A Constant Learning Rate Self-Organizing Map (CLRSOM) Learning Algorithm

VIKAS CHAUDHARY\textsuperscript{1,2}, R. S. BHATIA\textsuperscript{2} AND ANIL K. AHLAWAT\textsuperscript{3}
\textsuperscript{1,2}National Institute of Technology
Kurukshetra, 136119 Haryana, India
\textsuperscript{3}Krishna Institute of Engineering and Technology
Ghaziabad, 201206 India
Email: vikas.jnu@yahoo.co.in

In a conventional SOM, it is of utmost importance that a certain and consistently decreasing learning rate function be chosen. Decrease the learning rate too fast, the map will not get converged and the performance of the SOM may take a steep fall, and if too slow, the procedure would take a large amount of time to get carried out. For overcoming this problem, we have hereafter proposed a constant learning rate self-organizing map (CLRSOM) learning algorithm, which uses a constant learning rate. So this model intelligently chooses both the nearest and the farthest neuron from the Best Matching Unit (BMU). Despite a constant rate of learning being chosen, this SOM has still provided a far better result. The CLRSOM is applied to various standard input datasets and a substantial improvement is reported in the learning performance using three standard parameters as compared to the conventional SOM and Rival Penalized SOM (RPSOM). The mapping preserves topology of input data without sacrificing desirable quantization error and neuron utilization levels.

Keywords: self-organizing map (SOM), constant learning rate, winning frequency, 1-neighborhood neurons, rank

1. INTRODUCTION

Data Visualization is widely accepted and used to by scientists all over the world to solve problems related to pattern recognition, image processing, etc. Sammon mapping [1] and PCA [2] are two classical algorithms used for visualizing multivariate data and for data dimension reduction. For linear data the classical PCA is more effectively used [3, 4], as compared to with non-linear data [4, 5]. Sammon mapping, on the other hand, is more effective than PCA when it comes to analysis of data structure [1]. The Sammon algorithm successfully preserves the data structure, but whenever new data points are added, the projection has to be calculated again from scratch [6], thus leading to the computational complexity being large.

Neural Networks [7-9] is another non-linear data analysis method. The Self-organizing map (SOM) [10, 11] is an unsupervised learning algorithm invented by T. Kohonen. A SOM contains a set of artificial neurons, each with a weight vector. These weight vectors are used to cluster or classify additional data similar to the input data used [12]. The SOM often finds use as a visualization technique for dimensional reduction [13, 14]. The topology of input data is preserved by the SOM by assigning each datum to the most similar neuron, and data with similar properties are mapped into adjacent neurons [15].

Received February 6, 2014; revised June 25 & August 28, 2014; accepted October 30, 2014.
Communicated by Zhi-Hua Zhou.
A SOM is based on competitive learning. In competitive learning, neuron activation is a function of the distance between neuron weight and input data [16]. An activated neuron learns the most and its weights are modified. If a similar kind of pattern is found again, then the same neuron has more chances of being activated again. This means that a particular neuron wins repeatedly. Thus this neuron learns more. Conscience learning is a way proposed by Desieno [17] which has been found effective in preventing this side effect. Further, Rival penalized competitive learning (RPCL) [18] and Rival penalized controlled competitive learning (RPCCL) [19, 20], and a new SOM called shooting SOM (SSOM) algorithm [21] were also proposed. The important feature of SSOM is that first the neurons aim at a target, usually near the area of concentration of input. But it was found that some of the 1-neighborhood neurons of the winner neuron move a bit outside the cluster, while some neurons remained stationary. Due to this, neurons in SSOM tend to crowd in the region of concentration of the input. Using the idea of the rival penalized competitive learning (RPCL) and rival penalization controlled competitive learning (RPCCL), a rival-model penalized self-organizing map (RPSOM) [22] was proposed. The adaptive RPSOM chooses some rivals of the winner for each input, and punishes their weights a bit farther from the data. It is found that in RPSOM, the topographic error is increased, which means topology is not better preserved, making it the most critical drawback of RPSOM. A new technique has been developed which involves the display of input in Gray scale using genetic approach and SSOM [23]. In this the number of times the neuron becomes the winner is reflected by shading. The information of the input that is difficult to judge by appearance can be simplified and visualized through the proposed method.

In general, in conventional SOM, a learning rate is initialized and then gradually reduced to warrant the convergence of the map. However, when the initial value of learning rate is too small, the map formed is often not well established. To circumvent this, the learning rate is initialized at quite a large value, and then slowly reduced over time with the help of a consistently decreasing function. When the learning rate is reduced too slowly, the topology of inputs is learnt well by the map, but the process thus becomes quite lengthy. However, when the learning rate is reduced way too quickly, there are chances that the map will give a local suboptimal solution, thus leading to a relatively large quantization error. It is quite obvious, that the performance of the SOM depends largely on a correct ratio of the decrement of the learning rate by the chosen consistently decreasing function. It is therefore of utmost importance that this function be chosen wisely, without the need for trial-and-error, which is time-consuming and creates confusions. Now the problem that arises is the choice of the decrement function for learning rate.

In order to avoid this problem, the proposed CLRSOM does away with the consistently decreasing function for learning rate. The CLRSOM thus makes the neurons learn by keeping a constant learning rate, helping to better preserve the topology of the map with less quantization error and more neuron utilization in comparison to SOM and RPSOM. The nearest and farthest neurons in the 1-neighborhood of the winner neuron are found as shown in Fig. 1. Then each neuron is assigned a rank according to its distance from input and connection value in the updation process.

The remainder of this paper is organized as follows. In section 2, we explain the conventional SOM learning algorithm and the learning algorithm of the CLRSOM is ex-
Fig. 1. The nearest and farthest neurons of a winner neuron on a rectangular grid. Suppose \( C = 15 \), \( N_{c_1} = \{9, 14, 16, 21\} \). Among \( N_{c_1} \), suppose nearest = 14 and farthest = 16. If farthest = 16, then \( S_f = \{17, 18\} \). If farthest = 21, then \( S_f = \{27, 33\} \).

plained in section 3. In section 4, we conduct the experiments to demonstrate the performance of CLRSOM in comparison with the conventional SOM and RPSOM. In section 5, we discuss the conclusion.

2. SELF-ORGANIZING MAP (SOM)

The SOM consists of a set of artificial neurons, each having a weight vector. The SOM contains some neurons located on 2 dimensional grids, called “map”. The SOM consists of \( m \) neurons arranged on a regular map with low dimensions, usually a 2-D map, for example, \( m \times n \) neurons, that are interconnected with their neighbors to form a map. These neurons then connect to their neighbors as per certain neighborhood connections (also called topological connections) [24]. Two common types of topologies are used for SOM map: hexagonal and rectangular [25, 26]. A neuron has four 1-neighbor neurons and six 1-neighbor neurons in rectangular and hexagonal topologies respectively. Each neuron \( i \) has a \( d \)-dimensional weight vector \( w = (w_{i1}, w_{i2}, \ldots, w_{id}) \), where \( i = 1, 2, \ldots, m \), which is having same dimensions as the input space. In each iteration, \( t \), the input \( x(t) \) is randomly selected from the input dataset and the winner neuron which is called best matching unit (BMU) is found. The BMU neuron will be that neuron whose weight vector is nearest to the input vector \( x \) in Euclidean distance.

2.1 Learning Algorithm

The conventional SOM learning algorithm can be explained as follows:

**Step 1:** The weight vectors of all neurons are initialized randomly.

**Step 2:** Initialize epoch variable \( epo = 1 \) and step variable \( t = 1 \).

**Step 3:** While stop condition does not meet

(i) An input vector \( x(t) \) is randomly selected.

(ii) The BMU \( c \) is found using the given equation:

\[
  c = \arg \left( \min_{i=1}^{m} || w_i(t) - x(t) || \right).
\]
where \( || \cdot || \) is the Euclidean distance measure.

(iii) Update weights of BMU and its neighbors using the following:

\[
w_i(t+1) = w_i(t) + h_{c,i}(t) \cdot [x(t) - w_i(t)],
\]

where \( h_{c,i}(t) \) is called the neighborhood kernel. The common choice of a neighborhood kernel is Gaussian function [11]

\[
h_{c,i}(t) = \alpha(t) \cdot \left( \frac{||r_c - r_i||^2}{2\sigma^2(t)} \right).
\]

(iv) \( t = t + 1 \)

(v) \( epo = epo + 1 \)

**Step 4:** Stop process, if map converge or epoch completes.

The learning parameters \( \alpha(t) \) and \( \sigma(t) \) decrease monotonically using the following equation:

\[
\alpha(t) = \alpha(0) \cdot \left( \frac{\alpha(T)}{\alpha(0)} \right)^{\frac{1}{T}},
\]

\[
\sigma(t) = \sigma(0) \cdot \left( \frac{\sigma(T)}{\sigma(0)} \right)^{\frac{1}{T}},
\]

where \( T \) is the training length.

## 3. CONSTANT LEARNING RATE SOM (CLRSOM)

In order to avoid the problem of choosing a precise consistently decreasing learning function, CLRSOM has been proposed. The CLRSOM does away with this function and makes the neurons learn by keeping a constant learning rate, helping to better preserve the topology of the map with less quantization error and more neuron utilization in comparison to SOM and RPSOM. CLRSOM found the nearest and farthest neurons in the 1-neighborhood of the winner neuron and a rank is assigned to each neuron according to distance from the input.

### 3.1 CLRSOM Learning Algorithm

The procedure followed in CLRSOM is as follows:

**Step 1:** The weight vectors and winning frequency \( \eta_i = 0 \) of the neurons are initialized. Also the connection value \( C_{ij} = 0 \) between each neuron is also initialized.

**Step 2:** Initialize epoch variable \( epo = 1 \) and step variable \( t = 1 \).

**Step 3:** While stop condition does not meet
(i) An input vector \( x(t) \) is randomly selected.
(ii) Find the BMU \( c \) using Eq. (1). Then, the distance between input \( x(t) \) and weight vector is found and the rank \( rank_i \) is assigned to each neuron. The rank \( rank_i \) is taken to be 0 for the BMU, because of being nearest to the input vector. The winning frequency of the winner neuron is increased by 1.
(iii) The farthest neuron and the nearest neuron are found out from among the 1-neighborhood of BMU using the following equation:

\[
\text{farthest} = \arg \max_i ||w_i(t) - x(t)|| , \quad (6)
\]
\[
\text{nearest} = \arg \min_i ||w_i(t) - x(t)|| , \quad (7)
\]
where \( i \in N_c \), \( N_c \) is the set of 1-neighbors of BMU. Also find the set \( S_f \) neurons, which are located beyond to the farthest neuron as shown in Fig. 1. Also increase the connection value between BMU and farthest neuron by 1.
(iv) Except for the nearest neuron, the weight vectors of the winner neuron and its neighbors are updated using Eq. (2). The function \( h_c(t) \) is the neighborhood function and described as follows:

\[
h_{c}(t) = \alpha \cdot (1 - \lambda_t) \cdot \exp \left( -\frac{\gamma_{(c,t)}}{2 \sigma(t)} \right), \quad (8)
\]
\[
\gamma_{(c,t)} = rank_i + (||r_i - r_c||^2 + C_{(i,c)}). \quad (9)
\]
The relative winning frequency [27] of the neuron \( i \) is calculated using \( \lambda_i = \eta_i \Sigma_{j=1}^{M} \eta_j \). The learning parameter \( \sigma(t) \) decrease monotonically using Eq. (5).
(v) The weight vector of the nearest neuron is updated using the following equation:

\[
w_q(t + 1) = w_q(t) + h_{cq}(t) \cdot [x(t) - w_q(t)], \quad (10)
\]
\[
h_{cq}(t) = \alpha \cdot (1 - \lambda_t) \cdot \exp \left( -\frac{\delta_{(c,q)}}{2 \sigma(t)} \right), \quad (11)
\]
\[
\delta_{(c,q)} = ||w_q(t) - x(t)||^2 + C_{(c,q)}/m. \quad (12)
\]
The learning parameter \( \sigma(t) \) decrease monotonically using Eq. (5).
(vi) \( t = t + 1 \)
(vii) \( epo = epo + 1 \)

**Step 4:** Stop process, if map converge or epoch completes.

**4. EXPERIMENTAL RESULTS**

The CLRSOM is compared with the SOM and RPSOM. For this, one synthetic and two real datasets have been used from UCI machine learning repository.
4.1 Quality Criteria

The following three well known measurements are taken to compare the learning performance of CLRSOM with the conventional SOM and RPSOM. These parameters are widely used to check the performance of machine learning algorithms [27, 28].

(i) Quantization Error (QE): It calculates the average distance from each input data to its winner.
(ii) Topographic Error (TE) [15]: It describes how well the SOM preserves the topology of the input dataset.
(iii) Neuron Utilization (U) [13]: It calculates the winner neurons percentage, which are the winner for some input data,

4.2 Experimental Result on Synthetic Dataset

The CLRSOM is applied to the Target synthetic dataset which is taken from the UCI repository [29] for machine learning to determine the performance of learning.

(a) Target Dataset

We carry out the learning experiment on Target dataset. This dataset has 770 points of 2-D and has an outliers clustering problem [29]. We do the experiment using different parameters as shown in Table 1. It is not possible to visualize all the simulation results. The simulation results for map size 13*13, alpha (0) = 0.7 and epoch = 25 of both algorithms using random initialization is shown in Fig. 2. As can be observed from Fig. 2, the neurons reach to the outlier’s input data in CLRSOM, while in others it is not the case. Also the CLRSOM covers the input data more effectively in comparison to the conventional SOM.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Algorithm</th>
<th>Efficiency Parameter</th>
<th>Parameter</th>
<th>QE</th>
<th>TE</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map Size = 5*5, α = 0.5, Epoch = 15</td>
<td>SOM</td>
<td>0.2558</td>
<td>0.1753</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.2319</td>
<td>0.2863</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>0.2064</td>
<td>0.2662</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map Size = 10*10, α = 0.6, Epoch = 20</td>
<td>SOM</td>
<td>0.0967</td>
<td>0.1506</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.0913</td>
<td>0.2062</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>0.0892</td>
<td>0.0584</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map Size = 13*13, α = 0.7, Epoch = 25</td>
<td>SOM</td>
<td>0.0700</td>
<td>0.1403</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.0697</td>
<td>0.1948</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>0.0638</td>
<td>0.0701</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map Size = 17*17, α = 0.7, Epoch = 35</td>
<td>SOM</td>
<td>0.0474</td>
<td>0.1519</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.0469</td>
<td>0.2031</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>0.0456</td>
<td>0.0442</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map Size = 25*25, α = 0.6, Epoch = 45</td>
<td>SOM</td>
<td>0.0266</td>
<td>0.0961</td>
<td>0.9984</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.0262</td>
<td>0.1274</td>
<td>0.9984</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>0.0252</td>
<td>0.0506</td>
<td>0.9996</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
From this figure, it can be concluded that CLRSOM obtains a more effective map organized in every corner of the data as compared to the conventional SOM. The topology of data is better preserved by the CLRSOM even with a constant learning rate, with less quantization error and more neuron utilization as compared to the conventional SOM and RPSOM, as shown in Table 1. It is observed from Table 1 that there is an improvement in topographic and quantization error respectively. Also our approach leads to more neurons becoming the winner. So our approach leads to better learning and better convergence of the map in comparison to the conventional SOM and RPSOM.

4.3 Experimental Results on Real Dataset

We do experiments on two real datasets from UCI machine learning repository [29]. These datasets are widely accepted to compare the performance of learning algorithms.
(a) Iris Dataset

Fisher’s Iris dataset [29], one of the known benchmark dataset for pattern recognition, made of 150 4-D points consists of three clusters each of them has 50 points. The experiment is done using different parameters as shown in Table 2. From these results, it can be concluded that the neurons in CLRSOM learn input data distribution more precisely and efficiently. The maximum number of neurons learns the input distribution. The small value of topographic error indicates that the topology of input data is better preserved. The topographic and quantization errors of our approach, even after using a constant learning rate, are smaller than SOM and RPSOM as shown in Table 2.

### Table 2. Experimental results: Iris dataset.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Algorithm</th>
<th>Efficiency Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris Dataset; No. of Patterns =150; Dimension = 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>QE</td>
<td>TE</td>
<td>U</td>
</tr>
<tr>
<td>Map Size = 10*10, (\alpha = 0.4), Epoch = 15</td>
<td>SOM</td>
<td>0.2168 0.2133 0.96</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.2015 0.2957 0.96</td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>0.1869 0.0933 0.97</td>
</tr>
<tr>
<td>Map Size = 13*13, (\alpha = 0.7), Epoch = 30</td>
<td>SOM</td>
<td>0.1451 0.2267 0.9527</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.1287 0.2810 0.9527</td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>0.1120 0.0933 0.9527</td>
</tr>
<tr>
<td>Map Size = 17*17, (\alpha = 0.8), Epoch = 40</td>
<td>SOM</td>
<td>0.0939 0.1400 0.8858</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.0813 0.1887 0.8858</td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>0.0587 0.0533 0.9031</td>
</tr>
<tr>
<td>Map Size = 25*25, (\alpha = 0.4), Epoch = 50</td>
<td>SOM</td>
<td>0.0757 0.1331 0.6640</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.0567 0.0600 0.6736</td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>0.0459 0.0267 0.2780</td>
</tr>
<tr>
<td>Map Size = 40*40, (\alpha = 0.4), Epoch = 50</td>
<td>SOM</td>
<td>0.0548 0.0786 0.5742</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>0.0549 0.0267 0.2780</td>
</tr>
</tbody>
</table>

### Table 3. Experimental results: Wine dataset.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Algorithm</th>
<th>Efficiency Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wine Dataset; No. of Patterns =178; Dimension = 13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>QE</td>
<td>TE</td>
<td>U</td>
</tr>
<tr>
<td>Map Size = 10*10, (\alpha = 0.4), Epoch = 15</td>
<td>SOM</td>
<td>11.139 0.6629 1.0</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>10.547 0.7753 1.0</td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>8.8252 0.6326 1.0</td>
</tr>
<tr>
<td>Map Size = 13*13, (\alpha = 0.7), Epoch = 30</td>
<td>SOM</td>
<td>7.2953 0.6180 0.994</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>6.2313 0.6704 1.0</td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>4.3872 0.5708 0.9822</td>
</tr>
<tr>
<td>Map Size = 17*17, (\alpha = 0.8), Epoch = 40</td>
<td>SOM</td>
<td>4.9492 0.6517 0.958</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>3.8792 0.7254 0.958</td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>2.6032 0.6101 0.958</td>
</tr>
<tr>
<td>Map Size = 25*25, (\alpha = 0.4), Epoch = 50</td>
<td>SOM</td>
<td>4.0297 0.6348 0.824</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>3.0171 0.6973 0.8328</td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>2.0691 0.5652 0.8328</td>
</tr>
<tr>
<td>Map Size = 30*25, (\alpha = 0.6), Epoch = 30</td>
<td>SOM</td>
<td>3.6143 0.6371 0.6631</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>2.2878 0.4978 0.6097</td>
</tr>
<tr>
<td></td>
<td>CLRSOM</td>
<td>4.1440 0.5674 0.836</td>
</tr>
<tr>
<td>Map Size = 25*20, (\alpha = 0.7), Epoch = 35</td>
<td>SOM</td>
<td>4.3971 0.6286 0.857</td>
</tr>
<tr>
<td></td>
<td>RPSOM</td>
<td>3.8805 0.4157 0.880</td>
</tr>
</tbody>
</table>
(b) Wine Dataset

The wine dataset consists of 178 points of 13-D that are in three clusters [29]. The numbers of data points in each cluster are 59, 71 and 48. The wine dataset is applied using different map size, learning rate and epoch as shown in Table 3. The obtained results demonstrate how the learning capabilities are significantly improved. There is an improvement in quantization and topographic error. The CLRSOM learns better with much lower quantization and topographic error in comparison to the SOM and RPSOM even when the learning rate is constant. The neuron utilization is also seen to improve.

5. CONCLUSION

The proposed CLRSOM learning algorithm does away with the consistently decreasing learning rate function. The decreasing learning rate function is quite a nontrivial requirement regarding the performance of the conventional SOM model. Both the nearest and farthest neurons are found from among the 1-neighborhood of the winner neuron on the map and a constant learning rate is used. For the experiments, SOM, RPSOM and CLRSOM are applied on many standard input data sets and the learning performance is evaluated using error, topographic and quantization errors and neuron utilization.

The experimental results have shown that in the CLRSOM, more neurons reach to the outlier’s data and a more effective map is achieved in comparison to conventional SOM and RPSOM. A better learning is achieved along with much lower quantization and topographic errors in comparison to SOM and RPSOM. Also more neurons become winner in CLRSOM, and so the neuron utilization is also improved in comparison to SOM and RPSOM.

REFERENCES


Vikas Chaudhary had completed his M.C.A. from J.N.V. University, Jodhpur and M.Tech. in Computer Engineering from Maharshi Dayanand University, Rohtak. He is pursuing Ph.D. from National Institute of Technology, Kurukshetra. His research area is artificial neural networks.

R. S. Bhatia had completed his B.Tech. from Guru Nanak Engineering College, Ludhiana in 1987 and M.Tech., Ph.D. from NIT, Kurukshetra in 1993 and 2008 respectively. Presently, he is a Professor in NIT, Kurukshetra. His present research interests include adhoc networks, artificial neural networks, FACTS and software engineering.

Anil K. Ahlawat obtained his M.Phil from IIT Roorkee, M.Tech in Computer Science and Engineering from Kurukshetra University and Ph.D. from Guru Gobind Singh Indraprastha University, New Delhi. Presently, he is a Professor in KIET, Ghaziabad. He is in the editorial and reviewer board of several international journals. Currently he is guiding 5 Ph.D. students. His research interests include artificial neural networks, artificial intelligence, algorithm design and modeling and simulation.