
Article:
Liu, G.P.P., Kadirkamanathan, V. and Billings, S.A. (1999) Variable neural networks for adaptive control of nonlinear systems. IEEE Transactions on Systems Man and Cybernetics Part C: Applications and Reviews, 29 (1). pp. 34-43. ISSN 1094-6977

https://doi.org/10.1109/5326.740668

Variable Neural Networks for Adaptive Control of Nonlinear Systems

Guoping P. Liu, *Member, IEEE*, Visakan Kadirkamanathan, *Member, IEEE*, and Stephen A. Billings

***Abstract*— This paper is concerned with the adaptive control of continuous-time nonlinear dynamical systems using neural networks. A novel neural network architecture, referred to as a variable neural network, is proposed and shown to be useful in approximating the unknown nonlinearities of dynamical systems. In the variable neural networks, the number of basis functions can be either increased or decreased with time according to specified design strategies so that the network will not overfit or underfit the data set. Based on the Gaussian radial basis function (GRBF) variable neural network, an adaptive control scheme is presented. The location of the centers and the determination of the widths of the GRBF's in the variable neural network are analyzed to make a compromise between orthogonality and smoothness. The weight adaptive laws developed using the Lyapunov synthesis approach guarantee the stability of the overall control scheme, even in the presence of modeling error. The tracking errors converge to the required accuracy through the adaptive control algorithm derived by combining the variable neural network and Lyapunov synthesis techniques. The operation of an adaptive control scheme using the variable neural network is demonstrated using two simulated examples.**

***Index Terms*— Adaptive control, neural networks, nonlinear systems, radial basis functions.**

I. Introduction

NEURAL networks are capable of learning and reconstructing complex nonlinear mappings and have been widely studied by control researchers in the identification analysis and design of control systems. A large number of control structures have been proposed, including supervised control [55], direct inverse control [34], model reference control [39], internal model control [13], predictive control [14], [56], [29], gain scheduling [12], optimal decision control [10], adaptive linear control [7], reinforcement learning control [1], [3], variable structure control [30], indirect adaptive control [39], and direct adaptive control [19], [45], [50], [51]. The principal types of neural networks used for control problems are the multilayer perceptron (MLP) neural networks with sigmoidal units [34], [39], [48] and the radial basis function (RBF) neural networks [41], [43], [47].

Manuscript received September 4, 1995; revised September 9, 1996 and November 8, 1997. This work was supported by the Engineering and Physical Sciences Research Council (EPSRC) of the United Kingdom under Contract GR/J46661.

G. P. Liu is with the ALSTOM Energy Technology Centre, LE8 6LH Leicester, U.K. (e-mail: guoping.liu@energy.alstom.com).

V. Kadirkamanathan and S. A. Billings are with the Department of Automatic Control and Systems Engineering, University of Sheffield, S1 3JD Sheffield, U.K.

Publisher Item Identifier S 1094-6977(99)00100-5.

Most of the neural network-based control schemes view the problem as deriving adaptation laws using a fixed structure neural network. However, choosing this structure, such as the number of basis functions (hidden units in a single hidden layer) in the neural network, must be done *a priori*. This can often lead to either an overdetermined or an underdetermined network structure. In the discrete-time formulation, some approaches have been developed to determine the number of hidden units (or basis functions) using decision theory [4] and model comparison methods, such as *minimum description length* [54] and *Bayesian methods* [33]. The problem with these methods is that they require all observations to be available and hence are not suitable for online control tasks, especially adaptive control. In addition, the fixed structure neural networks often need a large number of basis functions even for simple problems.

Another type of neural network structure developed for learning systems is to begin with a larger network and then to prune this [32], [36], or to begin with a smaller network and then to expand this [9], [42] until the optimal network complexity is found. Among these dynamic structure models, the *resource allocating network* (RAN) developed by Platt [42] is an online or sequential identification algorithm. The RAN is essentially a growing Gaussian radial basis function (GRBF) network whose growth criteria and parameter adaptation laws have been studied and extended further [20], [21], [31] and applied to time-series analysis [24] and pattern classification [23]. The RAN and its extensions addressed the identification of only autoregressive systems with no external inputs and hence stability was not an issue. Recently, the growing GRBF neural network has been applied to sequential identification and adaptive control of dynamical continuous nonlinear systems with external inputs [8], [22], [27], [28]. Though the growing neural network is much better than the fixed neural network in reducing the number of basis functions, it is still possible that this network will induce an overfitting problem. There are two main reasons for this. It is difficult to know how many basis functions are really needed for the problem, and secondly, the nonlinearity of a nonlinear function to be modeled is different when its variables change their value ranges. Normally, the number of basis functions in the growing neural network may increase to the one that the system needs to meet the requirement for dealing with the most complicated nonlinearity (the worst case) of the nonlinear function. Thus, it may lead to a network that has the same size as the fixed neural networks.

To overcome the above limitations, a new network structure, referred to as the variable neural network, is proposed in this paper. The basic principle of the variable neural network is that the number of basis functions in the network can be either increased or decreased over time according to a design strategy in an attempt to avoid overfitting or underfitting. In order to model unknown nonlinearities, the variable neural network starts with a small number of initial hidden units and then adds or removes units located in a variable grid. This grid consists of a number of subgrids composed of different sized hypercubes that depend on the novelty of the observation. Since the novelty of the observation is tested, it is ideally suited for online control problems. The objective behind the development is to gradually approach the appropriate network complexity that is sufficient to provide an approximation to the system nonlinearities and consistent with the observations being received. By allocating GRBF units on a variable grid, only the relevant state-space traversed by the dynamical system is spanned, resulting in considerable savings on the size of the network.

The parameters of the variable neural network are adjusted by adaptation laws developed using the Lyapunov synthesis approach. Combining the variable neural network and Lyapunov synthesis techniques, the adaptive control algorithm developed for continuous dynamical nonlinear systems guarantees the stability of the whole control scheme and the convergence of the tracking errors between the reference inputs and the outputs.

The remainder of the paper is organized as follows. In Section II, the modeling of nonlinear dynamical systems by the GRBF network is discussed and a one-to-one mapping of the state-space to form a compact network input space is introduced. In Section III, a variable neural network is developed, based on a proposed variable grid. The selection of the GRBF's for the variable neural network is discussed. The adaptive control scheme using the variable neural networks and the Lyapunov synthesis techniques is developed in Section IV. The stability of the overall control scheme and the convergence of the tracking errors are also analyzed. The operation of the adaptive control scheme is demonstrated by two simulated examples in Section V.

II. Nonlinear System Modeling

Consider a class of continuous nonlinear dynamical systems that can be expressed in the canonical form [18], [40], [53]

$$y^{(n)}(t)+F(y^{(n-1)}(t),\ldots,y^{(1)}(t),y(t)) = G(y^{(n-1)}(t),\ldots,y^{(1)}(t),y(t))u(t) \quad (1)$$

where $y(t)$ is the output, $u(t)$ is the control input, $y^{(i)}$ is the ith derivative of the output with respect to time, and $F(\cdot)$ and $G(\cdot)$ are unknown nonlinear functions. The above system represents a class of continuous-time nonlinear systems, called affine systems. The above equation can also be transformed to the state-space form

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} - \mathbf{b}F(\mathbf{x}) + \mathbf{b}G(\mathbf{x})u \quad (2)$$

$$y = x_1 \quad (3)$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & \mathbf{I}_{n-1} \\ 0 & 0 \end{bmatrix}. \quad (4)$$

$\mathbf{b}=[0,0,\ldots,1]^T$, $\mathbf{I}_{n-1}$ is an $(n-1)\times(n-1)$ identity matrix, and $\mathbf{x}=[x_1,x_2,\ldots,x_n]^T$ is the state vector.

Due to some desirable features, such as local adjustment of weights and mathematical tractability, RBF networks have recently attracted considerable attention (see, for example, [2], [5], [6], and [26]). Their importance has also greatly benefited from the work of Moody and Darken [35] and Poggio and Girosi [44], who explored the relationship between regularization theory and RBF networks. The good approximation properties of the RBF's in interpolation have been well studied by Powell and his group [47]. With the use of Gaussian activation functions, each basis function in the RBF network responds only to inputs in the neighborhood determined by the center and width of the function. It is also known that, if the variables of a nonlinear function are in compact sets, the continuous function can be approximated arbitrarily well by GRBF networks [43]. Here, the GRBF networks are used to model the nonlinearity of the system.

If x_i is not in a certain range, we introduce the following one-to-one (1-1) mapping [27]:

$$\bar{x}_i = \frac{b_{xi}x_i}{|x_i|+a_{xi}} \quad \text{for } i=1,2,\ldots,n \quad (5)$$

where a_{xi},b_{xi} are positive constants, which can be chosen by the designer (e.g., a_{xi},b_{xi} are one). Thus, it is clear from (5) that $\bar{x}_i \in [-b_{xi},b_{xi}]$ for $x_i \in (-\infty,+\infty)$. The above one-to-one mapping shows that, in the n-dimensional space, the entire area can be transferred into an n-dimensional hypercube denoted by the compact set $\mathcal{X}$. Clearly, if $\mathbf{x}$ is already in the desired area, we only need to set $\bar{\mathbf{x}}=\mathbf{x}$.

Thus, the nonlinear part $G(\mathbf{x})u-F(\mathbf{x})$ of the system can be described by the following GRBF network:

$$G(\mathbf{x})u-F(\mathbf{x}) = (\mathbf{g}^*(K)u-\mathbf{f}^*(K))^T\Phi(\bar{\mathbf{x}},K)+\varepsilon(K) \quad (6)$$

where

$$\Phi(\bar{\mathbf{x}},K)=[\phi(\bar{\mathbf{x}};\mathbf{c}_1,d_1),\phi(\bar{\mathbf{x}};\mathbf{c}_2,d_2),\ldots,\phi(\bar{\mathbf{x}};\mathbf{c}_K,d_K)]^T \quad (7)$$

$$\phi(\bar{\mathbf{x}};\mathbf{c}_i,d_i)=\exp\left\{-\frac{1}{d_i^2}\|\bar{\mathbf{x}}-\mathbf{c}_i\|^2\right\}, \quad \text{for } i=1,2,\ldots,K. \quad (8)$$

$\mathbf{f}^*(K)=[f_1^*,f_2^*,\ldots,f_K^*]^T$ and $\mathbf{g}^*(K)=[g_1^*,g_2^*,\ldots,g_K^*]^T$ are the optimal weight vectors, $\bar{\mathbf{x}}=[\bar{x}_1,\bar{x}_2,\ldots,\bar{x}_n]^T$ is the variable vector, $\mathbf{c}_i$ is the ith center, d_i is the ith width, $\varepsilon(K)$ is the modeling error, and K is the number of the basis functions.

It is known from approximation theory that the modeling error can be reduced arbitrarily by increasing the number K, i.e., the number of the linear independent basis functions $\phi(\bar{\mathbf{x}};\mathbf{c}_i,d_i)$ in the network model. Thus, it is reasonable to assume that the modeling error $\varepsilon(K)$ is bounded by a constant ε_K, which represents the accuracy of the model, and this is defined as

$$\varepsilon_K = \sup_{t\in\Re^+}|\varepsilon(K)|. \quad (9)$$

Although ε_K can be reduced arbitrarily by increasing the number of the independent basis functions, generally, when the number is greater than a small value, the modeling error ε_K is improved very little by increasing the number further. It also results in a large-sized network even for a simple problem. In practice, this is not realistic. In most cases, the required modeling error can be given by considering the design requirements and specifications of the system. Thus, the problem now is to find a suitable-sized network to achieve the required modeling error. In other words, it is how to determine the number, centers, widths, and weights of the GRBF's in the network.

III. Variable Neural Networks

Two main neural network structures that are widely used in online identification and control are the fixed neural network and the growing neural network. The fixed neural network usually needs a large number of basis functions in most cases even for a simple problem. Though the growing network is much better than the fixed network in reducing the number of the basis functions for a number of problems, it is still possible that this network will lead to an overfitting problem for some cases, and this is explained in Section I. To overcome the above limitations of the fixed and growing neural networks, a new network structure, called the variable neural network, is proposed in this section.

A. Variable Grid

In GRBF networks, the very important parameter is the location of the centers of the GRBF's over the compact set $\mathcal{X}$, which is the approximation region. Usually, an n-dimension grid is used to locate all centers in the gridnodes [51]. Thus, the distance between the gridnodes affects the size of the networks and the approximation accuracy. In other words, a large distance leads to a small network and a coarser approximation, while a small distance results in a large size network and a finer approximation. However, even if the required accuracy is given, it is very difficult to know how small the distance should be since the underlying function is unknown. Also, the nonlinearity of the system is not uniformly complex over the set $\mathcal{X}$. So, here a variable grid is introduced for the location of the centers of all GRBF's in the network.

The variable grid consists of a number of different subgrids. Each subgrid is composed of equally sized n-dimensional hypercuboids. It implies that the number of the subgrids can increase or decrease with time in the grid according to a design strategy. All subgrids are named, the initial grid is named the first-order subgrid, then the second-order subgrid, and so on. In each subgrid, there are a different number of nodes, which are denoted by their positions. Let $\mathcal{N}_i$ denote the set of nodes in the ith-order subgrid. Thus, the set of all nodes in the grid with m subgrids is

$$\mathcal{N} = \bigcup_{i=1}^{m} \mathcal{N}_i. \tag{10}$$

To increase the density of the gridnodes, the edge lengths of the hypercubes of the ith-order subgrid will always be less than those of the $(i-1)$th-order subgrid. Hence, the higher order subgrids have more nodes than the lower order ones. On the other hand, to reduce the density of the gridnodes, always remove some subgrids from the grid until a required density is reached.

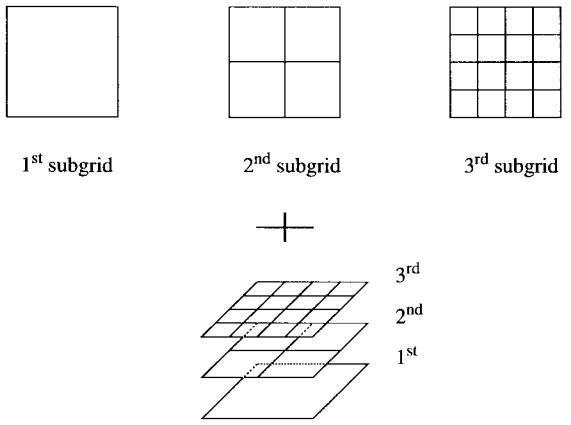


Fig. 1. Variable grid with three subgrids.

Let all elements of the set $\mathcal{N}$ represent the possible centers of the network. So, the more subgrids, the more possible centers. Since the higher order subgrids probably have some nodes that are the same as the lower order subgrids, the set of the new possible centers provided by the ith order subgrid is defined as

$$\mathcal{P}_i = \{\mathbf{c} : \mathbf{c} \in \mathcal{N}_i \text{ and } \mathbf{c} \notin \mathcal{P}_j \text{ for } j = 1, 2, \ldots, i-1\} \tag{11}$$

where $\mathcal{P}_0$ is an empty set. It shows that the possible center set $\mathcal{P}_i$ corresponding to the ith subgrid does not include those that are given by the lower order subgrids, i.e.,

$$\bigcap_{j=1}^{i} \mathcal{P}_j = \emptyset. \tag{12}$$

For example, in the two-dimensional (2-D) case, let the edge length of rectangulars on the ith subgrid be half of the $(i-1)$th subgrid. The variable grid with three subgrids is shown in Fig. 1.

B. Variable Network

The variable neural network has the property that the number of the basis functions in the network can be either increased or decreased over time according to a design strategy.

For the problem of nonlinear modeling with neural networks, the variable network is initialized with a small number of basis function units. As observations are received, the network grows by adding some new basis functions or is pruned by removing some old ones.

To add new basis functions to the network the following two conditions must be satisfied: 1) the modeling error must be greater than the required accuracy and 2) the period between the two adding operations must be greater than the minimum response time to the adding operation.

To remove some old basis functions from the network, the following two conditions must be satisfied: 1) the modeling

error must be less than the required accuracy and 2) the period between the two removing operations must be greater than the minimum response time of the removing operation.

It is known that if the grid consists of the same size n-dimension hypercubes with the edge length vector $\rho = [\rho_1, \rho_2, \ldots, \rho_n]$, the accuracy of approximating a function is in direct proportion to the norm of the edge length vector of the grid [46], i.e.,

$$\varepsilon_K \propto ||\rho||. \tag{13}$$

Therefore, based on the variable grid, the structure of a variable neural network is proposed here. The network selects the centers from the node set $\mathcal{N}$ of the variable grid. When the network needs some new basis functions, a new higher order subgrid (say, $(m+1)$th subgrid) is appended to the grid. The network chooses the new centers from the possible center set $\mathcal{P}_{m+1}$ provided by the newly created subgrid. Similarly, if the network needs to be reduced, the highest order subgrid (say, mth subgrid) is deleted from the grid. Meanwhile, the network removes the centers associated with the deleted subgrid. In this way, the network is kept to a suitable size. How to locate the centers and determine the widths of the GRBF's is discussed in the next section.

C. Selection of Basis Functions

It is also known that the GRBF has a localization property that the influence area of the kth basis function is governed by the center $\mathbf{c}_k$ and width d_k. In other words, once the center $\mathbf{c}_k$ and the width d_k are fixed, the influence area of the GRBF $\phi(\bar{\mathbf{x}}; \mathbf{c}_k, d_k)$ is limited in the state-space to the neighborhood of $\mathbf{c}_k$.

On the basis of the possible center set $\mathcal{N}$ produced by the variable grid, there are large number of basis function candidates, denoted by the set $\mathcal{B}$. During the system operation, the state vector $\bar{\mathbf{x}}$ will gradually scan a subset of the state-space set $\mathcal{X}$. Since the basis functions in the GRBF network have a localized receptive field, if the neighborhood of a basis function $\phi \in \mathcal{B}$ is located "far away" from the current state $\bar{\mathbf{x}}(t)$, its influence to the approximation is very small and could be ignored by the network. On the other hand, if the neighborhood of a basis function $\phi \in \mathcal{B}$ is near to or covers the current state $\bar{\mathbf{x}}(t)$, it will play a very important role in the approximation. Thus, it should be kept if it is already in the network or added into the network if it is not in.

Given any point $\bar{\mathbf{x}}$, the nearest node $\bar{\mathbf{x}}_i^+ = [\bar{x}_{i1}^+, \bar{x}_{i2}^+, \ldots, \bar{x}_{in}^+]^T$ to it in the ith subgrid can be calculated by

$$\bar{x}_{ij}^+ = \text{round}\left(\frac{\bar{x}_j}{\delta_{ij}}\right)\delta_{ij} \tag{14}$$

for $j = 1, 2, \ldots, n$, where round$(\cdot)$ is an operator for rounding the number $(\cdot)$ to the nearest integer; for example, round$(2.51) = 3$, and δ_{ij} is the edge length of the hypercube corresponding to the jth element of the vector $\bar{\mathbf{x}}$ in the ith subgrid. Without lose of generality, let $\delta_i = \delta_{i1} = \delta_{i2} = \cdots = \delta_{in}$.

Define m hyperspheres corresponding to the m subgrids, respectively

$$\mathcal{H}_i(\bar{\mathbf{x}}_i^+, \sigma_i) = \left\{ \bar{\mathbf{x}} : \sum_{j=1}^{n} \left(x_{ij} - \bar{x}_{ij}^+\right)^2 \le \sigma_i^2 \right\} \tag{15}$$

for $i = 1, 2, \ldots, m$, where σ_i is the radius of the ith hypersphere. In order to get a suitable-sized variable network, choose the centers of the basis functions from the nodes contained in the different hyperspheres $\mathcal{H}_i(\bar{\mathbf{x}}_i^+, \sigma_i)$, which are centered in the nearest nodes $\bar{\mathbf{x}}_i^+$ to $\bar{\mathbf{x}}$ in the different subgrids with radius σ_i, for $i = 1, 2, \ldots, m$. For the sake of simplicity, it is assumed that the basis function candidates whose centers are in the set $\mathcal{P}_i$ have the same width d_i and $d_i < d_{i-1}$. Thus, for the higher order subgrids, use the smaller radius, i.e.,

$$\sigma_m < \sigma_{m-1} < \cdots < \sigma_1. \tag{16}$$

Usually, choose

$$\sigma_i = \gamma_1 \sigma_{i-1} \tag{17}$$

where γ_1 is a constant and less than one. Thus, the chosen centers from the set $\mathcal{P}_i$ are given by the set

$$\mathcal{C}_i = \left\{ \mathbf{c} : \mathbf{c} \in \mathcal{P}_i \text{ and } \mathbf{c} \in \mathcal{H}_i\left(\bar{\mathbf{x}}_i^+, \sigma_i\right) \right\}. \tag{18}$$

In order that the basis function candidates in the set $\mathcal{P}_i$ that are less than an activation threshold to the nearest grid node $\bar{\mathbf{x}}_i^+$ in the ith subgrid are outside the set $\mathcal{H}_i(\bar{\mathbf{x}}_i^+, \sigma_i)$, it can be deduced from (8) and (15) that the σ_i must be chosen to be

$$\sigma_i \ge \sqrt{\log(\delta_{\min}^{-1})}\, d_i \tag{19}$$

for $i = 1, 2, \ldots, m$, where $\delta_{\min} \in (0, 1)$ represents the activation threshold.

Thus, the center set of the network is given by the union of the center sets $\mathcal{C}_i$, for $i = 1, 2, \ldots, m$, that is

$$\mathcal{C} = \bigcup_{i=1}^{m} \mathcal{C}_i. \tag{20}$$

For example, in the 2-D case, the radii are chosen to be the same as the edge lengths of the squares in the subgrids, that is

$$\sigma_i = \delta_i, \quad \text{for } i = 1, 2, \ldots, m. \tag{21}$$

The chosen centers in the variable grid with four subgrids are as shown in Fig. 2.

Now, consider how to choose the width d_k of the kth basis function. The angle between the two GRBF's $\phi(\mathbf{x}; \mathbf{c}_i, d_i)$ and $\phi(\mathbf{x}; \mathbf{c}_j, d_j)$ is defined as in (22), shown at the bottom of the

$$\theta_{ij} = \cos^{-1}\left(\frac{\langle \phi(\mathbf{x}; \mathbf{c}_i, d_i), \phi(\mathbf{x}; \mathbf{c}_j, d_j) \rangle}{\langle \phi(\mathbf{x}; \mathbf{c}_i, d_i), \phi(\mathbf{x}; \mathbf{c}_i, d_i) \rangle^{1/2} \langle \phi(\mathbf{x}; \mathbf{c}_j, d_j), \phi(\mathbf{x}; \mathbf{c}_j, d_j) \rangle^{1/2}} \right) \tag{22}$$

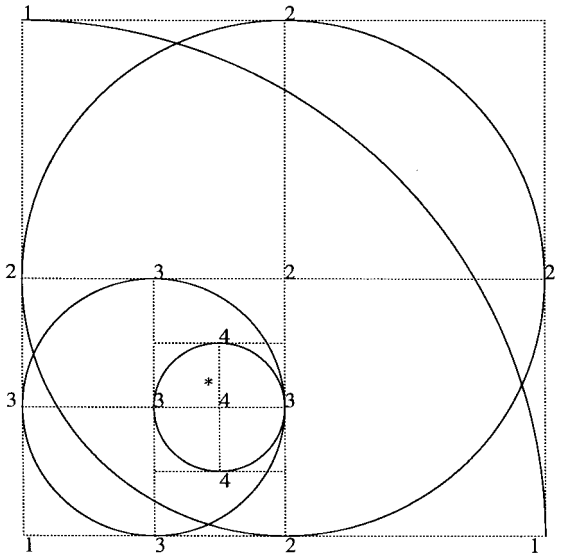


Fig. 2. Location of centers in the variable grid with four subgrids. The number i $(i = 1, 2, 3, 4)$ denotes the centers chosen from the ith subgrid.

previous page, where $\langle \cdot, \cdot \rangle$ is the inner product in the space of square-integrable functions, which is defined as

$$\langle \phi(\mathbf{x}; \mathbf{c}_i, d_i), \phi(\mathbf{x}; \mathbf{c}_j, d_j) \rangle = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \phi(\mathbf{x}; \mathbf{c}_i, d_i) \phi(\mathbf{x}; \mathbf{c}_j, d_j)\, dx_1\, dx_2 \cdots dx_n. \tag{23}$$

The angle can be given by [20]

$$\theta_{ij} = \cos^{-1}\left(\left(\frac{2\sqrt{\xi}}{\xi + 1} \right)^{\frac{n}{4}} \phi(\mathbf{c}_j; \mathbf{c}_i, d_i)^{\frac{\xi}{2\xi+2}} \right) \tag{24}$$

where $\xi = d_i^2/d_j^2$. It shows from the above that the $\cos(\theta_{ij})$ depends on three factors: the dimension n, the width ratio ξ, and the output of a basis function at the center of the other basis function $\phi(\mathbf{c}_j; \mathbf{c}_i, d_i)$.

If the centers of the two basis functions are chosen from the same subgrid, i.e., $\xi = 1$, it is clear from (24) that

$$\cos(\theta_{ij}) = \phi(\mathbf{c}_j; \mathbf{c}_i, d_i)^{\frac{1}{4}}. \tag{25}$$

On the other hand, if the centers of the two basis functions are from different subgrids, it is possible that their centers are very close. The worst case will be when $\phi(\mathbf{c}_j; \mathbf{c}_i, d_i)$ is near to one. In this case, the angle between the two basis functions can be written as

$$\cos(\theta_{ij}) \leq \left(\frac{2\sqrt{\xi}}{\xi + 1} \right)^{\frac{n}{4}}. \tag{26}$$

Given the center $\mathbf{c}_k$, in order to assign a new basis function $\phi(\mathbf{x}; \mathbf{c}_k, d_k)$ that is nearly orthogonal to all existing basis functions, the angle between the GRBF's should be as large as possible. The width d_k should therefore be reduced. However, reducing d_k increases the curvature of $\phi(\mathbf{x}; \mathbf{c}_k, d_k)$ that in turn gives a less smooth function and can lead to *overfitting* problems. Thus, to make a tradeoff between the orthogonality and the smoothness, it can be deduced from (25) and (26) that the width d_k, which ensures the angles between GRBF units are not less than the required minimum angle $\theta_{\min}$ should satisfy

$$\xi \geq \left(\frac{1 + \sqrt{1 - \cos^{8/n}(\theta_{\min})}}{\cos^{4/n}(\theta_{\min})} \right)^2 \tag{27}$$

or

$$\xi \leq \left(\frac{1 - \sqrt{1 - \cos^{8/n}(\theta_{\min})}}{\cos^{4/n}(\theta_{\min})} \right)^2 \tag{28}$$

and

$$d_k \leq \sqrt{\log^{-4}\cos(\theta_{\min})}\,\delta_k. \tag{29}$$

For example, assume that the s_0 satisfies (27). If the width of the basis functions whose centers are located in the set $\mathcal{C}_i$, which corresponds to the ith subgrid with $\delta_i = \xi_0 \delta_{i-1}$, is chosen to be $d_i = \xi_0 d_{i-1}$ and the width d_1 of the basis functions associated to the initial grid satisfies

$$d_1 \leq \sqrt{\log^{-4}\cos(\theta_{\min})}\,\delta_1 \tag{30}$$

then the smallest angle between all basis functions are not less than the required minimum angle $\theta_{\min}$.

Therefore, based on a variable grid with m subgrids, the nonlinear function approximated by the GRBF network in (6) can also be expressed by

$$G(\mathbf{x})u - F(\mathbf{x}) = \sum_{i=1}^{m} \sum_{j=1}^{m_i} (f_{i+j}^* + g_{i+j}^* u) \phi(\bar{\mathbf{x}}, \mathbf{c}_{i+j}, d_i) + \varepsilon(K) \tag{31}$$

where

$$K = \sum_{i=1}^{m} m_i. \tag{32}$$

$\mathbf{c}_{i+j}$ is the jth element of the set $\mathcal{C}_i$, m_i is the number of its elements, and f_{i+j}^* and g_{i+j}^* are the optimal weights. So, the next step is how to obtain the estimates of the weights.

IV. Adaptive Control

The stability of the overall control scheme is an important issue in the design of the system. The overall stability depends not only on the particular control approach that is chosen but also on the control laws that are used. In practice, one of the design objectives for a system is that the tracking error between inputs and outputs should converge to the required accuracy. Those problems are solved here by developing a stable adaptive control law based on Lyapunov stability techniques [25] and the variable GRBF network discussed in Section III.

A. Adaptation Laws

We assume that the basis functions $\phi(\bar{\mathbf{x}};\mathbf{c}_k,d_k)$ for $k=1,2,\ldots,K$ are given. Section IV-B will discuss how the basis functions of the network model are chosen.

The control objective is to force the plant state vector $\mathbf{x}$ to follow a specified desired trajectory $\mathbf{y_d}=[y_d,y_d^{(1)},\ldots,y_d^{(n-1)}]^T$. The tracking error vector and the weight error vectors, respectively, are defined as

$$\mathbf{e}=\mathbf{x}-\mathbf{y_d} \tag{33}$$

$$\tilde{\mathbf{f}}(K)=\mathbf{f}^*(K)-\mathbf{f}(K) \tag{34}$$

$$\tilde{\mathbf{g}}(K)=\mathbf{g}^*(K)-\mathbf{g}(K) \tag{35}$$

where $\mathbf{f}(K)$ and $\mathbf{g}(K)$ are the estimated weight vectors. From (1), it can be shown that

$$\begin{aligned}\dot{\mathbf{x}}=&\mathbf{A}\mathbf{x}+\mathbf{b}(\mathbf{g}^T(K)u-\mathbf{f}^T(K))\Phi(\bar{\mathbf{x}},K)\\&+\mathbf{b}(\tilde{\mathbf{g}}^T(K)u-\tilde{\mathbf{f}}^T(K))\Phi(\bar{\mathbf{x}},K)+\mathbf{b}\varepsilon(K).\end{aligned} \tag{36}$$

Hence, from (33)–(36), the dynamical expression of the tracking error is

$$\begin{aligned}\dot{\mathbf{e}}=&\mathbf{A}\mathbf{e}-by_d^{(n)}+\mathbf{b}(\mathbf{g}^T(K)u-\mathbf{f}^T(K))\Phi(\bar{\mathbf{x}},K)\\&+\mathbf{b}(\tilde{\mathbf{g}}^T(K)u-\tilde{\mathbf{f}}^T(K))\Phi(\bar{\mathbf{x}},K)+\mathbf{b}\varepsilon(K).\end{aligned} \tag{37}$$

One approach to this problem is to take the control input satisfying

$$\mathbf{g}^T(K)\Phi(\bar{\mathbf{x}},K)u=y_d^{(n)}+\mathbf{f}^T(K)\Phi(\bar{\mathbf{x}},K)+\mathbf{a}^T\mathbf{e} \tag{38}$$

where the vector $\mathbf{a}=[a_1,a_2,\ldots,a_n]^T$ makes the following matrix stable:

$$\mathbf{A}_a=\begin{bmatrix}0&1&0&\cdots&0\\0&0&1&\cdots&0\\\vdots&\vdots&\vdots&\vdots&\vdots\\-a_1&-a_2&-a_3&\cdots&-a_n\end{bmatrix} \tag{39}$$

i.e., all eigenvalues are in the open left plane. The control input consists of a linear combination of the tracking errors $\mathbf{a}^T\mathbf{e}$, the adaptive part $\mathbf{f}^T(K)\Phi(\bar{\mathbf{x}},K)$ that will attempt to estimate, and cancel, the unknown function $F(\cdot)$, and $y_d^{(n)}$ is a feedforward of the nth derivative of the desired trajectory.

Consider the following Lyapunov function:

$$V(\mathbf{e},\tilde{\mathbf{f}},\tilde{\mathbf{g}})=\mathbf{e}^T\mathbf{P}\mathbf{e}+\frac{1}{\alpha}\tilde{\mathbf{f}}^T(K)\tilde{\mathbf{f}}(K)+\frac{1}{\beta}\tilde{\mathbf{g}}^T(K)\tilde{\mathbf{g}}(K) \tag{40}$$

where $\mathbf{P}$ is chosen to be a positive definite matrix so that the matrix $\mathbf{Q}=-\mathbf{P}\mathbf{A}_a-\mathbf{A}_a^T\mathbf{P}$ is also a positive definite matrix and α and β are positive constants that will appear in the sequential adaptation laws, also referred to as the learning or adaptation rates. Using (37), the derivative of the Lyapunov function V with respect to time is given by

$$\begin{aligned}&\dot{V}(\mathbf{e},\tilde{\mathbf{f}},\tilde{\mathbf{g}})\\&\quad=-\mathbf{e}^T\mathbf{Q}\mathbf{e}+2\mathbf{P}_n^T\mathbf{e}((\tilde{\mathbf{g}}^T(K)u-\tilde{\mathbf{f}}^T(K))\Phi(\bar{\mathbf{x}},K)+\varepsilon(K))\\&\qquad+2\alpha^{-1}\tilde{\mathbf{f}}^T(K)\dot{\tilde{\mathbf{f}}}(K)+2\beta^{-1}\tilde{\mathbf{g}}^T(K)\dot{\tilde{\mathbf{g}}}(K)\end{aligned} \tag{41}$$

where the vector $\mathbf{P}_n$ is the nth row of the matrix $\mathbf{P}$, i.e., $\mathbf{P}_n=[p_{n1},p_{n2},\ldots,p_{nn}]$.

Since $\mathbf{f}^*$ is a constant vector, we have that $\dot{\tilde{\mathbf{f}}}=-\dot{\mathbf{f}}$, similarly, $\dot{\tilde{\mathbf{g}}}=-\dot{\mathbf{g}}$. If there is no modeling error, i.e., $\varepsilon(K)=0$, the weight vectors $\mathbf{f}$ and $\mathbf{g}$ can simply be generated according to the following standard adaptation laws: $\dot{\mathbf{f}}(K)=-\alpha\mathbf{P}_n^T\mathbf{e}\Phi(\bar{\mathbf{x}};K)$ and $\dot{\mathbf{g}}(K)=\beta\mathbf{P}_n^T\mathbf{e}u\Phi(\bar{\mathbf{x}};K)$. In the presence of a modeling error $\varepsilon(K)$, to ensure the stability of the system, a lot of algorithms, e.g., the fixed or switching σ-modification [16], [17], ε-modification [37], and the dead-zone methods [38], [52], can be applied to modify the above standard adaptation laws.

Define the following sets:

$$\mathcal{F}_1=\{\mathbf{f}:\|\mathbf{f}\|<M_1 \text{ or } (\|\mathbf{f}\|=M_1 \text{ and } \mathbf{P}_n^T\mathbf{e}\mathbf{f}^T\Phi(\bar{\mathbf{x}})\geq 0)\} \tag{42}$$

$$\mathcal{F}_2=\{\mathbf{f}:\|\mathbf{f}\|=M_1 \text{ and } \mathbf{P}_n^T\mathbf{e}\mathbf{f}^T\Phi(\bar{\mathbf{x}})<0\} \tag{43}$$

$$\begin{aligned}\mathcal{G}_1=\{&\mathbf{g}:\|\mathbf{g}\|<M_2 \text{ or}\\&(\|\mathbf{g}\|=M_2 \text{ and } \mathbf{P}_n^T\mathbf{e}u\mathbf{g}^T\Phi(\bar{\mathbf{x}})\leq 0)\}\end{aligned} \tag{44}$$

$$\mathcal{G}_2=\{\mathbf{g}:\|\mathbf{g}\|=M_2 \text{ and } \mathbf{P}_n^T\mathbf{e}u\mathbf{g}^T\Phi(\bar{\mathbf{x}})>0\} \tag{45}$$

where M_1 and M_2 are positive constants.

Here, in order to avoid parameter drift in the presence of modeling error, the application of the projection algorithm [11], [15], [45] gives the following adaptive laws for the parameter estimates $\mathbf{f}$ and $\mathbf{g}$, as in (46) and (47), shown at the bottom of the page. It is clear that, if the initial weights are chosen such that $\mathbf{f}(K,0)\in\mathcal{F}_1\cup\mathcal{F}_2$ and $\mathbf{g}(K,0)\in\mathcal{G}_1\cup\mathcal{G}_2$, the weight vectors $\mathbf{f}$ and $\mathbf{g}$ are confined to the sets $\mathcal{F}_1\cup\mathcal{F}_2$ and $\mathcal{G}_1\cup\mathcal{G}_2$, respectively. With use of the adaptive laws (46) and (47), (41) becomes

$$\dot{V}(\mathbf{e},\tilde{\mathbf{f}},\tilde{\mathbf{g}})\leq-\mathbf{e}^T\mathbf{Q}\mathbf{e}+2\sum_{i=1}^{n}|p_{ni}||e_i|\varepsilon_K. \tag{48}$$

For the sake of simplicity, the positive definite matrix $\mathbf{Q}$ is assumed to be diagonal, i.e., $\mathbf{Q}=\mathrm{diag}[q_1,q_2,\ldots,q_n]$, where $q_i>0$, for $i=1,2,\ldots,n$. Also define

$$\Theta(\zeta)=\left\{\mathbf{e}:\sum_{i=1}^{n}q_i\left(|e_i|-\frac{|p_{ni}|}{q_i}\zeta\right)^2\leq\sum_{i=1}^{n}\frac{p_{ni}^2\zeta^2}{q_i}\right\} \tag{49}$$

where ζ is a positive variable, i.e., $\zeta\geq 0$.

$$\dot{\mathbf{f}}(K)=\begin{cases}-\alpha\mathbf{P}_n^T\mathbf{e}\Phi(\bar{\mathbf{x}};K), & \text{if } \mathbf{f}(K)\in\mathcal{F}_1\\-\alpha\mathbf{P}_n^T\mathbf{e}\Phi(\bar{\mathbf{x}};K)+\alpha M_1^{-2}\mathbf{P}_n^T\mathbf{e}\mathbf{f}^T(K)\Phi(\bar{\mathbf{x}};K)\mathbf{f}(K), & \text{if } \mathbf{f}(K)\in\mathcal{F}_2\end{cases} \tag{46}$$

$$\dot{\mathbf{g}}(K)=\begin{cases}\beta\mathbf{P}_n^T\mathbf{e}u\Phi(\bar{\mathbf{x}};K), & \text{if } \mathbf{g}(K)\in\mathcal{G}_1\\\beta\mathbf{P}_n^T\mathbf{e}u\Phi(\bar{\mathbf{x}};K)-\beta M_2^{-2}\mathbf{P}_n^T\mathbf{e}u\mathbf{g}^T(K)\Phi(\bar{\mathbf{x}};K)\mathbf{g}(K), & \text{if } \mathbf{g}(K)\in\mathcal{G}_2\end{cases} \tag{47}$$

If there is no modeling error (i.e., $\varepsilon_K = 0$), (48) can be written as

$$\dot{V}(\mathbf{e}, \tilde{\mathbf{f}}, \tilde{\mathbf{g}}) \leq \sum_{i=1}^{n} q_i c_i^2. \tag{50}$$

The above clearly shows that $\dot{V}$ is negative semidefinite. Hence, the stability of the overall identification scheme is guaranteed and

$$\mathbf{e} \to 0, \quad \tilde{\mathbf{f}} \to 0, \quad \tilde{\mathbf{g}} \to 0. \tag{51}$$

On the other hand, in the presence of modeling error, (48) can be expressed as

$$\dot{V}(\mathbf{e}, \tilde{\mathbf{f}}, \tilde{\mathbf{g}}) \leq -\sum_{i=1}^{n} q_i \left(|e_i| - \frac{|p_{ni}|}{q_i} \varepsilon_K \right)^2 + \sum_{i=1}^{n} \frac{p_{ni}^2 \varepsilon_K^2}{q_i}. \tag{52}$$

It is easy to show from the above that, if $\mathbf{e} \notin \Theta(\varepsilon_K)$, $\dot{V}$ is still negative and the tracking errors will converge to the set $\Theta(\varepsilon_K)$. But, if $\mathbf{e} \in \Theta(\varepsilon_K)$, it is possible that $\dot{V} > 0$, which implies that the weight vectors $\mathbf{f}(K)$ and $\mathbf{g}(K)$ may drift to infinity over time. The adaptive laws (46) and (47) avoid this drift by limiting the upper bounds of the weights. Thus, the tracking error always converges to the set $\Theta(\varepsilon_K)$ and the overall control scheme will remain stable in the case of modeling error.

B. Adaptive Control Algorithm

From the set $\Theta(\varepsilon_K)$ that gives a relationship between the tracking and modeling errors, it can be shown that the tracking error depends on the modeling error. If the modeling error ε_K is known, the set $\Theta(\varepsilon_K)$ to which the tracking error will converge is also known. However, in most cases, the upper bound ε_K is unknown.

In practice, control systems are usually required to keep the tracking errors within prescribed bounds, that is

$$|e_i| \leq \varepsilon_{i0}, \quad \text{for } i = 1, 2, \ldots, n \tag{53}$$

where ε_{i0} is the required accuracy. At the beginning, it is very difficult to know how many neural network units are needed to achieve the above control requirements. In order to find a suitable-sized network for this control problem, first set lower and upper bounds for the tracking errors, which are functions of time t, and then try to find a variable network such that

$$|e_i| \in \left[\Delta_i^L(t), \Delta_i^U(t) + \varepsilon_{i0} \right], \quad \text{for } i = 1, 2, \ldots, n \tag{54}$$

where $\Delta_i^L(t), \Delta_i^U(t)$ are monodecreasing functions of time t, respectively. Those bounds are usually defined as

$$\Delta_i^U(t) = \beta_U^t \Delta_i^U(0) \tag{55}$$

$$\Delta_i^L(t) = \beta_L^t \Delta_i^L(0) \tag{56}$$

where β_U, β_L are constants and less than one, $\Delta_i^U(0), \Delta_i^U(0)$ are the initial values. It is clear that $\Delta_i^U(t), \Delta_i^L(t)$ decrease with time t. As $t \to \infty$, $\Delta_i^U(t), \Delta_i^L(t)$ approach zero. Thus, in this way, the tracking errors reach the required accuracies given in (53).

According to the relationship between the modeling error and the tracking error, it is easy to know that given the lower and upper bounds $\Delta_i^U(t), \Delta_i^L(t) + \varepsilon_{i0}$ of the tracking errors the modeling error corresponding to the above should be

$$\varepsilon_K(t) \in [\varepsilon_L(t), \varepsilon_U(t)]. \tag{57}$$

It is easy to know that the area that the set $\Theta(\zeta)$ covers is a hyperellipsoid with the center

$$\left(\frac{|p_{n1}|}{q_1} \zeta, \frac{|p_{n2}|}{q_2} \zeta, \ldots, \frac{|p_{nn}|}{q_n} \zeta \right). \tag{58}$$

Thus, it can be deduced from the set $\Theta(\varepsilon_K(t))$ given by (49) that the upper bound $\varepsilon_U(t)$ and the lower bound $\varepsilon_L(t)$ are given by

$$\varepsilon_L(t) = \max_{i=1,2,\ldots,n} \left(\frac{|p_{ni}|}{q_i} + \left(\sum_{j=1}^{n} \frac{p_{nj}^2}{q_i q_j} \right)^{0.5} \right)^{-0.5} \Delta_i^L(t) \tag{59}$$

$$\varepsilon_U(t) = \min_{i=1,2,\ldots,n} \left(\frac{|p_{ni}|}{q_i} + \left(\sum_{j=1}^{n} \frac{p_{nj}^2}{q_i q_j} \right)^{0.5} \right)^{-0.5} \times \left(\Delta_i^U(t) + \varepsilon_{i0} \right). \tag{60}$$

Hence, if the tracking error $\mathbf{e} \notin \Theta(\varepsilon_U(t))$, the network needs more basis functions. Add the $(m+1)$th order subgrid to the grid. The parameters associated with the GRBF units are then changed as follows:

$$\sigma_{m+1} = \gamma_1 \sigma_m \tag{61}$$

$$\delta_{m+1} = \gamma_2 \delta_m \tag{62}$$

$$d_{m+1} = \gamma_3 d_m \tag{63}$$

$$\mathcal{P} = \bigcup_{i=1}^{m+1} \mathcal{P}_i \tag{64}$$

$$\mathcal{C} = \bigcup_{i=1}^{m+1} \mathcal{C}_i \tag{65}$$

$$K = \sum_{i=1}^{m+1} m_i \tag{66}$$

where γ_i, for $i = 1, 2, 3$, is a constant and less than one.

But, if the tracking error $\mathbf{e} \in \Theta(\varepsilon_L(t))$, the network needs to remove some basis functions. Just remove the units associated with the mth subgrid. The parameters associated with the GRBF units are then changed as follows:

$$\mathcal{P} = \bigcup_{i=1}^{m-1} \mathcal{P}_i \tag{67}$$

$$\mathcal{C} = \bigcup_{i=1}^{m-1} \mathcal{C}_i \tag{68}$$

$$K = \sum_{i=1}^{m-1} m_i. \tag{69}$$

In both above cases, the adaptive laws of the weights are still given in the form of (46) and (47), based on the above changed

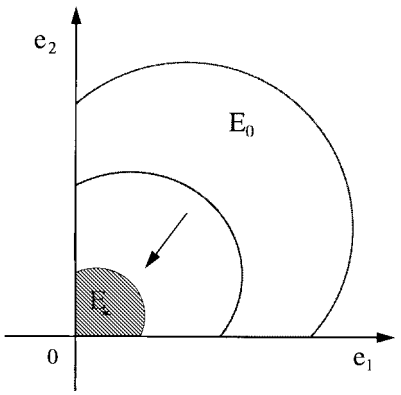


Fig. 3. Two-dimensional convergence area.

parameters. For the 2-D case, the convergence area is shown in Fig. 3. At the beginning, the convergence area of the tracking area is E_0. Finally, it approaches to the expected convergence area E_∞, that is, $|e_i| \leq \varepsilon_{i0}$, for $i = 1, 2$.

V. Simulation Results

This section considers two examples. The first is concerned with adaptive control of a time-invariant nonlinear system. The second considers adaptive control of a time-variant nonlinear system.

Example 1: The dynamical system used in the simulation example is given in [51]

$$\ddot{y} - 4\left(\frac{\sin(4\pi y)}{\pi y}\right)\left(\frac{\sin(\pi\dot{y})}{\pi\dot{y}}\right)^2 = (2 + \sin(3\pi y - 1.5\pi))u \tag{70}$$

which is a second-order time-invariant nonlinear system.

The parameter values used in this example are as follows: the reference input $y_d = \sin(t)$; the initial value of the output $y(0) = 0.5$; the initial value of the output derivative $\dot{y}(0) = 0$; the required accuracy of the tracking error vector $[\varepsilon_{10}, \varepsilon_{20}] = [0.05, 0.1]$; the constants $\beta_U = \beta_L = 0.96$; the initial values $\Delta_i^U(0) = 0.005, \Delta_i^L(0) = 0.05$, for $i = 1, 2$; the required minimum angle between the GRBF's $\cos(\theta_{\min}) = 0.951$; the edge length of the rectangles in the first subgrid is $\delta_1 = 0.5$; the radius of center selection in the first subgrid $\sigma_1 = 0.99$; the width of the GRBF units corresponding to the first subgrid $d_1 = 1.11$; the activation threshold $\delta_{\min} = 0.45$; the initial number of the variable networks is 45; the vector $a = [1, 1]$; the matrix $P = [[0.75, 0.5]^T, [0.5, 1]^T]$; and the adaptation rates $\alpha = 1.5$ and $\beta = 3$.

The parameters associated with the variable network are

$$\delta_i = 0.618\delta_{i-1}, \quad \sigma_i = 0.618\sigma_{i-1}, \quad d_i = 0.618d_{i-1} \tag{71}$$

for $i = 2, 3, \ldots, m$. The maximum of m (the number of the subgrids) is limited to be 11.

The weights are adaptively adjusted by the laws in (46) and (47). The adaptive control law is given by (38). The results of the simulation are shown in Figs. 4–6. Though the difference between the system output and the desired output is very large at the beginning, the system is still stable and the tracking error asymptotically converges to the expected range, which

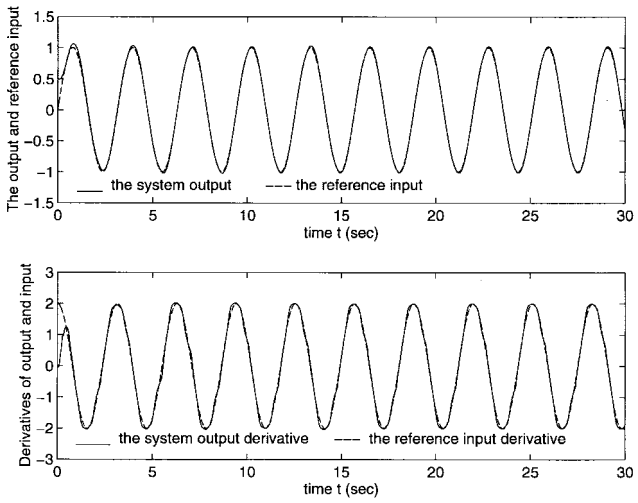


Fig. 4. Reference input $y_d(t)$, output $y(t)$, reference input derivative $\dot{y}_d(t)$, and output derivative $\dot{y}(t)$ of the system.

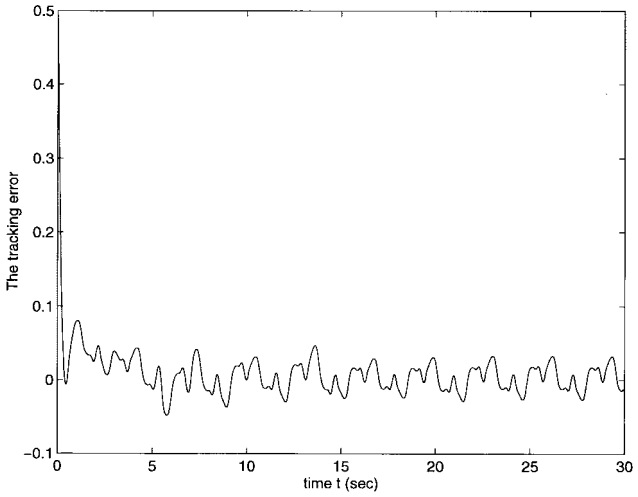


Fig. 5. Tracking error $y(t) - y_d(t)$ of the system.

is also shown in Fig. 5. As it can be seen from Fig. 6, the number of GRBF units in the neural network also converges in a period of time.

Example 2: Consider a time-variant nonlinear dynamical system given by

$$\ddot{y} - 4(2e^{-0.05t} - 1)\left(\frac{\sin(4\pi y)}{\pi y}\right)\left(\frac{\sin(\pi\dot{y})}{\pi\dot{y}}\right)^2 = (2 + \cos(0.1t)\sin(3\pi y - 1.5\pi))u. \tag{72}$$

This plant is different from that in Example 1. The functions F and G in Example 1 are time-invariant nonlinear functions. While, here the functions F and G are time variant.

All parameter values, the structure of variable networks, the weight learning laws, and the adaptive control laws used in this example are exactly the same as Example 1. The tracking error between the reference input and the output of the system is shown in Fig. 7. Although the plant to be controlled is time variant, the convergence of the tracking error in this example is still similar to that in Example 1. This shows that the scheme developed in this paper for adaptive control using variable

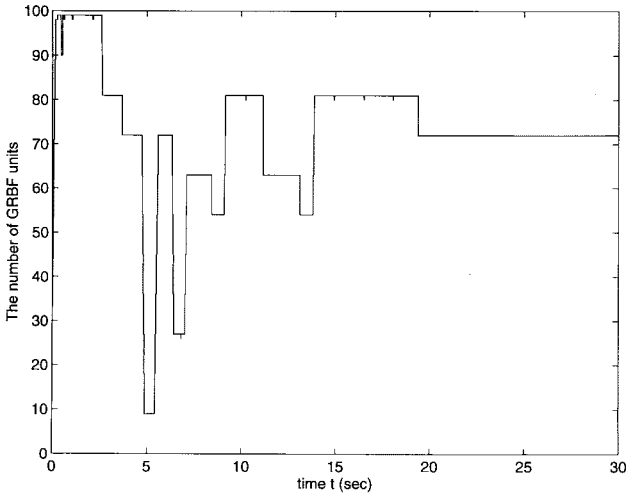


Fig. 6. Number K of GRBF units in the variable neural network.

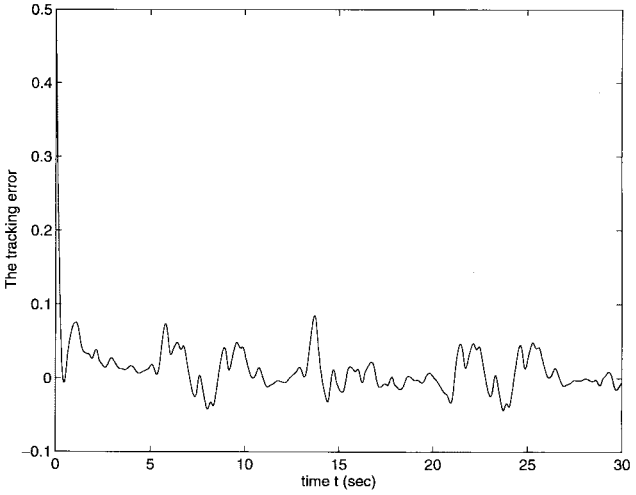


Fig. 7. Tracking error $y(t) - y_d(t)$ of the system.

neural networks also works well for time-variant nonlinear systems.

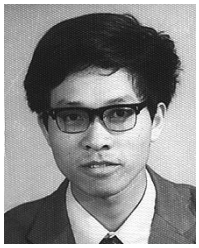
VI. Conclusion

A variable neural network structure has been proposed, in which the number of the basis functions in the network can be either increased or decreased over time according to some design strategy to avoid either overfitting or underfitting. In order to model unknown nonlinearities of nonlinear systems, the variable neural network starts with a small number of initial hidden units, then adds or removes units on a variable grid consisting of a variable number of subgrids with different sized hypercubes, based on the novelty of observation. The adaptive control algorithm, developed by combining the variable GRBF network and Lyapunov synthesis techniques, guarantees the stability of the control system and the convergence of the tracking errors. The number of GRBF units in the neural network also converges by introducing the monodecreasing upper and lower bounds of the tracking errors. The results of the simulation examples illustrate the operation of the variable neural network for adaptive nonlinear system control.

References

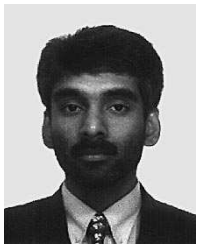
[1] C. W. Anderson, "Learning to control an inverted pendulum using neural networks," *IEEE Contr. Syst. Mag.*, vol. 9, pp. 31–37, 1989.
[2] P. J. Antsaklis, Ed., "Special issue on neural networks in control systems," *IEEE Contr. Syst. Mag.*, vol. 10, no. 3, 1990.
[3] G. Barto, *Neural Network for Control*. Cambridge, MA: MIT Press, 1990.
[4] E. Baum and D. Haussler, "What size net gives valid generalization," *Neural Computat.*, vol. 1, no. 1, pp. 151–160, 1989.
[5] S. A. Billings and S. Chen, "Neural networks and system identification," in *Neural Networks for Systems and Control*, K. Warwick *et al.*, Eds. London, U.K.: Peregrinus, 1992, pp. 181–205.
[6] S. Chen, S. A. Billings, and P. M. Grant, "Nonlinear system identification using neural networks," *Int. J. Contr.*, vol. 51, no. 6, pp. 1191–1214, 1990.
[7] S. R. Chi, R. Shoureshi, and M. Tenorio, "Neural networks for system identification," *IEEE Contr. Syst. Mag.*, vol. 10, pp. 31–34, 1990.
[8] S. Fabri and V. Kadirkamanathan, "Dynamic structure neural networks for stable adaptive control of nonlinear systems," *IEEE Trans. Neural Networks*, vol. 7, no. 5, pp. 1151–1167, 1996.
[9] S. E. Fahlman and C. Lebiere, "The cascade-correlation architecture," in *Advances in Neural Information Processing Systems 2*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1990.
[10] K. S. Fu, "Learning control systems—Review and outlook," *IEEE Trans. Automat. Control*, vol. 16, pp. 210–221, Jan. 1970.
[11] G. C. Goodwin and D. Q. Mayne, "A parameter estimation perspective of continuous time model reference adaptive control," *Automatica*, vol. 23, no. 1, pp. 57–70, 1987.
[12] A. Guez, J. L. Elibert, and M. Kam, "Neural network architecture for control," *IEEE Contr. Syst. Mag.*, vol. 8, pp. 22–25, 1988.
[13] K. J. Hunt and D. Sbararo, "Neural networks for nonlinear internal model control," *Proc. Inst. Elect. Eng. D*, vol. 138, pp. 431–438, 1991.
[14] K. J. Hunt, D. Sbararo, R. Zbikowski, and P. J. Gawthrop, "Neural networks for control systems—A survey," *Automatica*, vol. 28, no. 6, pp. 1083–1112, 1992.
[15] P. A. Ioannou and A. Datta, "Robust adaptive control: Design, analysis and robustness bounds," in *Foundations of Adaptive Control*, P. V. Kokotovic, Ed. Berlin, Germany: Springer-Verlag, pp. 71–152, 1991.
[16] P. A. Ioannou and P. V. Kokotovic, *Adaptive Systems with Reduced Models*. New York: Springer-Verlag, 1983.
[17] P. A. Ioannou and Tsakalis, "A robust direct adaptive control," *IEEE Trans. Automat. Control*, vol. AC-31, pp. 1033–1043, Sept. 1986.
[18] A. Isidori, *Nonlinear Control Systems: An Introduction*. Berlin, Germany: Springer-Verlag, 1989.
[19] A. Karakasoglu, S. L. Sudharsanan, and M. K. Sundareshan, "Identification and decentralized adaptive control using neural networks with application to robotic manipulators," *IEEE Trans. Neural Networks*, vol. 4, pp. 919–930, June 1993.
[20] V. Kadirkamanathan, "Sequential learning in artificial neural networks," Ph.D dissertation, Univ. Cambridge, Cambridge, U.K., 1991.
[21] ———, "A statistical inference based growth criteria for the RBF network," in *Proc. IEEE Workshop Neural Networks Signal Processing*, 1994, pp. 12–21.
[22] V. Kadirkamanathan and G. P. Liu, "Robust identification with neural networks using multiobjective criteria," in *Preprints 5th IFAC Symp. Adaptive Syst. Contr. Signal Processing*, Budapest, Hungary, 1995, pp. 237–242.
[23] V. Kadirkamanathan and M. Niranjan, "Application of an architecturally dynamic network for speech pattern classification," in *Proc. Inst. Acoust.*, 1992, vol. 14, part 6, pp. 343–350.
[24] ———, "A function estimation approach to sequential learning with neural networks," *Neural Computation*, vol. 5, pp. 954–957, 1993.
[25] I. D. Landau, *Adaptive Control—The Model Reference Approach*. New York: Marcel Dekker, 1979.
[26] G. P. Liu and V. Kadirkamanathan, "Learning with multiobjective criteria," in *Proc. 4th Int. Conf. Artif. Neural Networks*, Cambridge, U.K., 1995, pp. 35–40.
[27] G. P. Liu, V. Kadirkamanathan, and S. A. Billings, "Identification of nonlinear systems using growing RBF networks," in *Proc. 3rd Eur. Contr. Conf.*, Rome, Italy, 1995, pp. 2408–2413.
[28] ———, "Stable sequential identification of continuous nonlinear dynamical systems by growing RBF networks," *Int. J. Contr.*, vol. 65, no. 1, pp. 53–69, 1996.
[29] ———, "Nonlinear predictive control using neural networks," in *Proc. Control'96*, U.K.
[30] ———, "Neural network based variable structure system nonlinear control," in *Proc. 4th Eur. Contr. Conf.*, Brussels, Belgium, 1997.

[31] _____, "On-line identification of nonlinear systems using Volterra polynomial basis function neural networks," in *Proc. 4th Eur. Contr. Conf.*, Brussels, Belgium, 1997.
[32] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Advances in Neural Information Processing Systems 2*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1990, pp. 598–605.
[33] D. J. C. MacKay, "Bayesian interpolation," *Neural Computat.*, vol. 4, no. 3, pp. 415–447, 1992.
[34] T. M. Miller, R. S. Sutton III, and P. J. Werbos (Eds.), *Neural Networks for Control*. Cambridge, MA: MIT Press, 1990.
[35] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural Computat.*, vol. 1, pp. 281–294, 1989.
[36] M. C. Mozer and P. Smolensky, "Skeletonization: A technique for trimming the fat from a network via relevance assignment," in *Advances in Neural Information Processing Systems 1*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1989.
[37] K. S. Narendra and A. M. Annaswamy, "A new adaptive law for robust adaptation with persistent excitation," *IEEE Trans. Automat. Contr.*, vol. 32, no. 2, pp. 134–145, 1987.
[38] _____, *Stable Adaptive Systems*. Engelwood Cliffs, NJ, Prentice-Hall, 1989.
[39] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks*, vol. 1, pp. 4–27, Jan. 1990.
[40] H. Nijmeijer and A. J. van der Schaft, *Nonlinear Dynamical Control Systems*. New York: Springer-Verlag, 1990.
[41] M. Niranjan and F. Fallside, "Neural networks and radial basis functions for classifying static speech patterns," *Comput. Speech Language*, vol. 4, pp. 275–289, 1990.
[42] J. Platt, "A resource allocating network for function interpolation," *Neural Computat.*, vol. 4, no. 2, pp. 213–225, 1991.
[43] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proc. IEEE*, vol. 78, pp. 1481–1497, Sept. 1990.
[44] _____, "Regularization algorithms for learning that are equivalent to multilayer networks," *Science*, vol. 247, pp. 978–982, 1990.
[45] M. M. Polycarpou and P. A. Ioannou, "Identification and control of nonlinear systems using neural network models: Design and stability analysis," Dept. Elect. Eng. Syst., Univ. Southern California, Tech. Rep. 91-09-01, 1991.
[46] M. J. D. Powell, *Approximation Theory and Methods*. Cambridge, U.K., Cambridge Univ. Press, 1981.
[47] _____, "Radial basis functions for multivariable interpolation: A review," in *Algorithms for Approximation*, J. C. Mason and M. G. Cox, Eds. Oxford, U.K.: Oxford Univ. Press, 1987, pp. 143–167.
[48] D. Psaltis, A. Sideris, and A. A. Yamamura, "A multilayered neural network controller," *IEEE Contr. Syst. Mag.*, vol. 8, pp. 17–21, 1988.
[49] S. Z. Qin, H. T. Su, and T. J. McAvoy, "Comparison of four net learning methods for dynamic system identification," *IEEE Trans. Neural Networks*, vol. 3, pp. 122–130, Jan. 1992.
[50] N. Sadegh, "A perceptron network for functional identification and control of nonlinear systems," *IEEE Transactions Neural Networks*, vol. 4, pp. 982–988, June 1993.
[51] R. M. Sanner and J. J. E. Slotine, "Gaussian networks for direct adaptive control," *IEEE Trans. Neural Networks*, vol. 3, pp. 837–863, June 1992.
[52] S. Sastry and M. Bodson, *Adaptive Control: Stability, Convergence and Robustness*. Engelwood Cliffs, NJ: Prentice-Hall, 1989.
[53] J. J. E. Slotine and W. Li, *Applied Nonlinear Control*. Engelwood Cliffs, NJ: Prentice-Hall, 1991.
[54] P. Smyth, "On stochastic complexity and admissible models for neural network classifiers," in *Advances in Neural Information Processing Systems 3*, R. P. Lippmann, J. Moody, and D. S. Touretzky, Eds. San Mateo, CA: Morgan Kaufmann, 1991.
[55] R. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, pp. 1550–1560, Dec. 1990.
[56] M. J. Willis, G. A. Montague, C. Di. Massimo, M. T. Tham, and A. J. Morris, "Artificial neural networks in process estimation and control," *Automatica*, vol. 28, no. 6, pp. 1181–1187, 1992.



Guoping P. Liu (M'97) was born in Changsha, China, in November 1962. He received the B.Eng. and M.Eng. degrees in electrical and electronic engineering from the Central South University of Technology, China, in 1982 and 1985, respectively, and the Ph.D. degree in control systems engineering from the University of Manchester Institute of Science and Technology, Manchester, U.K., in 1992.

He was appointed as a Lecturer in the Department of Automatic Control Engineering, Central South University of Technology, in 1985 and as a Professor in the Institute of Information Engineering, Central South University of Technology, in 1994. He received the Alexander von Humboldt Research Fellowship in 1992. From 1992 to 1993, he worked as a Postdoctoral Research Assistant in the Department of Electronics, University of York, York, U.K. From 1993 to 1996, he was a Research Associate in the Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, U.K. He was a Senior Engineer in the Mechanical Engineering Centre, GEC-Alsthom, Leicester, U.K. from 1996 to 1998. He is currently with the ALSTOM Energy Technology Centre, Leicester. He developed an eigenstructure assignment toolbox for use with MATLAB. He is a co-author of three books: *Critical Control Systems: Theory, Design and Applications* (New York: Research Studies Press and Wiley, 1993), *Advanced Adaptive Control* (Oxford, U.K.: Pergamon, 1995), and *Eigenstructure Assignment for Control System Design* (Chichester, U.K.: Wiley, 1998). His main research interests include nonlinear system identification and control, adaptive control, eigenstructure assignment for multivariable control systems, multiobjective optimization and control, robust control, critical control systems, and real-time computer control and applications.



Visakan Kadirkamanathan (S'89–M'90) was born in Jaffna, Sri Lanka, on October 15, 1962. He received the B.A. degree in electrical and information sciences (with first-class honors) in June 1987 and the Ph.D. degree in February 1992, both from Cambridge University, Cambridge, U.K.

He was a Postdoctoral Research Assistant in the Electrical and Electronic Engineering Department, University of Surrey, Surrey, U.K., from 1991 to 1992 and in the Engineering Department, Cambridge University. Since 1993, he has been a Lecturer in the Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, U.K. His research interests include nonlinear systems, neural networks, adaptive estimation and control, medical signal analysis, engine fault diagnosis, and applications of neural networks.

Stephen A. Billings was born in the United Kingdom in 1951. He received the B.Eng. degree in electrical engineering (with first-class honors) from the University of Liverpool, Liverpool, U.K., in 1972, the Ph.D. degree in control systems engineering from the University of Sheffield, Sheffield, U.K., in 1976, and the D.Eng. degree from the University of Liverpool, Liverpool, U.K., in 1990.

He was appointed Professor in the Department of Automatic Control and Systems Engineering, University of Sheffield in 1990 and leads the Signal Processing and Machine Vision Research Group. His research interests include system identification and signal processing for nonlinear systems, time series analysis, spectral analysis, adaptive noise cancellation, nonlinear systems analysis and design, fault detection, neural networks, machine vision, and brain imaging.

Dr. Billings is a Chartered Engineer (C.Eng.), Chartered Mathematician (C.Math.), Fellow of the Institute of Electrical Engineers (U.K.), and Fellow of the Institute of Mathematics and its Applications.