iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 6 **Issue:** V **Month of publication:** May 2018

DOI: http://doi.org/10.22214/ijraset.2018.5196

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue V, May 2018- Available at www.ijraset.com

Back Propagation Algorithm to Solve Ordinary Differential Equations

Tejal B.Shah[1]
[1]*Science & Humanity Department, Vadodara Institute of Engineering*

Abstract: ***As Artificial neural network has collective approximation competence, it is used to find the solution of ordinary differential equation with given initial conditions .This paper finds the solution of ordinary differential equation using Back Propagation Algorithm. It adds many attractive features to the solution. The solution obtained by Back Propagation Algorithm is differentiable in a closed analytic form and can be used in subsequent calculation.***
Keywords: ***Artificial Neural Network, Multilayer Perceptron, Back Propagation Algorithm, Ordinary differential equations***

I. INTRODUCTION

In Artificial Neural Networks, when continuous function estimates are required to a high degree of accuracy, multilayered structures are used.Neurons must move from the linear to the sigmoidal system. Back propagation algorithm is a supervised learning algorithm for training Multi-layer Perceptron .It is successfully used to solve some difficult and diverse problems. This algorithm is also known as the Error correction learning rule. It consists of two passes. One is forward pass in which an input pattern is applied to the sensual nodes of the network, and its effect propagates through the network layer by layer. A set of outputs is produced as the actual response of the network and during this pass weights remain fixed. Other is backward pass. During this d pass, the synaptic weights are all adjusted in accordance with an error-correction rule. This error signal is propagated backward through the network, against the direction of synaptic connections.

II. BACK PROPAGATION ALGORITHM TO FIND SOLUTION OF FIRST ORDER DIFFERENTIAL EQUATION

Let us consider the general form of differential equations with initial or boundary conditions which represent both ordinary and partial differential equations:

$$\varphi(x, y(x), Dy(x), D^2y(x), \dots) = 0 \ , x \in E \ \dots\dots\dots (1)$$

Where $x = (x_1, x_2, \dots, x_n) \in R^n$, $E \in R^n$ denotes the domain and $y(x)$ denotes the solution to be computed .D is a differential operator. To find the solution of the differential equation discretized given domain E into finite set of points. Thus the equation (1) become

$$\varphi(x_i, y_t(x_i), Dy_t(x_i), D^2y_t(x_i), \dots) = 0 \ \dots\dots\dots (2)$$

Here $y_t(x, p)$ denotes the trial solution with adjustable parameters P like weights and biases. Thus equation (2) can be written as

$$\varphi(x_i, y_t(x_i, P), Dy_t(x_i, P), D^2y_t(x_i, P), \dots) = 0 \ \dots\dots\dots (3)$$

Now, $y_t(x, p)$ may be written as sum of two terms

$$y_t(x, p) = A(x) + F(x, N(x, P)) \ \dots\dots\dots (4)$$

Here $A(x)$ satisfies initial or boundary conditions and contains no adjustable parameters.
$N(x, P)$ is the output of feed forward neural network with adjustable parameter P and input x. The second term $F(x, N(x, P))$ has no contribution to initial or boundary conditions but it is used to minimize error by adjusting weights and biases.
Let us consider a multilayered neural network with one input node, a hidden layer with m nodes and one out unit. For the given inputs $x = (x_1, x_2, \dots, x_n)$ the output is given by

$$N(x, p) = \sigma(Y_k) \ where \ Y_k = \sum_{j=1}^{m} V_{kj}(Z_j) \quad \dots\dots\dots (5)$$

$$\text{Where } Z_j = \sigma(Z_j) = \sigma\left(\sum_{i=1}^{n} w_{ji} x_i + u_j\right) \dots\dots\dots (6)$$

Where w_{ji} denotes the weight from input unit i to the hidden unit$j$, V_j denotes weights from the hidden unit j to the output unit, u_j denotes the biases and $V_j\sigma(Z_j)$ is the sigmoid activation function.
The form of first order differential equation is

$$\frac{dy}{dx} = f(x, y), x \in [a, b] \ \dots\dots\dots (7)$$

with initial condition $y\ (a) = A$
The trial solution using ANN is

©IJRASET: All Rights are Reserved

$$y_t(x,p) = A + (x-a)N(x,p) \text{ ..... (8)}$$

Where $N(x,p)$ is the neural network output of the feed forward network with one input data x with parameter P.
The trial solution $y_t(x,p)$ satisfies the initial condition.

$$\frac{dy_t(x,p)}{dx} = N(x,p) + (x-a)\frac{dN(x,p)}{dx} \text{ ..... (9)}$$

Now to find $\frac{dN(x,p)}{dx}$,following derivatives are required.
From equation (5)

$$\frac{dN(x,p)}{dx} = \sum_{j=1}^{m} V_j \sigma'(Z_j)\, w_{ji} \text{ ..... (10)}$$

$$\frac{dy_t(x_i,p)}{dx_i} = \sum_{j=1}^{3} V_j Z_j + x_i \frac{d}{dx_i}\left(\sum_{j=1}^{3} V_j Z_j\right) \text{ ..... (11)}$$

$$\frac{\partial}{\partial w_{ji}}\left[\frac{dy_t(x_i,p)}{dx_i}\right] = \frac{\partial}{\partial w_{ji}}\left[\sum_{j=1}^{3} V_j Z_j\right] + \frac{\partial}{\partial w_{ji}}\left[x_i \frac{d}{dx_i}\left(\sum_{j=1}^{3} V_j Z_j\right)\right]$$

$$= \sum_{j=1}^{3} V_j Z_j (1-Z_j)\, x_i + \frac{d}{dw_{ji}}\left(x_i \sum_{j=1}^{3} V_j Z_j (1-Z_j) w_{ji}\right)]$$

$$= \sum_{j=1}^{3} V_j Z_j (1-Z_j)\, x_i + x_i \sum_{j=1}^{3} V_j Z_j (1-Z_j) + x_i \sum_{j=1}^{3} V_j Z_j (1-Z_j)^2 w_{ji} - x_i \sum_{j=1}^{3} V_j {Z_j}^2 (1-Z_j) w_{ji}$$

$$= 2\sum_{j=1}^{3} V_j Z_j (1-Z_j)\, x_i + x_i \sum_{j=1}^{3} V_j Z_j (1-Z_j)^2 w_{ji} - x_i \sum_{j=1}^{3} V_j {Z_j}^2 (1-Z_j) w_{ji} \text{ ..... (12)}$$

$$\frac{\partial}{\partial v_j}\left[\frac{dy_t(x,p)}{dx}\right] = \frac{\partial}{\partial v_j}\left[\sum_{j=1}^{3} V_j Z_j + x_i \frac{d}{dx_i}\sum_{j=1}^{3} V_j Z_j\right]$$

$$= \sum_{j=1}^{3} Z_j + \frac{\partial}{\partial v_j}\left[x_i \frac{d}{dx_i}\sum_{j=1}^{3} V_j Z_j\right]$$

$$= \sum_{j=1}^{3} Z_j + \frac{\partial}{\partial v_j}\left[x_i \sum_{j=1}^{3} V_j Z_j (1-Z_j) w_{ji}\right]$$

$$= \sum_{j=1}^{3} Z_j + x_i \sum_{j=1}^{3} V_j Z_j (1-Z_j) w_{ji} \text{ ..... (13)}$$

$$\frac{\partial}{\partial u_j}\left[\frac{dy_t(x,p)}{dx}\right] = \frac{\partial}{\partial u_j}\left[\sum_{j=1}^{3} V_j Z_j + x_i \frac{d}{dx_i}\sum_{j=1}^{3} V_j Z_j\right]$$

$$= \sum_{j=1}^{3} Z_j (1-Z_j) + x_i \sum_{j=1}^{3} (1-Z_j)^2 Z_j w_{ji} - \sum_{j=1}^{3} {Z_j}^2 (1-Z_j) w_{ji} \text{ ..... (14)}$$

The error function is formulated as

$$E(w,u_j) = \sum_{j=1}^{3} \left(\frac{dy_t(x,p)}{dx} - f(x_i, y_t(x,p))\right)^2 \text{ ..... (15)}$$

$$\frac{\partial}{\partial w_{ji}} f(x_i, y_t(x,p)) = \frac{\partial}{\partial w_{ji}} y_t(x,p)$$

$$= \frac{\partial}{\partial w_{ji}}[xN(x,p)]$$

$$= \frac{\partial}{\partial w_{ji}}\left[x_i \sum_{j=1}^{3} V_j Z_j\right]$$

$$= x_i \sum_{j=1}^{3} V_j Z_j (1-Z_j)\, x_i] \text{ ..... (16)}$$

The weight from input to hidden are modified according to the following rule:

$${w_{ji}}^{r+1} = {w_{ji}}^{r} - \alpha\left(\frac{\partial E}{\partial w_{ji}}\right) \text{ ..... (17)}$$

α is a learning rate.r is an iteration step.

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 6.887
Volume 6 Issue V, May 2018- Available at www.ijraset.com

III.EXAMPLE

Let us solve first order ordinary differential equation using Back Propagation Algorithm.

$$\frac{dy}{dx} = x - y \ , y(0) = 0 \ \ in \ \ [0,1]$$

At ten equidistance points in the given domain, the solution of given differential equation is obtained. According to Table-1 solution using analytical method and ANN are same. Also here MATLAB ODE45 tool is used to compare the solution in given domain, which is also same as analytical solution.

TABLE I Comparison of Analytical solution, ANN solution and solution using MATLAB ODE45

Input data x	Analytical Solution	Solution using BPA	Solution using ODE45 at different point in [0,1]				
0.1	0.0048	0.0048	0	0.0288	0.1065	0.2224	0.3679
0.2	0.0187	0.0187	0.0003	0.0346	0.1166	0.2357	
0.3	0.0408	0.0408	0.0012	0.0408	0.1269	0.2493	
0.4	0.0703	0.0703	0.0027	0.0475	0.1377	0.2632	
0.5	0.1065	0.1065	0.0048	0.0547	0.1488	0.2774	
0.6	0.1488	0.1488	0.0075	0.0623	0.1603	0.2919	
0.7	0.1966	0.1966	0.0107	0.0703	0.172	0.3066	
0.8	0.2493	0.2493	0.0145	0.0788	0.1842	0.3215	
0.9	0.3066	0.3066	0.0187	0.0876	0.1966	0.3367	
1	0.3679	0.3679	0.0235	0.0969	0.2093	0.3522	

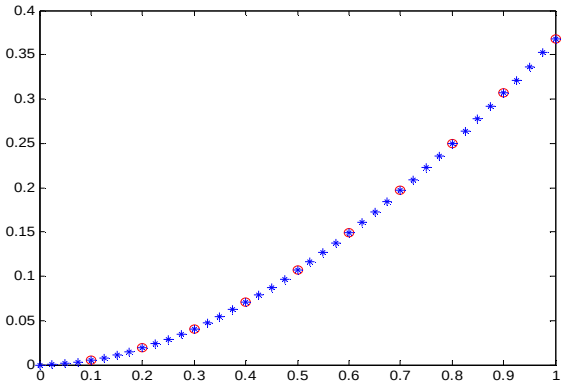


Fig. 1 Comparison of solution of ordinary differential equation using Back Propagation Algorithm and ODE45 tool in MATLAB.

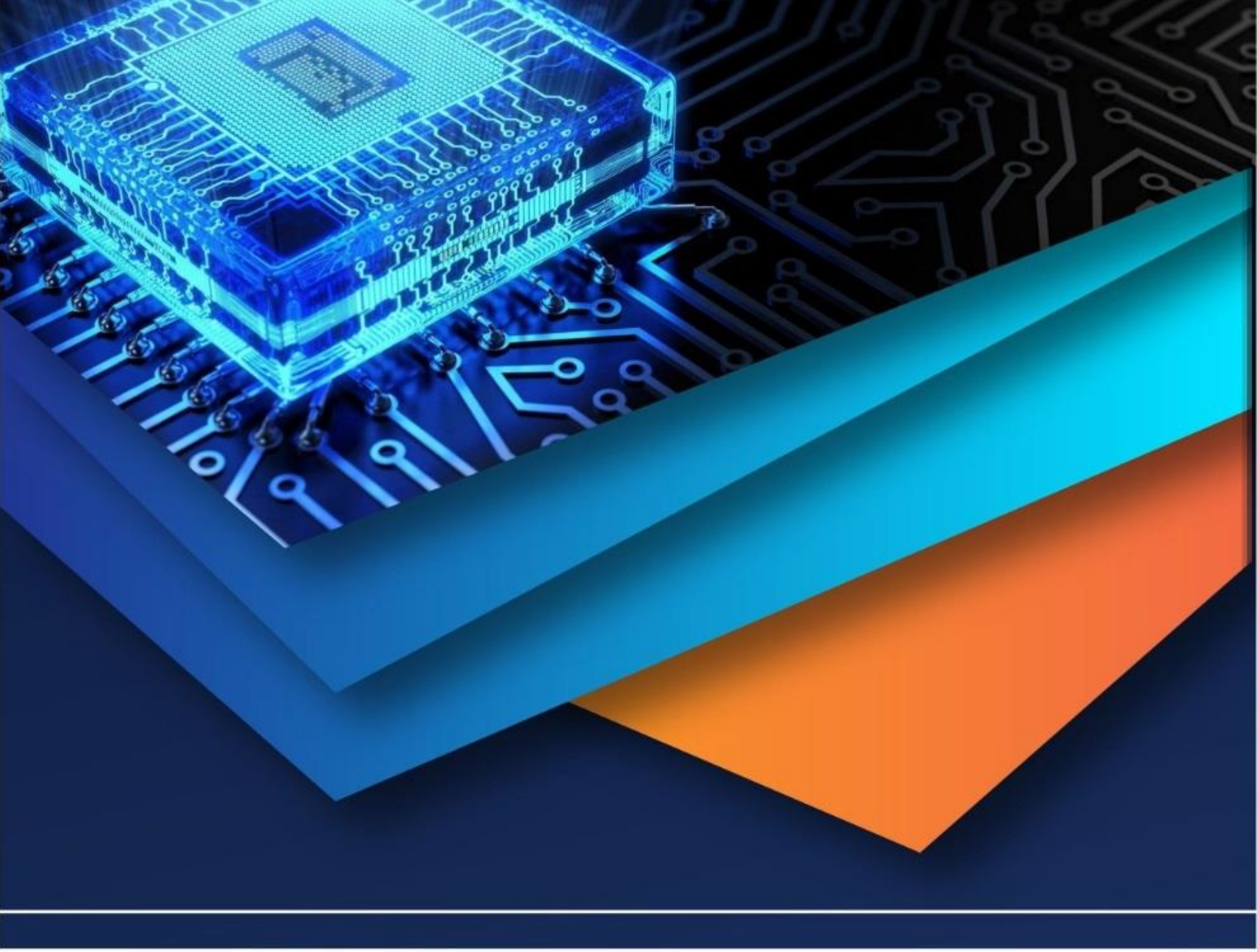
IV. CONCLUSIONS

A new approach to solve ordinary differential equation is shown in this paper. Back Propagation Algorithm is very easy and straight forward to use. Using this algorithm computationally efficient and accurate solution is possible.

REFERENCES

[1] I.E.Lagaris,A.Likas,Artificial neural networks for solving ordinary and partial differential equations,IEEE Transactions on Neural Networks 9(1998)987-1000
[2] M.Kumar,N.Yadav,Multilayer perceptions and radial basis function neural network methods for the solution of differential equations: A survey, ELSEVIER Computers and Mathematics with Applications 62(2011)3796-3811
[3] J.M.ZURADA,Introduction to artificial neural systems,165-205
[4] S.Kumar,Neural networks, A classroom approach,167-214

©IJRASET: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)