Nonlinear Fading Channel Equalization of BPSK Signals Using Multiplicative Neuron Model

Kavita Burse, R. N. Yadav and S. C. Shrivastava

***Abstract*—A high order feed forward neural network architecture with optimum number of nodes is used for adaptive channel equalization in this paper.The replacement of summation at each node by multiplication results in more powerful mapping because of its capability of processing higher-order information from training data. The equalizer is tested on Rayleigh fading channel with BPSK signals. Performance comparison with recurrent radial basis function (RRBF) neural network show that the proposed equalizer provides compact architecture and satisfactory results in terms of bit error rate performance at various levels of signal to noise ratios for a Rayleigh fading channel.**

***Index Terms*—channel equalization, BPSK signal, multiplicative neuron, Rayleigh channel.**

I. INTRODUCTION

As higher-level modulation becomes more desirable to cope with the need for high-speed data transmission, nonlinear distortion becomes a major factor, which limits the data carrying capacity of digital communication sytems. Thermal noise, impulse noise, cross talk and the nature of the channel itself distort the transmitted data in amplitude and phase due to which temporal spreading and consequent overlap of individual pulses occurs. The presence of inter symbol interference (ISI) in the system introduces errors in the decision device at the receiver output. Therefore, in the design of the transmitting and receiving filters, the objective is to minimize the effects of ISI, and thereby deliver the digital data to its destination with the smallest error possible. Equalizers modelled as adaptive digital filters which shape the receiver's transfer function are ubiquitous in todays signal processing applications to combat ISI in dispersive channels. Adaptive filters achieve desired spectral characteristics of a signal by altering the filter coefficients and thereby the filter response according to a recursive optimization algorithm. Adaptive coefficients are required since some parameters of the desired processing operation (for instance, the properties of some noise signal) are not known in advance [1].

When significant noise is added to the transmitted signal linear boundaries are not optimal. The received signal at each sample instant may be considered as a nonlinear function of the past values of the transmitted symbols. Further, since the nonlinear distortion varies with time and from place to place, effectively the overall channel response becomes a nonlinear dynamic mapping and the problem is tackled using classification techniques. As shown in a wide range of engineering applications, neural network (NN) has been successfully used for modeling complex nonlinear systems and forecasting signal with relatively simple architecture [2]-[4]. A wide range of neural architectures are available for modeling the nonlinear phenomenon of channel equalization. Feed forward networks like multilayer perceptron (MLP) which contain an input layer, an output layer and one or more hidden layers possess nonlinear processing capabilities and universal approximation characteristic and have been successfully implemented as channel equalizers [5]-[7]. The back propagation which is a supervised learning algorithm is used as a training algorithm [8]. These neuron models process the neural inputs using the summing operation.

Recently, higher-order networks have drawn great attention from researchers due to their superior performance in nonlinear input-output mapping, function approximation, and memory storage capacity. Some examples are Product unit neural network (PUNN), Sigma-Pi network (SPN), Pi-Sigma network (PSN) etc. They allow neural networks to learn multiplicative interactions of arbitrary degree. Multiplication plays an important role in neural modeling of biological behavior and in computing and learning with artificial neural networks. The multiplicative neuron contains units which multiply their inputs instead of summing them and thus allow inputs to interact nonlinearly. Multiplicative node functions allow direct computing of polynomials inputs and approximate higher order functions with fewer nodes. Thus they may present better approximation capability and faster learning times than the classical MLP (which incorporate additive neurons only) because of their capability of processing higher-order information from training data [9]-[11]. The remaining of the paper is organized as follows: section II describes the basic adaptive channel equalizer scheme. In section III learning rule for multiplicative neuron is derived, section IV provides the simulation and results and section V concludes the paper.

Manuscript received may 15, 2009.

Kavita burse is a research scholar in department of electronics and communication at maulana azad national institute of technology, bhopal, india. (phone: +919893141968; fax: +91-755-2734694).

Dr. R.n. Yadav and dr. S.c. Shrivastava are with the department of electronics and communication, maulana azad national institute of technology, bhopal, india

II. ADAPTIVE CHANNEL EQUALIZATION

The block diagram of adaptive equalization in Fig. 1 is described as follows. The external time dependant inputs consist of the sum of the desired signal $d(k)$, the channel nonlinearity NL and the interfering noise $v(k)$. The adaptive filter has a finite impulse response (FIR) structure. The impulse response is equal to the filter coefficients.The coefficients for a filter of order *p* are defined as

$$w_k = [w_k(0), w_k(1),....., w_k(p)]^T \tag{1}$$

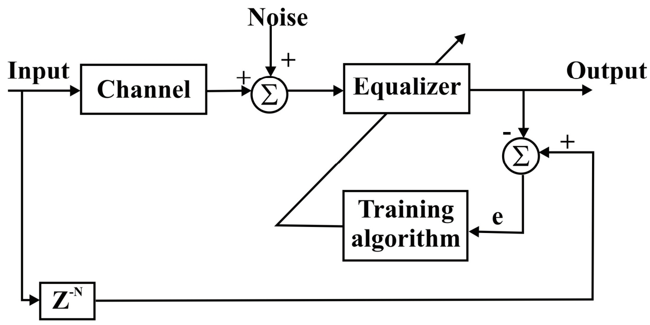

Fig. 1 Block diagram of an adaptive channel equalizer

A predefined delayed version of the original signal forms the training sequence to provide reference points for the adaptation process. The criterion for optimization is a cost function or the error signal which is the difference between the desired and the estimated signal given by

$$e(k) = d(k) - y(k) \tag{2}$$

The desired signal is estimated by convolving the input signal with the impulse response expressed as

$$d(k) = w_k^T x(k) \tag{3}$$

where, $x(k) = [x(k), x(k-1),....., x(k-p)]^T$ is the input signal vector. The filter coefficients are updated at every time instant as

$$w_{k+1} = w_k + \Delta w_k \tag{4}$$

Δw_k is a correction factor for the filter coefficients.

The optimization algorithm can be linear or nonlinear. The adaptive neural network equalizer is implemented using a feed forward multiplicative neural network (MNN). Fig. 2 shows the general architecture of the MNN.

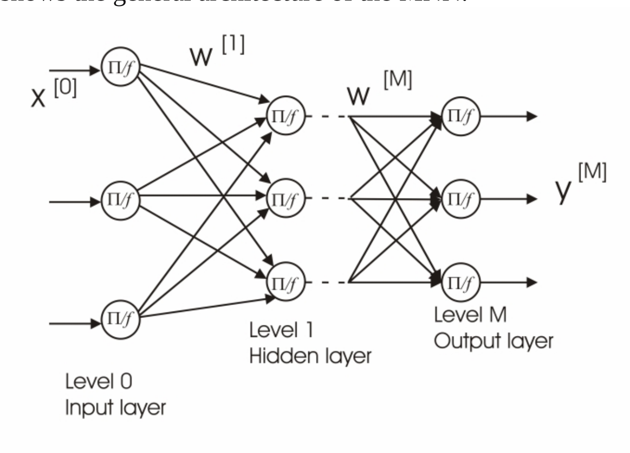

Fig. 2 Multiplicative neural network

The transmitter sends a known training sequence to the receiver. The discrete-time BPSK signal sampled at a rate of f_s is generated by the following equation:

$$r_k(kT_s) = A\exp\left\{ j\frac{p}{2}[1 - m(kT_s)] \right\}$$
$$(k = 0, \pm 1, \pm 2, ...) \tag{5}$$

In order to obtain integral number of samples in each bit interval, the sampling frequency f_s is equal to $\frac{m_s}{T_b}$ where m_s is an integer denoting number of samples per bit duration. If m_k is defined as the discrete time sampled version of the binary sequence $m(t)$, equation 5 becomes

$$r_k = A\exp\left\{ j\frac{p}{2}(1 - m_k) \right\}$$
$$= \begin{cases} A \\ -A \end{cases} \text{for } m_k = 1 \text{ and for } m_k = -1 \tag{6}$$

A sequence of 6000, equiprobable, BPSK complex valued symbol set, in which the input signal takes one of the values {-1, 1} is generated. In the absence of the noise the output signal occupies well-defined states of the BPSK signal constellation.

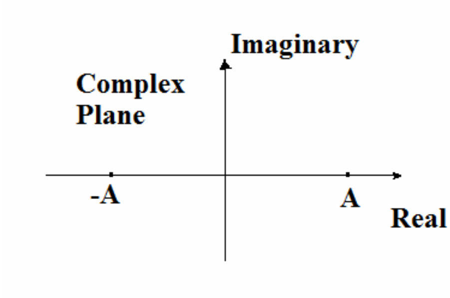

Fig. 3 BPSK signal in complex plane

When the signal is passed through the nonlinear channel, it becomes a stochastic random process. Decision boundaries can be formed in the observed pattern space to classify the observed vectors. For equalization, the adaptive filter is used in series with the unknown system on the test signal $d(k)$ by minimizing the squared difference between the adaptive equalizer output and the delayed test signal. The task of the equalizer is to set its coefficients in such a way that the output $y(k)$ is a close estimate of the desired output $d(k)$. Depending on the value of the channel output vector, the equalizer tries to estimate an output, which is close to one of the transmitted values. The neural equalizer separately processes the real and imaginary part using the multiplicative, split complex, neural network model [12]-[13]. The block diagram of a channel equalizer using MNN is shown in Fig. 4.

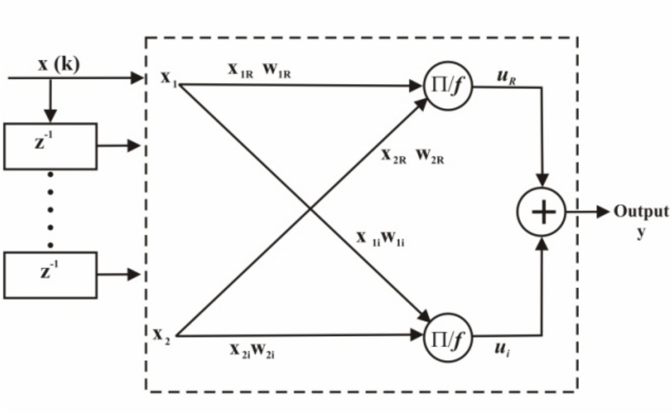

Fig. 4 Multiplicative neural network based channel equalizer

The real R and imaginary I parts of the input signal are split as

$$F(x(t)) = f(x_{1R}(t), x_{2R}(t)) + if(x_{1I}(t), x_{2I}(t))$$

(7)

Where, the input $x_1(t) = x_{1R}(t) + ix_{1I}(t)$

and $x_2(t) = x_{2R}(t) + ix_{2I}(t)$ (8)

III. Learning Rule for MultiplicativE Neuron

A bipolar sigmoidal activation function is used at each node. This kind of neuron itself looks complex in the first instance but when used to solve a complicated problem needs less number of parameters as compared to the existing conventional models. An error back propagation (BP) based learning using a norm-squared error function is described as follows [14]-[15].The symbols used are:

N_o is the number of inputs in the input layer.

n is the number of hidden layers in the FF network.

N_n is the number of neurons in the nth hidden layer.

K is the number of outputs in the output layer.

j_n is the *jth* neuron of the *nth* hidden layer.

y_{jn}^{n} is the output of the *jth* neuron of the *nth* hidden layer.

y_{dk} is the desired output of the *kth* neuron in the output layer.

y_k is the actual output of the *kth* neuron in the output layer.

w_{jnjn-1} is the weight of the connection between *jth* neuron of the *(n-1)th* layer and the *jth* neuron of the *nth* layer.

b_{jnjn-1} is the bias of the connection between *jth* neuron of the *(n-1)th* layer and the *jth* neuron of the *nth* layer.

The output of the *jth* neuron in the first hidden layer is given as

$$y_{j1}^{1} = f\left(\prod_{j0=1}^{N0}(w_{j1j0}x_{j0} + b_{j1j0})\right) \quad (9)$$

for $j_1=1,2,\ldots,N_1$ and x_{j0} represents *jth* input in the input layer and *f*(.) is the activation function defined by

$$f(y) = \frac{1-e^{-y}}{1+e^{-y}} \quad (10)$$

The output of the *jth* neuron in the second hidden layer is given as

$$y_{j2}^{2} = f\left(\prod_{j1=1}^{N1} w_{j2j1}y_{j1}^{1} + b_{j2j1})\right); \text{ for } j_2=1,2,\ldots,N_2$$

(11)

The output of the *jth* neuron in the *nth* hidden layer is given as:

$$y_{jn}^{n} = f\left(\prod_{jn-1=1}^{Nn-1}(w_{jnjn-1}y_{jn-1}^{n-1} + b_{jnjn-1})\right); \text{ for } j_n=1,2,\ldots,N_n \quad (12)$$

The output of the *kth* neuron in the output layer is given as

$$y_k = f\left(\prod_{jn=1}^{Nn-1}(w_{kjn}y_{jn}^{n} + b_{kjn})\right); \text{ for } k=1,2,\ldots,k \quad (13)$$

A simple gradient descent rule, using a mean square error function is used for computation of weight update.

$$E_{MSE} = \frac{1}{2PK}\sum_{k=1}^{K}\sum_{p=1}^{P}(y_{dk}^{p} - y_{k}^{p})^2 \quad (14)$$

Where y_k^p and y_{dk}^p are the actual and desired values, respectively, of the output of the *kth* neuron for the *pth* pattern in the output layer. P is the number of training patterns in the input space. The weights are updated as below. Weights between output layer and the *nth* hidden layer are given by:

$$\Delta w_{kjn} = -h\frac{\partial E_{MSE}}{\partial w_{kjn}}$$

$$= h\, d_k \frac{\left[\prod_{jn=1}^{Nn}(w_{kjn}y_{jn}^{n} + b_{kjn})\right]}{(w_{kjn}y_{jn}^{n} + b_{kjn})} . y_{jn}^{n} \quad (15)$$

$$d_k = \frac{1}{PK}\left[\sum_{k=1}^{K}\sum_{p=1}^{P}(y_{dk}^{p} - y_{k}^{p}).\left[(1/2)(1+y_{k}^{p})(1-y_{k}^{p})\right]\right] \quad (16)$$

$$\Delta b_{kjn} = h\, d_k \frac{\left[\prod_{jn=1}^{Nn}(w_{kjn}y_{jn}^{n} + b_{kjn})\right]}{(w_{kjn}y_{jn}^{n} + b_{kjn})}$$

$$= \frac{\Delta w_{kjn}}{y_{jn}^{n}} \quad (17)$$

Weights between *nth* and *(n-1)th* hidden layer

$$\Delta w_{jnjn-1} = -h\frac{\partial E_{MSE}}{\partial w_{jnjn-1}}$$

$$= \frac{h}{PK}\left[\sum_{k=1}^{K}\sum_{p=1}^{P}(y_{dk}^{p} - y_{k}^{p}).\frac{\partial y_{k}^{p}}{\partial y_{jn}^{n}}\right].\frac{\partial y_{jn}^{n}}{\partial w_{jnjn-1}}$$

$$= h\, d_k \frac{\left[\prod_{jn=1}^{Nn}(w_{kjn}y_{jn}^{n} + b_{kjn})\right]}{(w_{kjn}y_{jn}^{n} + b_{kjn})} . w_{kjn} . \frac{\partial y_{jn}^{n}}{\partial w_{jnjn-1}} \quad (18)$$

$$\Delta b_{jnjn-1} = \frac{\Delta w_{jnjn-1}}{y^{n-1}_{jn-1}} \tag{19}$$

Similarly, we can write equations for weight change between the hidden layer 1 and the input layer.
The weights and biases are updated as

$$w_i^{new} = w_i^{old} + \Delta w_i \tag{20}$$

$$b_i^{new} = b_i^{old} + \Delta b_i \tag{21}$$

IV. Simulation and Results

To study the BER performances the equalizer structure was trained with 3000 iterations and tested over 10000 samples. A fading channel is a communication channel that experiences fading due to multipath propagation. In wireless communications, the presence of reflectors in the environment surrounding the transmitter and receiver create multiple paths that the transmitted signal can traverse. At the receiver there is a superposition of these multipath signals which experience different attenuation, delay and phase shift. This can result in either constructive or destructive interference, amplifying or attenuating the signal power seen at the receiver. Strong destructive interference is known as deep fade. The fading process is characterised by a Rayleigh distribution for a non-line-of-sight path. The coherence time of the channel is related to a quantity known as Doppler spread of the channel. When the user or the reflectors in the environment are mobile, the user's velocity causes a shift in the frequency of the signal transmitted along each signal path. The difference in Doppler shifts between different signal components contributing to a single fading channel tap is known as Doppler spread. The coherence time is inversely proportional to the Doppler spread and is given by:

$$T_c = \frac{k}{D_s} \tag{22}$$

where T_c is the coherence time, D_s is the Doppler spread and k is constant taking on values between 0.25 to 0.5. In flat fading, the coherence bandwidth of the channel is larger than the bandwidth of the signal. Therefore, all frequency components of the signal will experience the same magnitude of fading. In our experiments we have simulated a frequency-flat ("single path") Rayleigh fading channel object as a linear FIR filter, with tap weights given by:

$$g_n = \sum_k \text{sinc}\left(\frac{t_k}{T-n}\right) h_k \quad \text{for} - N_1 \le n \le N_2 \tag{23}$$

The summation has one term for each major path. t_k is the set of path delays and T is the input sample period. N_1 and N_2 are chosen so that $|g(n)|$ is small. N_1 determines the channel filter delay. h_k is the set of complex path gains which are not correlated to each other.
The received signal in Rayleigh fading channel is of the form:

$$y = hx + n$$
(24)

y is the received symbol, h is the complex scaling factor corresponding to Rayleigh multipath channel, x is the BPSK transmitted symbol and n is the AWGN noise. The envelope power of Rayleigh channel model at a Doppler frequency of 75Hz is shown in Fig. 5.

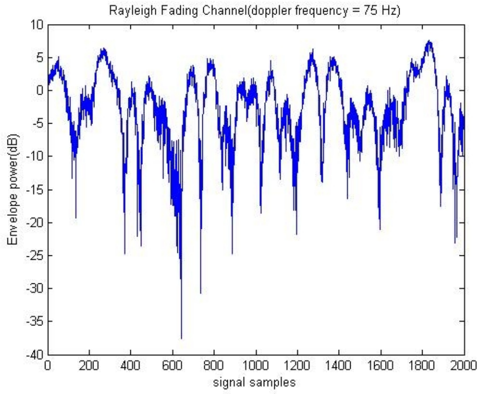

Fig. 5 Rayleigh channel model at Doppler frequency of 75Hz

The received noisy constellation at 15 dB SNR is plotted in Fig. 6.The signal after equalization is reclassified into values of 1 and -1 as shown in Fig. 7.

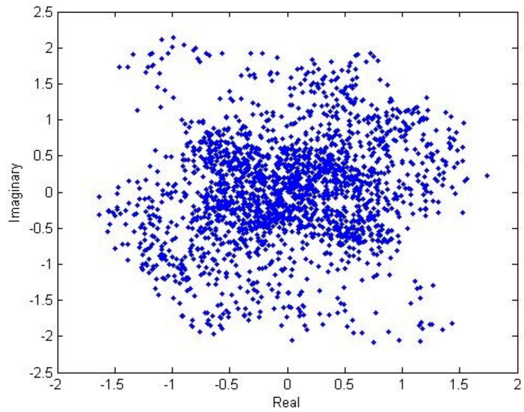

Fig. 6 Received noisy constellation at 15dB SNR

The BER for various SNR is plotted in Fig. 8 and compared with RRBF equalizer. The MNN equalizer gives better performance with optimum number of nodes with 6 neurons in the input layer, 6 neurons in the hidden layer and 2 neurons in the output layer as compared to RRBF network architecture which requires 500 neurons in the input layer, 375 neurons in the hidden layer and 1 neuron in the output layer. We have used the tanh bipolar sigmoidal activation function as compared to the sigmoidal activation function at the nodes [16].

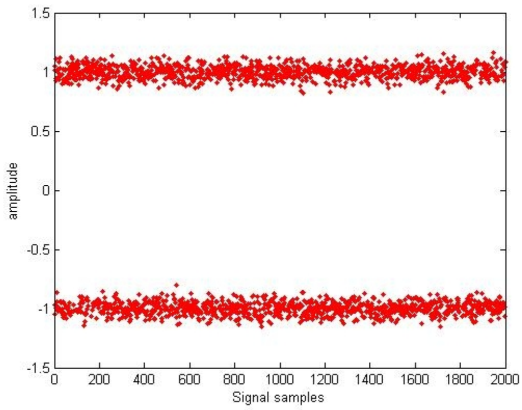

Fig. 7 Equalized signal samples at 15 dB SNR

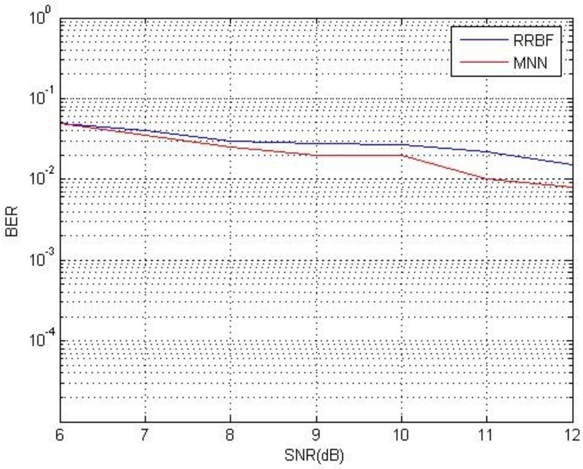

Fig. 8 BER vs. SNR for Rayleigh channel

V. Conclusions

A high order feed forward neural network equalizer with multiplicative neuron is proposed in this paper. Use of multiplication allows direct computing of polynomial inputs and approximation with fewer nodes. Performance comparison in terms of network architecture and BER performance suggest the better classification capability of the proposed MNN equalizer over RRBF.

References

[1] S. Haykin, *Adaptive Filter Theory*, Pearson Education, 2005, pp. 22-25.

[2] D.C. Park, M. A. El-Sharkawi, and R. J. Marks II, "Adaptively trained neural network," *IEEE Trans. Neural Networks*, vol. 2, pp. 334–345, May 1991.

[3] Pham DT, Liu X, *Neural networks for identification, prediction and control*, London: Springer, 1995.

[4] S. Bang, S. H. Sheu, and J. Bing, "Neural network for detection of signals in communication," *IEEE Trans. Circuits Syst. I*, vol. 43(8), pp. 644-655, Aug. 1996.

[5] S. Chen, G. Gibson, C. Cown, and P. Grant, "Adaptive equalization of finite nonlinear channels using multilayer perceptrons," *Signal Processing*, vol. 20, pp. 107-119, June 1990.

[6] G. Gibson, S. Siu, and C. Cowan, "Multilayer perceptron structures applied to adaptive equalizers for data communications," in *Proc. ICASSP*, May 1989, Glasgow, U.K., pp. 1183-1186

[7] T. Kim, T. Adali," Fully complex multi-layer perceptron network for nonlinear signal processing," *J. VLSI Signal Process.*, vol. 32, No. 1, pp. 29-43, 2002.

[8] Q. Zhang, "Adaptive equalization using the back propagation algorithm," *IEEE Trans. Circuits Syst.*, vol. 37, pp. 848–849, June 1990.

[9] C.L. Giles and T. Maxwell," Learning, invariance, and generalization in high-order neural networks," *Applied Optics*, vol. 26, no. 23, pp. 4972-4978, 1987.

[10] E.M. Iyoda, K. Hirota, and F. J. Von Zuben," Sigma-Pi cascade extended hybrid neural networks," *Journal of Advanced Computational Intelligence*, vol.6, no. 3, pp. 126-134, 2002.

[11] M Schmitt," On the complexity of computing and learning with multiplicative neurons," *Neural Comput.* vol. 14, no.2, pp. 241-301, Feb. 2002.

[12] A. Kantsila, M. Lehtokangas, J. Saarinen," Complex RPROP-algorithm for neural network equalization of GSM data bursts", *Neurocomputing*, vol. 61, pp. 339- 360, 2004.

[13] Kavita Burse, R.N. Yadav and S.C. Shrivastava," Complex Channel Equalization using Polynomial Neuron Model," in *Proc. IEEE 3rd Int. Symposium on Information Technology*, Kuala Lumpur, Malaysia, Aug. 26-29, 2008, pp. 771-775.

[14] R.N. Yadav, P.K. Kalra and J. John," Time series prediction with single multiplicative neuron model," *Applied soft computing*, vol. 7, pp 1157-1163, 2007.

[15] Kavita Burse, R.N. Yadav, S.C. Shrivastava, Vishnu Pratap Singh Kirar," A compact pi network for reducing bit error rate in dispersive FIR channel noise model," in *Proc. of WCSET:2009,World Congress on Science, Engineering and Technology*, pp. 235-238, Feb. 25-27, 2009, Penang, Malaysia.

[16] Gowrishankar and Satyanarayana P.S.," Recurrent neural network based bit error rate prediction for 802.11 wireless local area network," *International Journal of Computer Sciences and Engineering Systems*, vol.2, no.3, pp. 177-184, July 2008.

Kavita Burse received her Bachelor of Engineering (Electronics and Communication) from Shri Govind Ram Seksaria Institute of Science and Technology, Indore, India in 1992 and M.Tech (Digital Communication) from Maulana Azad National Institute of Technology, Bhopal, India. Currently she is pursuing Ph.D. degree from the same Institute. She is faculty with the Department of Electronics and Communication at Truba Institute of Engineering and Technology, Bhopal. She is associate member CSI and life member ISTE.

R. N. Yadav received his Bachelor of Engineering degree from Motilal Nehru Regional Engineering College Allahabad, India, M.Tech degree from Maulana Azad College of Technology, Bhopal, India and P.hD. degree from Indian Institute of Technology, Kanpur, India in 1993, 1997 and 2005 respectively. Currently he is Assistant Professor in the Department of Electronics and Communication Engineering, Maulana Azad National Institute of Technology, Bhopal, India. He is a life member of IETE and IE(I). He has authored and reviewed more than twenty papers in international Journals and conferences of repute.

S.C. Shrivastava received his Bachelor and Master degree in Engineering from Government Engineering College, Jabalpur, India in 1968 and 1970 respectively and P.hD. degree in 1994 from Barkatullah University, Bhopal, India. He is a life member of IETE, IE(I) and ISTEE. Currently he is Professor and Head, Department of Computer Science and Engineering, Maulana Azad National Institute of Technology, Bhopal, India. He has authored several papers in national and international Journals and conferences.