# On the Development and Performance Evaluation of Improved Radial Basis Function Neural Networks

Sashmita Panda, *Student Member, IEEE*, and Ganapati Panda, *Senior Member, IEEE*

***Abstract*—This article deals with the development of four modified radial basis function neural network (RBFNN) models. The corresponding learning algorithms associated with the updating of internal parameters of the models are derived. The conventional inputs are used in the first and second modified RBFNN models (models 3 and 4) whereas exponential nonlinear inputs are used in the fifth and sixth RBFNN models to provide additional nonlinearity for achieving a better solution of nonlinear classification, and direct and inverse modeling problems. To assess and compare the performance potentiality of the proposed four new RBFNN models, one classification problem, one direct modeling problem, and one inverse modeling problem are solved through computer simulation-based experiments. For comparison and to assign the performance rank of each of the four modified RBFNN models, two conventional and commonly used RBFNN models (models 1 and 2) are also simulated. To access the performance of different models during the training phase of Examples 1 and 2, the root mean-square error (RMSE) value, mean absolute deviation (MAD), and the number of iterations required to achieve convergence are obtained. For the third example, only the first two performance measures are found. During the testing or validation phase, the output responses of the different models of Example 2 are compared with the desired response analysis. For Example 3, the bit-error rate (BER) plots are compared. The observation of all the results demonstrates consistent ranks of all models in the case of all three examples. It is, in general, found that the ranks of the models 1–6 are 6, 4, 3, 2, 5, and 1, respectively. In essence, in terms of all performance measures, model M-6 with an exponential version of inputs with weights on both layers occupies the first position whereas model M-4 with conventional inputs as the second position.**

***Index Terms*—Bit-error rate (BER), classification, direct modeling, inverse modeling, learning algorithm, mean absolute deviation (MAD), radial basis function neural network (RBFNN), root mean-square error (RMSE).**

## I. Introduction

The RADIAL basis function neural network (RBFNN) essentially comprises input and output layers and a single hidden layer and an output stage. The hidden layer contains a set of radial basis functions at each node, and the hidden to output layer contains adjustable weights to produce the desired output. The RBFNN finds applications in areas of function approximation [1], clustering [2], classification [3], [4], forecasting [5], [6], estimation [7], direct system modeling [8], inverse system modeling [9], and adaptive control [10]. The two domains of research works in RBFNN are theoretical development in architectures and learning algorithms, as well as potential applications in different areas of science, engineering, and economics. A review of reported work in these two domains is presented in the sequel.

In [11], it is reported that for the classification problem, the RBFNN exhibits slower convergence and involves more computation time for large datasets. An error correction algorithm is suggested [12] to reduce the computational efforts by selecting each time the most violating input vector as the center of an additional hidden unit until convergence is achieved. A new RBFNN [13] using the multicolumn deep technique is proposed to improve the accuracy and reduce training and testing time. The placement of centers is important in RBFNN. In [14], the bound for the gradient of the error is estimated and considered as a function of centers, and is used to change the centers to reduce the overall error. A supervised method to select the center and width of kernel functions of the hidden neuron is developed [15] by employing the class membership of training samples. This approach reduces the error and provides a stable performance. The selection of the number of neurons of the hidden layer in RBFNN is a tedious task. This problem is sorted out in [16] using the principle of preservation of data structure. A novel median algorithm is developed in [17] using the marginal median for the estimation of kernel location and the median of absolute deviations for the estimation of scale parameters. In addition to the conventional training algorithm of RBFNN, the weights between input and hidden units are incorporated and adjusted in a novel RBFNN structure [18]. This strategy is reported to provide faster and efficient training of RBFNN. A new learning rule of RBFNN is reported [19] by constructing one subnetwork to approximate the probability density function of each class of objects in the training dataset. It is shown that for the classification task, the new algorithm offers accuracy similar to that of the support vector machine (SVM). For efficient training of RBFNN [20], a nonsymmetric fuzzy partitioning of the input space and optimization of the number of fuzzy sets in each dimension using particle swarm optimization (PSO) are introduced. This hybrid approach offers less error during the testing phase.

Manuscript received March 17, 2020; revised December 7, 2020; accepted April 23, 2021. Date of publication May 5, 2021; date of current version May 18, 2022. This article was recommended by Associate Editor L. Wang. *(Corresponding author: Sashmita Panda.)*

Sashmita Panda is with the Department of G. S. Sanyal School of Telecommunications, Indian Institute of Technology Kharagpur, Kharagpur 721302, India (e-mail: ysashmita786@gmail.com).

Ganapati Panda is with the Department of Electronics and Telecommunications Engineering, C. V. Raman Global University, Bhubaneswar 752054, India (e-mail: ganapati.panda@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TSMC.2021.3076747.

Digital Object Identifier 10.1109/TSMC.2021.3076747

To circumvent the problem of longer training time and choice of the number of hidden units, the orthogonal least square-based algorithms are suggested in [21]–[24]. To overcome the two constraints of RBFNN, alternative methods, including constructive decay [25], resource allocating networks [26], principle of minimum description length [27], and fuzzy partition of input space [28], are dealt in the literature. To enable real-time adaptation of RBFNN, an online fuzzy means (FM) algorithm is reported [29]. A nonsymmetric partition of the input space is employed [30] to develop an efficient RBFNN with a fewer number of hidden nodes. The optimization of width parameters is a challenge in RBFNN, which contributes to the accuracy of classification or estimation at the output. In [31], this issue is addressed and a concurrent subspace width optimization (CSWO) technique is proposed, and a corresponding RBFNN model is developed to provide improved performance. A new structure of RBFNN is proposed [32] by taking into account linear regression weight and changing the Gaussian function according to the interval distance measure. The effectiveness of the new structure is shown by conducting few experiments. A fuzzy RBFNN model is reported [33], which contains fuzzy weights and uses fuzzy inputs to produce fuzzy outputs. The proposed model is shown to perform as a universal function approximator. In an interesting paper [34], the authors investigate the generalization performance of RBFNN using the structural risk minimization principle and general loss function. Empirical studies reveal the potentiality of their approach. A new learning algorithm of RBFNN based on the sequential projection-based metacognitive principle is reported in [35]. For a classification problem, the proposed model is shown to exhibit better generalization performance with minimum computation. A sequential learning algorithm of RBFNN is suggested in [36]. The simulation results show that the proposed scheme provides comparable generalization performance with lesser training time and network complexity. A novel complex-valued RBFNN is reported in [37], which employs the metacognitive learning framework to control the learning rate. It is observed that the new learning scheme improves the approximation and classification performance. The fuzzification of the output layer of RBFNN is introduced in [38]. Self-organized learning is used for training the parameters of hidden units, and supervised learning is used for updating the weights. The simulation study is carried out to show the efficacy of the approach. A modified RBFNN is reported [39], in which regression weights obtained by the expectation–maximization algorithm are used in the output layer. It is shown that the performance of the proposed approach is better than the conventional RBFNN. A variable projection-based scheme is suggested [40] to estimate the model parameters of RBFNN. It is shown that through the modeling of chaotic time series, the proposed approach is computationally more efficient than the other reported methods. In a recent work [41], a self-organizing RBFNN is developed using an adaptive PSO. By simulating few nonlinear problems, the proposed method is found to be more effective than other self-organizing RBFNNs. A novel machine learning-based training algorithm for RBFNN is proposed in [5] and is employed for electric load forecasting. It is shown that the new model outperforms other methods based on extreme learning machine and support vector machine-based approaches. Some of the recent applications of RBFNN include modeling of antenna array for directivity prediction [42], [43], removal of haze for visibility restoration [44], identification of black plastic material [45], portfolio selection [46], sliding mode control of NPC converters [47], adaptive control of wind energy conversion system [48], real-time tuning of power system stabilizer [49], and classification of process faults [50]. In a recent article [51], the RBFNN is suggested to be used in the last layer of the convolutional neural network (CNN) in place of conventional ANN for classification purpose. The Gaussian radial basis function associated with the RBFNN is replaced by a new activation function, and the training process is introduced. The new CNN-RBFNN structure has been successfully applied to computer vision. In another interesting paper [52], the authors have proposed an alternative approach using the direct method of multipliers (ADMM) and minimax concave penalty (MCP), which selects the centers of RBF to provide good fault tolerance through a simulation study. It is shown that the proposed approach exhibits improved performance under fault conditions compared to other methods. The remaining useful life of an external gear pump has been efficiently predicted [53] by employing Bayesian regularized RBFNN. The proposed method has demonstrated an improved evaluation of the residual useful life of the gear pump. A novel method of detection of contactless insulator contamination levels is reported in [54] using features of infrared images and the RBFNN model. It is shown through a simulation study that the suggested prediction model performs better than the conventional backpropagation-based ANN and the generalized regression neural network.

The conventional RBFNN model is another form of ANN, which consists of two layers and one set of hidden nodes that essentially controls the overall performance of the network. Each hidden node contains a radial basis function, which is mostly a Gaussian function. In the conventional RBFNN, the parameter of the Gaussian function and the weights of the output layers are adapted to suit many engineering applications requiring operations, such as prediction, classification, filtering, detection, function approximation, control, direct, and inverse modeling. Because of its superior performance in terms of accuracy for any operation and generalization potentiality, it is preferred over the conventional backpropagation-based ANN model. Many articles have dealt with achieving faster and improved learning of the model. But still, there is further scope to improve its training time and overall accuracy by suitably modifying its structure.

### A. Motivation and Objectives of Research Work

The study on literature survey reveals that the publications related to RBFNN pertain to: 1) perform various operations [1]–[10]; 2) modifications of architecture or structure [16], [18], [33]–[35], [39]; 3) improvement in accuracy of performance [13]–[15], [19], [31]; 4) parameters optimization [17], [40], [41]; 5) reduction of training time and number of hidden units [25]–[29]; 6) tunning of learning rate [37], [38]; and 7) different practical applications [42]–[50]. These observations motivate that still more work can be carried out relating to structure, reduction in training time, and accuracy enhancement for basic operations, such as classification, direct, and inverse modeling of nonlinear systems. These two issues have been addressed in this article. In the first part of the investigation, four modifications in the existing structure leading to four modified RBFNN models have been suggested keeping in view that the performance in terms of faster training and improvement inaccuracy should be achieved. In each case, the new parameters update equations are derived. Furthermore, the basic examples of classification, direct, and inverse modeling problems are solved through simulation-based experiments: 1) using the new sets of learning algorithms during the training phase and 2) evaluating and comparing the performance of the proposed RBFNN models with the standard two models. The results in terms of the mean absolute deviation (MAD), root mean-square error (RMSE), and the number of iterations required when convergence is achieved and the response matching potentiality are obtained and compared and accordingly the rank of each RBFNN model is assigned.

### B. Organization of This Article

The remainder of this article proceeds as follows. Section II deals with two conventional basic models: the first one with update equations of weights of last layer and center updates of hidden nodes. In the second model, the update provision of the standard deviation of the Gaussian basis function is included. The development of two modified RBFNN models with the organization of the article proceeds as follows. Section II deals with two conventional basic models: the first one with update equations of weights of the last layer and center updates of hidden nodes. In the second model, the update provision of the standard deviation of the Gaussian basis function is included. The development of two modified RBFNN models with the provision of change of weights in the two layers as well as a center update in one and, in addition, the standard deviation update of the Gaussian function in the other is presented in Section III. The development of another two modified RBFNNs is dealt with in Section IV, in which the inputs used are an exponential form of actual inputs. In the first model, the conventional RBFNN is employed whereas in the second model, the weights of both layers are updated. The illustration of simulation examples using proposed RBFNN models is dealt with in Section V. In Section VI, the simulation study of three examples using all four modified RBFNN and two conventional RBFNN models is carried out. To assess the performance of these six models, simulation study is carried out, and the results are compared in the three cases. The analysis of simulation results is made and based on that rank of the models is assigned. Finally, the conclusion of the investigation, the contribution of the research work, and the scope for further extension of the current work are presented in Section VII.

TABLE I
INPUT AND ADAPTATION OF PARAMETERS USED IN DIFFERENT RBFN MODELS

| RBFN Models | Inputs | | Update of Weights of Layers | | Update Parameters of Radial Basis Function | |
|---|---|---|---|---|---|---|
| | Conventional $x_i$ | Exponential $e^{-x_i}$ | Output | Input | Centre | Standard Deviation |
| Model-1 (M-1) | Yes | _ | Yes | _ | Yes | _ |
| Model-2 (M-2) | Yes | _ | Yes | _ | Yes | Yes |
| Model-3 (M-3) | Yes | _ | Yes | Yes | Yes | _ |
| Model-4 (M-4) | Yes | _ | Yes | Yes | Yes | Yes |
| Model-5 (M-5) | _ | Yes | Yes | _ | Yes | _ |
| Model-6 (M-6) | _ | Yes | Yes | Yes | Yes | Yes |

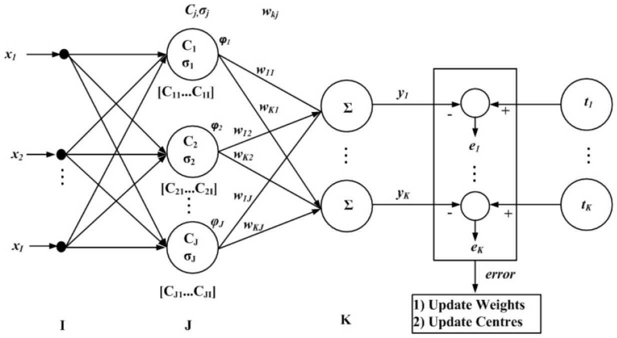

Fig. 1. RBFNN model (M-1) with updates of weights and centers.

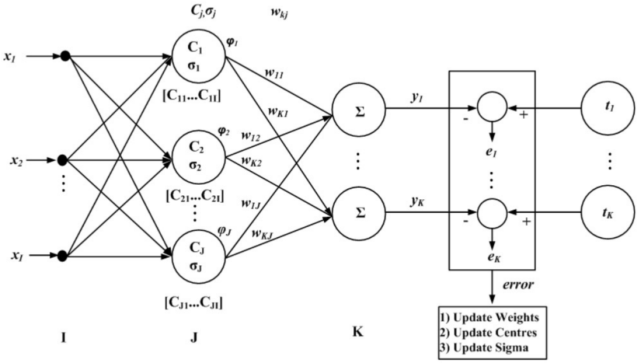

Fig. 2. RBFNN model (M-2) with updates of weights, centers, and standard deviations.

## II. CONVENTIONAL RBFNN MODEL WITH UPDATES OF WEIGHTS AND CENTERS

In this section, two conventional RBFNN models (M-1 and M-2) are presented. The block diagrams and the key equations relating to these models are also dealt with in this section, which serves as basics for the development of four new RBFNN models (M-3, M-4, M-5, and M-6). In Table I, the details of the types of input and adaptation of different parameters used in all the six models are listed.

The block diagrams of two conventional models M-1 and M-2 are shown in Figs. 1 and 2, respectively. The corresponding key equations are also provided for a better understanding of the newly developed four modified RBFNN models in this article.

TABLE II
INTERPRETATION OF SYMBOLS USED

| Symbols Used | Interpretation of Symbols |
|---|---|
| I | Total number of inputs |
| J | Total number of hidden nodes |
| K | Total number of output nodes |
| X=[$x_1$, $x_2$, ........$x_i$, $x_I$] | Input vector of I inputs |
| $C_j$=[$C_j$ $C_{j2}$ ........$C_{JI}$] | I number of Centre values of Gaussian (radial basis) function of $j_{th}$ hidden node |
| $\phi_j$ | Output of any $j_{th}$ hidden node |
| $W_{ji}$ | Weight value between $i_{th}$ input node to $j_{th}$ hidden node $1 \leq i \leq I$, $1 \leq j \leq J$ |
| $W_{kj}$ | Weight value between $j_{th}$ hidden node to $k_{th}$ output node, $1 \leq k \leq K$ |
| $\sigma_j$ | Standard deviation of Gaussian function at $j_{th}$ node $1 \leq j \leq J$ |
| $y_k$ | Output at $k_{th}$ output node, $1 \leq k \leq K$ |
| $t_k$ | Target value applied at $k_{th}$ output node , $1 \leq k \leq K$ |
| $e_k$ | Error value at $k_{th}$ output node during training phase of the model |
| $\eta_w$ | Learning rate for update of weights |
| $\eta_c$ | Learning rate for update of centres |
| $\eta_\sigma$ | Learning rate for update of sigmas |

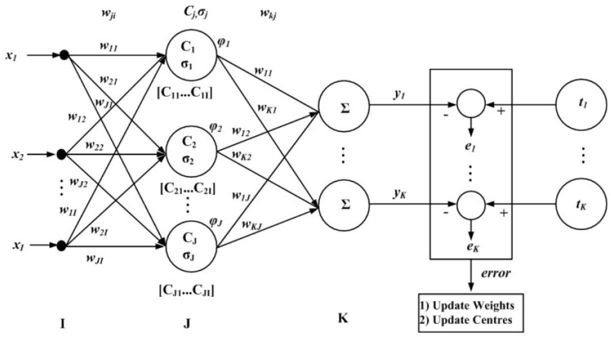

Fig. 3. RBFNN model (M-3) with updates of weights in both layers and centers.

## A. Conventional RBFNN Model 1 (M-1)

The details of the various symbols used in all the models and their corresponding meanings are presented in Table II.

The key equations of the standard RBFNN model (M-1) are presented as follows:

$$y_k = \sum_{j=1}^{J} \phi_j w_{kj}. \tag{1}$$

The output of the $j$th hidden node is given by

$$\phi_j = e^{\frac{-z_j^2}{2\sigma_j^2}} \tag{2}$$

$$z_j = \left\| X - C_j \right\| = \sqrt{\sum_i \left(x_i - c_{ji}\right)^2} \tag{3}$$

where $||.||$ stands for the Euclidean distance between $X$ and $C_j$ and $X = [x_1\, x_2, \ldots, x_i, \ldots, x_I]^T$, $C_j = [c_{j1}\, c_{j2}, \ldots, c_{ji}, \ldots, c_{JI}]^T$. The cost function in (4) is minimized to obtain different update equations of the models defined

$$\text{Cost function}: E = \frac{1}{2}\sum_k (t_k - y_k)^2. \tag{4}$$

The gradient descent learning for updating the weights of the output layer and centers of the radial basis function is represented as follows:

$$w_{kj}(t+1) = w_{kj}(t) - \eta_w \frac{\partial E}{\partial w_{kj}} \tag{5}$$

$$c_{ji}(t+1) = c_{ji}(t) - \eta_c \frac{\partial E}{\partial c_{ji}} \tag{6}$$

where for M-1

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial w_{kj}} = -(t_k - y_k)\phi_j \tag{7}$$

$$\begin{aligned}\frac{\partial E}{\partial c_{ji}} &= \left[\sum_k \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial \phi_j}\right].\frac{\partial \phi_j}{\partial z_j}.\frac{\partial z_j}{\partial c_{ji}} \\ &= \left[\sum_k -(t_k - y_k)w_{kj}\right] \times \frac{\phi_j}{\sigma_j^2} \times -\left(x_i - c_{ji}\right).\end{aligned} \tag{8}$$

Equations (1) and (2) represent the outputs of the $k$th output layer and $j$th hidden layer, respectively. The update equations relating to the weights of the output layer and the centers of the radial basis function of the hidden layer are given in (5)–(8).

## B. Conventional RBFNN Model 2 (M-2)

For model M-2 of Fig. 2, the key equations are provided in this section. The significance of the symbols used is presented in Table II, and can also be observed from Fig. 3. For this model, (1)–(8) are the same as in model M-1. However, since the standard deviation of the Gaussian center of this model are also updated, the learning equations relating to the update of the standard deviation are presented in (9) and (10). The gradient descent learning for updating the standard deviation is expressed as

$$\sigma_{ji}(t+1) = \sigma_{ji}(t) - \eta_\sigma \frac{\partial E}{\partial \sigma_{ji}} \tag{9}$$

where

$$\frac{\partial E}{\partial \sigma_j} = \left[\sum_k \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial \phi_j}\right]\frac{\partial \phi_j}{\partial \sigma_j}$$

$$= \left[\sum_k -(t_k - yk)w_{kj}\right]\frac{z_j^2 \times e^{\frac{-z_j^2}{2\sigma_j^2}}}{\sigma_j^3}. \qquad (10)$$

In the next section, an attempt has been made to improve the performance of models M-1 and M-2 by incorporating weights in the input layer and making provision of change in weights during the training phase. Hence, model M-3 is similar to model M-1 except that in model M-3, the weights of the input layer are updated in addition to the update of weights of the output layer and the center values of the Gaussian radial basis function. Similarly, in model M-4, the weights of the input layer are changed in addition to the change in center values and standard deviations of the Gaussian centers, as well as the weights of the output layer.

## III. Development of Improved RBFNN Models 3 and 4 (M-3 and M-4)

The development of two RBFNN modified models with the provision of updation of weights in both layers as well as the center values of the Gaussian function is dealt with in this section.

### A. Development of Improved RBFNN Model 3 (M-3)

The block diagram of the model M-3 is depicted in Fig. 3. The symbols used in the diagram as well as in the derivation of the update equations are explained in Table II.

Referring to Fig. 3, the output at the *j*th hidden node before passing to the Gaussian basis function is given by

$$a_j = \sum_{i=1}^{I} x_i w_{ji} \qquad (11)$$

$$z_j = \|A - C_j\| = \sqrt{\sum_i (a_i - c_{ji})^2}. \qquad (12)$$

where ||.|| stands for the Euclidean distance between $A$ and $C_j$ and $A = [a_1\, a_2 \cdots a_j \cdots a_J]^T$, $C_j = [c_{j1}\, c_{j2} \cdots c_{ji} \cdots c_{JI}]^T$. The equations for computing the outputs of the hidden nodes as well as the output nodes are identical to (2) and (1), respectively. Similarly, the cost function, $E$ defined in (4), is to be minimized for obtaining the various update equations for the weights of both layers as well as the centers of the Gaussian radial basis functions. The gradient descent learning rule for the centers is the same as (6). The assumption made in the derivation of the learning algorithms of any RBFNN model is that the expected value of the product of two variables is equal to the product of two variables. This assumption is identical to the assumption made for LMS or BP learning algorithms. Then, the gradient descent learning of the weights of the two layers is given by

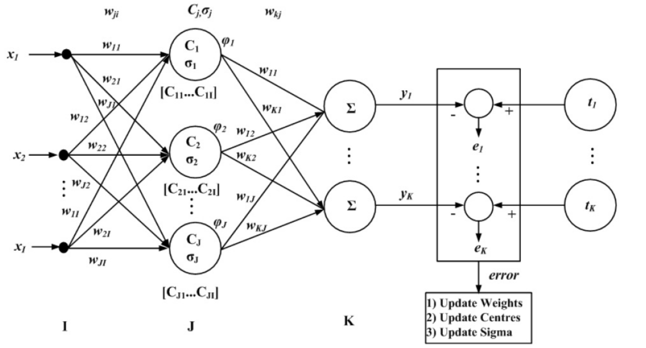

Fig. 4. RBFNN model (M-4) with updates of weights in both layers, centers, and standard deviations.

$$w_{kj}(t+1) = w_{kj}(t) - \eta_{w_1}\frac{\partial E}{\partial w_{kj}} \qquad (13)$$

$$w_{ji}(t+1) = w_{ji}(t) - \eta_{w_2}\frac{\partial E}{\partial w_{ji}} \qquad (14)$$

where the symbols $\eta_{w_1}$ and $\eta_{w_2}$ are explained in Table II. Using the chain rule, the values of $(\partial E/\partial c_{ij})$, $(\partial E/\partial w_{kj})$, and $(\partial E/\partial w_{ji})$ are derived. These values are obtained as

$$\frac{\partial E}{\partial c_{ji}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial \phi_j}.\frac{\partial \phi_j}{\partial c_j}$$

$$= \frac{-(t_k - y_k)w_{kj}\phi_j(a_j - c_j)}{\sigma^2} \qquad (15)$$

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial w_{kj}}$$

$$= -(t_k - y_k).\phi_j \qquad (16)$$

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial \phi_j}.\frac{\partial \phi_j}{\partial a_I}.\frac{\partial a_I}{\partial w_{ji}}$$

$$= \frac{-(t_k - y_k)w_{kj}\phi_j(a_I - c_{ji})x_i}{\sigma^2}. \qquad (17)$$

Equations (13)–(16) represent the learning equations for updating the Gaussian centers and the weights of two layers of M-3, respectively.

### B. Development of Improved RBFNN Model 4 (M-4)

The model M-4 shown in Fig. 4 is similar to model M-3 except that the standard deviation of the Gaussian radial basis functions is also updated in addition to the updates of the weights of both layers and the center values of the Gaussian functions. The derivation of the learning equations of various parameters of this model has been carried out in this section. Referring to Fig. 4, the computation of $a_j$, $z_j$, $\phi_j$, $y_k$, and cost function $E$ is carried out using (11), (12), (2), (1), and (4), respectively. The values of $(\partial E/\partial c_{ji})$, $(\partial E/\partial w_{kj})$, $(\partial E/\partial w_{ji})$, and $(\partial E/\partial \sigma_j)$ are obtained using the chain rule and are presented in (18)–(21)

$$\frac{\partial E}{\partial c_{ji}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial \phi_j}.\frac{\partial \phi_j}{\partial c_j}$$

$$= \frac{-(t_k - y_k)w_{kj}\phi_j(a_i - c_{ji})}{\sigma_j^2} \qquad (18)$$

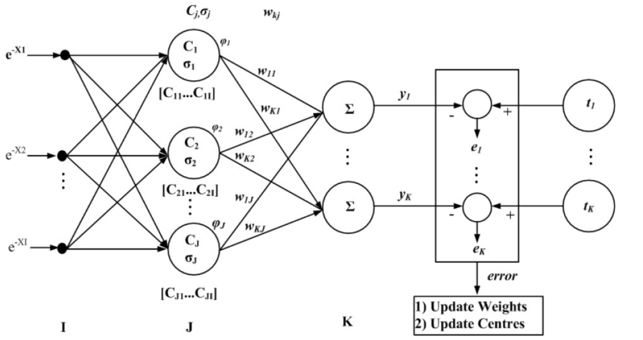

Fig. 5. RBFNN model (M-5) with updates of weights and centers using input $e^{-x}$.

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial w_{kj}} = -(t_k - y_k).\phi_{j]} \tag{19}$$

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial \phi_j}.\frac{\partial \phi_j}{\partial a_i}.\frac{\partial a_i}{\partial w_{ji}} = \frac{-(t_k - y_k)w_{kj}\phi_j\big(a_i - c_{ji}\big)x_i}{\sigma_j^2} \tag{20}$$

$$\frac{\partial E}{\partial \sigma_j} = \left[\sum_k -(t_k - y_k)w_{kj}\right].\frac{z_j^2\phi_j}{\sigma_j^3}. \tag{21}$$

The center values and the standard deviation associated with the Gaussian radial basis function of M-4 are updated using (6) and (18), as well as (9) and (21), respectively. Similarly, the weights of the output layers are updated using (13) and (19) whereas (14) and (20) are used to update the weights of the input layer.

## IV. Development of Improved RBFNN Model With Updates of Weights and Centers Using Input $e^{-x}$ (M-5 and M-6)

### A. *Improved RBFNN Model M-5*

The model M-5 shown in Fig. 5 is an improved RBFNN model using the input $e^{-x}$. In this model, the weights as well as the centers of the Gaussian radial basis function are updated.

Referring to Fig. 5, the key equations for this model are defined as in (1)–(4), respectively. This modified RBFNN (M-5) model is similar to the M-1 structure except that input is an exponential form of the conventional input. The weights of the output layer and the centers of the radial basis function are updated by using (13) and (6). The values of $(\partial E/\partial w_{kj})$ and $(\partial E/\partial c_{ji})$ are derived as follows:

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial w_{kj}} = -(t_k - y_k)\phi_j \tag{22}$$

$$\frac{\partial E}{\partial c_{ji}} = \left[\sum_k \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial \phi_j}\right].\frac{\partial \phi_j}{\partial z_j}.\frac{\partial z_j}{\partial c_{ji}} = \left[\sum_k -(t_k - y_k)w_{kj}\right] \times \frac{\phi_j}{\sigma_j^2} \times -\Big(e_i^{-x} - c_{ji}\Big)\Bigg]. \tag{23}$$

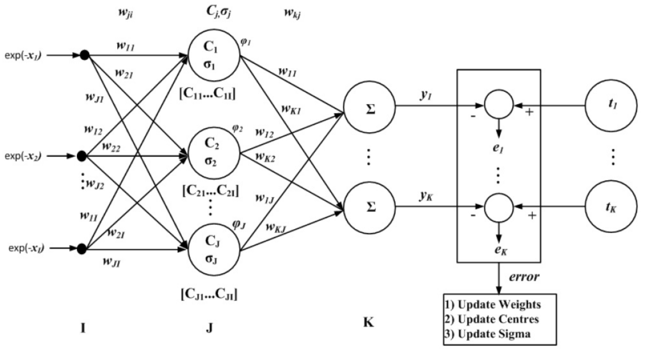

Fig. 6. RBFNN model (M-6) with updates of weights in both layers, centers, and standard deviations and using input $e^{-x}$.

Equations (13), (6), (22), and (23) represent the learning equations of weights and centers of the radial basis functions of model M-5.

### B. *Improved RBFNN Model M-6*

The model M-6 shown in Fig. 6 is similar to that of Fig. 5 except that the standard deviation of the Gaussian radial basis functions is also updated in addition to the updating of the weights of both layers and the centers of the radial basis function.

Referring to Fig. 6, $a_j$, $z_j$ ,$\phi_j$, $y_k$, and cost function $E$ are computed by using (11), (12), (2), (1), and (4), respectively. The values of $(\partial E/\partial c_{ji})$, $(\partial E/\partial w_{kj})$, $(\partial E/\partial w_{ji})$, and $(\partial E/\partial \sigma_j)$ are derived using

$$\frac{\partial E}{\partial c_{ji}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial \phi_j}.\frac{\partial \phi_j}{\partial c_j} = \frac{-(t_k - y_k)w_{kj}\phi_j\big(a_j - c_j\big)}{\sigma^2} \tag{24}$$

$$\frac{\partial E}{\partial w_{kj}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial w_{kj}} = -(t_k - y_k).\phi_j \tag{25}$$

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial y_k}.\frac{\partial y_k}{\partial \phi_j}.\frac{\partial \phi_j}{\partial a_I}.\frac{\partial a_I}{\partial w_{ji}} = \frac{(t_k - y_k)w_{kj}\phi_j\big(a_I - c_{ji}\big)e_i^{-2x}}{\sigma^2} \tag{26}$$

$$\frac{\partial E}{\partial \sigma_j} = \left[\sum_k -(t_k - y_k)w_{kj}\right].\frac{z_j^2\phi_j}{\sigma_j^3}. \tag{27}$$

Equations (6), (13), (14), (24), and (27) provide the updated equations of center values, standard deviation values of Gaussian radial basis functions, and the weights of both layers.

## V. Illustration of Simulation Examples

To demonstrate the potentiality of the proposed four modified RBFNN models, three standard examples are chosen and solved by a simulation study. These are: 1) classification; 2) nonlinear system identification (direct modeling); and 3) nonlinear channel equalization (inverse modeling). The performance of each of the four models is obtained and compared. In addition, the performance of the proposed model is

TABLE III
PARAMETERS USED FOR SIMULATION STUDY OF MODELS 1–3

| Applications | Structure of RBFN | Parameters used for simulation study of Model | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | M-1 | | M-2 | | | M-3 | | |
| | | $\eta_c$ | $\eta_{wo}$ | $\eta_c$ | $\eta_\sigma$ | $\eta_{wo}$ | $\eta_c$ | $\eta_{wi}$ | $\eta_{wo}$ |
| Classification | 2-2-1 | .0004 | .0003 | 0.004 | .0004 | .003 | 0.004 | .0005 | .003 |
| Direct Modelling of Nonlinear Systems (System Identification) | 3-3-1 | .00002 | .00002 | .00002 | .00001 | .00002 | .00005 | .001 | .00002 |
| Inverse Modelling of Nonlinear Systems (Channel Equalization) | 6-6-1 | .0005 | .0005 | .005 | .0001 | .005 | .005 | .0005 | .005 |

TABLE IV
PARAMETERS USED FOR SIMULATION STUDY OF MODELS 4–6

| Applications | Structure of RBFN | Parameters used for simulation study of Model | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | M-4 | | | | M-5 | | M-6 | | | |
| | | $\eta_c$ | $\eta_\sigma$ | $\eta_{wi}$ | $\eta_{wo}$ | $\eta_c$ | $\eta_{wo}$ | $\eta_c$ | $\eta_\sigma$ | $\eta_{wi}$ | $\eta_{wo}$ |
| Classification | 2-2-1 | .0004 | .0004 | .0005 | .003 | 0.0007 | .0006 | 0.0008 | .0001 | .0005 | .0006 |
| Direct Modelling of Nonlinear Systems (System Identification) | 3-3-1 | .00005 | 0.0005 | .001 | .00002 | .00005 | .00005 | .00005 | .0002 | .0005 | .00005 |
| Inverse Modelling of Nonlinear Systems (Channel Equalization) | 6-6-1 | .005 | .0001 | .0005 | .005 | .005 | .005 | .005 | .0001 | .0005 | .005 |

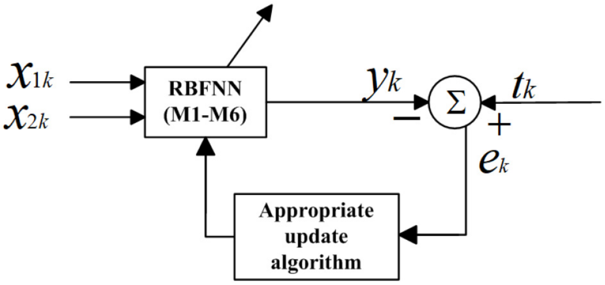

Fig. 7. Basic block diagram of implementation of Ex-OR classification problem (Ex-1).

also compared with the two standard basic models. The block diagram and details of each problem are explained in short in this section. The various parameters and constants used during the simulation study are presented in Tables III and IV for models 1–3 and 4–6, respectively. In addition, initial weight values in the two layers of all models are taken as zero, and all initial center values and standard deviation value of the radial basis functions in all models are chosen as 0.1.

### A. XOR Classification

The Ex-OR gate can be considered as a classifier to which four possible inputs are applied, and two possible outputs are obtained. When the inputs are identical, the output shows class 1, and when the inputs are dissimilar, the output shows another class. The truth table of Ex-OR gate is viewed as a two-class problem. When the inputs are similar (0, 0 or 1, 1), the output is zero, and when these are dissimilar (0, 1 or 1, 0), the output is one. In the simulation study, "0" is taken as 0.1 and "1" is considered as 0.9 because to get absolute "0" or "1" is difficult. In this case, there are four feature sets, each containing two features. Fig. 7 represents the block diagram of the Ex-OR classifier with training provision using learning equations of each RBFNN model. The XOR classification task using the RBF algorithm is carried to find which model is best suited to provide higher accuracy and speed. For this example, model M-6 (which contains both sided weights and center and sigma update) is the best model in terms of both speed and accuracy of performance.

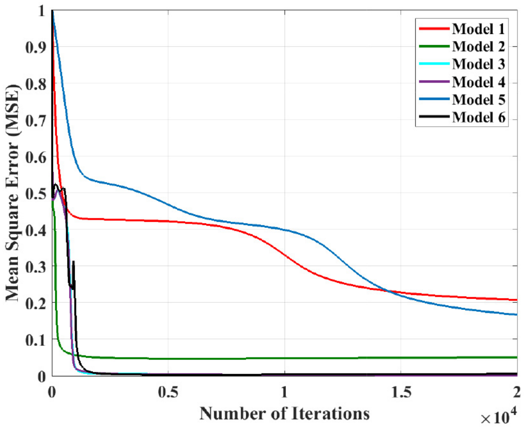

Fig. 8. Comparison of convergence characteristics of various models in classification problem (Ex. 1).

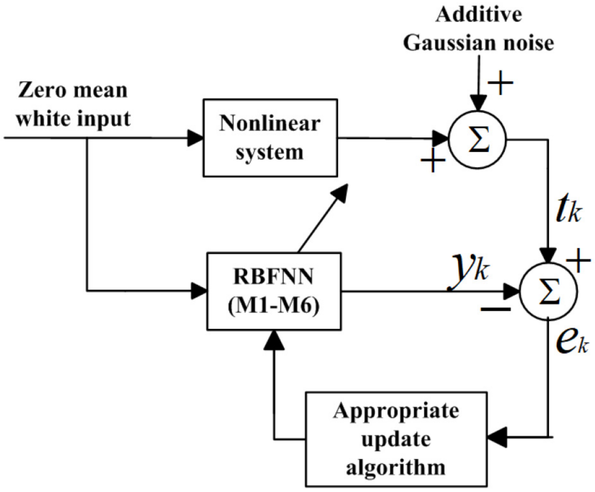

Fig. 9. Basic block diagram of implementation of nonlinear system identification problem (Ex-2).

### B. Nonlinear System Identification

Each of the RBFNN model is employed to develop a nonlinear system identifier using its corresponding learning rules. The basic block diagram to solve the system identification problem of Ex-2 is shown in Fig. 9. In this case, the same input is applied simultaneously to both the RBFNN model and the nonlinear system. The input applied is a zero-mean uniformly distributed random input, and the output of the nonlinear system is mixed with additive white Gaussian noise (AWGN) with an SNR of 30 dB. The outputs of both the system and model are compared to produce the error term $e_k$. The input considered is a zero-mean uniformly distributed white noise. The error value and the inputs are employed in the learning rules to update the internal parameters of each of the models. After successful completion of training, the performance of each of the models is compared and analyzed. The details of the linear and nonlinear parts of the system used for simulation are given in Table VII. The analysis of

TABLE V
SYSTEM SPECIFICATION OF SIMULATED PROBLEMS

| Problem | Type of Problem | Linear Part | Non-linear Part |
|---|---|---|---|
| Example 1 | Classification | - | - |
| Example 2 | System Identification (Direct Modelling) | $0.304z + 0.903z^{-1} + 0.304z^{-2}$ | $\tan(a(k))$ |
| Example 3 | Channel Equalization (Inverse Modelling) | $0.304z + 0.903z^{-1} + 0.304z^{-2}$ | $a(k) + 0.2a^2(k) - 0.1a^3(k)$ |

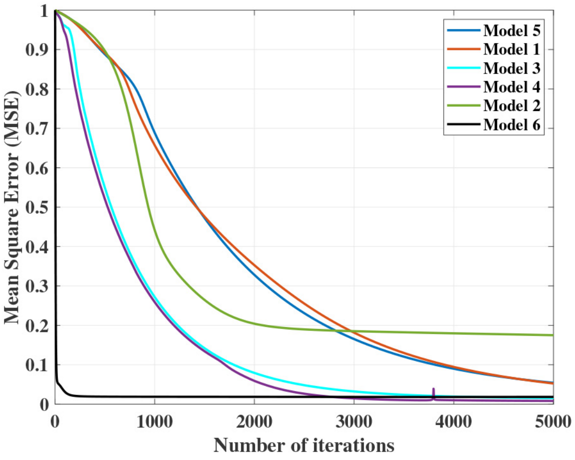

Fig. 10. Comparison of convergence characteristics of different models on system identification (Ex. 2).

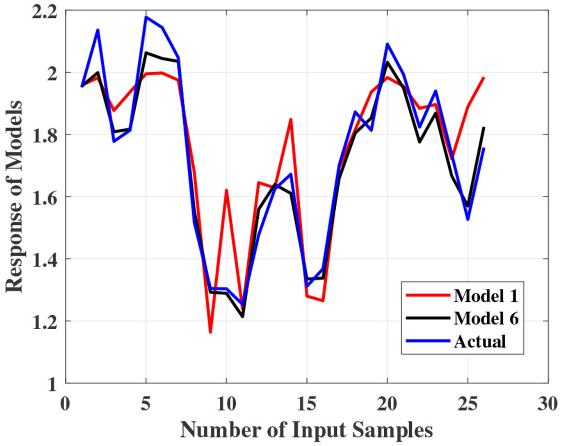

Fig. 11. Comparison of output response of different models on nonlinear system identification with input $e^{-x}$ (Ex. 2).

the results obtained from the simulation study is presented in the next section.

### C. Nonlinear Inverse Modeling Problem

The inverse model of the communication channel is developed using the proposed RBFNN models. The communication channel considered in this study is a nonlinear FIR channel presented in Table V. The block diagram of developing an inverse model is shown in Fig. 12. This diagram presents both the training and testing schemes. The input to this channel and RBFNN-based equalizers (M-1 to M-6) is a random binary sequence. The output of the channel is added with Gaussian white noise and the resultant sequence is passed to the RBFNN equalizer (M-1 to M-6) shown in Fig. 12. The error term $e_k$ produced after comparing the delayed binary sequence $t_k$ and the equalizer output is used in the update equations to adjust the parameters of the model. The order of the delay is usually a half order of the equalizer. After the MSE is reduced satisfactorily, the training phase becomes complete. Then, the testing of the equalizer is carried out by varying the SNR of the model. The training characteristics have been obtained for each model and the bit-error rates (BERs) of all models are obtained from the simulation study, and are plotted and compared with the models.

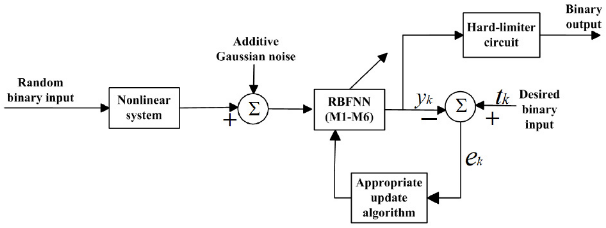

Fig. 12. Basic block diagram of implementation of nonlinear inverse model [adaptive channel equalizer (Ex-3)].

## VI. RESULTS AND DISCUSSION

In this section, the results of the simulation study of the proposed RBFNN model for three typical examples are presented. These examples are classification, direct modeling or nonlinear system identification, and inverse modeling or channel equalization. The various learning parameters used for the simulation study of models are presented in Table III. The performance measures used for comparison of the three examples are: MAD, RMSE, and the number of iterations to converge during the training phase of the six models. A learning algorithm is robust if the testing error is consistent with training error. This aspect has been studied during the simulation study of all four modified models. It is observed that all four models are mostly robust as these two errors for various runs are consistent in all three examples simulated. A learning algorithm is robust if the testing error is consistent with the training error. This aspect has been studied during the simulation study of all four modified models. It is shown that all the four modified models are observed to be robust as each of the models shows consistent residual error after training under different initial conditions. This is also true for all the three examples studied. The MAD is defined as

$$\mathrm{MAD} = \sum_{P=1}^{P} \frac{|\mathrm{AO} - \mathrm{MO}|}{P}. \tag{28}$$

The RMSE is defined as

$$\mathrm{RMSE} = \sqrt{\sum_{P=1}^{P} \frac{(\mathrm{AO} - \mathrm{MO})^2}{P}} \tag{29}$$

where AO = actual output and MO = model output, whereas $P$ = number of input patterns for EX-OR Classification problem/number of test samples used for system response plot. The convergence characteristics of the models for the classification task are presented in Fig. 8. For a classification problem,

TABLE VI
COMPARISON OF PERFORMANCE FOR CLASSIFICATION PROBLEM (EX. 1)

| Models | RMSE | MAD | Iterations to Converge | Assigned Rank |
|---|---|---|---|---|
| Model 1 | 0.2078 | 0.8903 | 3960 | 6 |
| Model 2 | 0.0508 | 0.1590 | 190 | 4 |
| Model 3 | 0.0020 | 0.1362 | 130 | 3 |
| Model 4 | 0.0017 | 0.0921 | 110 | 2 |
| Model 5 | 0.0581 | 0.2648 | 3110 | 5 |
| Model 6 | 0.0016 | 0.0303 | 70 | 1 |

TABLE VII
COMPARISON OF PERFORMANCE OF DIFFERENT MODELS OF NONLINEAR SYSTEM IDENTIFICATION PROBLEM (EX. 2)

| Models | RMSE | MAD | Iterations to Converge | Assigned Rank |
|---|---|---|---|---|
| Model 1 | 0.2308 | 1.8910 | 3452 | 6 |
| Model 2 | 0.0524 | 1.7903 | 1791 | 4 |
| Model 3 | 0.0153 | 0.5488 | 722 | 3 |
| Model 4 | 0.0081 | 0.3834 | 689 | 2 |
| Model 5 | 0.1750 | 1.8235 | 2085 | 5 |
| Model 6 | 0.0040 | 0.0538 | 42 | 1 |

the results presented in Table VI show that the proposed model M-6 yields the fastest (70 iterations) convergence whereas in the conventional model, M-1 takes the longest (3960) iterations to converge. This is also evident from the learning characteristics presented in Fig. 10. The MAD value is 0.0303, which is the lowest for M-6, whereas it is the highest for M-1. In terms of RMSE, M-6 offers the least value but this value is the highest, which is 0.2078 for M-1. Taking into consideration all the three performance indices, the ranking of the models is observed to be 6, 4, 3, 2, 5, and 1 for M-1, M-2, M-3, M-4, M-5, and M-6, respectively. It may be noted that the proposed modified RBFNN models M-6, M-4, and M-3 stand in positions 1–3. The second example implemented using all the six models is the identification of a nonlinear system. The convergence characteristics of these models obtained during the training phase of these models are presented in Fig. 10. The observation from this figure is presented in Table VII. The various output responses have been obtained in all cases. However, in this article, the output responses of M-1 (standard model) and M-6 (best model) are compared with the actual response presented in Fig. 11. By examining the values of MAD and RMSE, as well as the number of iterations to converge, the models (M-1 to M-6) are ranked as 6, 4, 3, 2, 5, and 1, respectively. This ranking of the models is identical to that obtained for Ex 1.

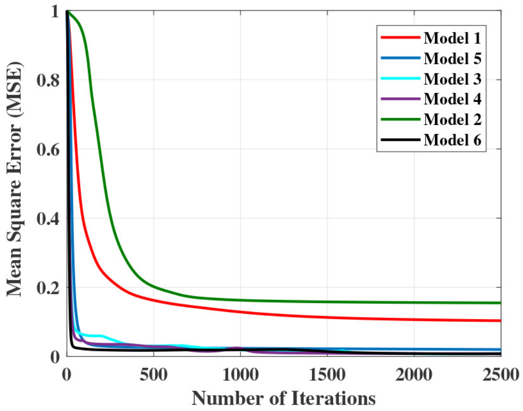

Fig. 13. Comparison of convergence characteristics of different models on channel equalization (Ex. 3).

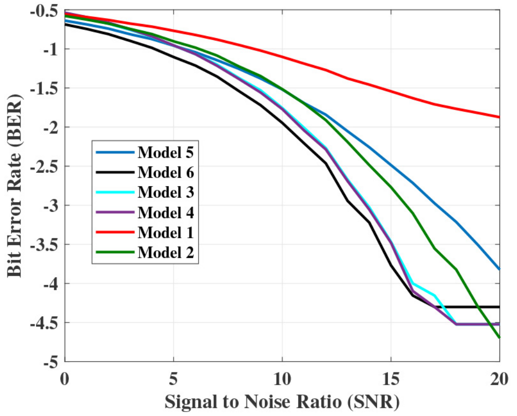

Fig. 14. Comparison of BER of different models for nonlinear channel equalization problem (Ex. 3).

TABLE VIII
COMPARISON OF PERFORMANCE OF DIFFERENT MODELS OF NONLINEAR CHANNEL EQUALIZATION PROBLEM (EX. 3)

| Models | RMSE | Iterations to Converge | Assigned Rank |
|---|---|---|---|
| Model 1 | 0.2176 | 394 | 6 |
| Model 2 | 0.0199 | 59 | 4 |
| Model 3 | 0.0092 | 36 | 3 |
| Model 4 | 0.0081 | 31 | 2 |
| Model 5 | 0.1519 | 214 | 5 |
| Model 6 | 0.0076 | 20 | 1 |

In the case of nonlinear inverse modeling (channel equalization), Ex. 3 is simulated by using all six models. In Fig. 13, the learning characteristics obtained from these models have been compared. The number of iterations to converge in each case is presented in Table VIII. The RMSE values are also computed and presented in the same table. It is observed from the table that the assigned ranks of the models remain consistent as have been obtained in the previous two examples. The BER graphs obtained during the testing phase of the inverse models are compared in Fig. 14. The observation of BER plots of Fig. 14 also demonstrates that the proposed M-6 model outperforms all other five RBFNN models. It is also noticed that in terms of the BER plot, the ranking of this model remains the same. It has been observed in the case of system identification and classification examples.

It is noticed from Table IX that the best performing modified model (M-6) requires the highest number of total operations. It needs additional operations of I to the number of exponential calculations, J(7I+K+4) number of multiplications, and J(I+K−1) number of additions per iteration compared to basic standard model M-1. The next highest number of additional operations is required by model M-4. The model M-5 needs

TABLE IX
COMPARISON OF THE NUMBER OF ADDITIONAL MATHEMATICAL OPERATIONS FOR EACH ITERATION DURING TRAINING PHASE (MODEL-1 IS TAKEN AS THE REFERENCE MODEL)

| RBFN Model | Input (Number of Exponentiation) | Number of Multiplication in Input layer | Number of Operations in Updating Input layer weights | | Number of Operations in Updating the Standard deviation | | Total Additional Operations | | | Rank Based on minimum number of additional Operations |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Multiplication | Addition | Multiplication | Addition | Exponentiation | Multiplication | Addition | |
| M-2 | - | - | - | - | JK+4J | J(K-1) | - | J(K+4) | J(K-1) | III |
| M-3 | - | JI | 6JI | JI | - | - | - | 7JI | JI | IV |
| M-4 | - | JI | 6JI | JI | JK+4J | J(K-1) | - | J(7I+K+4) | J(I+K-1) | V |
| M-5 | I | JI | - | - | - | - | I | - | - | II |
| M-6 | I | JI | 6JI | JI | JK+4J | J(K-1) | I | J(7I+K+4) | J(I+K-1) | VI |

TABLE X
COMPARISON OF EXECUTION TIME OF TWO CONVENTIONAL (M-1 AND M-2) AND PROPOSED MODIFIED (M-3–M-6) MODELS

| Existing/Proposed Models | Models | Execution Time in milliseconds (ms) per iteration during training phase | | | Ranking Based on minimum Execution Time |
|---|---|---|---|---|---|
| | | EX1 (Classification) | EX2 (Direct Modeling) | EX3 (Inverse Modeling) | |
| Existing | M-1 | 1.07 | 3.69 | 4.71 | I |
| | M-2 | 1.09 | 4.90 | 6.50 | III |
| Proposed Modified | M-3 | 1.10 | 5.20 | 6.90 | IV |
| | M-4 | 1.12 | 5.50 | 7.10 | V |
| | M-5 | 1.08 | 4.80 | 6.40 | II |
| | M-6 | 1.30 | 6.00 | 7.30 | VI |

only I number of exponential calculations, which is the minimum amongst all proposed models. Similar observations are noticed from Table X, which compares the execution time per single iteration during the training phases. A ranking of the models based on minimum time requirement has been made and presented in these two tables. The analysis of execution time indicates that the best performing model M-6 takes a maximum time of 7.3 ms and is ranked VI whereas the proposed model M-5 takes the least time for executing one iteration. But the range of execution time of modified four models (M-3 to M-6) lies in between 6.4 and 7.3 ms, which is small. Thus, in general, it can be concluded that the best two performing models M-6 and M-4 do not consume more execution time but offer better accuracy of performance and faster training.

The major contributions of the article are: it develops four modified RBFNN structures by introducing adaptive weights in the input layer in two models (M-3 and M-4) and by applying an exponential form of actual inputs, and then varying the weights of both layers in another two models (M-5 and M-6). The learning equations of associated weights of both layers, the center values and standard deviations of the radial basis functions located at the hidden nodes, are derived for each of the four modified models (M-3 to M-6). The four modified models (M-3 to M-6) can be employed in various applications, such as filtering, clustering, prediction, direct modeling, inverse modeling, control, and detection operations. But in this article, simulation results of only three standard applications, such as: 1) classification; 2) direct modeling (nonlinear system identification); and 3) inverse modeling (nonlinear channel equalization), are chosen. The various performance measures are evaluated for all models and all the applications. These key measures of all four models are compared along with corresponding measures of two standard RBFNN models (M-1 and M-2). Based on the superiority of performance measures, ranks are assigned to all the six RBFNN models (M-1 to M-6). The ranks stand as 3, 2, 5, and 1 for proposed models (M-3 to M-6). It is demonstrated that the ranks of all six models are consistent for the three standard applications studied. It is observed that all the proposed RBFNN models except model-5 (M-5) offer superior performance compared to the two standard models in terms of faster training time and accuracy offered.

Compared with the two conventional RBFNN models (M-1 and M-2), the proposed four modified RBFNN models have two distinct advantages. The first advantage is that the proposed models achieve a faster convergence rate during the training phase. The residual error after completion of the training phase is also lower in these models. The second advantage is the achievement of improved accuracy and better mean square error during the training phase in all three applications. This observation is consistent in terms of ranking assigned to the models.

## VII. CONCLUSION

In this article, four modified RBFNN models are presented, and the corresponding learning equations of various parameters of the models are derived. Also, the update equations of two conventional RBFNN models are presented for comparison purposes. To demonstrate the efficacy of the four models, simulation-based experiments of three standard applications have been carried out. The various performance indices are obtained from the results, and based on their potentialities of the models, appropriate rank of each model has been assigned. It is observed that the proposed models (M-1, M-4, and M-3) stand in ranks 1, 2, and 3 whereas the standard models M-1 and M-2 stand in ranks 6 and 4, respectively. Hence, it is, in general, shown that out of the proposed three modified RBFNN models offer superior performance compared to other existing RBFNN models 1 and 2. However, it is also observed that one of the new model M-5 occupies the fifth position but it performs better than its counterpart model M-1 because of its input exponential form with expected improved performance. The difficulty observed from the results of the three applications of the modified RBFNNs is that the performance of model M-5 with no provision of weights in the input layer but with the exponential version of the input is inferior to model-2 (M-2), but better than M-1. This is because no provision of weight update has been made in the input layer of M-5. Therefore, it shows that the newly introduced adaptive weights at the input layer have improved the learning rate and accuracy of performance.

These new models can also be applied for other different applications, such as prediction, detection, and filtering operations. In addition, to achieve further improved performance, the parameters, such as weights, center values, and standard deviation, of the Gaussian radial basis function can be optimized by employing suitable biologically inspired optimization techniques. As an extension of the work, other modified RBFNN models can be developed using the recursive least square (RLS) and Levenberg–Marquardt (LM) algorithm-based learning of weights, and the center values and the performance can be assessed and compared with the proposed modified RBFNN models. The two modified RBFNN models can be cascaded to provide better performance and faster training. Furthermore, the work can also be carried out using these proposed models for different control applications. The objective and focus of the current work are to develop modified improved RBFNN models, and to assess and compare their performances through few standard applications. Accordingly, three standard applications were chosen for comparison of the performance of proposed models. However, it was not objective of the paper to use the proposed models dedicatedly for nonlinear control applications, even though these models can be employed. For such purpose, out of three applications, we have chosen nonlinear system identification and nonlinear channel equalization to evaluate the performance measures of all models. The potentiality of all four modified RBFNN models for nonlinear control applications can be taken up as a separate work (as a future extension), and the performance obtained from these models can be compared with other recently reported nonlinear control applications.

## References

[1] C.-S. Leung, W. Y. Wan, and R. Feng, "A regularizer approach for RBF networks under the concurrent weight failure situation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 6, pp. 1360–1372, Jun. 2017.

[2] J. Raitoharju, S. Kiranyaz, and M. Gabbouj, "Training radial basis function neural networks for classification via class-specific clustering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 12, pp. 2458–2471, Dec. 2016.

[3] E. Bayro-Corrochano, E. Vazquez-Santacruz, E. Moya-Sanchez, and E. Castillo-Muñis, "Geometric bioinspired networks for recognition of 2-D and 3-D low-level structures and transformations," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 10, pp. 2020–2034, Oct. 2016.

[4] A. Alexandridis, E. Chondrodima, N. Giannopoulos, and H. Sarimveis, "A fast and efficient method for training categorical radial basis function networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2831–2836, Nov. 2017.

[5] C. Cecati, J. Kolbusz, P. Różycki, P. Siano, and B. M. Wilamowski, "A novel RBF training algorithm for short-term electric load forecasting and comparative studies," *IEEE Trans. Ind. Electron.*, vol. 62, no. 10, pp. 6519–6529, Oct. 2015.

[6] A. Alexandridis, E. Chondrodima, E. Efthimiou, G. Papadakis, F. Vallianatos, and D. Triantis, "Large earthquake occurrence estimation based on radial basis function neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 9, pp. 5443–5453, Sep. 2014.

[7] S. Mishra, R. N. Yadav, and R. P. Singh, "Directivity estimations for short dipole antenna arrays using radial basis function neural networks," *IEEE Antennas Wireless Propag. Lett.*, vol. 14, pp. 1219–1222, 2015.

[8] X. Meng, P. Rozycki, J.-F. Qiao, and B. M. Wilamowski, "Nonlinear system modeling using RBF networks for industrial application," *IEEE Trans. Ind. Informat.*, vol. 14, no. 3, pp. 931–940, Mar. 2018.

[9] I. Cha and S. A. Kassam, "Channel equalization using adaptive complex radial basis function networks," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 1, pp. 122–131, Jan. 1995.

[10] H. Jafarnejadsani, J. Pieper, and J. Ehlers, "Adaptive control of a variable-speed variable-pitch wind turbine using radial-basis function neural network," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 6, pp. 2264–2272, Nov. 2013.

[11] W. Kaminski and P. Strumillo, "Kernel orthonormalization in radial basis function neural networks," *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 1177–1183, Sep. 1997.

[12] H. Yu, P. D. Reiner, T. Xie, T. Bartczak, and B. M. Wilamowski, "An incremental design of radial basis function networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 10, pp. 1793–1803, Oct. 2014.

[13] A. O. Hoori and Y. Motai, "Multicolumn RBF network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 4, pp. 766–778, Apr. 2018.

[14] C. Panchapakesan, M. Palaniswami, D. Ralph, and C. Manzie, "Effects of moving the center's in an RBF network," *IEEE Trans. Neural Netw.*, vol. 13, no. 6, pp. 1299–1307, Nov. 2002.

[15] L. Bruzzone and D. F. Prieto, "A technique for the selection of kernel-function parameters in RBF neural networks for classification of remote-sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 37, no. 2, pp. 1179–1184, Mar. 1999.

[16] K. Z. Mao and G.-B. Huang, "Neuron selection for RBF neural network classifier based on data structure preserving criterion," *IEEE Trans. Neural Netw.*, vol. 16, no. 6, pp. 1531–1540, Nov. 2005.

[17] A. G. Bors and I. Pitas, "Median radial basis function neural network," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1351–1364, Nov. 1996.

[18] T. Xie, H. Yu, J. Hewlett, P. Rózycki, and B. Wilamowski, "Fast and efficient second-order method for training radial basis function networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 4, pp. 609–619, Apr. 2012.

[19] Y.-J. Oyang, S.-C. Hwang, Y.-Y. Ou, C.-Y. Chen, and Z.-W. Chen, "Data classification with radial basis function networks based on a novel kernel density estimation algorithm," *IEEE Trans. Neural Netw.*, vol. 16, no. 1, pp. 225–236, Jan. 2005.

[20] A. Alexandridis, E. Chondrodima, and H. Sarimveis, "Radial basis function network training using a nonsymmetric partition of the input space and particle swarm optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 2, pp. 219–230, Feb. 2013.

[21] S. Chen, C. F. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 302–309, Mar. 1991.

[22] X. Hong, "A fast identification algorithm for box–Cox transformation based radial basis function neural network," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 1064–1069, Jul. 2006.

[23] S. Chen, X. Hong, B. L. Luk, and C. J. Harris, "Construction of tunable radial basis function networks using orthogonal forward selection," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 457–466, Apr. 2009.

[24] Y. W. Wong, K. P. Seng, and L.-M. Ang, "Radial basis function neural network with incremental learning for face recognition," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 41, no. 4, pp. 940–949, Aug. 2011.

[25] M. Hou and X. Han, "Constructive approximation to multivariate function by decay RBF neural network," *IEEE Trans. Neural Netw.*, vol. 21, no. 9, pp. 1517–1523, Sep. 2010.

[26] L. Yingwei, N. Sundararajan, and P. Saratchandran, "A sequential learning scheme for function approximation using minimal radial basis function neural networks," *Neural Comput.*, vol. 9, no. 2, pp. 461–478, 1997.

[27] A. Leonardis and H. Bischof, "An efficient MDL-based construction of RBF networks," *Neural Netw.*, vol. 11, no. 5, pp. 963–973, 1998.

[28] H. Sarimveis, A. Alexandridis, G. Tsekouras, and G. Bafas, "A fast and efficient algorithm for training radial basis function neural networks based on a fuzzy partition of the input space," *Ind. Eng. Chem. Res.*, vol. 41, no. 4, pp. 751–759, 2002.

[29] A. Alexandridis, H. Sarimveis, and G. Bafas, "A new algorithm for online structure and parameter adaptation of RBF networks," *Neural Netw.*, vol. 16, no. 7, pp. 1003–1017, 2003.

[30] A. Alexandridis, H. Sarimveis, and K. Ninos, "A radial basis function network training algorithm using a non-symmetric partition of the input space–application to a model predictive control configuration," *Adv. Eng. Softw.*, vol. 42, no. 10, pp. 830–837, 2011.

[31] W. Yao, X. Chen, Y. Zhao, and M. van Tooren, "Concurrent subspace width optimization method for RBF neural network modeling," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 2, pp. 247–259, Feb. 2012.

[32] S.-F. Su, C.-C. Chuang, C.-W. Tao, J.-T. Jeng, and C.-C. Hsiao, "Radial basis function networks with linear interval regression weights for symbolic interval data," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 42, no. 1, pp. 69–80, Feb. 2012.

[33] D. Zhang, L.-F. Deng, K.-Y. Cai, and A. So, "Fuzzy nonlinear regression with fuzzified radial basis function network," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 6, pp. 742–760, Dec. 2005.

[34] Y. Lei, L. Ding, and W. Zhang, "Generalization performance of radial basis function networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 3, pp. 551–564, Mar. 2015.

[35] G. S. Babu and S. Suresh, "Sequential projection-based metacognitive learning in a radial basis function network for classification problems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 2, pp. 194–206, Feb. 2013.

[36] G.-B. Huang, P. Saratchandran, and N. Sundararajan, "An efficient sequential learning algorithm for growing and pruning RBF (GAP-RBF) networks," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 34, no. 6, pp. 2284–2292, Dec. 2004.

[37] R. Savitha, S. Suresh, and N. Sundararajan, "Metacognitive learning in a fully complex-valued radial basis function neural network," *Neural Comput.*, vol. 24, no. 5, pp. 1297–1328, 2012.

[38] C.-B. Cheng and E. S. Lee, "Fuzzy regression with radial basis function network," *Fuzzy Sets Syst.*, vol. 119, no. 2, pp. 291–301, 2001.

[39] R. Langari, L. Wang, and J. Yen, "Radial basis function networks, regression weights, and the expectation-maximization algorithm," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 27, no. 5, pp. 613–623, Sep. 1997.

[40] M. Gan, H.-X. Li, and H. Peng, "A variable projection approach for efficient estimation of RBF-ARX model," *IEEE Trans. Cybern.*, vol. 45, no. 3, pp. 462–471, Mar. 2015.

[41] H.-G. Han, W. Lu, Y. Hou, and J.-F. Qiao, "An adaptive-PSO-based self-organizing RBF neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 1, pp. 104–117, Jan. 2018.

[42] A. Rawat, R. Yadav, and S. Shrivastava, "Neural modeling of antenna array using radial basis neural network for directivity prediction," *Int. J. Model. Optim.*, vol. 3, no. 1, pp. 95–97, 2013.

[43] A. H. El Zooghby, C. G. Christodoulou, and M. Georgiopoulos, "Performance of radial-basis function networks for direction of arrival estimation with antenna arrays," *IEEE Trans. Antennas Propag.*, vol. 45, no. 11, pp. 1611–1617, Nov. 1997.

[44] B.-H. Chen, S.-C. Huang, C.-Y. Li, and S.-Y. Kuo, "Haze removal using radial basis function networks for visibility restoration applications," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3828–3838, Aug. 2018.

[45] S.-B. Roh, S.-K. Oh, and W. Pedrycz, "Identification of black plastics based on fuzzy RBF neural networks: Focused on data preprocessing techniques through Fourier transform infrared radiation," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 1802–1813, May 2018.

[46] Z.-R. Lai, D.-Q. Dai, C.-X. Ren, and K.-K. Huang, "Radial basis functions with adaptive input and composite trend representation for portfolio selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6214–6226, Dec. 2018.

[47] Y. Yin *et al.*, "Observer-based adaptive sliding mode control of NPC converters: An RBF neural network approach," *IEEE Trans. Power Electron.*, vol. 34, no. 4, pp. 3831–3841, Apr. 2019.

[48] M. A. Mayosky and I. Cancelo, "Direct adaptive control of wind energy conversion systems using Gaussian networks," *IEEE Trans. Neural Netw.*, vol. 10, no. 4, pp. 898–906, Jul. 1999.

[49] R. Segal, M. Kothari, and S. Madnani, "Radial basis function (RBF) network adaptive power system stabilizer," *IEEE Trans. Power Syst.*, vol. 15, no. 2, pp. 722–727, May 2000.

[50] J. A. Leonard and M. A. Kramer, "Radial basis function networks for classifying process faults," *IEEE Control Syst. Mag.*, vol. 11, no. 3, pp. 31–38, Apr. 1991.

[51] M. Amirian and F. Schwenker, "Radial basis function networks for convolutional neural networks to learn similarity distance metric and improve interpretability," *IEEE Access*, vol. 6, pp. 123087–123097, 2020.

[52] H. Wang, Z. Shi, H. T. Wong, C.-S. Leung, H. C. So, and R. Feng, "An $l_0$-norm-based centers selection for failure tolerant RBF networks," *IEEE Access*, vol. 7, pp. 151902–151914, 2019.

[53] R. Guo, Y. Li, L. Zhao, J. Zhao, and D. Gao, "Remaining useful life prediction based on the Bayesian regularized radial basis function neural network for an external gear pump," *IEEE Access*, vol. 8, pp. 107498–107509, 2020.

[54] H. Y. He, D. Luo, W.-J. Lee, Z. Zhang, Y. Cao, and T. Lu, "A contactless insulator contamination levels detecting method based on infrared images features and RBFNN," *IEEE Trans. Ind. Appl.*, vol. 55, no. 3, pp. 2455–2463, May/Jun. 2018.

**Sashmita Panda** (Student Member, IEEE) received the B.Tech. degree in electronics and communication engineering from Sanjay Memorial Institute of Technology, Berhampur, India, in 2009, and the M.Tech. degree in electrical engineering from the National Institute of Technology Rourkela, Rourkela, India, in 2015. She is currently pursuing the Ph.D. degree with G. S. Sanyal School of Telecommunications, Indian Institute of Technology Kharagpur, Kharagpur, India.

From 2009 to 2013, she was a Lecturer with the Roland Institute of Technology, Berhampur. From 2015 to 2016, she was an Assistant Professor with KL University, Vijayawada, India. Her research interests include biological neural communications and artificial neural networks.

**Ganapati Panda** (Senior Member, IEEE) received the Ph.D. degree in electronics and communication engineering from the Indian Institute of Technology (IIT) Kharagpur, Kharagpur, India, in 1982.

He was a Postdoctoral Researcher with the University of Edinburgh, Edinburgh, U.K., from 1984 to 1986. He is currently working as a Research Advisor with C. V. Raman College of Engineering, Bhubaneswar, India. He is also a Professorial Fellow with the Department of Electrical Sciences, IIT Bhubaneswar, Bhubaneswar, India. He is superannuated as a Senior Professor and a Deputy Director of IIT Bhubaneswar. He is currently a Distinguished Professor of AICTE-INAE of India. He has supervised 42 Ph.D. students. His research interests include digital signal processing, digital communication, and machine learning.

Prof. Panda received the Samanta Chandra Sekhar Award and the Biju Pattnaik Award from the Government of Odisha. He has been awarded the Outstanding Teacher Award by INAE in 2017 for outstanding contribution to engineering, education, and research. He is a Fellow of the National Academy of Engineering India and the National Academy of Science India.