

# Accelerated Backpropagation Learning: Extended Dynamic Parallel Tangent Optimization Algorithm

Ali A. Ghorbani<sup>1</sup> and Lila Bayat<sup>2</sup>

<sup>1</sup> University of New Brunswick, Fredericton, NB, Canada

<sup>2</sup> Iran University of Science and Technology, Tehran, Iran

**Abstract.** The backpropagation algorithm is an iterative gradient descent algorithm designed to train multilayer neural networks. Despite its popularity and effectiveness, the orthogonal steps (zigzagging) near the optimum point slows down the convergence of this algorithm. To overcome the inefficiency of zigzagging in the conventional backpropagation algorithm, one of the authors earlier proposed the use of a deflecting gradient technique to improve the convergence of backpropagation learning algorithm. The proposed method is called Partan backpropagation learning algorithm[3]. The convergence time of multilayer networks has further improved through dynamic adaptation of their learning rates[6]. In this paper, an extension to the dynamic parallel tangent learning algorithm is proposed. In the proposed algorithm, each connection has its own learning as well as acceleration rate. These individual rates are dynamically adapted as the learning proceeds. Simulation studies are carried out on different learning problems. Faster rate of convergence is achieved for all problems used in the simulations.

**Keywords:** Artificial neural networks, Backpropagation, Gradient descent, Parallel tangent, Dynamic parallel tangent.

## 1 Introduction

Backpropagation (BP) is the most popular and widely used learning algorithm for multilayer feedforward neural networks. The main limitation of BP is the slow pace at which it learns from examples. This is due to the fact that the standard backpropagation method uses fixed learning steps, and as the result slows down in flat areas and starts to take orthogonal steps near the optimum point. Over the last number of years, many new accelerating techniques have been developed to speedup the rate of convergence in the backpropagation training algorithm. A global error gradient adaptation technique called parallel tangent (Partan) training algorithm is proposed by one of the authors[3]. The proposed method can be used to accelerate the training process in multilayer neural networks.

In [6], we have proposed a dynamic parallel tangent learning algorithm that further improves the speed of training multi-layer neural networks. The improvement is done through the dynamic adaptation of the learning rates during the

training process. Faster rate of convergence is achieved for all learning problems used in the simulation studies of dynamic Partan.

In this paper, we present an extension to the dynamic Partan learning algorithm. In the extended dynamic Partan, the learning and the accelerating rates are adapted dynamically as the training proceeds. Other acceleration techniques can also be incorporated in this method to further improve the rate of convergence.

The outline of the paper is as follows. The concept of gradient descent and parallel tangent gradient is briefly reviewed in the following section. In Section 3, Partan backpropagation learning algorithm is explained. Dynamic Partan, extended dynamic Partan, and rate adaptation strategies are presented in Section 4. Subsequently, the results of the simulation studies are summarized in Section 5. Finally, conclusions of the present study are summarized.

## 2 Parallel Tangent (Partan) Gradient

The method of gradient descent is one of the most fundamental procedures for minimizing a differentiable function of several variables. In general, the gradient algorithm takes a point  $p_i \in S \subset E^n$  and computes a new point  $p_{i+1} \in S \subset E^n$ , where  $S$  represents an arbitrary set and  $E$  represents the Euclidean space. The new point is defined by making

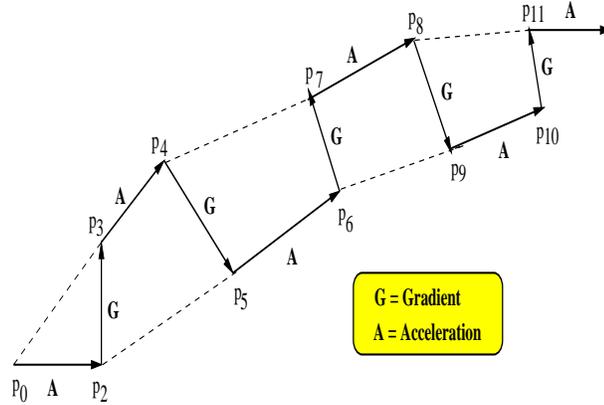
$$p_{i+1} = p_i + \eta g$$

where,  $\eta > 0$  for minimization or  $\eta < 0$  for maximization. Further,  $p_i$  is the origin of the line and  $g$  is the gradient vector,  $\nabla f(p_i)$ , determining the direction and  $\eta$  is the step-size parameter to be estimated. The gradient algorithm usually behaves poorly near an optimum point where small orthogonal steps are taken (zigzagging phenomena). To illustrate the zigzagging phenomena, let us consider an objective function with concentric ellipsoidal contours as shown in Figure 1. If the initial point for a gradient search happens to be precisely on one of the axes of the systems of ellipses, the gradient line will pass right through the optimum (peak) and the search will be over in one descent (ascent). Otherwise, the search will follow a zigzag course such as the one from  $p_0$  to  $p_2$  to  $p_3$  to  $p_4$ , etc. (in order to be consistent with the convention adopted in the next section, after  $p_0$  the next point is denoted as  $p_2$ , instead of  $p_1$ ).

It can be seen that the crooked path is bounded by two straight lines which intersect at the optimum. This suggests that the search from point  $p_3$  be conducted, not in the gradient direction toward  $p_4$ , but along the straight line from  $p_0$  through  $p_3$ . In this way, the peak,  $p^*$  would be located after three steps: first from  $p_0$  to  $p_2$  along the gradient at  $p_0$ , then from  $p_2$  to  $p_3$  along the gradient at  $p_2$ , and finally from  $p_3$  along the line through  $p_0$  and  $p_3$ . This is the two dimensional version of a method which accelerates along a ridge and usually is called gradient parallel tangents (gradient Partan)[2, 8, 9].

Parallel tangent has many forms and gradient Partan is one form which combines many desirable properties of the simple gradient methods[8]. This technique represents a distinct improvement over the method of steepest descent.





**Fig. 2.** Parallel Tangent Gradient.

### 3 Partan Backpropagation

In practice, the backpropagation training algorithm has proved to be a suitable method in computing a weight vector that enables the network to perform certain input-output mapping. It teaches a network iteratively. In order to properly train the network, an objective function which is the result of the contributions of all training samples, is simultaneously minimized. The BP algorithm minimizes the network's error by continuously adjusting the networks connection weights. The weights are updated using the gradient of the cost function, as follows:

$$w_{i+1} = w_i + \eta g,$$

where  $g$  represents the gradient of the cost function and  $\eta$  represents the learning rate, respectively. Standard backpropagation algorithm (generalized delta rule) has the reputation of being very slow. It suffers from the major drawbacks which are associated with steepest descent technique. As explained in Section 2, the gradient descent method starts to zigzag near the optimum point. This is mainly due to the fact that, using a fixed learning rate to determine the step size may not be appropriate for all regions of the error surface.

The parallel tangent (Partan) overcomes the difficulties of the zigzagging phenomena by deflecting the gradient steps. The Partan technique combines many desirable properties of the simple gradient method. It uses an accelerating step after each gradient step, and can be used as an alternative for momentum term to accelerate the convergence. In Partan, connection weights are updated as follows:

```

PROCEDURE Partan(input[p0, ε], output[p*]);
/* η and μ are appropriate step sizes for climbing and acceleration,
respectively. Γ represents the function that is used to compute a proper
step size and n represents the number of independent variables[2].
*/

BEGIN
  p* = p0;
  REPEAT
    pk = p0 = p*;
    p* = pk - η∇f(pk);
    FOR i = 1 to n DO
      BEGIN
        η = Γ(η, ∇f(p*), n);
        g = p* - η∇f(p*);
        δ = g - pk;
        pk = p*;
        p* = g + μδ;
        μ = Γ(μ, δ, n);
      END;
    UNTIL (|| p0 - p* || < ε);
  RETURN (p*);
END.

```

**Fig. 3.** Parallel tangent gradient optimization algorithm.

$$w_{i+1} = w_i + \eta g + \mu s,$$

where  $s$  represents a direction based on two previous gradient steps and  $\mu$  is the accelerating rate. The general framework of the new training technique is defined as follows. This procedure can be restarted every  $n$  steps, however, global convergence is not tied to this restart.

```

Begin
  Do one gradient step.
  While (error >= Desired Threshold)
    Do one gradient step.
    Do one accelerating step.
  End
End

```

A proposed detailed algorithm for the Partan backpropagation is given in Figure 4. This procedure starts with an arbitrary starting point and searches for an optimum using a positive termination scalar,  $\epsilon$ . Starting at point  $w_0$ , point  $w_2$  is found by a standard gradient step. Following the initial step, the optimization is continued for  $n$  iterations and may restart from another random initial point until the optimum weight vector  $w^*$  is found. After  $n$  iterations, one has the choice of either continuing the cycle of backpropagation search and acceleration

```

PROCEDURE Partan_backprop(input[I, D, w0, ε], output[w*]);
/* I and D are the input and the desired output vectors, respectively.
w0 is the starting point, w* is the optimum weight vector, and ε is a
termination scalar which is chosen to be greater than zero. η and μ are
appropriate step sizes for climbing and acceleration, respectively.
Γ represents the function that is used to compute a proper step size and
n represents the number of independent variables. M is the number of
epochs. backprop is the standard backpropagation procedure that returns
the gradient of the criterion function at a given point and the amount of
existing error.
*/

BEGIN
  j = 1;
  w* = w0;
  REPEAT
    wk = w0 = w*;
    Call backprop(input[I, wk, ε, D], output[∇wk, error]);
    w* = wk - η∇wk;
    FOR i = 1 to n DO
      BEGIN
        Call backprop(input[I, w*, ε, D], output[∇w*, error]);
        IF (error < ε) RETURN(w*), EXIT;
        η = Γ(η, ∇w*, n);
        g = w* - η∇w*;
        δ = g - wk;
        wk = w*;
        w* = g + μδ;
        μ = Γ(μ, δ, n);
      END;
    UNTIL (error < ε OR j ≥ M);
  RETURN (w*);
END.

```

**Fig. 4.** Parallel tangent backpropagation learning algorithm.

or starting over again. In Figure 4, we have presented the latter choice.

The learning rate,  $\eta$ , plays an important role in the convergence of a network. Choosing appropriate value for the learning rate can speed up the training process. A large learning rate is efficient in the flat regions of the error surfaces, but, usually causes oscillation near the optimum point. On the other hand, small learning rates tend to slow down the convergence of a network. We have shown that the adaptation of the learning rate during the training process speeds up the convergence[6]. The learning method proposed in [6] is called dynamic Partan. In the dynamic Partan, the learning rate is adapted for each gradient step. The adaptation is done with respect to the properties of the error surfaces. The framework of the dynamic Partan training algorithm is as follows:

```

Begin
  Do one gradient step.
  While (error >= Desired Threshold)
    Adapt learning rate.
    Do one gradient step.
    Do one accelerating step.
  End
End

```

In this paper, we propose the use of dynamic accelerating rate,  $\mu$ , in the Partan backpropagation. In the proposed method, the accelerating rate is adapted prior to taking the acceleration step. The adaptation technique is similar to that of the learning rate adaptation. The extended dynamic Partan algorithm is as follows:

```

Begin
  Do one gradient step.
  While (error >= Desired Threshold)
    Adapt learning rate.
    Do one gradient step.
    Adapt accelerating rate.
    Do one accelerating step.
  End
End

```

### 3.1 Rates Adaptation

In standard parallel tangent backpropagation algorithm, the learning as well as the accelerating rates are fixed during the process of training the network. Moreover, all the connection weights use the same learning and accelerating rates. In dynamic Partan, the learning rates are adapted continuously over time. The adaptation is done with respect to the shape of the error surfaces. The sign of the gradient is used to adapt the learning rates. If consecutive changes (local gradients) of a connection weight possess the same signs, the learning rate for that connection is increased [1]. This increase helps to take a longer step in the next iteration. If consecutive gradients possess opposite signs, it shows that the previous learning rate has been too large and a jump over the local minima has occurred. Thus, the next step should be carried out with smaller learning rate. This is done by removing the effect of the previous step (i.e., backtracking one step) and decreasing the learning rate appropriately.

In the dynamic Partan, the accelerating rate is fixed during the training of the network, whereas, in the extended dynamic Partan proposed in this paper, each connection has its own accelerating rate. The accelerating rates are also adapted dynamically as the training proceeds. The adaptation of the accelerating rates is done similar to that of the learning rates. The accelerating rate is increased or decreased whenever the corresponding learning rate is increased or decreased.

Four dynamic Partan schemes called Partan1, Partan2, Partan3, and Partan4 are presented in this paper. Dynamic Partan1 and 2 use variable learning rates and fixed accelerating rates during the training process. In Dynamic Partan1, the learning rates are adapted as follows,

$$\begin{aligned}\eta_{i+1} &= \eta_i * \nu^+, \\ \eta_{i+1} &= \eta_i * \nu^-, \end{aligned}$$

where  $\nu^+$  and  $\nu^-$  are the adaptation rates used to increase and decrease the learning rates, respectively. Whereas in dynamic Partan2, adaptation rates ( $\nu^+$ ,  $\nu^-$ ) are added/subtracted to/from the learning rates.

Dynamic Partan3 and 4 are extended versions of dynamic Partan1 and 2. In these schemes, dynamic learning rates as well as dynamic accelerating rates are used during the training process. The accelerating rates are adapted similar to the adaptation of the learning rates in the dynamic Partan1 and 2. In dynamic Partan3, the accelerating rates are adjusted as follows,

$$\begin{aligned}\mu_{i+1} &= \eta_i * \tau^+, \\ \mu_{i+1} &= \eta_i * \tau^-, \end{aligned}$$

where  $\tau^+$  and  $\tau^-$  are the adaptation rates used to increase and decrease the accelerating rates, respectively. In dynamic Partan4, adaptation rates are added/subtracted to/from the accelerating rates.

## 4 Simulation

In order to evaluate the performance of the extended dynamic Partan learning algorithm, simulation studies are carried out on different learning problems. The learning problems are chosen so that they possess different error surfaces and collectively represent an environment that is suitable to determine the effect of the proposed learning algorithms. For all the methods presented in this paper, the backpropagation procedure is used to calculate the partial derivatives of the error with respect to each weight.

The network architecture are predetermined, specifying the number of hidden units, the step sizes  $\eta$ s and  $\mu$ s, the number of patterns in the training set, and the convergence criterion,  $\epsilon$ , which was set so that the average error per pattern in the training set is below some threshold. For the standard backpropagation networks, we have selected the architectures and learning parameters (i.e., learning and momentum rates) that resulted in good performance. The same parameters and architectures are used for different Partan schemes. Ideal architectures for Partan algorithms may even show faster rate of convergence. The simulation studies are carried out using a large number of learning problems. The results for Sin function, Sonar classification problem and a character recognition problem are summarized in Tables 1-3.

At the start of each simulation, the weights are initialized to random values between  $+r$  to  $-r$ . Since the backpropagation algorithm is sensitive to different starting points, we carried out our simulation with various runs starting from different random initializations for the weights of the network. For each algorithm, 25 simulations were attempted.

Four schemes of the dynamic Partan namely Partan1, Partan2, Partan3, and Partan4 are implemented. Dynamic Partan1 and 2 use variable learning rates and fixed accelerating rates, whereas, dynamic Partan3 and 4 use dynamic learning rates as well as dynamic accelerating rates during the training process.

The results of the simulations for the above problems are summarized in Tables 1-3. In these tables,  $\alpha$  represents the momentum rate used in the standard backpropagation training algorithm. The results shown in these tables clearly indicate that dynamic Partan3 and dynamic Partan4 exhibit a faster rate of convergence comparing to the standard backpropagation and the standard Partan.

#### 4.1 Sin Problem

In this problem, neural networks are used for the function approximation. The network architecture used for this problem consists of one input unit, one output unit with linear activation function, and two hidden layers with 8 and 3 sigmoidal units, respectively. The training is considered complete when the cumulative error is below 0.01. The training and test sets consist of 60 and 360 points in the range  $[-\pi, +\pi]$ , respectively.

The results of our simulation studies are summarized in Table 1. It is seen that on the average, the standard backpropagation with  $\eta = 0.6$  and  $\alpha = 0.5$  converges after 218 epochs. The dynamic Partan1 with the same learning rate as that of the standard backpropagation and  $\mu = 0.8$  converges after 125 epochs. Dynamic Partan4 converges after 85 epochs. The results show that, on the average, the dynamic Partan4 converges twice as fast as the standard Partan and 2.56 times faster than the standard BP algorithm.

**Table 1.** The training results for Sin function.

Learning Parameters	$r$	$\eta$	$\alpha$	$\mu$	$\nu^+$	$\nu^-$	$\tau^+$	$\tau^-$	Avg. Epochs
Backpropagation	0.5	0.6	0.5	-	-	-	-	-	218
Standard Partan	0.5	0.19	-	0.76	-	-	-	-	167
Dynamic Partan 1	0.5	0.6	-	0.8	1	0.7	-	-	125
Dynamic Partan 2	0.5	0.95	-	0.8	0	0.05	-	-	126
Dynamic Partan 3	0.5	0.6	-	0.5	1	0.7	0.01	0.02	88
Dynamic Partan 4	0.95	0.6	-	0.6	1	0.7	0.95	0.83	85

#### 4.2 Sonar Problem

This problem is the classification of sonar signals using neural networks. The task is to train a network to discriminate between sonar signals bounced off a metal cylinder and those bounced off a roughly cylindrical rock. The problem contains of 104 training examples (49 Mine patterns and 55 Rock patterns) and 104 test

examples (62 Mine patterns and 42 Rock patterns). Each pattern consists of 60 numbers in the range of 0.0 to 1.0 as input and 2 binary values as output. The network architecture used for this problem consists of 60 input units, 24 hidden units and 2 output units.

The simulation results for this problem are given in Table 2. The training is considered complete when the error for one epoch is less than 0.1. It is seen that dynamic Partan3 converges about 2.5 times faster than standard BP and about 1.6 times faster than standard Partan.

**Table 2.** The training results for Sonar, Mines vs. Rocks Problem.

Learning Parameters	$r$	$\eta$	$\alpha$	$\mu$	$\nu^+$	$\nu^-$	$\tau^+$	$\tau^-$	Avg. Epochs
Standard BP	0.3	0.4	0.9	-	-	-	-	-	390
Standard Partan	0.3	0.2	-	0.67	-	-	-	-	257
Dynamic Partan 1	0.35	0.95	-	0.74	1	0.29	-	-	199
Dynamic Partan 2	0.35	0.98	-	0.7	0	0.205	-	-	203
Dynamic Partan 3	0.3	0.32	-	1.5	1	0.5	0	0.01	160
Dynamic Partan 4	0.3	0.98	-	1.1	1	0.29	1	0.98	180

After training the network, its memorization and generalization abilities are examined with tests as well as training patterns. The results of testing the network with  $\epsilon = 0.1$  and  $\epsilon = 0.07$  are given in Table 3. It is seen that dynamic Partan schemes show stronger memorization and generalization capabilities. When the error threshold is set to 0.07, the dynamic Partan3 is able to memorize all the training examples and correctly classify 91 percent of unseen patterns from the test set. For the similar cases, standard BP and standard Partan show 96% memorization capability; and 81% and 86% generalization abilities, respectively.

**Table 3.** The memorization and generalization results for Sonar problem.

Algorithm	Error Threshold = 0.1			Error Threshold = 0.07		
	Epochs	Memorize.	Generalize.	Epochs	Memorize.	Generalize.
Standard BP	338	94%	80%	390	96%	81%
Standard Partan	179	95%	85%	257	96%	86%
Dynamic Partan1	169	96%	85%	199	96%	82%
Dynamic Partan2	169	96%	86%	203	96%	86%
Dynamic Partan3	136	96%	90%	160	100%	91%
Dynamic Partan4	149	97%	89%	180	98%	90%

### 4.3 Character Recognition Problem

The network architecture used for solving this problem consists of 64 input units, twenty five hidden units, and one output unit. The task is to train a multilayer neural network to recognize English capital letters. Each letter is represented as an  $8 \times 8$  matrix of 0s and 1s. There are 24 different patterns for each letter. Six patterns represent positional movements of the letter inside the matrix (i.e., moving the letter up, down, left or right inside its  $8 \times 8$  matrix). Each pattern is represented with four different angles (0, 90, 180, 270 degrees). In other words, besides the original representations of the letters, there are 3 more patterns for each representation of a letter that show the state of that letter after being rotated 90, 180, and 270 degrees.

**Table 4.** The training results for character recognition problem.

Learning Parameters	$r$	$\eta$	$\alpha$	$\mu$	$\nu^+$	$\nu^-$	$\tau^+$	$\tau^-$	Avg. Epochs
Standard BP	0.5	0.01	0.5	-	-	-	-	-	3703
Standard Partan	0.5	0.007	-	0.7	-	-	-	-	1936
Dynamic Partan 2	0.5	0.001	-	0.9	0.003	0.01	-	-	663
Dynamic Partan 3	0.5	0.002	-	0.84	0.003	0.01	0.015	0.0025	543
Dynamic Partan 4	0.5	0.002	-	0.7	0.003	0.01	1.22	0.95	539

The training and test sets each consists of 628 patterns. The training is considered complete when the error of one epoch is less than 0.01. The training results for this problem are shown in Table 4. It is seen that the dynamic Partan4 converges about 7 times faster than the standard BP and about 3.6 times faster than the standard Partan.

## 5 Conclusion

Parallel tangent (Partan) gradient is a deflecting method that combines many desirable characteristics of the simple gradient method and has certain ridge-following properties which make it attractive. The Partan as well as the dynamic Partan are used to accelerate the convergence to the solution of backpropagation learning algorithm[3, 4, 6]. In this paper, we have proposed two extensions to dynamic Partan training algorithm.

The main features of parallel tangent technique are its simplicity, ridge-following, and ease of implementation. The most desirable property of Partan backpropagation, however, is its strong global convergence characteristics. Each step of the process is at least as good as the steepest descent; the additional move (acceleration) to  $p_{i+1}$  provides further decrease of the objective function.

In dynamic parallel tangent, the local information is used for the adaptation of the learning as well as the accelerating rates. The local adaptation of the rates is ‘similar’ to biological neural learning adaptation process and is more suitable

for parallel implementations. We have demonstrated through simulation that the dynamic adaptation of rates is an effective approach to speed up the training of multilayer neural networks. The networks energy function behaves differently in various dimensions during the training process. This concept is simulated by using and dynamically adapting different learning and accelerating rates for the connection weights.

In all the problems we have studied so far, the convergence of the dynamic Partan was faster than the standard BP as well as the standard Partan. Table 5 depicts the speedup achieved for the three learning problems studied in this paper. The results show that on average the rate of convergence of the standard Partan and the dynamic Partan1-4 are approximately 1.56, 2.70, 3.185, 4.36 and 4.3 times faster than that of the standard BP algorithm, respectively.

**Table 5.** Average speedup of standard and dynamic Partan versus standard backpropagation.

Learning problem	s-Partan	d-Partan1	d-Partan2	d-Partan3	d-Partan4
Sin function	1.30	1.74	1.73	2.47	2.56
Sonar problem	1.51	1.95	1.92	2.43	2.16
Char. recognition	1.91	-	5.58	6.81	6.78
Average speedup	1.56	2.70	3.18	4.36	4.3

## References

1. Baldi P., "Gradient Descending Learning Algorithm Overview: A General Dynamic systems Perspective", *IEEE Trans. on Neural Network*, No. 1, Vol. 6, Jan. 1995.
2. Bazaraa M.S. and Shatty C.M., *Nonlinear Programming Theory and Algorithms*, John Wiley & Sons, USA, pp. 253-290, 1979.
3. Ghorbani A.A. and Bhavsar V.C., "Parallel Tangent Learning Algorithm for Training Artificial Neural Networks", Technical Rep. No. *TR93-075*, University of New Brunswick, April 1993.
4. Ghorbani A.A. and Bhavsar V.C., "Accelerated Backpropagation Learning Algorithm: Parallel Tangent Optimization Algorithm", Proc. 1993 International Symposium on Nonlinear Theory and it's Applications (NOLTA'93), pp. 59-62, Hawaii, USA, Dec. 1993.
5. Ghorbani A.A., Nezami A.R. and Bhavsar V.C., "An Incremental Parallel tangent Learning Algorithm for Artificial Neural Networks", CCECE'97, pp. 301-304, Saint John's, Canada, May 1997.
6. Ghorbani A.A. and Bayat L., "Accelerated Backpropagation Learning Algorithm: Dynamic Parallel Tangent Optimization Algorithm", Proc. of IASTED International Conference on Computer Systems And Applications, pp. 116-119, Irbid, Jordan, March 30-April 2, 1998.
7. Gorman, R. P. and Sejnowski, T. J., Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets, *Neural Networks* 1, pp. 75-89, 1989.

8. Wilde D. J. and Beightler C. S., *Foundations of optimization*, Prentice-Hall, Englewood Cliffs, N.J., USA, 1967.
9. Wismer D. A. and Chattergy R., *Introduction to Nonlinear Optimization*, Elsevier North-Holland, Amsterdam, 1978.