A TWO-STAGE LEARNING METHOD TO CONFIGURE RBF CENTERS AND WIDTHS IN DYNAMIC ENVIRONMENT EMPLOYING IMMUNE OPERATIONS

Xiaofeng Ling, Xinbao Gong, Xiaogang Zang, Ronghong Jin[*]

Abstract: This paper proposes an immunity-based RBF training algorithm for nonlinear dynamic problems. Exploiting the locally-tuned structure of RBF network through immunological metaphor, a two-stage learning technique is proposed to configure RBF centers and widths in the hidden layer. Inspired by affinity maturation process of immune response, immune evolutionary mechanism (IEM) with memory operations is implemented in the learning stages to dynamically fine-tune the network performance. Experiment results also demonstrate that the algorithm has reached good performance with relatively low computational efforts in dynamic environments.

Key words: *RBF neural network, dynamic problems, immune evolutionary mechanism*

Received: December 30, 2010
Revised and accepted: August 8, 2011

1. Introduction

As a kind of typical feedforward artificial neural network, Radial Basis Function (RBF) networks consist of neurons which are locally tuned. Due to their simple structure and powerful approximation ability, RBF networks have been widely used in various fields [1]. In recent years, RBF networks have also found applications in nonlinear dynamic problems, and many RBF online training algorithms are proposed [2,3,4,5].

It is justifiable that the key issue in configuring RBF networks is the determination of the centers and widths in hidden units. There is also a tradeoff between the ability of an RBF network to learn a desired input-output mapping and its ability to generalize [6]. However, many RBF online training algorithms tend to

[*]Xiaofeng Ling, Xinbao Gong, Xiaogang Zang, Ronghong Jin
Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai 200240, China, xgzang@163.com

produce a local optimal network with excessive number of hidden units, degrading their ability to generalize.

With their promising ability to search the global optimum in parameter space, evolutionary algorithms (EAs) can produce a compact RBF network [7], and have been employed in dynamic optimization problems [8,9,10]. However, EA training strategy of RBF networks often incurs extensive computational requirements, making it especially impractical in dynamic environments, where parameters of RBF networks require to be continually adjusted according to changes in the training set.

With the properties of being adaptive, dynamic, diverse, inherently distributed, highly robust and self-organized, natural immune system has been used as a rich source of inspiration to solve complex engineering problems [11]. Algorithms incorporating immune operations have also been developed to design RBF networks. In some RBF training algorithms, immune clustering techniques [12] are incorporated to initialize the centers of RBF networks [13,14]. As a kind of improved evolutionary algorithms, immune algorithms [15,16] have also been implemented to design the whole RBF network [17,18]. However, these immunity-based RBF training algorithms did not fully exploit the locally-tuned nature of RBF network, and often treated the RBF network as a whole. That will greatly increase the computational overhead in dynamic environments.

Inspired by natural immunology, this paper presents a two-stage learning technique incorporating immune operations to configure RBF centers and widths in dynamic environment. Based on the analogy drawn between natural immune system and RBF neural network, the proposed algorithm incorporates immunological ideas that are suitable for dynamic problem solving. Compared to other immunity-based RBF training algorithms, the proposed algorithm fully exploits the locally-tuned nature of RBF network, and avoids the starting-from-scratch problem of nonlinear learning models through immune operations, such as immune evolutionary mechanism (IEM) and immune repertoire. In the algorithm, the RBF hidden units are determined in different learning stages with different adjusting frequencies, corresponding to the stationary part and perturbation part of the training set. This two-stage learning method also makes a good balance between network precision and generalization. The experimental results also demonstrate that the algorithm is a promising method for dynamic problems solving.

The paper is organized as follows. Section 2 reviews the fundamental knowledge of natural immune system and RBF neural network, and draws analogy between immune system and our proposed algorithm. Section 3 describes the algorithm in detail. In Section 4, the proposed algorithm is applied to dynamic problems, and the results are compared with those obtained by other algorithms. Conclusions are presented in Section 5.

2. Background of the Algorithm

2.1 Immune system overview

The immune system defends the body against an ever-changing cast of antigens. In order to succeed, it must perform pattern recognition tasks to distinguish molecules and cells of the body (called "self") from foreign ones (called "nonself") [19]. Natural immune system has many useful mechanisms from the viewpoint of information processing. Among these we focus on the mechanisms that guide the design of our algorithm.

1. ***Multi-layered:*** In natural immune system, multiple layers of different mechanisms are combined to provide high overall security. The immune system is composed of innate immune system and adaptive immune system. Innate immune system remains constant during one's lifetime, and can handle most common pathogens in the body. On the other hand, adaptive immune system is adaptively adjusted according to the environments and is more specific to corresponding antigens, eliminating new antigens as they appear.
2. ***Distributed:*** The adaptive immune system can be viewed as a highly distributed detection system which consists primarily of lymphocytes. Detection of nonself occurs when molecular bonds are formed between a pathogen and receptors that cover the surface of the lymphocyte. Each lymphocyte can bind with several different kinds of (structurally related) pathogens, thus covering a small part of antigen space. These lymphocytes function as small independent detectors that interact locally to provide global protection.
3. ***Dynamic:*** Lymphocytes are continually created, destroyed and circulated throughout the body, thus forming a dynamically changing coverage of the space of possible antigens.
4. ***Adaptable:*** The immune system can learn to recognize and respond to new antigens through adaptive immune response. If the immune system detects a pathogen that it has not encountered before, it evolves a set of lymphocytes with high affinity for that pathogen through a process called affinity maturation. These adapted lymphocytes are then retained as memory cells to facilitate future responses. On subsequent encounters with the same antigen pattern, the immune system mounts a secondary response, in which the memory cells are re-stimulated and quickly reproduced in great numbers. In this case, the pathogens are eliminated so rapidly that the symptoms of the infection are not noticeable by the individual.

2.2 RBF network

Typical RBF network has a feedforward network structure with a single layer of locally-tuned hidden units. Through linear combination of responses of these localized hidden units RBF network achieves universal approximation from given input-output patterns and produces a sparse coverage of all the pairs of the training set. For an RBF network with p-dimensional input vector and a scalar output, assume that a training set of L samples is available, where $\mathbf{Y} = \{y_i | i = 1, \ldots, L\}$ is the desired network output corresponding to the network input $\mathbf{X} = \{\mathbf{x}_i | i = 1, \ldots, L, \mathbf{x}_i \subset R^p\}$. The actual output of RBF network can be computed by:

$$\hat{y}_i = \sum_{k=1}^{N} w_k \exp(-\frac{1}{\sigma_k^2} \|\mathbf{x}_i - \mathbf{c}_k\|^2), \tag{1}$$

where $\mathbf{c}_k$ and σ_k denote the center and width of each RBF hidden unit, respectively; N is the number of centers; w_k is the weight associated with the kth function.

2.3 Analogy between immune system and proposed algorithm

RBF network has a locally-tuned architecture in which only a part of the nodes are affected by any given input [20]. So, during the training phase in dynamic environment, when the input-output pattern changes, only a portion of the model parameters may need to be adjusted, thus reducing the computational overhead.

Similar locally-tuned property can be found in immune system, although the immune system is much more complicated in architecture and behavior. Furthermore, by the continual circulation of lymphocytes through the body and a continual turnover of the lymphocyte population, immune system also provides a dynamically changing coverage of the space of antigen patterns.

Based on the locally-tuned architecture of RBF network similar to immune system, in our proposed algorithm RBF hidden units are treated as lymphocytes in immune system, and immunity-based strategies are implemented dynamically to fine-tune the parameters of RBF network according to changes in the training set. These immunity-based strategies include:

1) ***Two-stage learning:*** RBF hidden layer is adjusted through two stages, corresponding to innate immune response and adaptive immune response, respectively. In first-stage learning, some RBF hidden units are determined for overall generalization; while in second-stage learning, new hidden units are added and replaced according to those training data with large errors, thus improving the precision of the network to current training set.

2) ***Immune evolution:*** corresponding to affinity maturation process in adaptive immune response, immune evolutionary mechanism (IEM) is proposed to adjust the centers and widths of RBF network.

3) ***Memory repertoire:*** inspired by immune memory mechanism, RBF hidden units are extracted as memory cells to form the memory repertoire, which helps to avoid the starting-from-scratch problem in dynamic environments.

The analogy drawn between immune system and our proposed algorithm is shown in Tab. I.

3. The Proposed Algorithm

Flowchart of the proposed algorithm is depicted in Fig. 1, and the procedure for the algorithm can be described in Tab. II. Parameters in Tab. II are denoted as follows:

MSE: mean squared error (MSE) of RBF network for current training set. It is defined as:

$$MSE = \frac{1}{L}\sum_{i=1}^{L}(y_i - \hat{y}_i)^2 \qquad (2)$$

$\mathbf{w}$: $\mathbf{w} = [w_1, \ldots, w_N]^T$, RBF output weight vector.

T_{min}: threshold to determine whether RBF network needs adjustment. T_{min} reflects the desired performance of RBF network. A smaller T_{min} implies better

Prototype of Immune Mechanism	Analogy in Proposed Algorithm
Immune System	RBF Network
Innate Immune Response	First-Stage Learning
Adaptive Immune Response	Second-Stage Learning
Antigen	Training Sample
Antigen Pattern Space	Training Set
Lymphocyte	RBF Hidden Unit
Simulation Field of Lymphocyte	Width of RBF Hidden Unit
Concentration of Lymphocytes	RBF Output Weights
Cross-Reactive Response	Generalization of RBF Network
Immune Memory	Memory Repertoire
Affinity Maturation	Immune Evolutionary Mechanism

Tab. I *Analogy between immune system and proposed algorithm.*

precision of RBF network on current training set, but would incur greater computation efforts since RBF hidden layer has to be adjusted more frequently. To achieve better compromise between network performance and computation loads, T_{min} can be adjusted according to the training frequency of RBF hidden layer: if frequency of $MSE > T_{min}$ is relatively low in past detection phases, T_{min} can be decreased to achieve better network precision; on the contrary, T_{min} should be increased to reduce training frequency.

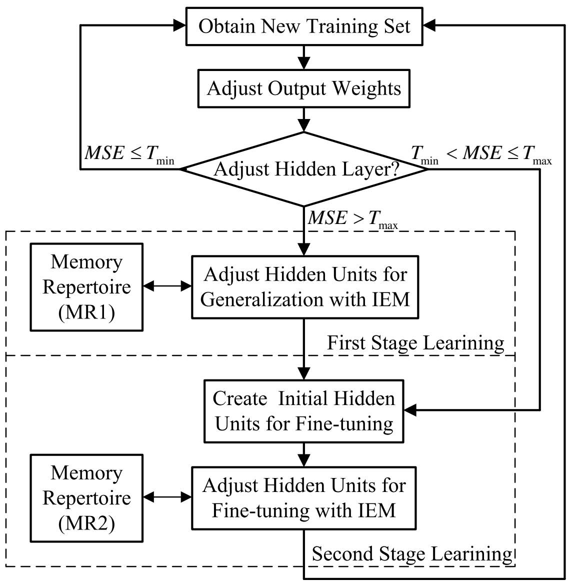

Fig. 1 *Flowchart of the proposed algorithm in dynamic environment.*

```
Begin
    %Step1: Detect changes in training set
    if $MSE \le T_{min}$, then return;
    else $\mathbf{w} = LS(\mathbf{X}, \mathbf{Y}, \mathbf{V}_p)$
    if $MSE \le T_{min}$, then return;
    else if $MSE > T_{max}$, go to step2;
    else go to step3;

    %Step2: First-Stage Learning
    $\mathbf{V}_p = \mathbf{V}_1$;
    $pop\mathbf{V}_1(0) = pop\mathbf{V}_1(S_1)$;
    Gen=1;
    repeat
        $\mathbf{V}_1 = \max\{affinity(pop\mathbf{V}_1(\text{Gen}))\}$
        $\text{MR1} = Update(\text{MR1}, \mathbf{V}_1)$
        $pop\mathbf{V}_1(\text{Gen}) = Clonal\_Selection(pop\mathbf{V}_1(\text{Gen}))$
        $pop\mathbf{V}_1(\text{Gen}) = Clonal\_Expansion(pop\mathbf{V}_1(\text{Gen}))$
        $pop\mathbf{V}_1(\text{Gen}) = Receptor\_Editing(pop\mathbf{V}_1(\text{Gen}), \text{MR1})$
        Gen=Gen+1
    until Gen $> S_1$

    %Step3: Second-Stage Learning
    $\mathbf{X}_m = \{\mathbf{x}_i | |y_i - \hat{y}_i| > \xi_1, i = 1, \ldots, L\}$
    $\mathbf{V}_2 = Clustering(\mathbf{X}_m)$
    $\mathbf{V}_p = [\mathbf{V}_1; \mathbf{V}_2]$;
    $pop\mathbf{V}_2(0) = Clonal\_Expansion(\mathbf{V}_p)$;
    Gen=1;
    repeat
        $\mathbf{V}_2 = \max\{affinity(pop\mathbf{V}_2(\text{Gen}))\}$
        $\text{MR2} = Update(\text{MR2}, \mathbf{V}_2)$
        $pop\mathbf{V}_2(\text{Gen}) = Clonal\_Selection(pop\mathbf{V}_2(\text{Gen}))$
        $pop\mathbf{V}_2(\text{Gen}) = Clonal\_Expansion(pop\mathbf{V}_2(\text{Gen}))$
        $pop\mathbf{V}_2(\text{Gen}) = Receptor\_Editing(pop\mathbf{V}_2(\text{Gen}), \text{MR2})$
        Gen=Gen+1
    until Gen $> S_2$ or $affinity(\mathbf{V}_p) > \varsigma$
    return;
    end if;
end
```

Tab. II *Pseudo code for the proposed algorithm.*

T_{max}: threshold to determine whether RBF network needs first-stage learning. T_{max} influences the frequency of first-stage learning. If $T_{max} \to T_{min}$, probability of first-stage learning tends to 1, i.e., every RBF hidden unit has to be adjusted; that would often incur additional computation loads. If $T_{max} \to \infty$, probability of first-stage learning tends to 0; that would reduce the adaptability of the algorithm to dynamic environments. In practice, $T_{max} \in [2T_{min}, 10T_{min}]$ is acceptable. Similar to T_{min}, T_{max} can be also adaptively adjusted according to the training frequency of first-stage learning.

$\mathbf{V}_1$: $\mathbf{V}_1 = [\mathbf{c}_k, \sigma_k]_{k=1}^{M}$, $M(M < N)$ hidden units determined in first-stage learning.

$\mathbf{V}_2$: $\mathbf{V}_2 = [\mathbf{c}_k, \sigma_k]_{k=M+1}^{N}$, $N-M$ hidden units determined in second-stage learning.

$\mathbf{V}_p$: RBF hidden layer. In first-stage learning $\mathbf{V}_p = \mathbf{V}_1 = [\mathbf{c}_k, \sigma_k]_{k=1}^{M}$, while in second-stage learning $\mathbf{V}_p = [\mathbf{V}_1; \mathbf{V}_2] = [\mathbf{c}_k, \sigma_k]_{k=1}^{N}$.

$pop\mathbf{V}_1$: $pop\mathbf{V}_1 = \{[\mathbf{V}_1]_j | j = 1, \ldots, n\}$, population of RBF hidden layers in first-stage learning.

$pop\mathbf{V}_2$: $pop\mathbf{V}_2 = \{[\mathbf{V}_1; \mathbf{V}_2]_j | j = 1, \ldots, n\}$, population of RBF hidden layers in second-stage learning.

MR1: memory repertoire for first-stage learning.

MR2: memory repertoire for second-stage learning.

S_1: maximum number of iterations in first-stage learning.

S_2: maximum number of iterations in second-stage learning.

$\mathbf{X}_m$: input pattern in current training set that needs further approximation. The size of $\mathbf{X}_m$ is determined by the dynamic environment that is reflected in the training set. The size of $\mathbf{X}_m$ is also relevant to ξ_1: for a specific training set, the smaller ξ_1, the more centers would be obtained for $\mathbf{X}_m$.

ξ_1: threshold to determine new pattern in current training set. In practice, ξ_1 can be set between $[1.5\sqrt{T_{min}}, 3\sqrt{T_{min}}]$.

ς: desired affinity for termination in second-stage learning.

Main steps of the proposed algorithm can be described as follows:

1) **Detect changes in current training set:** In this step, RBF output weight vector $\mathbf{w}$ is first adjusted with linear least squares (LS) method according to new training set in dynamic environment. Then, calculate MSE of RBF network for current training set. Preset threshold T_{min} and T_{max} are defined to test whether RBF hidden layer needs further adjustment.

2) **Adjust hidden units for generalization:** RBF hidden units are decomposed as $\mathbf{V}_1$ and $\mathbf{V}_2$. In first-stage learning $\mathbf{V}_1$ are updated with immune evolutionary mechanism (IEM) to produce a compact RBF network for generalization.

3) **Create initial hidden units for fine-tuning:** In second-stage learning initial $\mathbf{V}_2$ are generated to cover those training data with large errors denoted as $\mathbf{X}_m$. $\mathbf{X}_m$ reflects new input pattern in current training set that need further approximation. Initial $\mathbf{V}_2$ is produced from $\mathbf{X}_m$ using k-means clustering algorithm [21].

4) **Adjust hidden units for fine-tuning:** $\mathbf{V}_1$ and $\mathbf{V}_2$ are concatenated to form a prototype RBF hidden layer $\mathbf{V}_p$; and then IEMs are performed to adjust $\mathbf{V}_2$, while $\mathbf{V}_1$ remains constant during the process. After second-stage learning, go back to step 1.

3.1 Immune evolutionary mechanism for adjusting RBF hidden layers

In the first- and second-stage learning, immune evolutionary mechanism (IEM) is proposed to adjust $\mathbf{V}_1$ and $\mathbf{V}_2$, respectively. IEM corresponds to affinity maturation in natural immune system, a process very similar to adaptive biological evolution. In IEM algorithm, RBF hidden units are treated as "lymphocytes", and extracted as memory cells to form the memory repertoire (MR). IEM algorithm can be described as follows, where n is the population size, m is the size of memory repertoire, and P is the number of units for receptor editing:

1) **Create initial population:** In first-stage learning, the initial population of $\mathbf{V}_1$ comes from the last population in previous training epoch. In second-stage learning, the initial population of $\mathbf{V}$ comes from prototype $\mathbf{V}_p$ by clonal expansion, which is composed of reproduction of $\mathbf{V}_p$ and mutation on $\mathbf{V}_2$.

2) **Evaluate affinity:** Each RBF hidden layer in current population is decoded to form an RBF network, and corresponding output weights $\mathbf{w}$ is determined by LS method. Corresponding to antigenic affinity between antigen and lymphocyte in natural immune system, affinity is introduced here to evaluate the performance of RBF network, and is defined as:

$$affinity = \frac{1}{MSE} = \frac{1}{\frac{1}{L}\sum_{i=1}^{L}(y_i - \hat{y}_i)^2} \tag{3}$$

3) **Update memory repertoire:** Hidden units of the RBF hidden layer that has highest affinity are extracted to update memory repertoire (MR). Old memory cells in MR that have the highest similarities with these new hidden units are replaced. Here the similarity between hidden units is defined according to Euclidean distance:

$$similarity([\mathbf{c}_i, \sigma_i], [\mathbf{c}_j, \sigma_j]) = \frac{1}{\|\mathbf{c}_i - \mathbf{c}_j\| + |\sigma_i - \sigma_j|} \tag{4}$$

4) **Clonal selection:** Select k highest affinity RBF hidden layers from current population for further operation.

5) **Clonal expansion:** This process is composed of reproduction and mutation. For each RBF hidden layer, the reproduction rate is proportional to its affinity, and mutation rate is inversely proportional to its affinity.

6) **Receptor editing:** "Receptor editing" is an idea borrowed from natural immunology [22]. Here it means the operation of substituting certain hidden units in RBF hidden layer with those in the memory repertoire, providing the possibility for performance improvement. Randomly selected P memory cells from MR and P hidden layers in current population. Each selected memory cell is added in corresponding hidden layer, and replaces the hidden unit that has the highest similarity with it. If the affinity of corresponding hidden layer has improved, the operation of receptor editing is confirmed.

7) **Termination:** In first-stage learning, stopping criterion is predefined maximum number of iterations S_1; in second-stage learning, stopping criterion is met when desired affinity ς or maximum number of iterations S_2 has been reached.

3.2 Discussion

Our proposed algorithm is suitable for the training of RBF network in dynamic environments for the following reasons:

1) The adjustment of linear output weights is performed before the adjustment of nonlinear parameters in hidden layers, i.e., different type of parameters are adjusted by different methods with different frequencies. This method can improve the efficiency of the training algorithm.

2) The two-stage learning of RBF hidden units corresponds to stationary part and perturbation part in dynamic environments, respectively. This kind of partition reduces the parameter optimization space. Furthermore, second-stage learning is adjusted according to the changing part of the training set, and is performed more frequently than first-stage learning, thus reducing the computation load in dynamic environments.

3) IEM has global optimization abilities similar to evolutionary algorithms, which guarantees the performance of the algorithm. Immune memory operation helps to avoid the starting-from-scratch problem, thus improving the convergence speed and adaptability of the algorithm for dynamic problems.

4. Experiments

Our proposed method was applied in dynamic function approximation and channel estimation of orthogonal frequency division multiplexing (OFDM) communication systems to test its performance in dynamic environments.

4.1 Dynamic function approximation

Multimodal function approximation has been widely used to assess the performance of neural networks. In our experiment, the multimodal function used to evaluate the performance of the proposed algorithm can be expressed as follows:

$$\begin{aligned} z = f(x,y) &= 3(1-x+a)^2 e^{-x^2/2-(y+1)^2-a} \\ &\quad - (10+b)(\frac{x}{5} - x^3 - y^5) e^{-x^2-y^2+c} - \frac{1}{3} e^{-(x+1)^2-y^2}, \end{aligned} \tag{5}$$

where $a \in [-1,1]$, $b \in [-2,2]$, $c \in [-1,1]$, a, b, c are time-varying coefficients that are uniform distributed. In the experiment, parameters a, b, c were stochastically changed after each training phase, and new samples were regenerated: both training set and test set were equi-distributed in the input interval $[-3,+3]^2$; training set contains 289 data and test set contains 2401 data. All the samples had a signal-to-noise ratio (SNR) of 14 dB.

In simulation, our proposed RBF training algorithm was compared with other RBF training algorithms including: 1) clustering algorithm [21]; 2) least mean square (LMS) algorithm [3]; 3) memory enhanced genetic algorithm (MEGA) [9]; 4) genetic algorithm with memory-based immigrants (MIGA) [10].

Parameters in our proposed algorithm were set as follows: $T_{min} = 0.05$, $T_{max} = 0.1$, $N = 30$, $M = 10$, $\xi_1 = 0.5$; for first-stage learning, parameters of IEM were set as follows: $n = 15$, $m = 20$, $P = 3$, $S_1 = 50$; for second-stage learning, parameters of IEM were set as follows: $n = 40$, $m = 20$, $P = 4$, $S_2 = 80$; $\sigma_k(k \in 1,\ldots,N) \in [0.01, 6]$. These parameters were chosen in ad hoc fashion, being only adequate choices, but perhaps not the optimal ones.

Algorithm	Training Error(MSE)		Test Error(MSE)		Hidden	Training
	Mean	StD	Mean	StD	Units	Time(sec)
clustering	0.3231	0.0432	0.3751	0.2354	40	10.2
LMS	0.1238	0.0285	0.3493	0.2052	35	14.8
MEGA	0.0279	0.0223	0.0546	0.0803	31	39.7
MIGA	0.0176	0.0195	0.0339	0.0569	31	42.2
Proposed	0.0103	0.0145	0.0295	0.0321	30	34.2

Tab. III *Performance comparison for dynamic function approximation.*

All simulations were implemented with Matlab 7.0.4 and were executed on IBM compatible PC in which AMD Athlon 64 X2 2.2G CPU and 1GB RAM were mounted. 100 Monte Carlo were performed, and parameters a, b, c went through 50 permutations in each run of simulation. Experimental results of different RBF training algorithms are shown in Tab. III, where mean and standard deviation (StD) of MSE are compared. From Tab. III it can be seen that the proposed algorithm has reached very high precision with robustness. The training time of the algorithm is also less than other evolutionary algorithms for dynamic environments, such as MEGA and MIGA.

4.2 Channel estimation for OFDM systems

RBF networks have been widely used in adaptive channel equalization [3], which presents a typical nonlinear dynamic problem to evaluate the performance of RBF online training algorithms. In this section, our proposed algorithm is applied to channel equalization of the frequency-selective fading channels in orthogonal frequency division multiplexing (OFDM) systems.

Orthogonal frequency-division multiplexing is a multicarrier digital modulation technique for high-speed wireless communication systems. In OFDM systems, the entire channel is divided into many narrowband subcarriers, through which data are transmitted in parallel, thereby increasing the symbol duration and reducing the effect of intersymbol interference.

Accurate channel estimation is the key to reliable coherent OFDM communications. In [3] RBF network has been developed for pilot-symbol-aided channel estimation in OFDM systems in relatively fast multipath fading channels, where LMS algorithm is applied to adjust the parameters of RBF network.

In our experiment, we used the same channel model and parameters as in [3], i.e., a six-ray multipath fading channel in Standard ITU R M.1225-Vehicle test environment was considered; the parameters were set as follows: the relative delays are 0, 300, 8900, 12900, 17100 and 20000 (ns); the average powers are -2.5, 0, -12.8, -10 -25.2 and -16.0 (dB). The Rayleigh fading process is simulated by using the harmonic decomposition method [23]. The OFDM signal was also constructed according to [3], and quadrature phase shift keying (QPSK) modulation with coherent demodulation was used.

In simulation, our proposed RBF training algorithm was compared with other RBF training algorithms including: 1) 1-D RBFN in [3]; 2) 2-D RBFN in [3]; 3) MEGA [9]; 4) MIGA [10]. Except for 1-D RBFN, all RBF estimators used two-dimensional channel estimation as in [3]. The symbol error rate (SER) in different normalized Doppler frequency (f_dT) and SNR conditions is employed as the performance index.

Parameters in our proposed algorithm were set as follows: $T_{min} = 0.001/f_dT$, $T_{max} = 0.004/f_dT$,$N = 15$, $M = 6$, $\xi_1 = 2\sqrt{T_{min}}$; for first-stage learning, parameters of IEM were set as follows: $n = 15$, $m = 20$, $P = 3$, $S_1 = 40$; for second-stage learning, parameters of IEM were set as follows: $n = 40$, $m = 30$, $P = 3$, $S_2 = 60$; $\sigma_k(k \in 1,\ldots,N) \in [0.005, 10]$.

All simulations were implemented with Matlab 7.0.4 and were executed on IBM compatible PC in which AMD Athlon 64 X2 2.2G CPU and 1GB RAM were mounted. Fig. 2 shows the SER performance of the five RBF channel estimators in a channel with $f_dT = 0.003$. Fig. 3 shows the SER performance of the five RBF channel estimators in a channel with $f_dT = 0.008$. Fig. 4 shows the SER performance of the five RBF channel estimators as a function of the normalized Doppler f_dT, with SNR=12dB.

Tab. IV compares mean and standard deviation (StD) of SER and MSE for different RBF channel estimators, where $f_dT = 0.008$ and SNR=10dB. In all the above simulations 100, Monte Carlo (MC) realizations were done for each value.

Algorithm	SER		MSE		Hidden	Training
	Mean	StD	Mean	StD	Units	Time(sec)
1D-RBF	0.1404	0.2510	8.21e-3	1.21e-2	15	21.2
2D-RBF	0.1188	0.2140	7.62e-3	1.32e-2	15	20.8
MEGA	0.0868	0.0892	1.63e-3	0.87e-3	15	44.7
MIGA	0.0715	0.0642	1.03e-3	0.55e-3	15	50.2
Proposed	0.0298	0.0272	0.62e-3	0.24e-3	15	41.2

Tab. IV *Performance comparison for OFDM channel estimation.*

From the above simulations it can be seen that RBF network designed with the proposed algorithm is robust against environmental changes, and has reached best performance in various conditions. The proposed algorithm offers better performance especially in relatively fast fading channels, and its computation overhead is also less than that of MEGA and MIGA.

5. Conclusions

In this paper, a two-stage learning method based on immune operations is proposed to configure RBF centers and widths. The algorithm fully exploits the locally-tuned structure of the RBF network, and is specially designed to adapt to dynamic problems solving. Applications in dynamic problems demonstrate that the proposed algorithm has made good compromise between network precision and computation overhead. Experiments in channel estimation of OFDM systems prognosticate that the proposed algorithm can offer good performance especially in rapidly changing environments.

As future work for this paper, it would be interesting to apply the proposed algorithm to other dynamic signal processing problems, which include automatic control, wireless communication (such as channel equalization, multiuser detection, adaptive power control), dynamic system modeling and prediction, speech signal processing, dynamic function approximation, adaptive filter design, etc.

Another future task involves the parameter setting of the algorithm. As a key part of the proposed algorithm, the two-stage learning strategy introduces many user-defined parameters besides its improvement on the algorithm efficiency in dynamic environments. Although some parameters can be adaptively adjusted as suggested in the paper, it is interesting to study the selection of parameters in different kinds of dynamic environments (such as periodically returning dynamic

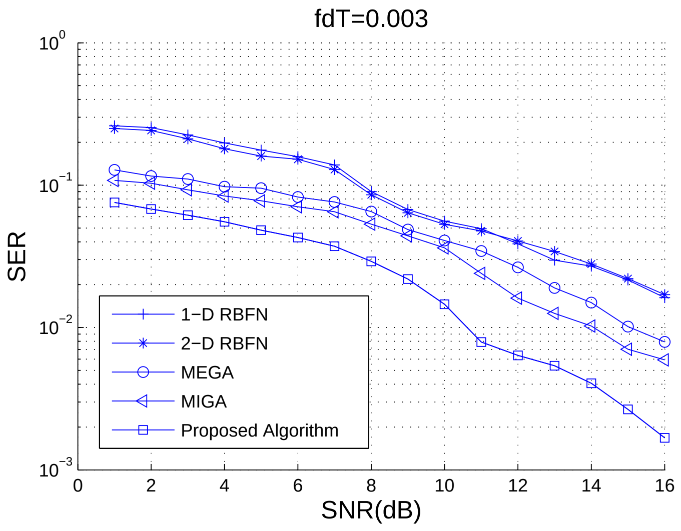

Fig. 2 *SER versus SNR with* $f_dT = 0.003$.

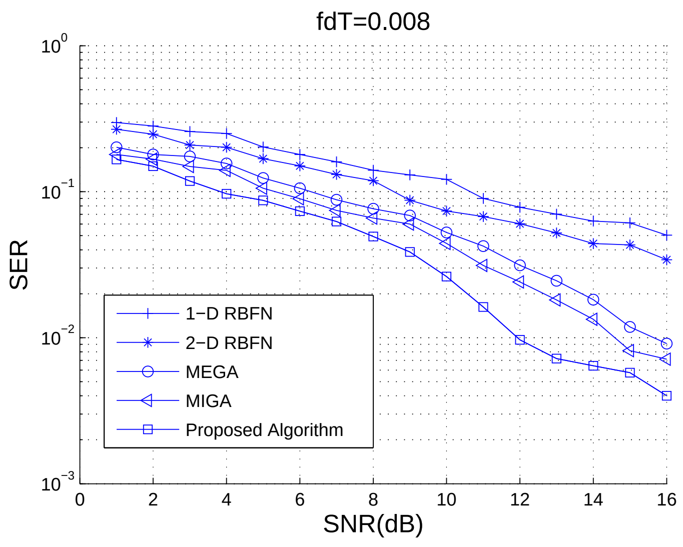

Fig. 3 *SER versus SNR with* $f_dT = 0.008$.

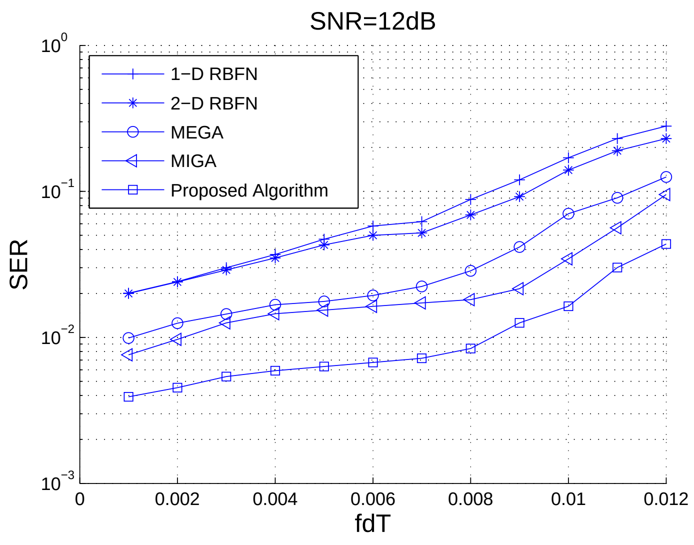

Fig. 4 *SER versus* f_dT *with SNR=12dB.*

environments, where memory schemes might be more useful) for better operation of the algorithm.

While in this paper immunity-based operations are incorporated in the design of RBF neural network, the interaction between neural networks and immunity-based systems is not confined to this architecture. Since both neural networks and immunity-based systems are biologically inspired techniques that exhibit compelling parallel adaptive information-processing abilities, this paper just prognosticates a tempting idea of incorporating neural networks and immune operations for solving complex dynamic information processing problems.

Acknowledgement

This work was supported by Shanghai-NRC-IRAP Joint Research Program (No. 06SN07112).

References

[1] Chen S., Xia H., Luk B. L., Harris C. J.: Construction of Tunable Radial Basis Function Networks Using Orthogonal Forward Selection, IEEE Transactions on Systems, Man, and Cybernetics, Part B, **39**, 2, 2009, pp. 457-466.

[2] Chen S., Lajos H., Tan S.: Symmetric Complex-Valued RBF Receiver for Multiple-Antenna-Aided Wireless Systems, IEEE Transactions on Neural Networks, **19**, 9, 2008, pp. 1659-1665.

[3] Zhou X. B., Wang X. D.: Channel Estimation for OFDM Systems Using Adaptive Radial Basis Function Networks, IEEE Transactions on Vehicular Technology, **52**, 1, 2003, pp. 48-59.

[4] Karayiannis N. B., Xiong Y. H.: Training Reformulated Radial Basis Function Neural Networks Capable of Identifying Uncertainty in Data Classification, IEEE Transactions on Neural Networks, **17**, 5, 2006, pp. 1222-1234.

[5] Huang G. B., Saratchandran P., Sundararajan N.: A Generalized Growing and Pruning RBF (GGAP-RBF) Neural Network for Function Approximation, IEEE Transactions on Neural Networks, **16**, 1, 2005, pp. 57-67.

[6] Karayiannis N. B., Randolph-Gips M. M.: On the Construction and Training of Reformulated Radial Basis Function Neural Networks, IEEE Transactions on Neural Networks, **14**, 4, 2003, pp. 835-846.

[7] Gonzalez J., Rojas I., Ortega J., etc.: Multiobjective Evolutionary Optimization of the Size, Shape, and Position Parameters of Radial Basis Function Networks for Function Approximation, IEEE Transactions on Neural Networks, **14**, 6, 2003, pp. 1478-1495.

[8] Jin Y. C., Branke J.: Evolutionary Optimization in Uncertain Environments – A Survey, IEEE Transactions on Evolutionary computation, **9**, 3, 2005, pp. 303-317.

[9] Yang S. X.: Genetic Algorithms with Memory – and Elitism-Based Immigrants in Dynamic Environments, Evolutionary Computation, **16**, 3, 2008, pp. 385-416.

[10] Yang S. X.: Memory-Based Immigrants for Genetic Algorithms in Dynamic Environments, Proceedings of 2005 Genetic and Evolutionary Computation Conference, **2**, 2005, pp. 1115-1122.

[11] Trojanowski K., Wierzchon S. T.: Immune-Based Algorithms for Dynamic Optimization, Information Sciences, **179**, 10, 2009, pp. 1495-1515.

[12] de Castro L. N., Timmis J. I.: Artificial Immune Systems as a Novel Soft Computing Paradigm, Soft Computing Journal, **7**, 8, 2003, pp. 526-544.

[13] Barra T. V., Bezerra G. B., de Castro L. N., Von Zuben F. J.: An Immunological Density-Preserving Approach to the Synthesis of RBF Neural Networks for Classification, 2006 International Joint Conference on Neural Networks, 2006, pp. 929-935.

[14] Zhou Y., Zheng D. L., Qiu Z. L., Dong G. Y.: The Application of RBF Networks Based on Artificial Immune Algorithm in the Performance Prediction of Steel Bars, Proceedings of the Third International Conference on Machine Learning and Cybemetics, 2004, pp. 3439-3443.

[15] Jiao L. C., Wang L.: A Novel Genetic Algorithm Based on Immunity, IEEE Transactions on Systems, Man and Cybernetics, Part A, **30**, 5, 2000, pp. 552-561.

[16] Aminifar F., Lucas C., Khodaei A., Fotuhi-Firuzabad M.: Optimal Placement of Phasor Measurement Units Using Immunity Genetic Algorithm, IEEE Transactions on Power Delivery, **24**, 3, 2009, pp. 1014-1020.

[17] Liu J. C., Zang X. G., Gong X. B.: Immune System Assisted Radial Basis Function Network for OFDM System Channel Tracking in Dynamic Environments, Eighth International Conference on Intelligent Systems Design and Applications, 2008, **1**, pp. 582-586.

[18] Wang L., Courant M.: Multiuser Detection Based on the Immune Strategy RBF Network, Proceedings of the 9th International Conference on Neural Information Processing, **3**, 2002, pp. 1485-1489.

[19] Dasgupta D.: Advances in Artificial Immune Systems, IEEE Computational Intelligence Magazine, **1**, 4, 2006, pp. 40-49.

[20] Moody J., Darken C. J.: Fast Learning in Networks of Locally-Tuned Processing Units, Neural Computation, **1**, 1989, pp. 281-294.

[21] Chen S., Mulgrew B., Grant P. M.: A Clustering Technique for Digital Communications Channel Equalization Using Radial Basis Function Networks, IEEE Transactions on Neural Networks, **4**, 4, 1993, pp. 570-590.

[22] de Castro L. N., Von Zuben F. J.: Learning and Optimization Using the Clonal Selection Principle, IEEE Transactions on Evolutionary Computation, **6**, 3, 2002, pp. 239-251.

[23] Crespo P. M., Jimenez J.: Computer Simulation of Radio Channels Using a Harmonic Decomposition Technique, IEEE Transactions on Vehicular Technology, **44**, 3, 1995, pp. 414-419.