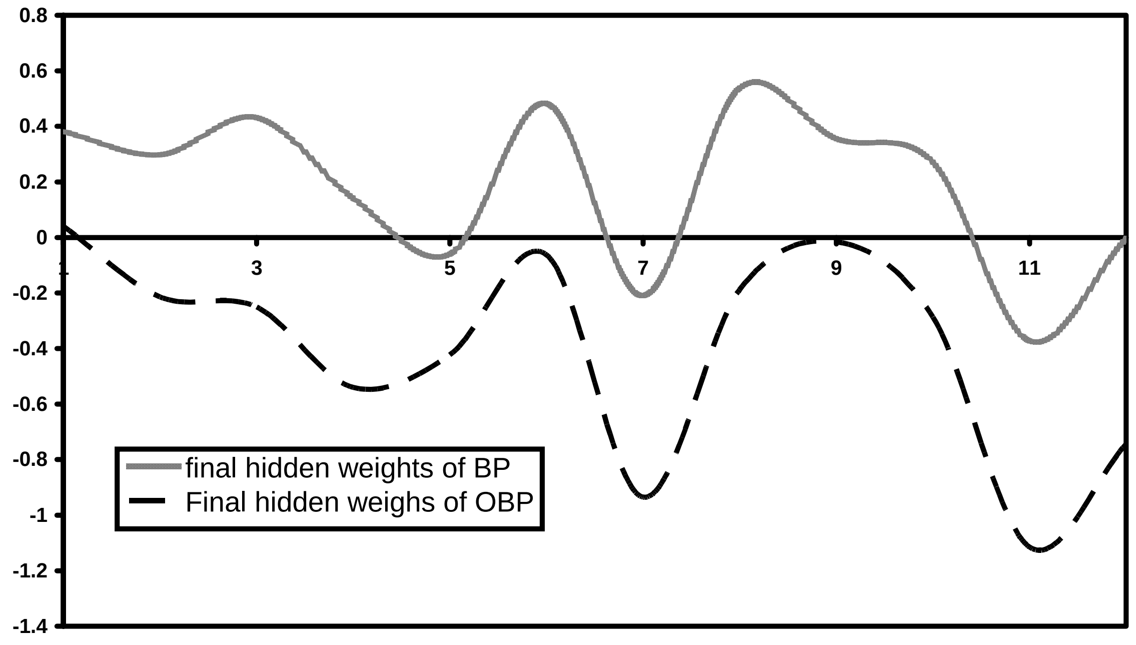
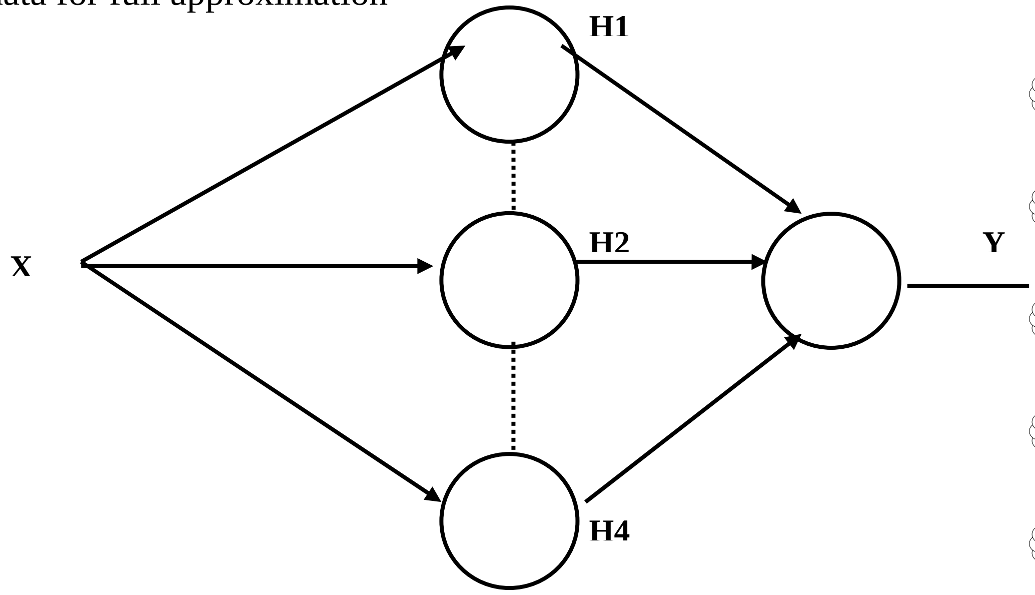


Fig. (3.3 b) Final weights of input to hidden layer,

<u>Example 2:</u> The multi layer network shown below, is used to approximate the following desired function (using back propagation and optical back propagation)

H(x) = x ***zero< x <=1***

- The number of hidden neuron =4
- Learning rate = 0.1
- Uni polar sigmoid activation for both hidden and output layer
- Use 9 data for full approximation



<u>Discussion in approximating the desired function</u>

Although the two functions h(x) and y(x) do not exactly overlap as shown in fig. (3.4 a) and in fig. (3.5 a), the example shows the potential that exists for function approximation using neural network using both standard BP and OBP. The simulation result showed the similar behavior and the effectiveness of both algorithm, but OBP has the priority in completing the learning process than the standard back propagation.

.

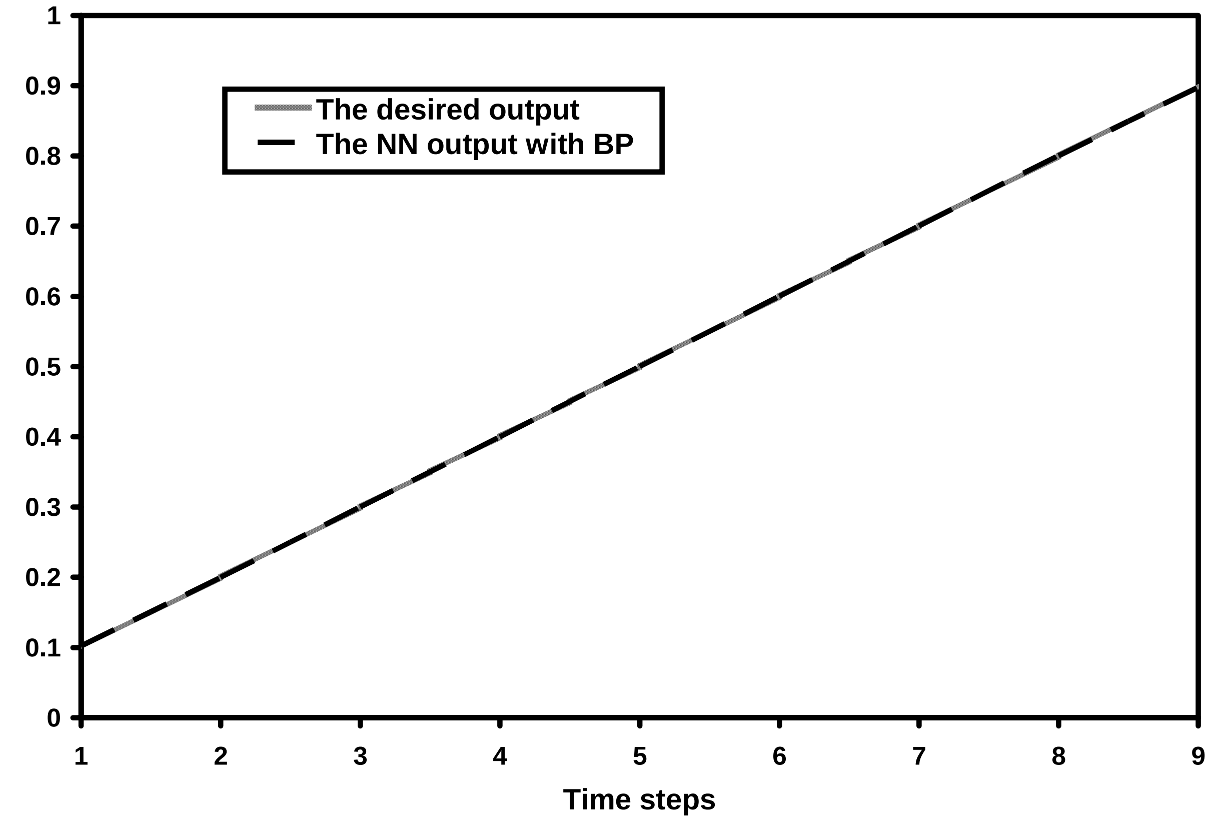


Fig. (3.4) the output and error of the network with BP algorithm and learning rate 0.1

Fig. (3.4 a) The desired and OBP outputs after NN takes 1619 Epoch

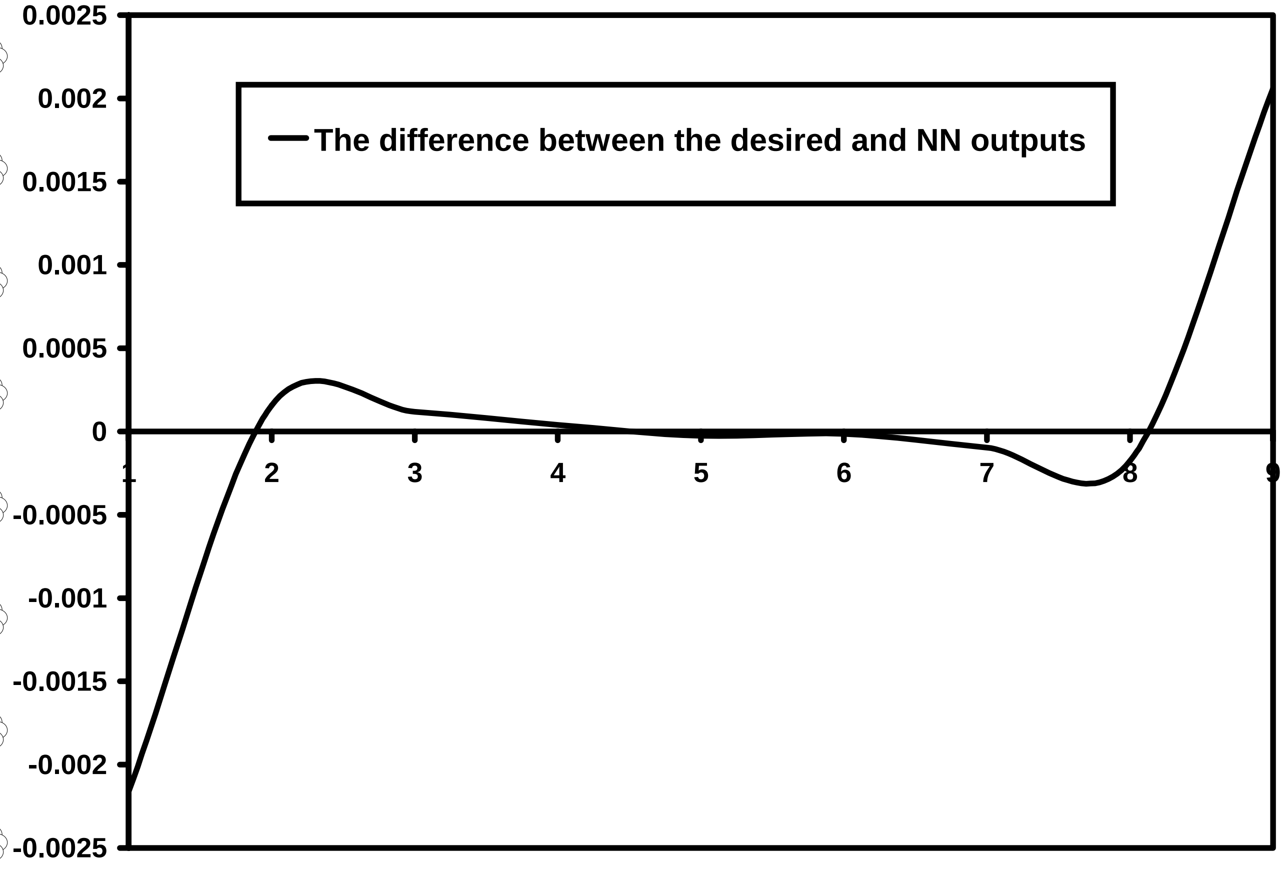


Fig. (3.4 b) The difference between the desired output and the output of the network (with BP) algorithm

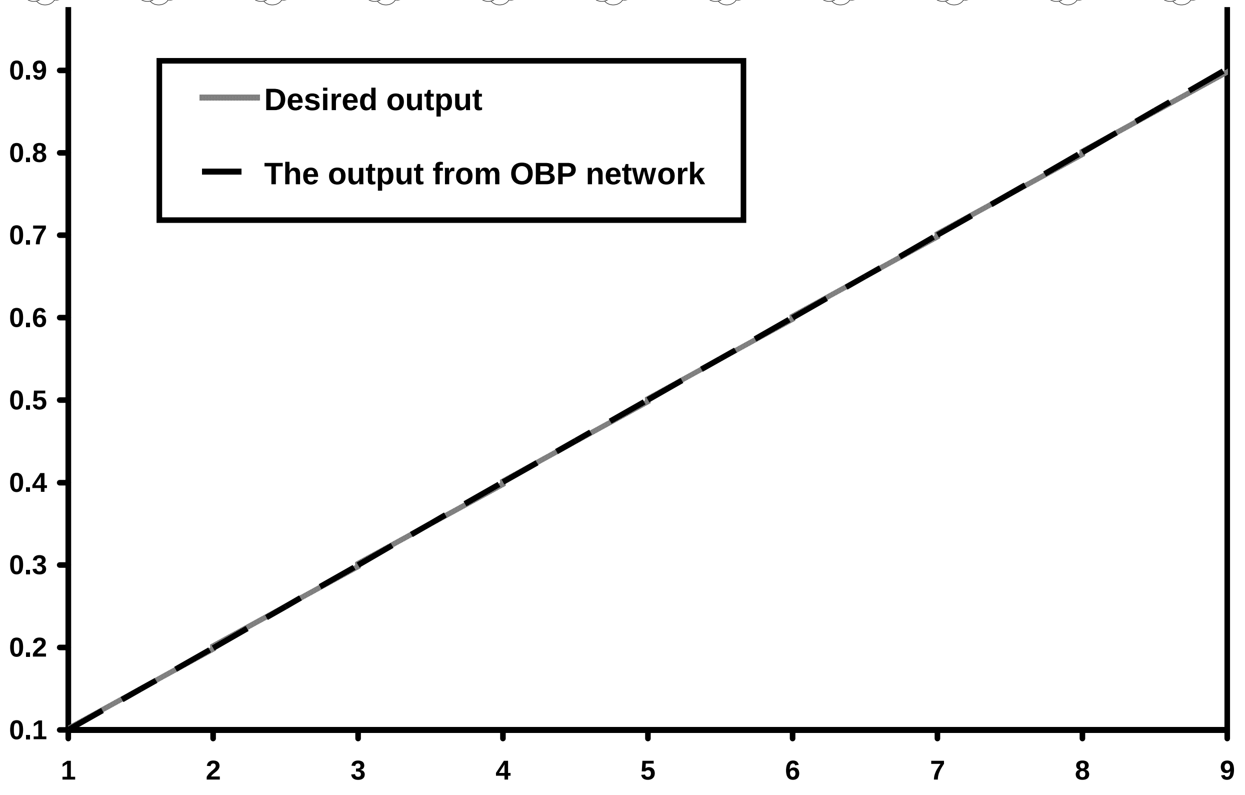


Fig. (3.5) the output and error of the network with BP algorithm and learning rate 0.1

Fig. (3.5 a) The desired and OBP outputs after NN takes 964 Epoch

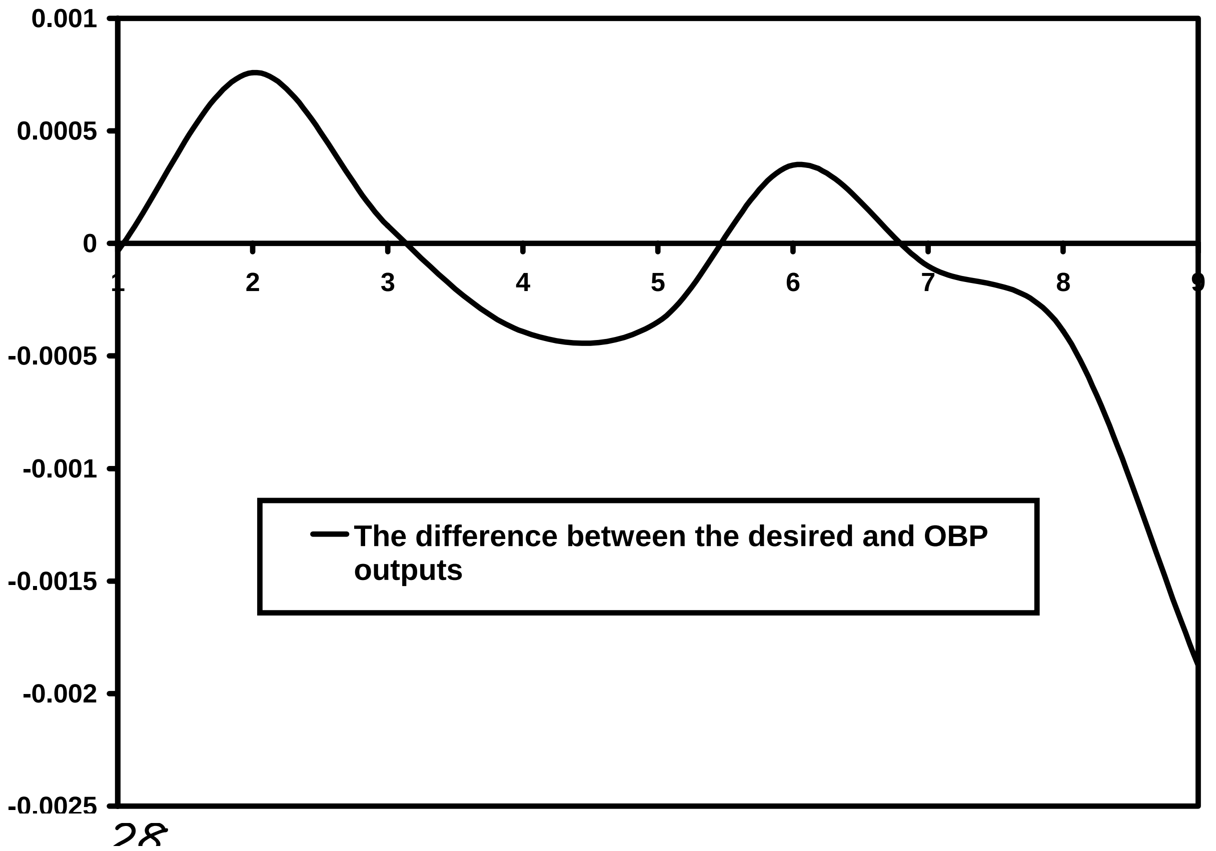


Fig. (3.5 b) The difference between the desired output and the output of the network (with OBP) algorithm

CHAPTER FOUR

CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

CHAPTER FOUR

Conclusions and suggestions for future work

4.1 Conclusions

This section includes some conclusions, which are:

- OBP algorithm has been proposed to train multilayer neural networks and it is an enhanced version of the standard BP.
- OBP is beneficial in speeding up the learning process when the learning rate is small. So, OBP is a good algorithm, because it can adapt all the weights with optical time.
- When a very small value is used for learning rate with OBP makes the adapted final weights very closed become to the final weights that introduced from BP.
- So, the last conclusion OBP can escape from the local minima.

4.2 Suggestions for future work

This project is considered as the base step for future work that includes:

- Using another types of neural network such as CMAC and make a comparison between them from the point of tracking ability, robustness, initial condition requirements.
- OBP algorithm can be implemented practically for a wide range of important applications such as in control, when it is required to control the behavior of complex non linear systems.

Ref er enc e

1. Phil Picton, "Neural Network", 2000

2. Abeer F. Shimal,"CMAC as an adaptive controller of non linear system ", M.Sc. Thesis, university of technology, 2001

3. Sigeru Omatu, Marzuki Khalid and Rubiyah Yousof,"Neuro – Control and its applications", Springer-verlay London limite 1996.

4. Jacek M. Zurada, "Introduction to Artificial Neural Systems", JAICO publishing house

5. Richard P. Lippmann, "An Introduction to computing with Neural Nets ", IEEE ASSP magazine April 1987

6. Robert J. Schalkoff, "Artificial Neural Networks", Co – published by the MIT press and the Mc Graw - Hill companies. Inco, 1997.

7. Ahmed Sabah, "A neural controller with a pre assigned performance index ", M.Sc. thesis, university of Technology, 2000

8. Mohammed A.Qtair, "Speeding up neural networks ", Jordan university of science and Technology, 2005