International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:2, No:4, 2008

Levenberg-Marquardt Algorithm for Karachi Stock Exchange Share Rates Forecasting

Syed Muhammad Aqil Burney, Tahseen Ahmed Jilani, Cemal Ardil

***Abstract*—** Financial forecasting is an example of signal processing problems. A number of ways to train/learn the network are available. We have used Levenberg-Marquardt algorithm for error back-propagation for weight adjustment. Pre-processing of data has reduced much of the variation at large scale to small scale, reducing the variation of training data.

***Keywords*—** Gradient descent method, jacobian matrix. Levenberg-Marquardt algorithm, quadratic error surfaces,

I. INTRODUCTION

Forecasting is a high noisy application because we are typically dealing with systems that receive thousands of inputs, which interact in a complex nonlinear fashion. The forecasting of future events from noisy time series data is commonly done using various forms of statistical models [10]. Usually with a very small subset of inputs or measurements available from the system, the system behavior is to be estimated and extrapolated into the future. Typically neural network models are closely related to statistical models [8], where the inputs are weighted and scalar outputs are processed to produce output, thus provide a method of functional mapping. Learning by example often operates in a vector space i.e. a functional mapping R^n *to* R^m is approximation [20]. Where for the positive integer n, R^n is the set of all ordered n-tuples of the form $\{x_1, x_2, \ldots, x_n\}$.

II. INTRODUCTION TO NEURAL NETWORKS

An Artificial Neural Network (ANN) is an information-processing paradigm that is inspired by the biological nervous systems, such as brain, process information [1]. The key element of this paradigm is the novel structure of the information processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as forecasting and prediction or reconstruction /recognition through learning process that involves adjustments to the synaptic connections that exist between neurons. Introductory ANN concepts are made in [3], [5] and [11]. A network is composed of a number of interconnected units; each unit has input/output (I/O) characteristics and implements a local computing. The output of any unit is determined by its I/O characteristics, it's interconnection to other units and external inputs.

Most of the present work in ANN concerns the development, characterization and extension of mathematical neural network models.

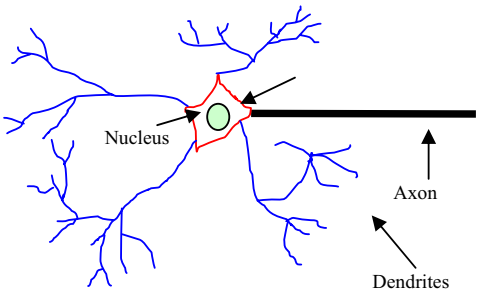


Fig. 1. Components of a neuron

A mathematical neural network model refers to the set of nonlinear n-dimensional (generally) equations that characterize the overall network operations, as well as structure and dynamics of the ANN (and training).

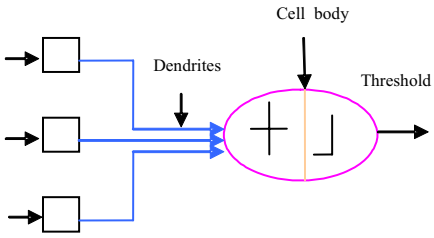


Fig. 2. An Artificial Neuron

III. FEEDFORWARD NEURAL NETWORK ARCHITECTURE AND MAPPING

Feed forward neural networks are composed of neurons in which the input layer of neurons is connected to the output layer through one or more layers of intermediate neurons, The

Manuscrot received on December, 28. 2003
Dr. S. M. Aqil Burney is Professor in the Department of Computer Science,
University of Karachi, Pakistan, Phone: 0092-21-9243131 ext. 2447, fax: 0092-21-9243203, Burney@computer.Org, aqil_burney@yahoo.com.
Tahseen Ahmed Jilani is lecturer in the Department of Computer Science, university of Karachi and is Ph.D. research associate in the Department of Statistics. University of Karachi, Pakistan. tahseenjilani@yahoo.com
Cemal Ardil is with the Azerbaijan National Academy of Aviation Baku, Azerbaijan. cemalardil@gmail.com

training process of neural networks involves adjusting the weights till a desired input/output relationship is obtained [7],[15]. The majority of adaptation learning algorithms are based on the Widrow-Hoffback-algorithm. The mathematical characterization of a multilayer feedforward network is that of a composite application of functions [21]. Each of these functions represents a particular layer and may be specific to individual units in the layer, e.g. all the units in the layer are required to have same activation function. The overall mapping

is thus characterized by a composite function relating feedforward network inputs to output [12]. That is

$$\underline{O} = \underline{f}_{composite}\left(\underline{i}\right)$$

Using p-mapping layers in a p+1 layer feedforward net yield

$$O = f^{L_p}\left(f^{L_{p-1}}....\left(f^{L_1}\left(\underline{i}\right).....\right)\right)$$

Thus the interconnection weights from unit k in L_1 to unit i in L_2 are $w^{L_1 \to L_2}$. If hidden units have a sigmoidal activation function, denoted f^{sig}

$$O_i^{L_2} = \sum_{k=1}^{H_1} w_{ik}^{L_1 \to L_2} \left\{ f_k^{sig}\left(\sum_{j=1}^{I} w_{kj}^{L_0 \to L_1} i_j \right) \right\} \quad (1)$$

Above equation illustrates neural network with supervision and composition of non-linear function.

The neural network forecasting can be described as follows

$$Z_{k+1} = NN\left(Z_k, Z_{k-1},....,Z_{k-d}, e_k, e_{k-1},...,e_{k-d}\right)$$

Where Z_i is either original observations or pre-processed data, and e_i are residuals. The processing of input data and the number of e's and Z's are needed to determine the performance of the forecast [9].

IV. BACKPROPAGATION ALGORITHM

Backpropagation was created by generalizing the Widrow-Hoff learning rule to multiple-layer networks and non-linear differentiable transfer functions. Input vectors and the corresponding target vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined by analyst [1]. Networks with biases, a sigmoid function, and a linear output layer capable of approximating any function with a finite number of discontinuities, see [13] and [23]. Standard backpropagation is a gradient descent algorithm, in which the network weights are moved along the negative of the gradient of the performance function. Thus for each movement we have,

$$X_{k+1} = X_k - a_k . g_k \quad (2)$$

There are a number of variations on the basic algorithm that are based on other standard optimization techniques, such as conjugate gradient [6]. Properly trained backpropagation networks tend to give reasonable answers when presented with inputs that they have never seen. Typically, a new input leads to an output similar to the correct output for input vectors used in training that are similar to the inputs being presented [1].

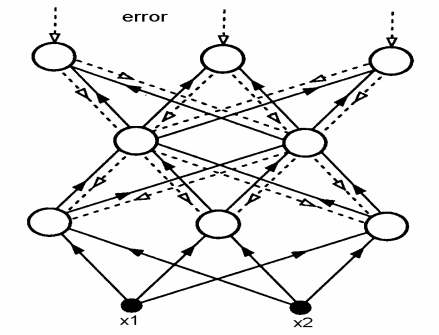


Fig. 3. Error backpropagation method for weight adjustment (A recursive operation)

This generalization property makes it possible to train a network on a representative set of inputs/targets pairs and get good results without training the network on all possible input/output pairs. Instances are represented by many attribute value pair. The target function to be learned is defined over instances that can be described by a vector of predefined features. These input attributes may be highly correlated or independent of one another. Input values can be any real values. The target function output may be discrete-valued, real-valued or a vector of several real or discrete–valued attributes [7] and [16].

The training examples may contain errors. ANN learning methods are quite robust to noise in the training data. Long training times are acceptable. Network training algorithms typically require longer training times then, say decision tree algorithm. Training times can range from a few seconds to many hours, depending on the factors such as the number of weights in the network, the number of training examples considered and the setting of various learning algorithm parameters [20]. Fast evaluation of the leaning target function may be required. Although ANN learning times are relatively long, evaluating the learned network, in to apply it to a subsequent instance is typically very fast [18], [23].

V. GRADIENT DESCENT TRAINING USING SIGMOIDAL AND LINEAR ACTIVATION FUNCTION

For networks having differentiable activation functions, there exist a powerful and computationally efficient method, called error backpropagation for fining the derivatives of an error function with respect to the weights and biases in the network. Gradient Descent algorithm is the most commonly used error backpropagation method. See for more in detailed discussion [12], [13] and [22]. If net is the network designed by taking the scalar product of the connections (weights) and the stimulus then we define, $net = \underline{w}^T \underline{i}$, is large and have a

differentiable activation function. Given a training set of input-target pairs of the form

$$H = \left\{ \left(\underline{i}^1, t^1 \right), \left(\underline{i}^2, t^2 \right), ,.. \left(\underline{i}^n, t^n \right) \right\}$$

Consider the sigmoidal activation function with outputs as

$$O_i = f\left(net_i\right) = \frac{1}{1+e^{-net_i}}$$

now, defining a single pair error measure based on $\left(t^p - O^p\right)$, where O^p is the output obtained from $\underline{i}^p$, we have

$$\left(e^p\right)^2 = \frac{1}{2}\left(t^p - O^p\right)^2 \tag{3}$$

The basic idea is to compute $\frac{d\left(e^p\right)^2}{d\,\underline{w}}$, we use this quantity to adjust $\underline{w}$. Let for the k^{th} element of $\underline{w}$, i.e., we have

$$\frac{d\left(e^p\right)^2}{dw_k} = \frac{d\left(e^p\right)^2}{dO^p}.\frac{dO^p}{d\,net}.\frac{d\,net}{d\,w_k}$$

$$= \left(O^p - t^p\right)\left[\frac{d\,f\left(net^p\right)}{d\,net^p}\right].i_k^p \tag{4}$$

which is the gradient of the pattern error with respect to weight w_k and forms the basis of the gradient descent training algorithm.

Specifically, we assign the weight correction, Δw_k such that

$$\Delta w_k = -\alpha \frac{d\left(e\right)^2}{d\,w_k}$$

yielding the correction:

$$w_k^{j+1} = w_k^j - \alpha\left[\left(O^p - t^p\right).\frac{d\,f\left(net^p\right)}{d\,net^p}.i_k^p\right] \tag{5}$$

and for linear activation function $\left(O = net\right)$

$$w_k^{j+1} = w_k^j - \alpha\left[.\frac{\left(O^p - t^p\right)}{-2e^p}.i_k^p\right]$$

$$w_k^{j+1} = w_k^j + \alpha' e^p\, i_k^p \tag{6}$$

VI. LEVENBERG-MARQUARDT ALGORITHM

Training neural network is essentially a non-linear least squares problem, and thus can be solved by a class of non-linear least squares algorithms. Among them, the Levenberg-Marquardt is a trust region based method with hyper-spherical trust region. This method work extremely well in practice, and is considered the most efficient algorithm for training median sized artificial neural networks.

Like Quasi-Newton methods, the Levenberg-Marquardt algorithm was designed to approach second order training speed with out having to compute Hessian matrix. When the performance function has the form of a sum of squares that is

$$F(\underline{\mathbf{w}}) = \frac{1}{2}e^T e = \frac{1}{2}e^T(w)\,e(w) = \frac{1}{2}\sum_i^k\sum_j^p\left(O_{ij} - t_{ij}\right)^2$$

where $\underline{W} = \left[w_1, w_2, w_3, ..., w_N\right]^t$ consists of all weights of the network, the function of sum of squared errors is defined as

$$F(\mathbf{W}) = \frac{1}{2}e^t\, e$$

Newton's method for minimizing objective function is generated using well known recurrence formula.

$$W_{i+1} = W_i - \mathbf{H}^{-1}\nabla F(\mathbf{W}) \tag{7}$$

When $F(\mathbf{W}) = \frac{1}{2}e^T e$ and $\nabla F(\mathbf{W})$ is the gradient of $F(\mathbf{W})$ then the Hessian matrix can be approximated as,

$$\mathbf{H} \approx \mathbf{J}^T\mathbf{J}$$

and the gradient can be computed as

$$g = \mathbf{J}^T\,\underline{e}$$

where the Jacobian matrix $\mathbf{J}$ contains the first derivatives of the network errors with respect to weights and biases, and $\underline{e}$ is a vector of network errors. The Gauss-Newton update formula can be

$$W_{i+1} = W_i - \left(\mathbf{J}_i^T\mathbf{J}_i\right)^{-1}\mathbf{J}_i^T\,\varepsilon_i \tag{8}$$

$$\mathbf{J} = \begin{bmatrix} \frac{\partial e_{11}}{\partial w_1} & \frac{\partial e_{11}}{\partial w_2} & . & . & \frac{\partial e_{11}}{\partial w_N} \\ \frac{\partial e_{21}}{\partial w_1} & \frac{\partial e_{21}}{\partial w_2} & . & . & \frac{\partial e_{21}}{\partial w_N} \\ . & . & . & . & . \\ . & . & . & . & . \\ \frac{\partial e_{k1}}{\partial w_1} & \frac{\partial e_{k1}}{\partial w_2} & . & . & \frac{\partial e_{.k1}}{\partial w_N} \\ . & . & . & . & . \\ . & . & . & . & . \\ \frac{\partial e_{1p}}{\partial w_1} & \frac{\partial e_{1p}}{\partial w_2} & . & . & \frac{\partial e_{1p}}{\partial w_N} \\ \frac{\partial e_{2p}}{\partial w_1} & \frac{\partial e_{2p}}{\partial w_2} & . & . & \frac{\partial e_{2p}}{\partial w_N} \\ \frac{\partial e_{kp}}{\partial w_1} & \frac{\partial e_{kp}}{\partial w_2} & . & . & \frac{\partial e_{kp}}{\partial w_N} \end{bmatrix}.$$

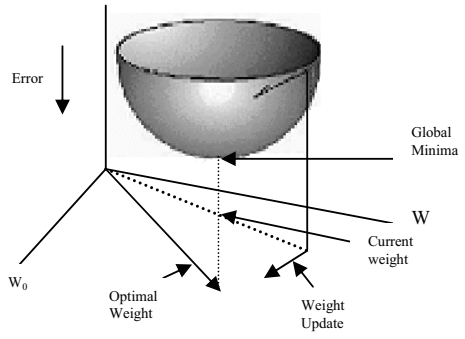
Where $\left(\mathbf{J}^T\mathbf{J}\right)$ is positive definite, but if it is not, then, we make some perturbation into it that will control the probability of being non positive definite.

Thus,

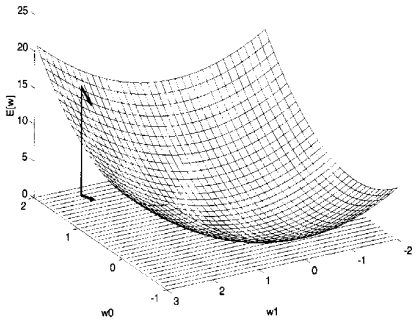
$$\mathbf{H} \approx \mathbf{J}^T\mathbf{J} + \mu\mathbf{I}$$

$$\mathbf{W}_{i+1} = \mathbf{W}_i - \left(\mathbf{J}_i^T\mathbf{J}_i + \lambda\mathbf{I}\right)^{-1}\mathbf{J}_i^T\,\varepsilon_i \tag{9}$$

Where in neural computing context the quantity λ is called the learning parameter, it ensures that $\mathbf{J}^T\mathbf{J}$ is positive definite. It is always possible to choose λ sufficiently large enough to ensure a descent step. The learning parameter is decreased as the iterative process approaches to a minimum. Thus, in Levenberg-Marquardt optimization, inversion of square matrix $\mathbf{J}^T\mathbf{J}+\lambda\mathbf{I}$ is involved which requires large memory space to store the Jacobian matrix and the approximated



(a)



(b)

Fig. 4. (a) Typical error function for linear networks, (b) Gradient descent used to search the space of possible weight vectors

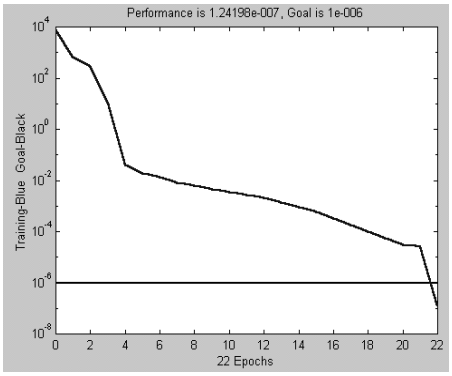
Hessian matrix along with inversion of approximate H matrix of order $N\times N$ in each iteration, thus for large networks, Levenberg-Marquardt algorithm is not suitable.

VII. KARACHI STOCK EXCHANGE SHARES

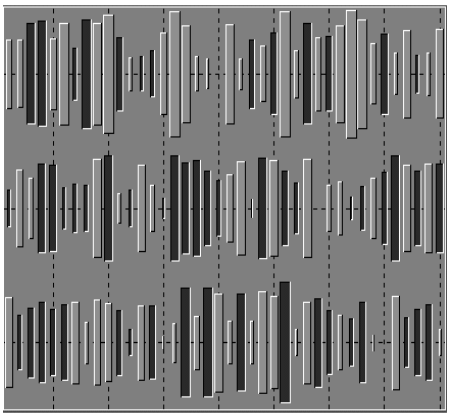
Stock exchange share prices exhibit very high noise, and significant non-stationarity. Here, we consider the prediction of the direction of change in the share price for the next business day. A number of researchers have applied neural network technology to exchange rates prediction and trading. This is for the very first time that ANN technique of forecasting is applied at K.S.E. The data set used has been taken from Karachi Stock Exchange Library and KSE brokers. We have taken 400 data points for training of neural networks. For prior work on Karachi Stock Exchange see [2].

VIII. NETWORK TRAINING AND RESULTS

We presented input data in concurrent patterns, this computation can be performed using [12] or [17]. For programming in C/C++ of ANN we also use the approach of [3] or [25]. The network performance value along with each epoch is presented in the figure-6 (a). We set the network performance measurement goal at 1E-06. Secondly in Figure-6 (b): the Hinton plot [10] or [12] of input layer weight is presented. The white boxes shows positive and grey are negative weights.



(a)



(b)

Fig.5. (a) Plot of Network Training, Performance function with epoch (b) Input Weights Hinton Plot.

We trained the network many times and the performance goal is achieved at different epochs. The gradient of error surface is also shown in table 1. BP nets initialized with small weights can develop large weights only after the number of updates is large. We have used only 100 iterations for training of network. Early stopping combined with BP is so effective that large nets can be trained where nets with large weights have more representation power [20]. For further discussion please concern [7], [15], [18] and [24]. Finally, the results of forecasting for 10 days are shown below. It is best to use the forecast of only next day because of high variability at Karachi Stock Exchange.

TABLE 1
NETWORK PERFORMANCE GOAL= E-06, ALONG WITH GRADIENT OF ERROR FUNCTION 1E-12.

Training	Goal achieved at epoch	M.S.E	Gradient
1	50	3.80E-10	0.03409
2	10	7.67E-07	2.01321
3	12	3.08E-08	0.767221
4	50	3.77E-10	0.110359
5	17	3.64E-08	0.110359
6	34	1.51E-10	0.034858
7	50	1.18E-09	0.038043
8	33	5.08E-09	0.807517
9	50	3.80E-10	0.03409
10	39	1.29E-16	5.93E-06

TABLE 2
A COMPARISON BETWEEN TARGETS AND NETWORK OUTPUTS

Date	Targets	ANN Output	Difference
7/16/2001	17.8	17.7991	0.0009
7/17/2001	17.1	17.0992	0.0008
7/18/2001	17.15	17.1492	0.0008
7/19/2001	17.15	17.1491	0.0009
7/20/2001	16.9	16.8992	0.0008
7/23/2001	16.9	16.8996	0.0004
7/24/2001	17.6	17.5992	0.0008
7/25/2001	16.95	16.9492	0.0008
7/26/2001	16.45	16.4492	0.0008
7/27/2001	16.45	16.4492	0.0008

IX. CONCLUSIONS

We have seen that neural networks produced fairly accurate forecast if compared with weighted exponential methods and ARIMA methods. Generalization performance is reasonable, indicating confidence in the training results and produced fairly accurate forecasts [1].

There are a number of available training algorithms that can be used with backpropagation method, but Levenberg-Marquardt has proven results specifically to a sum-of-squares error function for financial forecasts of share rate forecasting. The comparison ability of the different approaches is usually problem dependent.

It would be interesting to find out how would other architecture such as Regression neural networks, Bayesian neural networks and SOFM will behave. Some new ways of preprocessing and input selection methods will be our future work. We hope to extend our work to recurrent neural networks, Bayesian neural networks and iterative principle component neural networks.

REFERENCES

[1] S. M. Aqil Burney, Tahseen A. Jilani,. (2002). "Time Series forecasting using artificial neural network methods for Karachi Stock Exchange". A Project at Department of Computer Science, University of Karachi.
[2] S. M. Aqil Burney, “Measures of central tendency in News-MCB stock exchange”. Price index pp 12-31. Developed by Vital information services (pvt) limited, Karachi. c/o Department of Computer Science, University of Karachi.
[3] Adam Blum, (1992)."Neural Network in C++". John Wiley and Sons, New York.
[4] C. Brooks, (1997). "Linear and non-linear (non-) forecastability of high-frequency exchange rates". Journal of Forecasting, pp. 16:125-145.
[5] M. Caudill, and C. Butler. (1994)."Understanding neural network". MIT Press.
[6] Y. Chauvin, and D. Rumelhart, (1995). "Backpropagation: theory, architecture, and applications." Lawrence Erlbaum Association.
[7] Chung-Ming Kuan and Liu. Tung, (1995)."Forecasting exchange rates using feedforward and recurrent neural networks." Journal of Applied Econometrics, pp. 347-64.
[8] C. Lee Giles, Steve Lawrence and A. C. Tsoi, .(2001)."Time series prediction using a recurrent neural network and grammatical inference". Machine Learning, Vol. 44, Number ½ , July/August, pp. 161-183.
[9] C. Lee Giles, Steve Lawrence, A. C. Tsoi.(1997). "Rule inference for financial prediction using recurrent neural networks". Proceedings of IEEE/IAFE Conference on Computational Intelligence for Financial Engineering (CIFEr), IEEE, Piscataway, NJ, , pp. 253-259.
[10] C. M. Bishop.(1995). "Neural networks for pattern recognition". Clarendon Press. Oxford
[11] R. Dixon, and F. Bevan. (1994)."Neural network tutor." Advance Technology Transfer Group.
[12] T. M. Hagan, H. B. Demuth, (1996)."Neural network design", Brooks/Cole publishing company.
[13] R. Hecht-Nielsen, (1989). "Theory of the backpropagation neural network". International Joint Conference on Neural Networks, IEEE, New York, pp. 593-605.
[14] K. Hornik., M. Sunchcombe, and H. White. (1990). "Using multilayer feedforward networks for universal approximation". Neural Networks 3, pp. 551-60.
[15] Iebeling Kaastra and Milton S. Boyd. (1995). "Forecasting future trading volume using neural networks." Journal of Future Markets, pp. 953-70.
[16] C. M. Kuan, and T. Liu. (1995). "Forecasting exchange rates using feedforward and recurrent neural networks". Journal of Applied Econometrics, pp.10: 347-364.
[17] Lars Liden, (2000) "Neural network simulation package". Department of Cognitive and Natural Systems at Boston University
[18] Mir F. Atiya, Suzan M. El-Shoura, Samir I. Shaheen. "A comparison between neural network forecasting techniques- case study: river flow forecasting". IEEE Transactions on Neural Networks. Vol. 10, No. 2 March 1999
[19] Moody, J.(1994). "Prediction risk and architecture selection for neural networks". appears in, From Statistics to Neural Networks: Theory and Pattern Recognition Application, NATO ASI Series, Springer-Verlag.
[20] Rich caruana, Steven Lawrence, C. Lee Giles. (2000)."Overfitting in neural nets: backpropagation, conjugate gradient, and early stopping". Neural Information Processing Systems, Denver, Colorado, November pp. 28-30.
[21] B. D. Ripley, (1994)”Neural networks and related methods for classification”, Journal of the Royal Statistician Society, B 56(3),409-456.
[22] R. J. Schalkoff, (1997)."Artificial neural networks". McGraw-Hill International Edition, computer science series,
[23] N. Schraudolph, Nicol, and T. J. Sejnowski, (1996)."Tempering backpropagation networks: not all weights are created equal". Appear in Advances in Neural Information Processing Systems 8, MIT Press, Cambridge.
[24] W. Chu Chao-Hsien and Djohan Widjaja, (1994)."Neural network system for forecasting method selection". Decision support systems (August), pp. 13-24.
[25] T. Stephen Welstead, (1994). "Neural network and fuzzy logic applications in C/C++". John Wiley and Sons, Inc. N.Y.

S. M. Aqil Burney received the B.Sc.(Physics, Mathematics, Statistics) from D. J. College affiliated with the University of Karachi in 1970; First class first M.Sc. (Statistics,) from Karachi University in 1972 with M.Phil. in 1983 with specialization in Computing, Simulation and stochastic modeling in from risk management. Dr. Burney received Ph.D. degree in Mathematics from Strathclyde University, Glasgow with specialization in estimation, modeling and simulation of multivariate Time series models using algorithmic approach with software development.

He is currently professor and approved supervisor in Computer Science and Statistics by the High education Commission Government of Pakistan. He is also the project director of Umair Basha Institute of Information technology (UBIT). He is also member of various higher academic boards of different universities of Pakistan. His research interest includes AI, Soft computing neural networks, fuzzy logic, data mining, statistics, simulation and stochastic modeling of mobile communication system and networks and network security. He is author of three books, various technical reports and supervised more than 100software/Information technology projects of Masters level degree programs and project director of various projects funded by Government of Pakistan.

He is member of IEEE (USA), ACM (USA) and fellow of Royal Statistical Society United Kingdom and also a member of Islamic society of Statistical Sciences. He is teaching since 1973 in various universities in the field of Econometric, Bio-Statistics, Statistics, Mathematic and Computer Science He has vast education management experience at the University level. Dr. Burney has received appreciations and awards for his research and as educationist.

Tahseen A. Jilani received the B.Sc.(Computer Science, Mathematics, Statistics) from University of Karachi in 1998; First class second M.Sc. (Statistics,) from Karachi University in 2001. Since 2003, he is Ph.D. research fellow in the Department of Computer Science, University of Karachi.

His research interest includes AI, neural networks, soft computing, fuzzy logic, Statistical data mining and simulation. He is teaching since 2002 in the fields of Statistics, Mathematics and Computer Science.