Submitted to *IEEE Transactions on Systems, Man, and Cybernetics* (July 2000)

Immunity-Based Hybrid Learning Methods for Structure and Parameter Adjustment *

Yixin Diao and Kevin M. Passino

Department of Electrical Engineering
The Ohio State University
2015 Neil Avenue, Columbus, OH 43210

Abstract

From the point of view of information processing the immune system is a highly parallel and distributed intelligent system which has learning, memory, and associative retrieval capabilities. In this paper we present two immunity-based hybrid learning approaches for function approximation (or regression) problems that involve adjusting the structure and parameters of spatially localized models (e.g., radial basis function networks or Takagi-Sugeno fuzzy systems) to interpolate data. The number and centers of the receptive fields for local models are specified by immunity-based structure adaptation algorithms, while the parameters of the local models, which enter in a linear fashion, are tuned separately using a least squares method. The effectiveness of the procedure is demonstrated through a nonlinear function approximation problem and a nonlinear dynamical system modeling problem.

Index Terms: Hybrid learning, function approximation, radial basis function networks, fuzzy systems, artificial immune systems.

1 Introduction

When a neural network or fuzzy system is used to approximate a function by interpolating data from that function, the tuning flexibility of the approximator mapping is heavily dependent on the structure of the neural network (e.g., the number of nodes or layers) or fuzzy system (e.g., the number of rules or type of membership functions). Suppose that we use some appropriate measure of the size of an approximator structure (e.g., the number of tunable parameters, or the number of nodes or rules) to quantify the complexity of an approximator. Generally, a larger approximator structure provides more tuning flexibility with the potential to achieve higher approximation accuracy [1], but it can also cause convergence and complexity problems in training, overfitting and generalization problems, and perhaps require a larger training data set. Due to these trade-offs, we generally want the simplest approximator structure that can

*This work was supported by the NASA Glenn Research Center under grant NAG3-2084. Please address all correspondence to K. Passino (passino@ee.eng.ohio-state.edu, (614) 292-5716).

achieve our desired level of approximation accuracy.

Basically, approximator structure tuning algorithms can be classified into two broad categories: incremental (a.k.a., growing or constructive) algorithms and decremental (a.k.a., pruning or destructive) algorithms. Incremental algorithms start with a network that is too small to solve a problem. Parameter training methods are iterated until the training error does not decrease significantly, and then new units (weights, nodes, or rules) are added. Afterwards, either the parameters of the whole approximator are retrained, or only the parameters of the newly added tunable parameters are trained and the values of old parameters are frozen. The procedure is repeated until some terminating criteria is met [2]. The major incremental approaches include cascade correlation [3], projection pursuit regression [4], dynamic node creation [5], Meiosis networks [6], resource-allocating network [7], selective node activation [8], and so on. Decremental algorithms start with a large network, which is capable of good estimation, and trim it until the error becomes unacceptable. This may be achieved either by estimating the sensitivity of the error function to the removal of an element and then removing the elements with the least effect, or by adding terms to the objective function that reward choosing a small structure [9]. The main decremental approaches are the optimal brain surgeon [10], weight decay method [11], interactive pruning [12], and local/distributed bottlenecks [13], to name a few. Generally, it is straightforward to specify an initial network for incremental algorithms, while for decremental algorithms choosing an initial network that is definitely sufficient for an accurate mapping may require the use of an impractically large structure. Since incremental algorithms always search for small approximator solutions first, the training expense is low compared to the decremental algorithms, which can be computationally wasteful since the majority of the training time is spent on approximators that are larger than necessary and hence which are ultimately discarded. Furthermore, as many approximators with different sizes may be capable of implementing acceptable solutions, incremental algorithms are likely to find smaller approximator structure solutions than decremental algorithms, which may get stuck at one of the intermediate-sized solutions because of the shape of the error surface [14]. The structure of neural networks or fuzzy systems may also be tuned using genetic algorithms (see, e.g., [15]) or data clustering methods [16]. Methods to construct rules in fuzzy systems directly from training data have been studied extensively (see, e.g., [17, 18]).

In this paper we develop approaches to approximator structure and parameter tuning using a hybrid learning approach inspired from the natural immune system, a complex system which discriminates between own-body cells (self) and foreign pathogens (non-self), and defends against pathogens. From the point of view of information processing, the immune system is a highly parallel and distributed intelligent system, which has the capabilities of learning (to recognize relevant patterns, e.g., an antigenic peptide), memory (to memorize patterns that have been encountered previously), and associative retrieval (to construct pattern detectors to distinguish between self and non-self) by means of a network of local antigen-antibody interactions (mutual recognitions) at a system level. Inspired by the immune system, many intelligent computational methods, known as "artificial immune systems" or "immunity-based systems," have been studied and successfully used in a variety of applications such as pattern recognition, fault diagnosis, and computer security [19]. Of particular interest for the development of complex learning systems, adaptive both at the structural and parametric levels, is the "endogenous double plasticity" of the immune system [20]. "Parameter plasticity" is the adaptive mechanism allowing the system to adjust its parameters, and it is represented in the immune system by the variation of antibody concentrations. "Structural plasticity" is the adaptive mechanism allowing the immune system to have more or less elements, which amounts to the appearance or disappearance of antibody types, which are, moreover, conducted in a "collective" spirit, via adding new elements to replace the weakest ones or suppressing redundant elements from the system. Exploiting the endogenous double plasticity property of the immune network, in [21] the "evolving modular architecture for neural networks" (EMANN) algorithm is presented for self-structuring neural net classifiers. The ideas of the endogenous double plasticity of the immune network are also used in [22, 23] to enhance the reinforcement learning (Q-learning) for maze problems.

Here, we explore the computational power of immunity-based hybrid learning systems for spatially localized models with the goal of approximating unknown nonlinear functional mappings between input-output data. The biological model of the immune system is described first to understand the immunological mechanisms from which the motivation for our approach are derived (Section 2). In Section 3 spatially localized models in the form of radial basis function networks or Takagi-Sugeno fuzzy systems are discussed. In Section 4 we propose two immunity-based "hybrid" learning approaches capable of allocating the appropriate receptive fields or pruning the redundant ones, respectively, illustrated by a nonlinear function approximation problem. Here, "hybrid" refers to two essential characteristics of the learning procedure: (i) we use immunity-based concepts to specify the structure and some approximator parameters and a conventional least squares method to specify the remaining parameters, and (ii) we tune both parameters and structure of the approximators. A nonlinear dynamical system modeling problem is presented in Section 5 to demonstrate the potential of the proposed hybrid learning systems.

The proposed immunity-based hybrid learning approaches deploy the endogenous double plasticity property of the immune network (developed in [21] for structuring multilayer perceptrons) to spatially localized models and present a new measure of redundancy via the Euclidean distance between different receptive field units. Moreover, constructing hybrid learning systems based on spatially localized models extends the "hybrid premise/consequent training" methodology (see, e.g., [24]) so that less computational complexity (compared to tuning all parameters with the gradient methods) may be expected and the possibility of encountering local minima may be decreased.

2 Artificial Immune Systems

In the immune response the lymphocyte plays the main role. There are mainly two types of lymphocytes: T cells and B cells. Basically, B lymphocytes interact with antigens (foreign invaders) to stimulate the immune response, whereas T cells can either enhance or suppress the B cells' response to a stimulus. There are many antibodies on the surface of the B-lymphocyte that act as antigen detectors. The specialized portion of the antibody used for identifying other

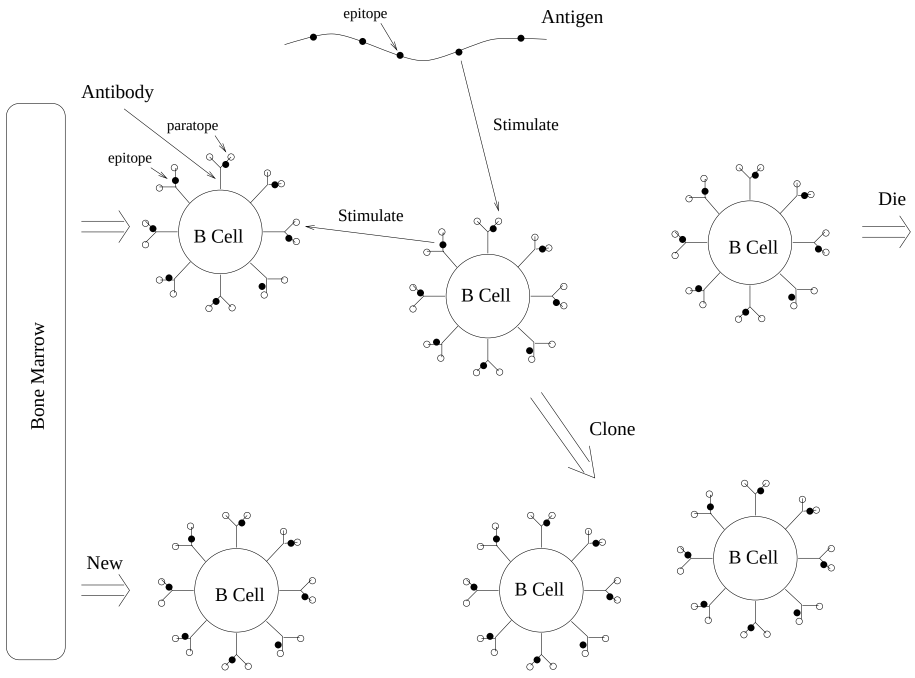

Figure 1: A schematic illustration of the immune system.

molecules (or cells) is called *paratope*. The regions on any molecule (of antigen or antibody) that can be recognized by the paratopes are called *epitopes*. If the epitope of an foreign antigen molecule matches the paratope of an antibody, the antibody can attach itself to the antigen, leading to its eventual demise. Note that the epitope of a body molecule may also match the paratope of an antibody. To prevent antibodies from destroying the tissue of the organism in which they reside, the immune system can eliminate all antibodies that react with the molecules of the host organism once they are produced; this is the so-called *self/non-self recognition process*.

Since antibodies, like other molecules, have epitopes, they may also be recognized by other antibodies. Therefore, different antibody types, together with antigens, compose a complex reaction network, which has been called the *immune network* by Jerne [25]. In the immune network, each node represents a type of antigen or antibody and the links between nodes represent the affinity between them, which is determined by both the “match degree” between the paratope and the epitope, and the population of each type of antibody (or antigen). An antibody type is thought to be stimulated when its paratope recognizes the epitope of antigens or other types of antibodies, so that the corresponding lymphocyte is stimulated to reproduce more lymphocytes, which then secrete more antibodies (with a certain of mutation rate). This process is called *clonal selection*, as shown in Figure 1. On the contrary, an antibody or antigen may be suppressed if its epitopes are recognized by others. Furthermore, the diversity of the immune system is maintained through the replacement of the dead lymphocytes (caused by aging or being suppressed by others) by new lymphocytes generated in the bone marrow (through the reshuffling of the host DNA that codes for the antibody genes).

Based on Jerne’s hypothesis that the immune system is a regulated network of molecules and cells (i.e., antibodies and antigens) that recognize one another, Perelson presented an immune network model [26, 27], which has received a lot of attention among researchers. Let i and j denote two nodes (two types of antibodies) in the immune network, which are represented by binary strings of length l. The matching index between them is given by

$$m_{ij} = \sum_k G\left(\sum_n e_i(n+k) \wedge p_j(n) - s + 1\right) \tag{1}$$

where n denotes the bit of a string, k denotes the shift alignment, $e_i(n+k)$ denotes the value of the $(n+k)^{th}$ bit of the epitope string on the i^{th} antibody type, $p_j(n)$ denotes the value of the n^{th} bit of the paratope string on the j^{th} antibody type, $\wedge$ denotes the *exclusive or* operation, s is a matching threshold below which two antibodies will not react at all, and $G(x) = x$ for $x > 0$ and $G(x) = 0$ otherwise. The connection strengths between different types of antibodies are determined not only by the matching indices, which are fixed (given the types of antibodies), but also by the concentrations of antibodies, whose dynamics are given by the following differential equations

$$\dot{x}_i = c\left[\sum_{j=1}^{N} m_{ji}x_i x_j - k_1 \sum_{j=1}^{N} m_{ij}x_i x_j + \sum_{j=1}^{n} m_{ji}x_i y_j\right] - k_2 x_i \tag{2}$$

where $x_i, i = 1, 2, \ldots, N$, denote the concentrations of different antibody types, and $y_j, j = 1, 2, \ldots, n$, denote the concentrations of antigen types. The first term represents the stimulation from other antibody types, the second term represents the suppression from other antibody types, and the third term represents the stimulation from other antigen types. The form of these terms is dictated by the fact that the probability of a collision between an antibody type i and an antibody type j (or an antigen type j) is proportional to $x_i x_j$ (or $x_i y_j$). The parameter c is a rate constant that depends on the number of collisions per unit time and the rate of antibody production stimulated by a collision. The constant k_1 represents a possible inequality between stimulation and suppression. The last tern models the tendency of cells to die in the absence of any stimulation, at a rate determined by k_2.

Besides the above differential equations that are used to describe the dynamics of the antibody concentrations, an essential aspect of the immune network model is that the list of antigen and antibody types is dynamic, changing as some types are added or removed, so that N and n change with time, but on a time scale that is slower than the evolution of the dynamics in the x_i. In particular, a minimum threshold on concentrations is placed so that a variable (a given antibody type) and all of its reactions are eliminated when its concentration drops below the threshold, which causes the death of unstimulated (useless) antibodies. Moreover, new types of variables may be added according to the reproduction of new antibodies.

Note that with the introduction of the dynamics of antibody concentrations, the dynamical behavior of the immune network becomes quite complicated because the connection parameters between the nodes are not independent, but are determined by the dynamics of the antibody concentrations (and the concentrations of lymphocytes), which implies that they cannot be

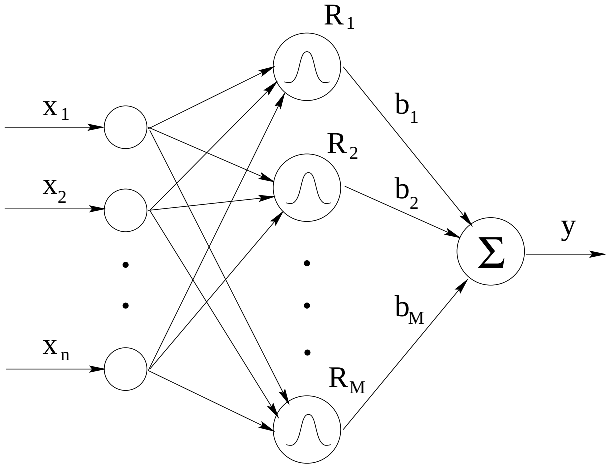

Figure 2: Radial basis function network.

specified and adjusted directly. This suggests that the immune network potentially possesses powerful capabilities, such as associative memory, but the learning ability may be weaker or more inefficient than those networks where the connection parameters are independent [28]. This may be one reason that the artificial immune system has not gained as much attention as the artificial neural networks on the aspects of learning and nonlinear modeling so far. However, the immune system is still an emerging field and provides a rich source of inspiration, and there are many computational aspects, such as pattern recognition and clonal selection, that can be and have been derived for practical use [19, 29, 30]. Here, we resort to inspirations from natural immune systems in order to address the problems of structure adaptation for learning systems.

3 Spatially Localized Model Architecture

In neurobiological studies, the concept of localized information processing in the form of receptive fields has been known and demonstrated by experimental evidence (e.g., locally tuned and overlapping receptive fields have been found in parts of the cerebral cortex, in the visual cortex, and in other parts of the brain), which suggests that such local learning offers alternative computational opportunities to learning with "global basis functions," such as the multilayer perceptron neural network with sigmoidal activation functions [31].

Inspired by these biological counterparts, the radial basis function neural network model has been presented, which, as shown in Figure 2, can be defined by

$$y = F_{rbf}(x, \theta) = \sum_{i=1}^{M} b_i R_i(x) \tag{3}$$

where y is the output of the radial basis function network, $x = [x_1, x_2, \ldots, x_n]^\top$ holds the n inputs, $i = 1, 2, \ldots, M$ represent M receptive field units, and θ holds the parameters of the "receptive field units," which consist of the "strength" parameters b_i and possibly the parameters of the "radial basis functions" $R_i(x)$ (a.k.a., radial response functions or kernel functions, defining the activation extents of the corresponding receptive fields, and having the characteristics that their responses decrease monotonically with distance from a central point). There are several possible choices for the receptive field functions $R_i(x)$. Typically, Gaussian-shaped functions are used for analytical convenience, that is,

$$R_i(x) = exp\left(-\frac{1}{2}(x - c_i)^\top D_i(x - c_i)\right) \tag{4}$$

where $c_i = [c_1^i, c_2^i, \ldots, c_n^i]^\top$ parameterize the locations of the receptive fields in the input space, and $D_i = diag\left(\left(\frac{1}{\sigma_1^i}\right)^2, \left(\frac{1}{\sigma_2^i}\right)^2, \ldots, \left(\frac{1}{\sigma_n^i}\right)^2\right)$ determine the shapes (or relative widths) of the receptive fields.

Note that rather than computing the output of the radial basis function network with the simple sum as in Equation (3), there are also alternatives, for instance, by computing a weighted average

$$y = F_{rbf}(x, \theta) = \frac{\sum_{i=1}^{M} b_i R_i(x)}{\sum_{i=1}^{M} R_i(x)} \tag{5}$$

Moreover, it is also possible to further define the strength parameters b_i to be parametric functions

$$b_i(x) = a_{i,0} + a_{i,1}x_1 + \cdots + a_{i,n}x_n \tag{6}$$

where $a_{i,j}$, $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, n$, are strength function parameters, so as to improve the modeling flexibility of the radial basis function network.

Another type of spatially localized model is the Takagi-Sugeno fuzzy system. The fuzzy system facilitates the emulation of human intelligence by modeling human cognitive processes in the form of rules and inference mechanism. Abstracted from the qualitative description of premise representation, inference, and defuzzification, the mathematical formula of the Takagi-Sugeno fuzzy system can be defined by

$$\begin{aligned} y &= F_{ts}(x, \theta) = \frac{\sum_{i=1}^{R} g_i(x)\mu_i(x)}{\sum_{i=1}^{R} \mu_i(x)} && (7) \\ g_i(x) &= a_{i,0} + a_{i,1}x_1 + \cdots + a_{i,n}x_n && (8) \\ \mu_i(x) &= \prod_{j=1}^{n} \exp\left(-\frac{1}{2}\left(\frac{x_j - c_j^i}{\sigma_j^i}\right)^2\right) && (9) \end{aligned}$$

where y is the output of the fuzzy system, $x = [x_1, x_2, \ldots, x_n]^\top$ holds the n inputs, and $i = 1, 2, \ldots, R$ represent R different rules [24]. The shapes of the membership functions are chosen to be Gaussian, and center-average defuzzification and product are used for the premise and implication in the structure of the fuzzy system. The $g_i(x)$, $i = 1, 2, \ldots, R$, are called consequent functions of the fuzzy system, where the $a_{i,j}$ are constants. The premise membership functions $\mu_i(x)$ are assumed to be well-defined so that $\Sigma_{i=1}^{R}\mu_i(x) \neq 0$ for all x. The parameters c_j^i and σ_j^i are the centers and relative widths of the membership functions, respectively, for the j^{th} inputs and the i^{th} rules.

Actually, the radial basis function neural network defined above by Equations (4, 5, 6) is functionally equivalent to the Takagi-Sugeno fuzzy system defined by Equations (7, 8, 9). To see this, suppose that we let the number of receptive field units equal to the number of rules (i.e., $M = R$), let the receptive field strength functions same as the consequent functions (i.e., $b_i(x) = g_i(x)$), and choose the parameters of the radial basis functions same as those of the premise membership functions (i.e., $R_i(x) = \mu_i(x)$). In this case the radial basis function network is identical to the Takagi-Sugeno fuzzy system.

Note that the tunable parameter vector θ in (5) or (7) can be composed of both radial basis function (or premise membership function) parameters c_j^i and σ_j^i and strength function (or consequent function) parameters $a_{i,j}$. This is referred to as *nonlinear in the parameter* case. A *nonlinear in the parameter* spatially localized model can be tuned by a variety of gradient methods such as the steepest descent method and Levenberg-Marquardt method. Alternatively, we may decompose the parameter vector into and a linear part θ_1 consisting of the strength (or consequent) function parameters, and a nonlinear part θ_2 composed of the radial basis function (or premise membership function) parameters. To see this, note that for radial basis function networks (and similar for Takagi-Sugeno fuzzy systems), we have

$$y = \frac{\sum_{i=1}^{M} a_{i,0} R_i(x)}{\sum_{i=1}^{M} R_i(x)} + \frac{\sum_{i=1}^{M} a_{i,1} x_1 R_i(x)}{\sum_{i=1}^{M} R_i(x)} + \cdots + \frac{\sum_{i=1}^{M} a_{i,n} x_n R_i(x)}{\sum_{i=1}^{M} R_i(x)}$$

By defining

$$\theta_1 = [a_{1,0}, a_{2,0}, \ldots, a_{M,0}, a_{1,1}, a_{2,1}, \ldots, a_{M,1}, \ldots\ldots, a_{1,n}, a_{2,n}, \ldots, a_{M,n}]^\top$$

$$\begin{aligned} \theta_2 \;=\; & [c_1^1, c_2^1, \ldots, c_n^1, c_1^2, c_2^2, \ldots, c_n^2, \ldots\ldots, c_1^M, c_2^M, \ldots, c_n^M, \\ & \sigma_1^1, \sigma_2^1, \ldots, \sigma_n^1, \sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2, \ldots\ldots, \sigma_1^M, \sigma_2^M, \ldots, \sigma_n^M]^\top \end{aligned}$$

and

$$\begin{aligned} \phi \;=\; & [\xi_1(x), \xi_2(x), \ldots, \xi_M(x), x_1\xi_1(x), x_1\xi_2(x), \ldots, x_1\xi_M(x), \ldots\ldots, \\ & x_n\xi_1(x), x_n\xi_2(x), \ldots, x_n\xi_M(x)]^\top \end{aligned}$$

where

$$\xi_j(x) = \frac{R_j(x)}{\sum_{i=1}^{M} R_i(x)}, j = 1, 2, \ldots, M$$

we have

$$y = F_{rbf}(x, \theta) = F_{ts}(x, \theta) = \theta_1^\top \phi(x, \theta_2) \tag{10}$$

In this way we separate the tunable parameters into two sets: a set θ_1 that enters linearly, and a set θ_2 that enters in a nonlinear fashion. This decomposition facilitates structure and parameter adaptation of the spatially localized model by way of hybrid learning approaches (as shown in the next section). In particular, we will apply the immunity-based structure adaptation approach to specify the number and location of the receptive field units, followed by using least squares to train the strength (or consequent) functions since they enter linearly.

4 Immunity-Based Hybrid Learning Algorithms

From the point of view of the biology, the function approximation problem may be thought of as a kind of artificial immune systems. The "antigens" represent the input-output data pairs which for the k^{th} experiment is $(x(k), y(k))$, and a collection of antigens defines the problem of function approximation. The "antibodies" represent the units of the approximator (e.g., the receptive field units of the radial basis function network or the rules of the Takagi-Sugeno fuzzy system), which are characterized by the tunable parameter vector θ, and a collection of antibodies defines the function approximator. A good immune system usually consists of a compact collection of antibodies capable of recognize all antigens, which is analogous to finding the simplest approximator structure that can achieve the desired level of approximation accuracy.

4.1 Incremental Approach

In the context of the spatially localized model, we propose an immunity-based hybrid learning approach, which is capable of allocating the appropriate receptive fields. The basic steps of this incremental approach can be summarized as follows:

1. Initialization: The initial approximator may be composed of one (or perhaps a few) receptive field unit(s) using *a priori* knowledge of the function to be approximated. Usually, the centers of the receptive field units c_j^i are uniformly distributed on the input space of the problem, and the relative widths σ_j^i are chosen to form local activation regions.

2. Recognition: The affinities between the data pairs $(x(k), y(k)), k = 1, 2, \ldots, P$, and the approximator $F_{rbf}(x, \theta)$ or $F_{ts}(x, \theta)$ are evaluated. The affinity measure is defined as the absolute value of the difference between the output data $y(k)$ and the output of the approximator.

3. Reinforcement: The "weakest" data pair $(x(w), y(w))$, i.e., the one whose value of affinity measure is the largest, is cloned. That is, we recruit a new receptive field unit, whose center is located at $x(w)$ and the relative width is a fixed value chosen in advance, and enlarge the parameter vector θ_2 accordingly (while keeping the values of already-existing parameters frozen). The structure of the parameter vector θ_1 is also enlarged, and all parameters in $theta_1$ will be retrained (in the next step) in order to exploit the newly-added design flexibility and optimize the performance of the approximator.

4. Parameter Adaptation: We use the training data set $G = \{(x(k), y(k)) : k = 1, 2, \ldots, P\}$ to tune the "linear" parameters θ_1 which consist of the strength (or consequent) function

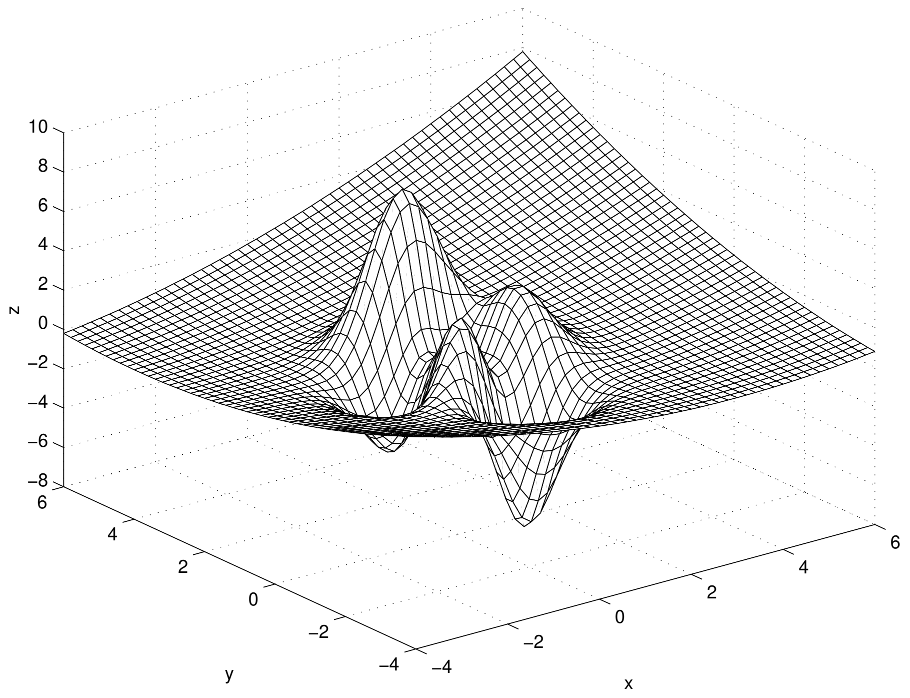

Figure 3: Nonlinear function we seek to approximate.

parameters $a_{i,j}$ only. This may be implemented by least squares method, that is,

$$\theta_1 = (\Phi^\top \Phi)^{-1} \Phi^\top Y \tag{11}$$

where $Y = [y(1), y(2), \ldots, y(P)]^\top$ and $\Phi = [\phi(x(1), \theta_2), \phi(x(2), \theta_2), \ldots, \phi(x(P), \theta_2)]^\top$.

5. Generalization: The performance of the approximator is tested by evaluating approximation error, which is defined as maximum absolute error (MAE), the absolute value of the difference between the output of the actual function and the output of the approximator. If the value of the fitness function is below a "satisfactory threshold," we claim that a satisfactory approximator has been generated. Otherwise, go to Step 2 for structure and parameter adaptation again.

To study the performance of the proposed immunity-based hybrid learning approach we seek to approximate a complicated nonlinear function

$$z = 0.1x + 0.05(x+y)^2 + 3(1-x)^2 e^{-x^2-(y+1)^2} - 10(\frac{x}{5} - x^3 - y^5) e^{-x^2-y^2} - \frac{1}{3} e^{-(x+1)^2-y^2} \tag{12}$$

with $x \in [-4, 6], y \in [-4, 6]$, which is shown in Figure 3. Note that the nonlinearity represented in this function is not uniform. The "frequency" of the nonlinearity is high in some regions, while the function is relatively smooth in other areas. It is generally not easy to approximate such functions.

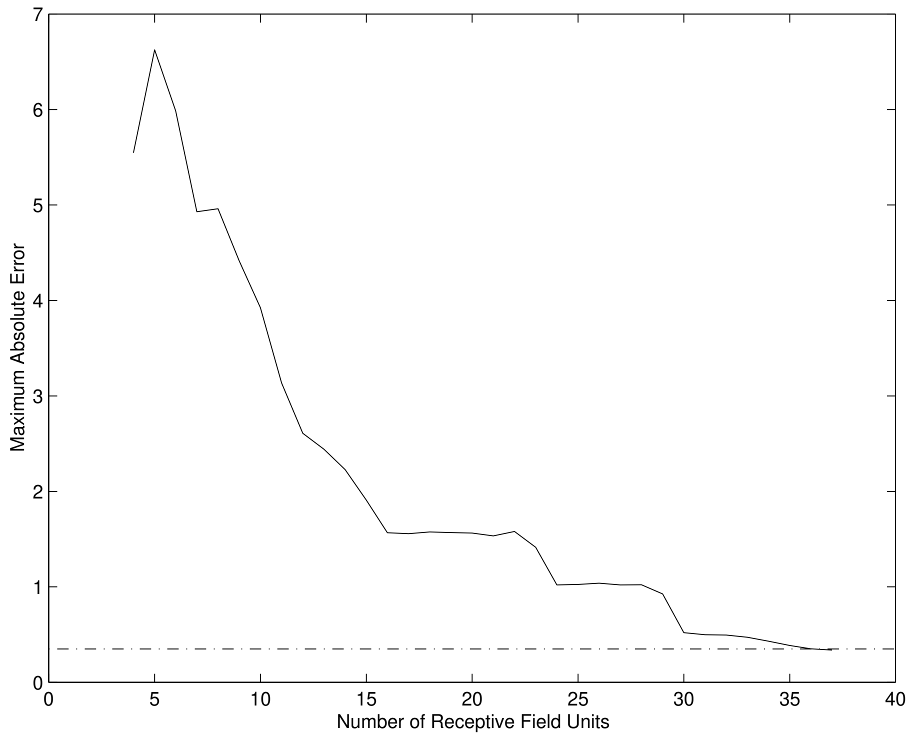

Figure 4: Structure adaptation via recruitment of new receptive field units.

We generate a training data set including 1000 data pairs $([x(k), y(k)], z(k)), k = 1, 2, \ldots, 2000$. The "satisfactory threshold" is chosen to be 0.35. We start the approximation process with 4 receptive field units, whose centers, located at $(-2, -2)$, $(-2, 4)$, $(4, -2)$, and $(4, 4)$, are uniformly distributed in the problem space. The performance of the approximator becomes better along with the structure adaptation when more receptive field units are recruited into the spatially localized model, as shown in Figure 4 (where the dash-dotted line represents the satisfactory threshold). An animation illustrating the structure adaptation process of incremental immunity based function approximator can also be found at http://eewww.eng.ohio-state.edu/∼diaoy/immune.html.

The immunity-based hybrid learning approach results in a spatially localized model with 37 receptive field units to satisfy the requirement given by the satisfactory threshold. As shown in Figure 5, there are more units in the region with high nonlinearities, which is quite reasonable. The function surface implemented by this immunity-based hybrid learning system, as shown in Figure 6, is very close to the function surface for which we seek to approximate, and the maximum absolute error is 0.3580 (which is slightly above 0.35 because we are using a different test data set consisting 2000 data pairs here, rather than the training data set for structure and parameter tuning).

The performance of a function approximator with both structure and parameter adaptation becomes remarkable when the function encountered is complicated enough. A function only with high frequency components may be easy to approximate, for instance, by a *linear in the*

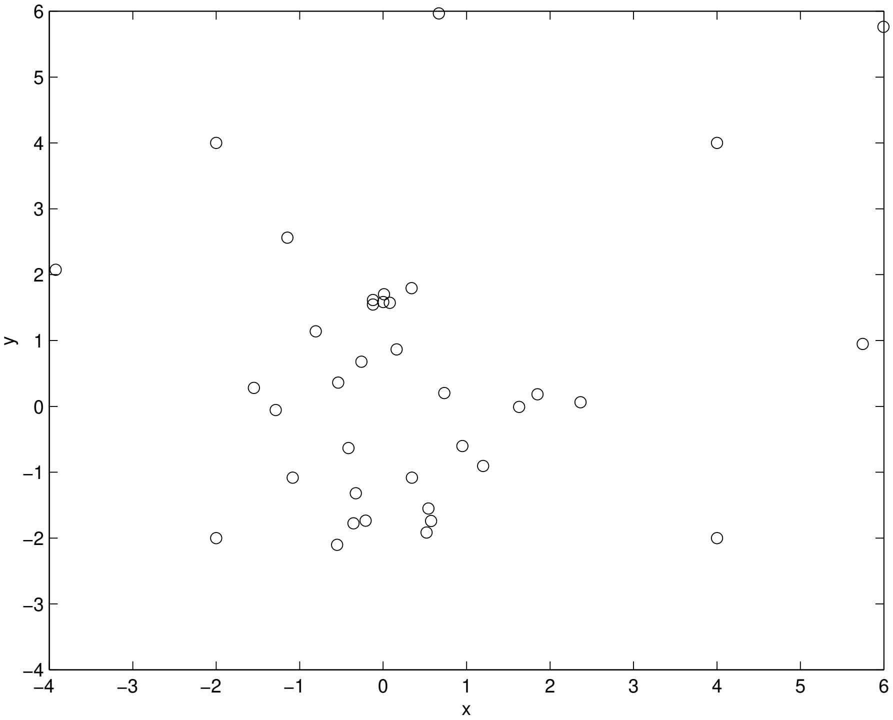

Figure 5: Centers of the resulting receptive field units (incremental approach).

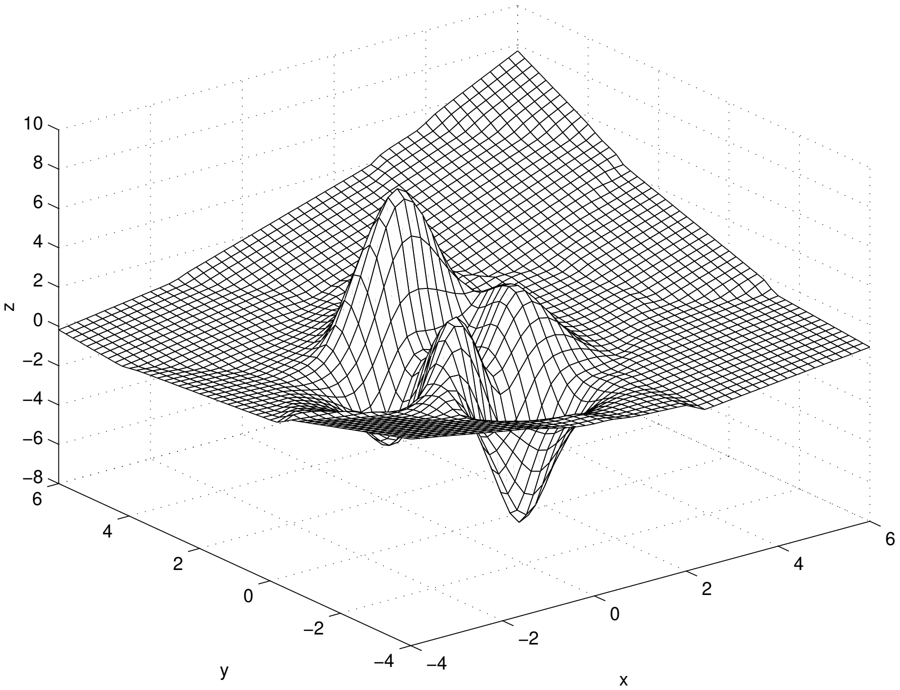

Figure 6: Function surface of the immunity-based approximator (incremental approach).

parameter Takagi-Sugeno fuzzy system with enough number of rules, given a large amount of well distributed training data. In this case we may simply add more rules to improve the approximation accuracy, and the effects of structure adaptation is straightforward. Here, we are trying to approximate a function which consists of both high frequency and low frequency components. Without *a priori* knowledge of the nonlinearity of the function, a very fine grid of receptive field units (or rules) is required for radial basis function networks (or Takagi-Sugeno fuzzy systems), which, actually, may result in a redundant approximation structure that wastes many computations. For example, in order to accurately approximate the function given in Equation (12), a Takagi-Sugeno fuzzy system with 64 rules (on an 8×8 grid) is required, while using 49 rules (a 7×7 grid) is not appropriate due to the poor error surface. The proposed immunity-based hybrid learning system, however, is capable of approximating the function accurately enough with only 36 receptive field units, so that the complexity of the approximator is greatly reduced. Moreover, there is no generalization or overtraining problem encountered.

For the above function approximation problem one may argue that a nonlinear in the parameter approximator tuned by gradient methods may obtain good performance with relatively small number of parameters. However, it is difficult, in general, to find the optimal solution, and the parameters may get distracted by the multiple local minima, which we will illustrate later by an example of using radial basis function networks tuned by the Levenberg-Marquardt method.

4.2 Decremental Approach

Note that a natural immune system performs a structure change according to its endogenous behavior (i.e., antibody concentrations) rather than exogenous inspiration (i.e., antigen concentrations). Inspired by this aspect of the immune system, we propose another immunity-based hybrid learning approach, which is described as follows:

1. Initialization: The decremental approach starts with a spatially localized model consisting of a large set of units, which is capable of good estimation. Sometimes, this can be obtained by a fine division of the input space and then by assigning a receptive field unit for each subregion (of course, if the input space is of high dimension, this may not be possible).

2. Recognition: We calculate the affinities between the receptive field units. The affinity measure is defined as

$$f_{i,k} = m_{i,k}\delta_{i,k} \tag{13}$$

$$m_{i,k} = \sqrt{\sum_{j=1}^{n}(c_j^i - c_j^k)^2} \tag{14}$$

$$\delta_{i,k} = \sqrt{\sum_{j=1}^{n}(a_{i,j} - a_{k,j})^2} \tag{15}$$

where i and k represent two different receptive field units, $m_{i,k}$ is the Euclidean distance between the centers of the i^{th} unit and the k^{th} unit, and $\delta_{i,k}$ is the Euclidean distance between the strength (or consequent) function parameters of the i^{th} unit and the k^{th} unit.

3. Refinement: This process serves as structure adaptation for the approximator by eliminating the redundant units. We find the pair of receptive field units whose value of affinity measure is the smallest, and eliminate (any) one of them. Accordingly, we reduce the size of parameter vector θ_2 while keeping other parameters frozen. The size of the parameter vector θ_1 is also diminished and its parameters will be retrained in the next step.

4. Parameter Adaptation: We use a least squares method to tune the parameter vector θ_1 in order to maintain good approximation accuracy by allocating the computational power optimally among the remaining receptive field units.

5. Generalization: The performance of the approximator is tested by calculating the fitness function of the hybrid learning system, which is defined as the MAE. If the value of the fitness function is above a "satisfactory threshold," that is, the error becomes unacceptable, we stop the learning process and come up with the last satisfied approximator. Otherwise, go to Step 2 for structure and parameter adaptation again.

To test the method we use the function defined in Equation (12) once again. We start the approximation process from a spatially localized model with 100 receptive field units, whose centers are uniformly distributed in a 10×10 grid. The "satisfactory threshold" is chosen to be 0.7 because the centers of the decremental learning system are distributed more regularly and an acceptable function approximation surface corresponds to a larger value of satisfactory threshold (compared to 0.35 in the incremental approach). The performance of the approximator degrades along with the structure adaptation when more units are eliminated from the learning system, and it finally results in a spatially localized model with 32 receptive field units, whose centers are shown in Figure 7. As shown in Figure 8, the function surface implemented by this immunity-based learning system is also very close to the function surface for which we seek to approximate, and the maximum absolute error for the testing data set is 0.6774. Note that the MAE here is much larger than that in the incremental approach but their function surfaces are very similar. This is because MAE is only one possible error measure that we are using to terminate the learning process and its effect may be different in different learning approaches. Thus, we view the function surface (as in Figure 6 or Figure 8) as a kind of general measure of the goodness of the approximator and we use it to determine the appropriate "satisfactory threshold" in the incremental and decremental approaches, respectively.

4.3 Remarks

Although both of these two learning systems are inspired by the natural immune system, their structure adaptation strategies are different. The incremental approach is capable of adding more receptive field units and the structure adjustment is driven by the exogenous criteria (i.e., the error function). The decremental learning system can eliminate redundant elements from the structure according to the endogenous behavior of the spatially localized model (i.e., the

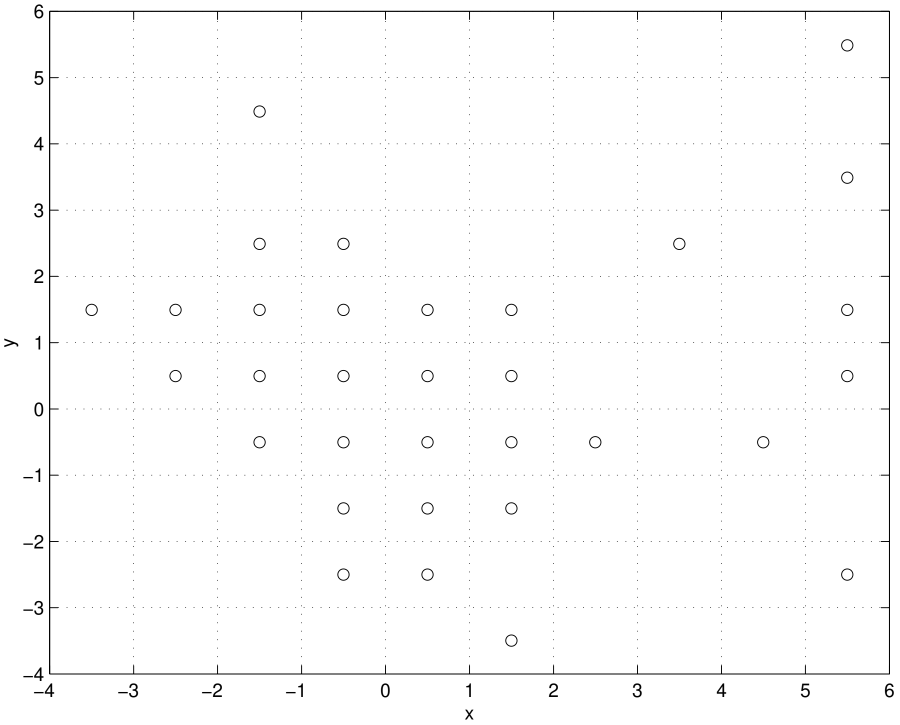

Figure 7: Centers of the resulting receptive field units (decremental approach).

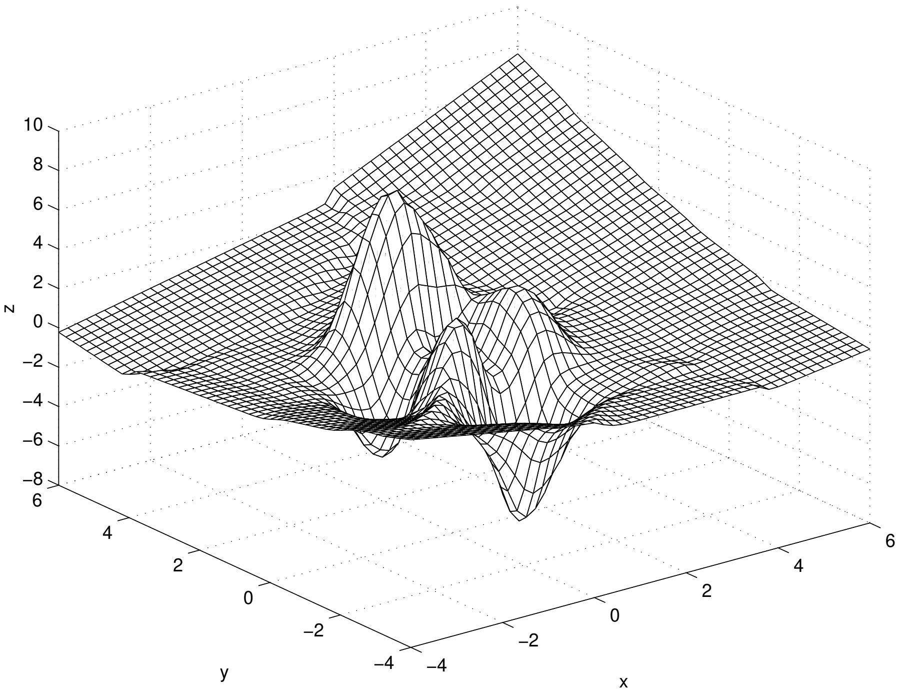

Figure 8: Function surface of the immunity based approximator (decremental approach).

similarities of the units).

The idea of using a heuristic measure of redundancy is also used in [21] for multilayer perceptron neural networks, where the authors take into account the weights of neurons. A neuron is eliminated from the network if the average value of the weights connected to it is below a "useless threshold," and if these average values for all neurons in the network are above a certain "settling threshold," which implies that the learning ability of the network is low because all neurons are almost saturated, a new neuron is then added to the network. Here, we use the Euclidean distance to measure the redundancy between different receptive field units for spatially localized models and remove the redundant units accordingly. Furthermore, constructing hybrid learning systems based on spatially localized models provides the advantages of design flexibility for structure and parameter adaptation so that we can apply the immunity-based structure adaptation approach to specify the architecture and the parameters that enter nonlinearly, followed by using least squares to train the linear part parameters (which also often implies less computational complexity compared to tuning all parameters with the gradient methods). In other words, it is a kind of combination of local tuning (where the new unit is added only to compensate for the worst part of the approximator) and global tuning (where all the parameters of strength (or consequent) function are retrained to achieve global optimization afterwards).

In addition, note that decremental algorithms inherently assume the availability of a complex approximator sufficient for accurate mapping. This assumption may be relatively easier to be achieved by a spatially localized model with a fine grid of units tuned by the least squares method rather than by a multilayer perceptron neural network which are often tuned by gradient methods such as backpropagation. However, we do not consider the approaches universal. For other applications, one of the other methods may provide a better solution. Indeed, none of the existing algorithms outlined in the Introduction are proven to be optimal in structure adaptation for learning systems and we could not show that either. For different problems and different measures of "best," different algorithms may give structures with a minimal number of units (neurons or rules).

Compared to the incremental immunity-based learning system, the centers of the decremental learning system are distributed more regularly, but its approximation performance is slightly worse. This is because the arrangement of its centers is restricted due to the initialization, while the assignment of the centers for the incremental approximator is more flexible so that it may capture more characteristics of the function.

However, note that as shown in Figure 5 some receptive field units generated in the incremental approach are very close to each other. This suggests that we integrate the incremental and decremental approaches together. For instance, we could execute the incremental approach first to add units at the positions that are necessary, and then execute the decremental approach to remove the redundant ones. We may expect the resulting approximator to have a smaller structure while still maintaining sufficient accuracy. Actually, by using this strategy for the above problem we reduce the number of units to 31 and can still have a function surface accurately enough.

Finally, note that from the view of computational complexity, the incremental approach is better because it starts with less number of units so that it may take less time to reach the final structure than the decremental approach. For the instance of the above function approximation case, the decremental approach takes about 9 times more computations than the incremental approach.

5 Application to Nonlinear Dynamical System Modeling

Function approximation is a generic problem that permeates many fields such as economic and business planning, weather forecasting, and systems and control. In this section, a nonlinear dynamical system modeling problem, in particular, model development for an aircraft jet engine (General Electric XTE46), is provided to demonstrate the potential of the proposed hybrid learning methods.

Model development for jet engines is complicated by the presence of engine-to-engine manufacturing differences and engine deterioration during normal operation, the complexity of an accurate engine model, and our inability to directly measure certain engine variables. Here, we work with a sophisticated component level model (CLM) simulation of a turbine engine (XTE46) that can simulate the effects of manufacturing and deterioration differences, in addition to a variety of failures. The component level model may be used as the "truth model" in the control simulation to represent the real engine. However, from the point of view of theoretical studies (in fault tolerant control), this model is too complicated. Although the CLM does provide a driver to trim the model to specified operating conditions and generate linearized models, studies show that the accuracy of the linear models is not adequate for our purposes (where we consider faults with significant nonlinear effects). Here, we develop a nonlinear analytical model with Takagi-Sugeno fuzzy systems and will view it as a "design model."

The General Electric XTE46 engine is a simplified, unclassified version of the original IHPTET engine [32]. The component level engine cycle model of the XTE46 engine is a thermodynamic simulation package, where each engine component is simulated. The CLM executes one pass within the digital control's sampling time and thermodynamic states are assumed to be in equilibrium after each pass through the simulation. The operating condition of the engine is defined by the altitude (ALT), mach number (XM), difference of temperature (DTAMB), and throttle setting represented by power code (PC). The health of the engine is described by ten "quality parameters" which include the flows and efficiencies of the fan, the compressor, and turbines. The model has three state variables, including the fan rotor speed (XNL), the core rotor speed (XNH), and the temperature at combustor inlet (TMPC). There are six actuators, but the major control variables are the combustor fuel flow (WF36), the exhaust nozzle area (A8), and the variable area bypass injector area (A16). For simplicity, we assume that the fundamental dynamic characteristics of the CLM can be represented by a single-input single-output system in the form

$$\dot{x} = f(x, u, c, p) \tag{16}$$
$$y = h(x, u, c, p) \tag{17}$$

where $x = [\text{XNL, XNH}]^\top$ is the state vector, $u = \text{WF36}$ is the input variable, $y = \text{XN2}$ is the output of the engine, $c = [\text{ALT, XM, DTAMB, PC}]^\top$ represents the operating condition of the engine, $p = [\text{ZSW2, SEDM2, ZSW7D, SEDM7D, ZSW27, SEDM27, ZSW41, ZSE41, ZSW49, ZSE49}]^\top$ represents the quality parameter vector, $f(\cdot)$ denotes the unknown function representing the nonlinear characteristics of the engine, and $h(\cdot) = \text{XNL}$ because the output variable XN2 is just the measurement of the state variable XNL.

To develop a computational model for such a complicated nonlinear system can be quite challenging because the nonlinearity of the engine is different at different operating conditions and for different quality parameters. Although it is theoretically possible to build one neural network or fuzzy system to approximate the engine dynamics by interpolating all engine data collected from the whole space of operating conditions and quality parameters, it is not feasible in practice because of the huge amounts of data (i.e., millions of data pairs). Instead, we would like to view it as a "distributed" modeling problem, that is, we perform learning to approximate local engine dynamics (specified by fixed values of operating conditions and quality parameters), followed by interpolating these local models to generate the "global" model (actually, it is a "regional" model valid in the "climb" region). Note that it is possible to obtain the "regional" model by fuzzy interpolation on a grid of these local models [33]. However, using a too coarse grid may only obtain limited model accuracy, while using a too fine grid may result in a redundant structure.

In this section we attempt to use the ideas of (incremental) immunity-based hybrid learning approach to generate the engine model. We use immunity-based concepts (i.e., the heuristics like "compensate for the weakest elements") to specify the structure and some parameters of the "regional" nonlinear model, and the Levenberg-Marquardt method to tune local model parameters and to optimize the remaining parameters of the regional model. To illustrate the basic ideas we only consider a simplified scenario representing the nominal behavior of the engine, where the engine dynamics are only varying with respect to three key operating condition variables (ALT, XM, and PC), while the variable DTAMB and quality parameters are fixed. The process of hybrid learning starts from generating a regional model consisting of only one local model, located in the center (ALT $= 15000$, XM $= 0.7$, and PC $= 47.5$) of the climb region (ALT $\in [12500, 17500]$, XM $\in [0.6, 0.8]$, and PC $\in [45, 50]$). Given a specific operating condition (c_i) and fixed values of quality parameters (p_i), the local model for the XTE46 engine can be obtained using nonlinear system identification technique as

$$\begin{aligned}
\widehat{\mathrm{XNLdot}}(k) &= F_{ts}^1\left(\widehat{\mathrm{XNL}}(k), \widehat{\mathrm{XNH}}(k), \mathrm{WF36}(k), \theta^1(c_i, p_i)\right) \\
\widehat{\mathrm{XNHdot}}(k) &= F_{ts}^2\left(\widehat{\mathrm{XNL}}(k), \widehat{\mathrm{XNH}}(k), \mathrm{WF36}(k), \theta^2(c_i, p_i)\right) \\
\widehat{\mathrm{XNL}}(k+1) &= \widehat{\mathrm{XNL}}(k) + T_s\widehat{\mathrm{XNLdot}}(k) \\
\widehat{\mathrm{XNH}}(k+1) &= \widehat{\mathrm{XNH}}(k) + T_s\widehat{\mathrm{XNHdot}}(k) \\
\widehat{\mathrm{XN2}}(k+1) &= \widehat{\mathrm{XNL}}(k+1)
\end{aligned}$$

where two multiple-input single-output (MISO) Takagi-Sugeno fuzzy systems (or, equivalently, radial basis function networks), F_{ts}^1 and F_{ts}^2, are specified with corresponding parameters θ^1 and θ^2, respectively. The variables $\widehat{\mathrm{XNL}}(k)$ and $\widehat{\mathrm{XNH}}(k)$ denote the estimated values of $\mathrm{XNL}(k)$ and $\mathrm{XNH}(k)$, $\widehat{\mathrm{XN2}}(k)$ is the estimated value of $\mathrm{XN2}(k)$, and $\widehat{\mathrm{XNLdot}}(k)$ and $\widehat{\mathrm{XNHdot}}(k)$ are the outputs of the fuzzy systems, which denote the estimated derivatives of $\mathrm{XNL}(k)$ and $\mathrm{XNH}(k)$, respectively. By means of Levenberg-Marquardt method, the fuzzy systems are trained using engine data generated by the transient driver of the component level model simulator of the XTE46 engine (provided by GE Aircraft Engines). One thousand engine input-output data pairs are collected which reflect the transient performance of the engine for 20 seconds (sampled every $T_s = 0.02$ second) at a specific "node" (of operating conditions and quality parameters). The performance of this local model is reasonably good for this specific point, as illustrated in Figure 9 (where the solid lines represent the system response of the "real" engine (i.e., the CLM) and the dashed lines represent the response of the analytical model), and the mean square error is 4.1394. Obviously, a regional model composed of only one local model results in poor approximation accuracy for other points in the region because the nonlinearity of the engine is different at different operating conditions. Here, we use the average mean square error (AMSE) as the affinity measure to evaluate the suitability of the model in the whole region, which is defined by the average value of the mean square errors representing model performance at different points in the region. For this one local model case the AMSE is 61.9364.

To improve the regional model performance, we recruit more local models which correspond to the "weakest" points with poorest approximation performance. The regional model is in the form of nonlinear interpolation between these local models

$$\widehat{\mathrm{XNLdot}}(k) = \frac{\sum_{i=1}^N F_{ts}^1(\widehat{\mathrm{XNL}}(k), \widehat{\mathrm{XNH}}(k), \mathrm{WF36}(k), \theta^1(c_i, p_i))\mu_i(z)}{\sum_{i=1}^N \mu_i(z)} \tag{18}$$

$$\widehat{\mathrm{XNHdot}}(k) = \frac{\sum_{i=1}^N F_{ts}^2(\widehat{\mathrm{XNL}}(k), \widehat{\mathrm{XNH}}(k), \mathrm{WF36}(k), \theta^2(c_i, p_i))\mu_i(z)}{\Sigma_{i=1}^N \mu_i(z)} \tag{19}$$

$$\mu_i(z) = \prod_{j=1}^{14} \exp\left(-\frac{1}{2}\left(\frac{z_j - c_j^i}{\sigma_j^i}\right)^2\right) \tag{20}$$

where $i = 1, 2, \ldots, N$ represent N different local models and $z = [c^\top, p^\top]^\top$ is the premise input vector for the interpolating functions $\mu_i(z)$, which are Gaussian-shaped functions. By recruitment of new local models, whose locations are specified using the heuristics from the

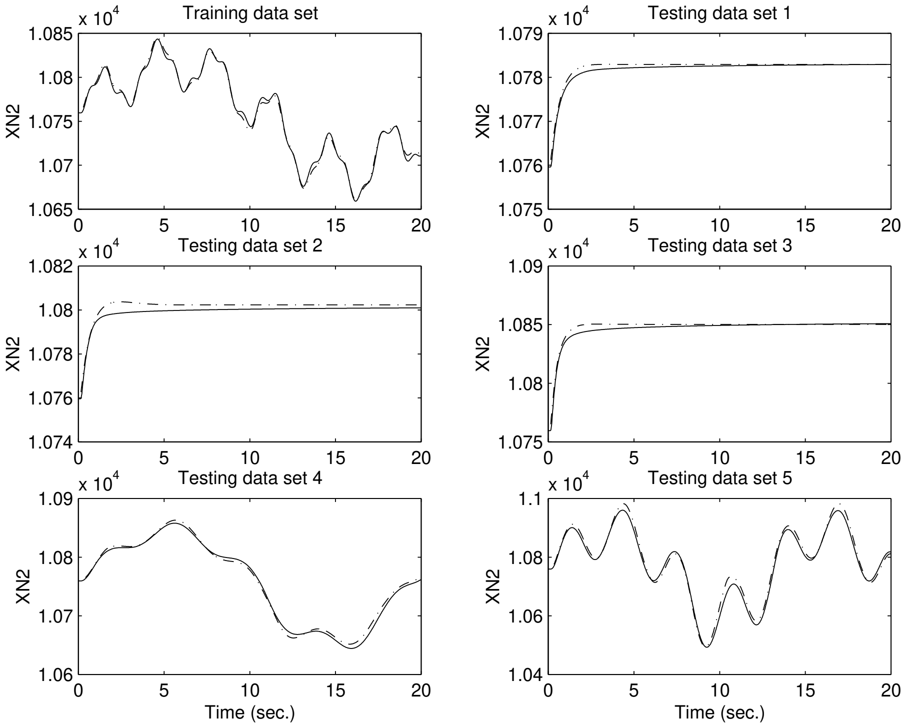

Figure 9: Performance of the local model.

immune system, the structure of the regional model is adjusted (i.e., enlarged) to accommodate more tunable parameters. The parameters of the newly-added local models are tuned by the Levenberg-Marquardt method. Furthermore, the shapes (or relative width) of the interpolating functions are optimized using the Levenberg-Marquardt method so as to obtain a well-allocated regional model.

The performance of the engine model becomes better along with the structure adaptation when more appropriate local models are recruited and the interpolating functions are optimized. This hybrid learning process finally results in a regional model with 64 local models, which we think is accurate enough to represent the engine dynamics for theoretical studies, and its AMSE is 15.3244. To illustrate the performance of the immunity-based hybrid learning process, we also implement a regional model by interpolating on a grid ($4 \times 4 \times 4$) of 64 local models, whose AMSE is 19.5777. A typical performance comparison of the regional model generated by the hybrid learning process and the grid process is also shown in Figure 10, where the solid line represents the system response of the CLM, the dashed line represents the response of the regional model generated by the hybrid learning process, and the dotted line represents the response of the model generated by the grid process.

Why does the estimation accuracy degrade over time? Note that the analytical model is running in an "open loop manner" (as the parallel identification model), that is, the outputs of the model will be fed back into the model as the inputs, and only the basic behavior of the engine may be obtained due to the drifting effects caused by the accumulation of approximation

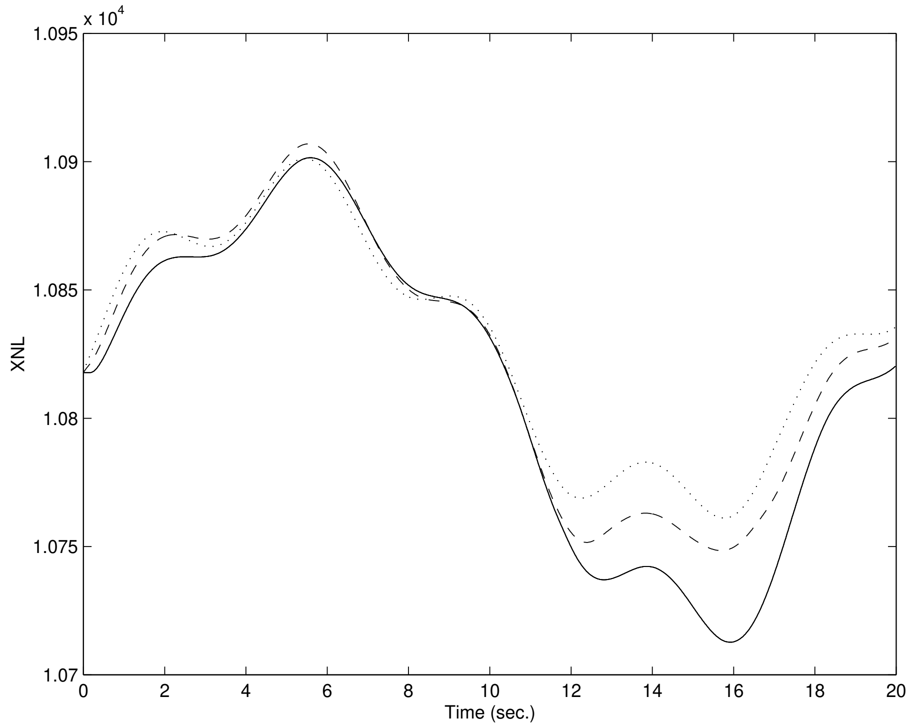

Figure 10: Performance of the regional model.

errors. We use this approach because sometimes we want this analytical model to be running as the "truth model" of the engine in the simulation study, where the CLM and thus the engine states XNL and XNH are not available. Of course, when we utilize the model (e.g., as the known dynamics of the nominal engine in our fault tolerant controller) to control the CLM, we may use the (measurable) real engine states to be the inputs of fuzzy systems and may expect an improvement in model accuracy.

6 Conclusion

Artificial immune systems have received increasing attention and have recently been successfully used in a variety of applications. From the point of view of information processing the immune system is a highly parallel and distributed intelligent system, which has the capabilities of learning, memory, and associative retrieval. In this paper we presented two immunity-based hybrid learning approaches for function approximation problems that interpolate data by means of spatially localized models (e.g., radial basis function networks or Takagi-Sugeno fuzzy systems). The number and centers of the receptive fields for local models are specified by immunity-based structure adaptation algorithms, while the parameters of the local models, which enter in a linear fashion, are tuned separately using a least squares method. As a result, we obtain a hybrid learning system that can perform structure and parameter adaptation. The effectiveness of the methods was demonstrated through a nonlinear function approximation problem and a nonlinear dynamical system modeling problem.

References

[1] A. Barron, "Universal approximation bounds for superpositions of a sigmoid function," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 930–945, 1993.

[2] T. Kwok and D. Yeung, "Constructive algorithms for structure learning in feedforward neural networks for regression problems," *IEEE Transactions on Neural Networks*, vol. 8, no. 3, pp. 630–645, 1997.

[3] S. Fahlman and C. Lebiere, "The cascade-correlation learning architecture," in *Advances in Neural Information Processing Systems 2, ed. D.S. Touretzky, San Mateo, CA: Morgan Kaufmann*, pp. 524–532, 1990.

[4] J. Friedman and W. Stutzle, "Projection pursuit regression," *Journal of the American Statistical Association, Theory and Models*, vol. 76, pp. 817–823, 1981.

[5] T. Ash, "Dynamic node creation in backpropagation networks," *Connection Science*, vol. 1, pp. 365–375, 1989.

[6] S. Hanson, "Meiosis networks," in *Advances in Neural Information Processing Systems 2, ed. D.S. Touretzky, San Mateo, CA: Morgan Kaufmann*, pp. 533–541, 1990.

[7] J. Platt, "A resource-allocating network for function interpolation," *Neural Computation*, vol. 3, pp. 213–225, 1991.

[8] S. Fabri and V. Kadirkamanathan, "Dynamic structure neural networks for stable adaptive control of nonlinear systems," *IEEE Transactions on Neural Networks*, vol. 7, no. 5, pp. 1151–1167, 1996.

[9] R. Reed, "Pruning algorithms - a survey," *IEEE Transactions on Neural Networks*, vol. 4, no. 5, pp. 740–747, 1993.

[10] B. Hassibi and D. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *Advances in Neural Information Processing Systems 5, ed. S.J. Hanson, J.D. Cowan, and C.L. Giles, San Mateo, CA: Morgan Kaufmann*, 1993.

[11] A. Weigend, D. Rumelhart, and B. Huberman, "Backpropagation, weight-elimination and time series prediction," in *Connectionist Models: Proceedings of the 1990 Summer School, ed. D.S. Touretzky, San Mateo, CA: Morgan Kaufmann*, pp. 105–116, 1991.

[12] J. Sietsma and R. Dow, "Creating artificial neural networks that generalize," *Neural Networks*, vol. 4, no. 1, pp. 67–69, 1991.

[13] J. Kruschke, "Creating local and distributed bottlenecks in hidden layers of back-propagation networks," in *Proceedings of the 1988 Connectionist Models Summer School, ed. D.S. Touretzky, G. Hinton, and T. Sejnowski, San Mateo, CA: Morgan Kaufmann*, pp. 120–126, 1988.

[14] T. Ash and G. Cottrell, "Topology-modifying neural network algorithms," in *The Handbook of Brain Theory and Neural Networks, ed. M. A. Arbib, MIT Press, Cambridge, MA*, pp. 990–993, 1998.

[15] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.

[16] S. Sin and R. Defigueriredo, "Fuzzy system design through fuzzy clustering and optimal predefuzzification," in *Proceedings of the 2nd IEEE Conference on Fuzzy Systems, San Francisco, CA*, pp. 190–195, 1993.

[17] E. Laukonen and K. Passino, "Training fuzzy systems to perform estimation and identification," *Engineering Applications of Artificial Intelligence*, vol. 8, no. 5, pp. 499–514, 1995.

[18] L. Wang and J. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 6, pp. 1414–1427, 1992.

[19] D. Dasgupta, ed., *Artificial Immune Systems and Their Applications.* Germany: Springer-Verlag, 1999.

[20] H. Bersini, "The endogenous double plasticity of the immune network and the inspiration to be drawn for engineering artifacts," in *Artificial Immune Systems and Their Applications, ed. D. Dasgupta, Springer-Verlag, Germany*, pp. 22–44, 1999.

[21] G. Seront and H. Bersini, "An algorithm for self-structuring neural net classifiers," in *Proceedings of the Second IEEE Conference on Neural Networks*, pp. 1307–1312, 1994.

[22] H. Bersini, "Reinforcement and recruitment learning for adaptive process control," in *Proceedings of the 1992 IFAC/IFIP/IMACS on Artificial Intelligence in Real Time Control*, pp. 331–337, 1992.

[23] H. Bersini and F. Varela, "The immune learning mechanisms: reinforcement, recruitment and their applications," in *Computing with Biological Metaphors, ed. R. Patton, Chapman and Hall, London*, pp. 166–192, 1994.

[24] K. M. Passino and S. Yurkovich, *Fuzzy Control.* Menlo Park, CA: Addison Wesley Longman, 1998.

[25] N. K. Jerne, "Towards a network theory of the immune system," *Annales Immunology*, vol. 125C, pp. 373–389, 1974.

[26] A. S. Perelson, "Immune network theory," *Immunological Review*, vol. 110, pp. 5–36, 1989.

[27] J. D. Farmer, N. H. Packard, and A. S. Perelson, "The immune system, adaptation, and machine learning," *Physica*, vol. 22D, pp. 187–204, 1986.

[28] J. D. Farmer, "A rosetta stone for connectionism," *Physica D*, vol. 42, pp. 153–187, 1990.

[29] J. E. Hunt, D. E. Cooke, and H. Holstein, "Case memory and retrieval based on the immune system," in *First International Conference on Case Based Reasoning*, pp. 205–216, 1995.

[30] A. Ishiguro, T. Kondo, Y. Watanabe, and Y. Uchikawa, "Immunoid: An immunological approach to decentralized behavior arbitration of autonomous mobile robots," *Lecture Notes in Computer Science*, vol. 1141, pp. 666–675, 1994.

[31] S. Schaal and C. Atkeson, "Constructive incremental learning from only local information," *Neural Computation*, vol. 10, pp. 2047–2084, 1998.

[32] S. Adibhatla and T. Lewis, "Model-based intelligent digital engine control," in *AIAA-97-3192, 33rd Joint Propulsion Conference*, 1997.

[33] Y. Diao and K. M. Passino, "Fault tolerant stable adaptive fuzzy/neural control for a turbine engine," *to appear in IEEE Transactions on Control Systems Technology.*