(3·16—22

基于动态神经网络的人工环境自适应控制*

Adaptive Control of Artificial Environments Based on Dynamic Neural Networks

TP2732

张邦礼 叶仲泉 杨映林 曹长修
Zhang Bangli Ye Zhongquan Yang Yinglin Cao Changxiu

(重庆大学自动化系,重庆,630044;第一作者 54 岁,女,副教授)

A 摘 要 本文将动态 BP 网络应用于双输入双输出非线性耦合系统的自适应控制,并利用递归最小二乘法(RLS)来训练该网络,RLS 算法具有收敛速度快、抗噪声能力强等优点,还避免了常规 BP 算法中学习率选取困难的缺点。

关键词 神经网络;人工环境;自适应控制
中国图书资料分类法分类号 TP273.2

ABSTRACT This paper investigates the adaptive control of nonlinear and coupling systems with dual inputs and dual outputs using dynamic BP neural network. Recursive Least Squares(RLS) algorithm is applied to train the network. It is well known that RLS algorithm has fast speed of convergence and has good robustness against noise and disturbance. The new algorithm also overcomes the difficulty of determine step size for conventional HP algorithm.

KEYWORDS neural networks; artificial enviroments; adaptive control

0 引 言

我们在研究"人工环境试验控制系统"时发现,密闭空间中的温度和湿度这两个变量是一类典型的双输入双输出系统,它集中了控制工程中的几大难点——非线性、滞后和耦合,而且在工业控制系统中很具有代表性。对于这样一类多输入多输出(MIMO)系统,由于问题的复杂性,要建立精确的数学模型和设计出稳定可靠的控制器是十分困难的。为这类 MIMO 非线性和耦合系统设计一种切实可行的,并且不完全依赖于对象数学模型的控制方法,正是本文的研究目的。

神经网络理论近年来取得了突破性进展,在许多领域得到了广泛的应用,这也为解决 MIMO 非线性耦合系统的控制提供了强有力的新手段。多层前向网络具有逼近任意非线性函数的功能[1],但它不是动态系统,因而仅仅实现了一个静态映射[2],它是目前研究得最多和应用最广泛的网络之一,误差反向传播算法(EBP)是多层前向网络中基本的算法,它本质

* 收文日期 1995-11-01
高等学校博士点基金资助课题

上是一种梯度最速下降算法，所以收敛速度太慢且存在局部极小问题。另外，BP 算法中学习率（步长）的选取十分困难，无任何理论指导，只能靠经验和仿真结果，如果步长取得太小，收敛速度就可能太慢；而步长取得过大，则网络可能不收敛。

本文利用动态 BP 网络[3]研究双输入双输出非线性耦合系统的自适应控制。并利用自适应控制和自适应滤波中广泛使用的递推最小二乘算法来训练多层前向网络。众所周知，RLS 算法具有收敛速度快，跟踪性能好和鲁棒性强等优点[8]，后面的仿真结果也证实了这点。此外，采用 RLS 算法还去除了常规 BP 算法中学习率的选择（这也常是 BP 算法出问题的原因），而用协方差矩阵代替学习率（步长）。由于协方差矩阵包含了到目前为止的所有输入输出信息，所以它提供了更好的参数估计和参数调节的方法，且具有更快的收敛速度。

利用 RLS 算法训练神经网络，目前已有一些结果[4,6]等，但多是用 RLS 调节隐层到输出层的权值矩阵。如用 RLS 调节隐层到输出层的权值，而输入层到隐层的权值仍用梯度下降法[6]. 用连续时间 RLS 训练多层前向网络[5]，输入层到隐层的权值矩阵也用 RLS 训练，其方法富有启发性。

本文用离散 RLS 来训练三层前向网络，隐层到输出层以及输入层到隐层的权值调节都用 RLS 算法。

本文的方法可以毫不困难地推广到 $n(>3)$ 层前向网络。

1　双输入双输出非线性耦合系统的描述

本文所讨论的温度和湿度对象，是一个相互耦合的非线性双输入双输出系统，可以用如下的离散时间方程组来描述：

$$\begin{aligned} y_1(t+1) &= f_1(\cdot) + g_1(\cdot)u_1(t) + g_2(\cdot)u_2(t) \\ y_2(t+1) &= f_2(\cdot) + g_3(\cdot)u_1(t) + g_4(\cdot)u_2(t) \end{aligned} \tag{1}$$

式中 $f_1(\cdot), g_1(\cdot), g_2(\cdot), f_2(\cdot), g_3(\cdot), g_4(\cdot)$ 等是变量 $(y_1(t), \cdots, y_1(t-n_1+1), y_2(t), \cdots, y_2(t-n_1+1), u_1(t-1), \cdots, u_1(t-m), u_2(t-1), \cdots, u_2(t-m))$ 的函数。y_1 是系统的一个输出（如温度），y_2 是另一个输出（如湿度）；u_1 是对 y_1 的控制量（如实际工程中，u_1 代表制冷压缩机工作情况下加热电阻丝导通和关闭的时间比），u_2 是对另外一个变量 y_2 的控制量（如实际工程中，u_2 代表除湿压缩机工作情况下加湿器工作和停止的时间比）；$f_1(\cdot), f_2(\cdot), g_1(\cdot), g_2(\cdot), g_3(\cdot), g_4(\cdot)$ 均为非零且非线性函数。

2　动态 BP 网络

只含一层隐层且隐层中节点的激活函数为 Sigmoid 函数的三层静态 BP 网络，可以逼近任意复杂的非线性函数[1]，在静态 BP 网络中引入反馈和时间延迟，因而静态模型变成了式（2）的动态模型。三层动态 BP 网络的结构如图 1.

$$\begin{aligned} \hat{y}_1(t+1) &= \hat{f}_1(\cdot) + \hat{g}_1(\cdot)u_1(t) + \hat{g}_2(\cdot)u_2(t) \\ \hat{y}_2(t+1) &= \hat{f}_2(\cdot) + \hat{g}_3(\cdot)u_1(t) + \hat{g}_4(\cdot)u_2(t) \end{aligned} \tag{2}$$

式中 $\hat{f}_1(\cdot), \hat{f}_2(\cdot), \hat{g}_1(\cdot), \hat{g}_2(\cdot), \hat{g}_3(\cdot)$ 和 $\hat{g}_4(\cdot)$ 均是变量 $(y_1(t), \cdots, y_1(t-n_1+1), y_2(t), \cdots, y_2(t-n_1+1), u_1(t-1), \cdots, u_1(t-m), u_2(t-1), \cdots, u_2(t-m))$ 的函数；$\hat{y}_1$、$\hat{y}_2$ 分别为动态 BP

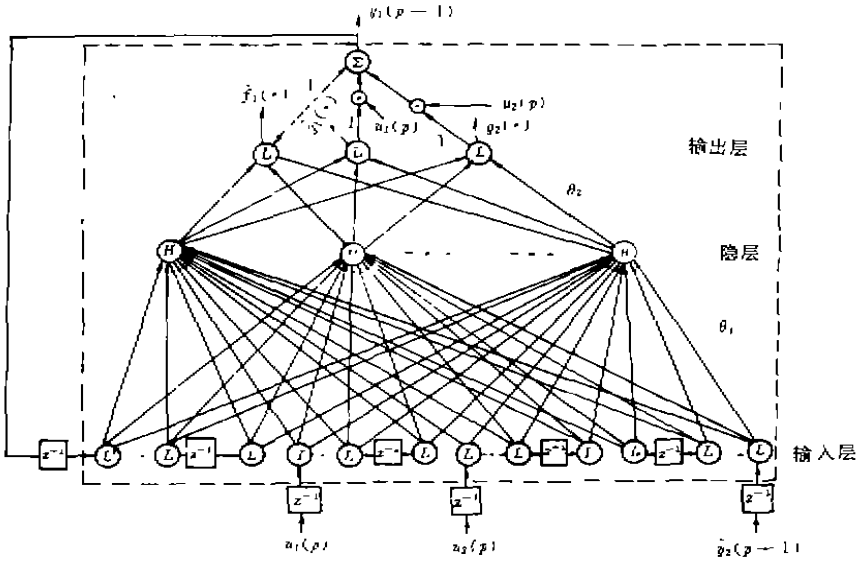

图 1 三层动态 BP 网络

网络 1,2 的输出:$\hat{f}_1(\cdot),\hat{f}_2(\cdot),\hat{g}_1(\cdot),\hat{g}_2(\cdot),\hat{g}_2(\cdot),\hat{g}_3(\cdot),\hat{g}_4(\cdot)$ 是通过网络学习分别对对象如式(1) 中的 $f_1(\cdot),f_2(\cdot),g_1(\cdot),g_2(\cdot),g_3(\cdot),g_4(\cdot)$ 的逼近。

用两个上述的动态 BP 网络来作式(1) 所示离散动态系统的辨识模型,其学习算法采用本文提出的 RLS 算法。

3 系统辨识及多层网络的 RLS 算法

3.1 递推最小二乘算法(RLS)

最小二乘法是参数估计的一项基本技术,如果模型在参数上是线性的,则最小二乘法就特别简单。

考察模型

$$y(t)=\varphi_1(t)\theta_1+\varphi_2(t)\theta_2+\cdots+\varphi_n(t)\theta_n=\varphi(t)^T\theta \tag{3}$$

式中 y 是观测变量,$\theta_1,\theta_2,\cdots,\theta_n$ 是未知参数,$\varphi_1,\varphi_2,\cdots,\varphi_n$ 是由其它已知变量决定的已知函数,成对的观测量和回归量 $\{[y(i),\varphi(i)],i=1,2,\cdots,t\}$ 可由实验得到。

令

$$Y(t)=[y(1)\quad y(2)\quad\cdots\quad y(t)]^T$$

$$\Phi(t)=\begin{bmatrix}\varphi^T(1)\\ \vdots\\ \varphi^T(t)\end{bmatrix}$$

$$P(t)=(\Phi^T(t)\Phi(t))^{-1}=\Big(\sum_{i=1}^{t}\varphi(i)\varphi^T(i)\Big)^{-1}$$

$$E(t) = [\varepsilon(1) \quad \varepsilon(2) \quad \cdots \quad \varepsilon(t)]^T$$
$$\varepsilon(i) = y(i) - \hat{y}(i) = y(i) - \varphi^T(i)\theta$$

最小二乘误差可表示为

$$V(\theta,t) = \frac{1}{2}\sum_{i=1}^{t}\varepsilon(i)^2 = \frac{1}{2}\sum_{i=1}^{t}(y(i) - \varphi^T(i)\theta)^2$$
$$= \frac{1}{2}E^TE = \frac{1}{2}\|E\|^2$$

RLS 估计[7]：设 $\hat{\theta}(t)$ 代表基于 t 次测量数据的最小二乘估计，$\Phi^T\Phi$ 对所有 t 都是正定的，则

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \frac{P(t-1)\varphi(t)}{1+\varphi^T(t)P(t-1)\varphi(t)}[y(t) - \varphi^T(t)\hat{\theta}(t-1)]$$
$$P(t) = P(t-1) - \frac{P(t-1)\varphi(t)\varphi(t)^TP(t-1)}{1+\varphi^T(t)P(t-1)\varphi(t)} \tag{4}$$

当(3)式中的 θ 为 $n \times p$ 矩阵时，有与(4)式完全类似的 RLS 估计[1]，只是(4)式中的 $\hat{\theta}$ 为 $n \times p$ 矩阵，$y(t)$ 为 $1 \times p$ 矩阵(p 维行向量)。

3.2　多层前向网络的 RLS 算法

考虑到三层前向网络中，输出向量与隐层的输出向量之间呈线性关系，以及输入向量与隐层的输入向量之间也呈线性关系，所以希望能用 RLS 算法来修改网络权值。但有一个问题，即隐层的输入向量的相应的期望向量难以确定。本文采用与[5]中类似的方法。

在图 1 中所示的网络，用 $\varphi(t) \in R^n$ 表示时刻 t 的输入向量，即 $\varphi(t) = [y_1(t),\cdots,y_1(t-n_1+1),y_2(t),\cdots,y_2(t-n_2+1),u_1(t-1),\cdots,u_1(t-m),u_2(t-1),\cdots,u_2(t-m)]^T$，$n = 2n_1 + 2m$；$\theta_1 \in R^{n\times p}$ 表示输入层到隐层的权值矩阵，p = 隐层中隐单元数目，$\theta_2 \in R^{p\times 3}$ 表示隐层到输出层的权值矩阵；输入层和输出层的激活函数皆为 $\sigma(x) = x$，隐层的激活函数为 $\sigma(x) = 1/1 + \bar{e}^{-x}$。

于是

$$\hat{y}_1(t) = [1 \quad u_1(t) \quad u_2(t)][\hat{f}_1(\cdot),\hat{g}_1(\cdot),\hat{g}_2]^T$$
$$= [1 \quad u_1(t) \quad u_2(t)]\theta_2^T\sigma(\theta_1^T\varphi(t)) \tag{5}$$

(注意：$\sigma\left(\begin{bmatrix}x_1\\ \vdots \\ x_p\end{bmatrix}\right) = \begin{bmatrix}\sigma(x_1)\\ \vdots \\ \sigma(x_p)\end{bmatrix}$)

神经网络的误差函数为：

$$V(\theta_1,\theta_2,t) = \frac{1}{2}\sum_{i=1}^{t}(y_1(i) - \hat{y}_1(i))^2$$
$$= \frac{1}{2}\sum_{i=1}^{t}(y_1(i) - [1 \quad u_1(t) \quad u_2(t)]\theta_2^T\sigma(\theta_1^T\varphi(t)))^2 \tag{6}$$
$$= \frac{1}{2}\sum\varepsilon(i)^2$$

$$\frac{\partial V}{\partial\theta_2} = -\sum_{i=1}^{t}\sigma(\theta_1^T\varphi(i))[1 \quad u_1(t) \quad u_2(i)]\varepsilon(i) \tag{7}$$

$$\frac{\partial V}{\partial\theta_1} = -\sum_{i=1}^{t}\varphi(i)e_1^T(i) \tag{8}$$

其中
$$\varepsilon_1(i) = \sigma'(\theta_1^T\varphi_1(i))\theta_2\begin{bmatrix}1\\ u_1(i)\\ u_2(i)\end{bmatrix}\varepsilon(i) \tag{9}$$

其中 $\sigma'(q) = \dfrac{\partial\sigma}{\partial q}$ (Jacob 矩阵)

即
$$\dot{\sigma}'\begin{bmatrix}x_1\\ x_2\\ \vdots\\ x_p\end{bmatrix} = \begin{bmatrix}\sigma'(x_1) & 0\cdots & & 0\\ 0 & \sigma'(x_2) & \cdots & 0\\ \cdots & \cdots & \cdots & \cdots\\ 0 & 0 & \cdots & \sigma'(x_p)\end{bmatrix}_{p\times p}.$$

误差 $\varepsilon_1(i)$ 与 BP 算法中隐层的误差完全一样，也是输出误差的反向传播。

三层前向网络的 RLS 算法：

$$\hat{\theta}_2(t) = \hat{\theta}_2(t-1) + \frac{P_2(t-1)\sigma(\hat{\theta}_1^T(t-1)\varphi(t))\begin{bmatrix}1 & u_1(t) & u_2(t)\end{bmatrix}e'(t)}{\alpha(t-1) + \sigma^T(\hat{\theta}_1^T(t-1)\varphi(t))P_2(t-1)\sigma(\hat{\theta}_1^T(t-1)\varphi(t))} \tag{10}$$

$$P_2(t) = \frac{1}{\alpha(t-1)}\left[P_2(t-1) - \frac{P_2(t-1)\sigma(\hat{\theta}_1^T(t-1)\varphi(t))\sigma^T(\hat{\theta}_1^T(t-1)\varphi(t))P_2(t-1)}{\alpha(t-1) + \sigma^T(\hat{\theta}_1^T(t-1)\varphi(t))P_2(t-1)\sigma(\hat{\theta}_1^T(t-1)\varphi(t))}\right] \tag{11}$$

其中

$$\begin{aligned}e'(t) &= y_1(t) - \dot{y}_1(t)\\ &= y_1(t) - \begin{bmatrix}1 & u_1(t) & u_2(t)\end{bmatrix}\hat{\theta}_2^T(t-1)\sigma(\hat{\theta}_1^T(t-1)\varphi(t))\end{aligned} \tag{12}$$

$$\hat{\theta}_1(t) = \hat{\theta}_1(t-1) + \frac{P_1(t-1)\varphi(t)\begin{bmatrix}1 & u_1(t) & u_2(t)\end{bmatrix}\hat{\theta}_2^T(t)\sigma'(\hat{\theta}_1^T(t-1)\varphi_1(t)\varepsilon''(t)}{\alpha(t-1) + \varphi^T P_1(t-1)\varphi(t)} \tag{13}$$

$$P_1(t) = \frac{1}{\alpha(t-1)}\left[P_1(t-1) - \frac{P_1(t-1)\varphi(t)\varphi^T(t)P_1(t-1)}{\alpha(t-1) + \varphi^T(t)P_1(t-1)\varphi(t)}\right] \tag{14}$$

其中

$$\varepsilon''(t) = y_1(t) - \begin{bmatrix}1 & u_1(t) & u_2(t)\end{bmatrix}\hat{\theta}_2^T(t)\sigma(\hat{\theta}_1(t-1)\varphi(t)) \tag{15}$$

$\alpha(t)$ 的引入使得在较新的数据点上有较大的加权，提高了系统跟踪性能，同时 $\alpha(t)$ 也有减小初始数据，提高抑制噪声能力的作用。本文中的例子采用了与[6]中一样的选择：

$$\alpha(t) = \alpha_0\cdot\alpha(t-1) + (1-\alpha_0)$$

其中 $\alpha(0) = 0.95, \alpha_0 = 0.99$

注 1° 此算法与标准 RLS 算法的差别在于误差项。

2° 输入信号的选择应使 $\{\varphi(t)\}$ 和 $\{\sigma(\theta_1(\varphi(t)))\}$ 是持续激励的，但如何做到这点却相当困难。这样，该算法的收敛性分析就很困难。

3.3 辨识过程

1) 初始化：$P_1(0) = I, P_2(0) = I$. $\hat{\theta}_1(1), \hat{\theta}_2(1)$ 取为范数很小的随机矩阵。

2) 读入一个样本，用公式(10) ～ (15) 修改权值矩阵 θ_2 和 θ_1.

3) 若还有样本，则读入下一个样本并重复以上步骤，否则结束。

很显然，另一个动态 BP 网络的学习过程完全类似。

4 自适应控制

对式(1) 所示系统，当期望为 $d_1(p+1), d_2(p+2)$ 时，若采用下面的控制，系统的输出

$y_1(p+1), y_2(p+1)$ 将准确跟踪 $d_1(p+1), d_2(p+1)$

$$u_1(p) = \frac{g_4(\cdot)[d_1(p+1) - f_1(\cdot)] - g_2(\cdot)[d_2(p+1) - f_2(\cdot)]}{g_1(\cdot)g_4(\cdot) - g_2(\cdot)g_3(\cdot)}$$
$$u_2(p) = \frac{g_3(\cdot)[d_1(p+1) - f_1(\cdot)] - g_1(\cdot)[d_2(p+1) - f_2(\cdot)]}{g_1(\cdot)g_4(\cdot) - g_2(\cdot)g_3(\cdot)} \quad (16)$$

式中 $f_1(\cdot), f_2(\cdot), g_1(\cdot), g_2(\cdot), g_3(\cdot)$ 和 $g_4(\cdot)$ 均是$(y_1(p), \cdots, y_1(p-n_1+1), y_2(p), \cdots, y_2(p-n_2-1), u_1(p-1), \cdots, u_1(p-m), u_2(p-1), \cdots, u_2(p-m))$ 的函数。因 $f_1(\cdot), f_2(\cdot), g_1(\cdot), g_2(\cdot), g_3(\cdot)$ 和 $g_4(\cdot)$ 是未知的，我们用网络来学习，当网络学习成功后，控制量可以如下所示：

$$u_1(p) = \frac{\hat{g}_4(\cdot)[d_1(p-1) - \hat{f}_1(\cdot) - \hat{g}_2(\cdot)[d_2(p+1) - \hat{f}_2(\cdot)]}{\hat{g}_1(\cdot)\hat{g}_4(\cdot) - \hat{g}_2(\cdot)\hat{g}_3(\cdot)}$$
$$u_2(p) = \frac{\hat{g}_3(\cdot)[d_1(p+1) - \hat{f}_1(\cdot)] - \hat{g}_1(\cdot)[d_2(p+1) - \hat{f}_2(\cdot)]}{\hat{g}_1(\cdot)\hat{g}_4(\cdot) - \hat{g}_2(\cdot)\hat{g}_3(\cdot)} \quad (17)$$

式中 $\hat{f}_1(\cdot), \hat{f}_2(\cdot), \hat{g}_1(\cdot), \hat{g}_2(\cdot), \hat{g}_3(\cdot)$ 和 $\hat{g}_4(\cdot)$ 均是$(\hat{y}_1(p), \cdots, \hat{y}_1(p-n_1-1), \hat{y}_2(p), \cdots, y_2(p-n_2+1), u_1(p-1), \cdots, u_1(p-m), u_2(p-1), \cdots, u_2(p-m))$ 的函数。

本文采用自适应控制算法的主要算法之一，自校正调节器。自校正调节器设计的基本思想是：如果系统的环境和模型中的参数已知，那么采用适当的设计方法可获得某种意义下的最优控制器；如果系统的参数未知，则可用参数在线估计（即递推辨识）来代替未知的真实参数值，即所谓的确定性等价原则。在计算出控制器的参数后，即可进行实时控制。

自适应控制的具体步骤如下：

1° 初始化 $y_1(0), y_2(0), P=0, \varepsilon > 0$

2° 网络开始学习，进行系统辨识

3° 网络学习好，用式(17) 计算 $u_1(p), u_2(p)$

4° 按仿真举例，计算实际对象的输出值：

$y_1(p+1), y_2(p+1)$

5° 若 $p < N$，则 $p = p+1$，进行 3°；否则进行 6°

6° 计算 $E_1 = \sum_{p=1}^{N} \frac{1}{2}[d_1(p) - y_1(p)]^2$

$E_2 = \sum_{p=1}^{N} \frac{1}{2}[d_2(p) - y_2(p)]^2$

$E = E_1 + E_2$

若 $E > \varepsilon$，即对象发生变化，进行 2°，否则进行 3°.

5　仿真实验

考察一阶双输入双输出非线性耦合系统：

$$y_1(p+1) = \sin(x) + 1.2\sin(x)u_1(p) - u_2(p)$$
$$y_2(p+1) = 0.8\cos(x) + 0.6u_1(p) - 1.8\sin(x/2) \quad (18)$$

式中　$x = 2(y_1(p) + y_2(p) + u_1(p-1) + u_2(p-1))$

网络输入节点 16 个，输出节点 3 个，隐层节点 60 个。

图 2 中的(a),(b)分别是网络对式(18)所示系统中的 y_1,y_2 辨识过程。在辨识时,u_1 为由 5 级移位寄存器产生的 M 序列,u_2 为由 6 级移位寄存器产生的 M 序列。

从图 2 中可以看出收敛速度很快,150 次时网络输出与期望输出几乎重合了。

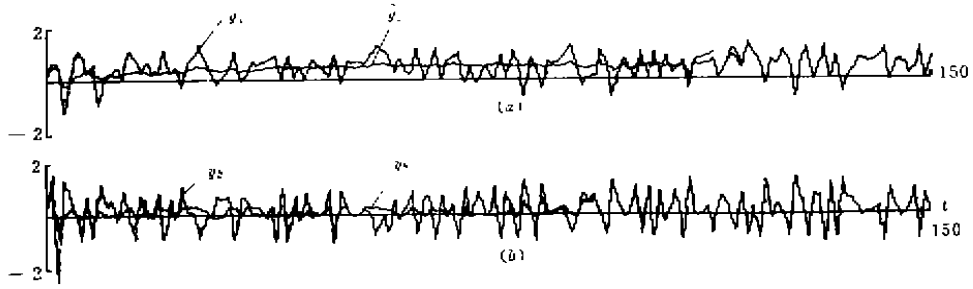

图 2 辨识过程

6 结 语

本文利用动态 BP 网络研究双输入双输出非线性耦合系统的自适应控制,反过来又利用自适应控制和自适应滤波中广泛使用的递推最小二乘算法来训练动态 BP 网络。这样就将神经网络和自适应控制这两个领域联系起来了。可以预见,神经网络具有的非线性变换特性和高度的并行运算能力为系统辨识和自适应控制提供了一条十分有效的途径;另一方面,自适应控制理论中丰富的学习算法对研究神经网络的学习算法有启示。如何将这两个领域统一起来是值得进一步研究的。

从理论分析和模拟结果来看,新方法克服了常规 BP 算法收敛速度慢,学习率选择困难的缺点,新方法具有收敛速度快、跟踪性能好及鲁棒性强等优点。同时,新方法的收敛性分析也较为困难,作者正致力于这方面的研究。

参 考 文 献

1 Funahashi K I. On the approxinate realization of continuous mapping by newral networks, Neural Networks. 1989 (2):183～192

2 Hunt K J, Sharbaro B et al. Neural Network for control system——A Survey. 1992,28(6):1083～1112

3 田明,戴汝为. 基于动态 BP 神经网络的系统辨识方法. 自动化学报,1993,19(4):450～453

4 S-K. Sin, deFigueiredo R J P. An evolution oriented learning algorithm for optimal interpolative net IEEE Trans Neural Networks, 1992,3(2):315～323

5 Loh A P, Fong K F. Backpropagation using generalized Least Squares. ICNN, 1993. 592～597

6 顾炜 等. 一种适合于快速非线性预测的前馈神经网络,C^2N^2-94,1994

7 Astrōm K J, Wittenmark B. 自适应控制,李清泉等译. 北京:科学出版社,1992

8 Slotine J E, Li W. Applied Non-Linear Control. Prentice Hall, 1991