文章编号: 1001-0920(2002)05-0517-05

基于回归神经网络的非线性时变系统辨识

邹高峰, 王正欧
(天津大学 系统工程研究所, 天津 300072)

摘 要: 为克服基于前馈神经网络的非线性时变系统辨识算法存在需预先估计系统输入输出滞后阶数的缺陷,提出一种基于回归神经网络的非线性时变系统的辨识算法。针对现有的回归网络学习算法大多采用梯度算法,收敛速度缓慢问题,提出一种具有快速收敛性的扩展卡尔曼滤波学习算法,大大提高了学习收敛速度;并推导了一种基于单个神经元的局部化算法,减少了计算量。仿真实例证明,所提出的算法是有效的。

关键词: 系统辨识; 回归神经网络; 非线性时变系统; 扩展卡尔曼滤波
中图分类号: TP 273 文献标识码: A

Identification of nonlinear time varying systems based on recurrent neural networks

ZOU Gao-feng, WANG Zheng-ou
(Institute of System Engineering, Tianjin University, Tianjin 300072, China)

Abstract: A common drawback of the existing identification algorithms based on feedforward neural networks for nonlinear time varying systems is that the input and output delay orders of a system must be estimated in advance. A new identification approach based on recurrent neural networks is presented to overcome this drawback. The learning algorithms based on extended Kalman filter are derived. Compared with the training algorithms of most existing recurrent networks based on the gradient approach, the proposed algorithms largely improve the learning convergence, and the local algorithm reduces the computation cost. The simulation results demonstrate the effectiveness of the proposed approach.

Key words: system identification; recurrent neural networks; nonlinear time varying system; extended Kalman filter

1 引 言

人工神经网络因其具有强大的非线性多变量系统辨识能力,现已广泛应用于系统辨识中。然而,非线性时变系统大量存在于工程、经济、社会和生物系统中,应用传统的辨识方法很难解决这些非线性时变系统的辨识问题[1]。已经提出的基于前馈神经网络的非线性时变系统辨识方法[1~3],可较好地解决此问题。但他们存在的共同缺陷是需要预先估计系统的输入和输出滞后阶数,为实际应用带来了很大

收稿日期: 2001-06-22; 修回日期: 2002-05-08
基金项目: 国家自然科学基金项目(69774033)
作者简介: 邹高峰(1978—),男,河南虞城人,博士生,从事应用神经网络系统辨识、系统建模和金融工程等研究; 王正欧(1938—),男,上海人,教授,博士生导师,从事神经网络理论及应用、系统建模和数据挖掘等研究。

困难。为此,本文提出一种基于回归网络的非线性时变系统辨识的新算法,可以克服上述缺陷。

现有的回归网络大多采用基于梯度的学习算法[4, 5],收敛缓慢,难以适应时变系统需快速跟踪的要求。对此,本文提出一种具有快速收敛性的扩展卡尔曼滤波学习算法。为了降低学习算法的计算复杂性,还推导了一种基于单个神经元的局部化算法,大大减少了每次迭代的计算量和存储量,能很好地适应非线性时变系统的辨识。

本文以两个仿真实例验证了所提出算法的有效性。仿真结果表明,新算法的计算精度大大高于现有同类算法。

2 系统描述和回归神经网络模型

离散的非线性时变系统可描述为

$$y(k)=f(y(k-1),\cdots,y(k-n_y),u(k-1),\cdots,u(k-n_u);\Theta(k))+v(k) \tag{1}$$

其中,$y(k)\in R^m$,$u(k)\in R^n$ 和 $v(k)\in R^u$ 分别为输出向量、输入向量和噪声向量;n_y 和 n_u 分别为输出和输入的最大延迟;$v(k)$ 为零均值高斯独立分布序列;$\Theta(k)$ 为系统时变参数向量;$f(\cdot)$ 为未知的非线性向量函数。在给定系统输入和输出数据下设计回归神经网络的学习算法,在线辨识非线性时变系统,无需预先估计 n_y 和 n_u。

本文采用文献[6, 7]中的Elman回归网络模型结构。考虑一个3层回归网络,如图1所示。

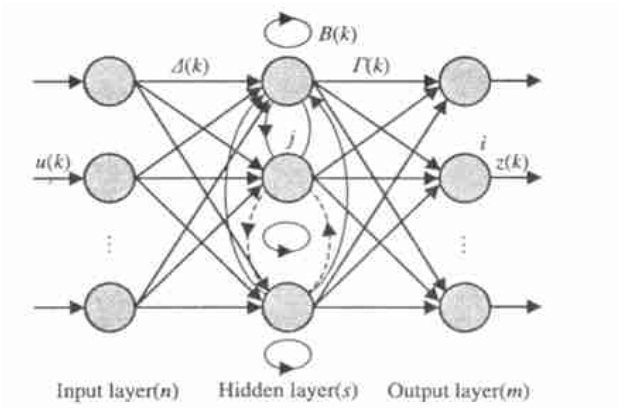


图1 回归神经网络模型

设 $z(k),h(k),u(k),g(k)$ 分别表示 m 维输出矢量、s 维隐含单元的状态矢量、n 维输入矢量和 s 维内部反馈状态矢量;$\Delta(k)\in R^{s\times(n+1)}$,$B(k)\in R^{s\times s}$,$\Gamma(k)\in R^{m\times s}$ 分别表示输入层与隐单元之间的连接权矩阵、回归单元连接权矩阵和隐单元与输出层之间的连接权矩阵。则回归神经网络模型可用如下方程描述

$$z(k)=\phi(\Gamma(k)h(k)) \tag{2}$$

$$h(k)=\varphi(\Delta(k)u(k)+B(k)g(k)) \tag{3}$$

$$g(k)=h(k-1) \tag{4}$$

ϕ,φ 表示第 k 时刻输出神经元、隐含神经元矢量函数(ϕ,φ 分别表示线性求和函数与 sigmoid 函数)。令 $\Theta(k)$ 表示所有神经网络连接权的 $M\times 1$ 向量,M 表示网络中所有连接权个数。

回归神经网络的回归变量,能将网络隐单元状态的信息以紧凑形式保留系统所有以前的输入信息[6],这是因为

$$\begin{aligned}g(k)&=h(k-1)=\\&\varphi(\Delta(k-1)u(k-1)+B(k-1)h(k-2))=\\&\cdots=\varphi(u^{k-1},\Theta^{k-1})\end{aligned} \tag{5}$$

其中

$$u^{k-1}=\{u(0),u(1),\cdots,u(k-1)\}$$

$$\Theta^{k-1}=\{\Theta(0),\Theta(1),\cdots,\Theta(k-1)\}$$

另外

$$\begin{aligned}z(k)&=\phi(\Gamma(k)\varphi(\Delta(k)u(k)+B(k)g(k)))=\\&\hat{f}(u(k),g(k),\Theta(k))\end{aligned} \tag{6}$$

$\hat{f}$ 表示神经网络的逼近函数,问题是在给定非线性时变系统的输入情况下估计网络输出 $z(k)$。

3 系统辨识的学习算法

用神经网络对非线性时变系统辨识就是将神经网络权值 $\Theta(k)$ 作为系统的时变参数,反映系统的输出变化。本文将 $\Theta(k)$ 作为系统的状态向量,用卡尔曼滤波进行估计。卡尔曼滤波已成功地应用于线性时变系统的辨识[8]。利用全局卡尔曼滤波算法能获得很高的计算精度,但需要很大的计算量和存储量。下面推导回归网络的卡尔曼滤波全局算法和局部化算法。

3.1 全局学习算法

当参数 $\Theta(k)$ 的变化规律未知时,假定服从随机游动规律

$$\Theta(k)=\Theta(k-1)+e(k) \tag{7}$$

其中 $e(k)$ 是系统噪声。进一步假定对所有 k,$E[e(k)]=0$,$E[e(k)e(j)^{\mathrm{T}}]=Q(k)\delta_{kj}$,$\delta$ 是狄拉克函数,$Q(k)$ 是正定矩阵。

由网络产生的模型误差 $\epsilon(k)$ 可写成

$$\begin{cases}y(k)-z(k)=\epsilon(k)\\y(k)=z(k)+\epsilon(k)=\hat{f}(u(k),\Theta(k))+\epsilon(k)\end{cases} \tag{8}$$

对所有 k 和 j, 有

$$E[\epsilon(k)] = 0, \quad E[\epsilon(k)\epsilon(j)^{\mathrm{T}}] = R(k)\delta_{kj}$$
$$E[e(k)\epsilon(j)^{\mathrm{T}}] = 0, \quad E[\Theta(0)e(k)^{\mathrm{T}}] = 0$$
$$E[\Theta(0)e(k)^{\mathrm{T}}] = 0, \quad E[\Theta(0)\epsilon(k)^{\mathrm{T}}] = 0$$

其中 $y(k)$ 和 $z(k)$ 表示期望输出和网络输出。在 $\hat{\Theta}(k-1)$ 处将非线性函数 $f(u(k);\Theta(k))$ 一阶泰劳展开, $\hat{\Theta}(k-1)$ 表示在 $k-1$ 时刻状态向量的估计值。

$$y(k) = \hat{f}(u(k), \hat{\Theta}(k-1)) + F(k)(\Theta(k) - \hat{\Theta}(k-1)) + \eta(k) + \epsilon(k) \tag{9}$$

其中

$$F(k) = \left.\frac{\partial \hat{f}(u(k), \Theta(k))}{\partial \Theta}\right|_{\Theta=\hat{\Theta}(k-1)} \tag{10}$$

$\eta(k)$ 是泰劳展开的高阶项, $F(k)$ 是一 $m \times M$ 阶矩阵。这样便可得到状态方程

$$\begin{cases} \Theta(k) = \Theta(k-1) + e(k) \\ y(k) = F(k)\Theta(k) + G(k) + \epsilon(k) \\ G(k) = f(u(k), \hat{\Theta}(k-1)) - F(k)\hat{\Theta}(k-1) + \eta(k) \end{cases} \tag{11}$$

$\eta(k)$ 在实际计算中可以忽略不计。$\Theta(k)$ 的最小方差无偏估计 $\hat{\Theta}(k)$ 和误差协方差矩阵 $P(k)$ 由扩展卡尔曼滤波直接得到

$$\begin{cases} \hat{\Theta}(k) = \hat{\Theta}(k-1) + K(k)[y(k) - F(k)\hat{\Theta}(k-1) - G(k)] \\ K(k) = P(k|k-1)F^{\mathrm{T}}(k)[F(k)P(k|k-1)F^{\mathrm{T}}(k) + R(k)]^{-1} \\ P(k|k-1) = P(k) + Q(k) \\ P(k) = [I - K(k)F(k)]P(k|k-1) \end{cases} \tag{12}$$

从式(12) 可以看到, 该算法需要存储 $M \times M$ 误差协方差矩阵 $P(k)$, 计算一个 $M \times m$ 卡尔曼滤波增益矩阵 $k(K)$ 和 $m \times m$ 矩阵的逆矩阵。算法初始化值可取 $P(0) = \delta^{-1}I$, δ 是一个任意小的正数, $\Theta(0)$ 是非零的随机参数。在卡尔曼滤波算法中 $P(k)$ 决定了参数估计的精度。假如误差协方差矩阵 $P(k)$ 按上式进行计算, 在某些条件下可能导致发散。这里采用一种称为恒定迹调整法[9]

$$\begin{cases} \bar{P}(k) = [I - K(k)F(k)]P(k|k-1) \\ P(k) = \dfrac{K_0}{\mathrm{trace}(\bar{P}(k))}\bar{P}(k), \quad K_0 > 0 \end{cases} \tag{13}$$

这样就设置了误差协方差矩阵特征值的上界。在随后的局部算法中也同样设置 $P_i(k)$ 和 $P_j(k)$。

3.2　局部学习算法

由于全局算法需要很大的计算量和存储量, 为简化全局算法, 将整个问题(最小化无偏估计误差的方差) 分解为一族可以掌握的小的单元。可以把问题分解到网络层, 网络节点, 甚至直到一个简单的连接权值[10,11]。在这里采用分解到单个神经元, 每次独立地更新每个神经元的连接权来解决整个问题, 从而得到所有与输出神经元的连接权和与隐单元的连接权的学习算法。

3.2.1　输出神经元连接权的学习算法

与输出层单元 i 的连接权向量(包括阈值 μ_i) 可表示为

$$\Theta_i^{(o)}(k) = [\mu_i(k), w_{i1}^{(o)}(k), \cdots, w_{is}^{(o)}(k)]^{\mathrm{T}}$$

隐层输出向量为

$$h(k) = [h_1(k), h_2(k), \cdots, h_s(k)]^{\mathrm{T}}$$

神经元 i 的输入向量为

$$H(k) = [1, h^{\mathrm{T}}(k)]^{\mathrm{T}}$$

按上述公式可得到 $\Theta_i^{(o)}$ 状态方程为

$$\begin{cases} \Theta_i^{(o)}(k) = \Theta_i^{(o)}(k-1) + v_i(k) \\ y_i(k) = H^{\mathrm{T}}(k)\Theta_i^{(o)}(k) + \epsilon(k) \end{cases} \tag{14}$$

对于 k 和 j, $E[\Theta_i^{(o)}(o)v_i(k)] = 0$, $E[\Theta_i^{(o)}(o)\epsilon^{\mathrm{T}}(k)] = 0$, $E[v_i(k)] = 0$, $E[\epsilon(k)] = 0$, $E[v_i(k)v_i^{\mathrm{T}}(j)] = Q_i(k)\delta_{kj}$, $E[\epsilon(k)\epsilon^{\mathrm{T}}(j)] = R_i(k)\delta_{kj}$, $E[v_i(k)\epsilon^{\mathrm{T}}(j)] = 0$, $Q_i(k)$ 和 $R_i(k)$ 均为正定矩阵, 则 $\Theta_i^{(o)}(k)$ 的估计 $\hat{\Theta}_i^{(o)}(k)$ 可以同时独立地进行计算如下

$$\begin{cases} \hat{\Theta}_i^{(o)}(k) = \hat{\Theta}_i^{(o)}(k-1) + K_i(k)[y_i(k) - H^{\mathrm{T}}(k)\hat{\Theta}_i^{(o)}(k-1)] \\ K_i(k) = P_i(k|k-1)H(k)[H^{\mathrm{T}}(k)P_i(k|k-1)H(k) + r_i(k)] \\ P_i(k|k-1) = P_i(k-1) + Q_i(k) \\ P_i(k) = [I - K_i(k)H(k)]P_i(k|k-1) \end{cases} \tag{15}$$

不需要矩阵求逆。$P_i(k)$ 初始化为 $P_i(0) = 1/\gamma$, γ 是小的正数, 初始化连接权 $\Theta_i^{(o)}(0)$ 是非零的随机向量。

3.2.2　隐单元连接权的学习算法

令 $\Theta_j^{(h)}(k) = [\mu^{(h)}(k), W_{j1}^{(h)}(k), \cdots, W_{jn}^{(h)}(k), W_{jn+1}^{(h)}(k), \cdots, W_{jn+s}^{(h)}(k)] = [\Theta_1^{(h)}(k)\ \Theta_2^{(h)}(k) \cdots \Theta_t^{(h)}(k)]$ 表示所有与隐单元 j 连接的连接权向量, $j = 1, 2, \cdots, s$; $\mu_j^{(h)}$ 表示该隐单元的阈值; 回归向量为 $k-1$ 时刻隐单元层的输出 $h(k-1)$; $t = 1 + n + s$。隐单元 j 对网络的输出 $z(k)$ 的影响可表述为

$$F_j(k) = \frac{\partial z(k)}{\partial \Theta_j^{(h)}} = \left.\frac{\partial \hat{f}(u(k), \Theta(k))}{\partial \Theta_j^{(h)}}\right|_{\Theta_j^{(h)}=\hat{\Theta}_j^{(h)}(k-1)}$$

这里仅考虑对 $\Theta_j^{(h)}(k)$ 线性化。类似于全局算法, 在

$\hat{\Theta}_j^{(h)}(k-1)$ 处一阶泰劳展开

$$\begin{cases} y(k) = F_j(k)\Theta_j^{(h)}(k) + G_j(k) + \epsilon(k) \\ G_j(k) = \hat{f}(u(k), \hat{\Theta}_j^{(h)}(k-1)) - \\ \qquad F_j(k)\hat{\Theta}_j^{(h)}(k-1) + \eta_j(k) \end{cases} \tag{16}$$

状态方程 $\Theta_j^{(h)}(k)$ 可简化为

$$\begin{cases} \Theta_j^{(h)}(k) = \Theta_j^{(h)}(k-1) + v_j(k) \\ y(k) = F_j(k)\Theta_j^{(h)}(k) + G_j(k) + \epsilon(k) \end{cases} \tag{17}$$

对于 k 和 j, $E[\epsilon_j(k)] = 0$, $E[\Theta_j^{(h)}(0)\epsilon_j^{\mathrm{T}}(k)] = 0$, $E[\epsilon_j(k)\epsilon_j^{\mathrm{T}}(j)] = R(k)\delta_{kj}$; $E[v_j(k)\epsilon_j^{\mathrm{T}}(j)] = 0$, $E[\Theta_j^{(h)}(0)v_j^{\mathrm{T}}(k)] = 0$, $E[v_j(k)v_j^{\mathrm{T}}(j)] = Q_j(k)\delta_{kj}$; $E[v_j(k)] = 0$, $Q_j(k)$ 和 $R(k)$ 都是正定矩阵, 则有

$$\begin{cases} \hat{\Theta}_j^{(h)}(k) = \hat{\Theta}_j^{(h)}(k-1) + K_j(k)[y(k) - \\ \qquad F_j(k)\hat{\Theta}_j^{(h)}(k-1) - G_j(k)] \\ K_j(k) = P_j(k|k-1)F_j^{\mathrm{T}}(k)[F_j(k)P(k|k- \\ \qquad 1)F_j^{\mathrm{T}}(k) - R(k)]^{-1} \\ P_j(k|k-1) = P_j(k) + Q_j(k) \\ P_j(k) = [I - K_j(k)F_j(k)]P_j(k|k-1) \end{cases} \tag{18}$$

通常情况下, $P_j(k)$ 初始化为 $P_j(0) = I \cdot 1/\gamma$, γ 是小的正数。$\Theta_j^{(h)}(0)$ 是非零的随机向量。此算法对每个隐单元需要存储 $t \times t$ 协方差矩阵 $P_j(k)$、计算 $t \times m$ 增益矩阵 $K_j(k)$ 和一个 $m \times m$ 逆矩阵, 每个隐单元都能同时独立计算。

4 仿真实例

在卡尔曼滤波中假定 Q 和 R 是已知的, 然而, 在很多实际情况下 Q 和 R 通常都是未知的。到目前为止, 许多在线估计 Q 和 R 的方法, 其收敛速度非常慢, 不能适应本算法的要求。这里根据经验确定 Q 和 R, Q 和 R 均取对角元素为相同常数的对角矩阵, 给定 Q, 用上述卡尔曼辨识算法估计 $\Theta(k)$, 计算网络系统输出 $z(k)$, $k = 1, 2, \cdots, N$, 其中 N 为样本数。均方误差 δ 为

$$\delta = \frac{1}{N}\sum_{k=1}^{N}[y(k) - z(k)]^2$$

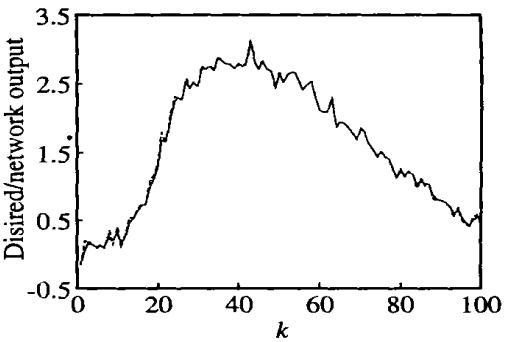
不同的 Q 对应不同的 δ, 取 Q 应使得对应的 δ 达到最小值, 详细算法参见文献[1]。R 的取值对卡尔曼滤波计算不敏感, 每次计算取为固定值。仿真实例说明这种取法具有很好的效果。下面用 2 个例子来验证这种算法的有效性。

例 1 考虑如下非线性时变系统[11]

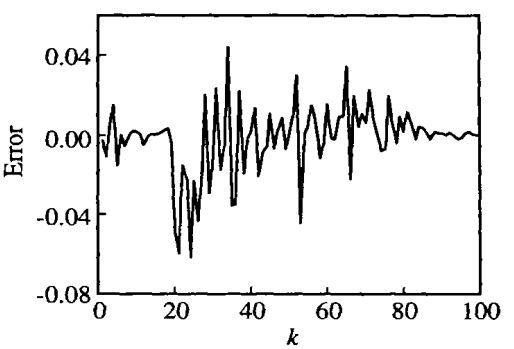
$$y(k+1) = \frac{y(k)}{1 + 0.68\sin(0.000\,5\pi k)y^2(k)} + 0.78u^3(k) + v(k)$$

其中 $v(k)$ 是零均值标准差为 0.1 的高斯白噪声。系统输入 $u(k) = \sin(0.01\pi k)$, 连接权值均初始化为 0 ~ 1 的均匀分布随机数。误差协方差矩阵初始化为 $P(0) = \mathrm{diag}(1, 1, \cdots, 1)$, R 阵取为 0.01 的对角矩阵。网络模型为 1–10–1, 输入单元为 $u(k)$。共有 130 个连接权系数。经启发式算法求得 $Q = \mathrm{diag}(0.05, 0.05, \cdots, 0.05)$。

仿真结果如图 2 所示, 图 2(a) 中实线表示系统的期望输出, 虚线表示网络的实际输出; 图 2(b) 为网络输出与系统的期望输出误差曲线。与文献[1]结果相比, 计算精度提高很多(本文算法误差范围为 [-0.06, 0.06], 而文献[1]误差范围为 [-0.4, 0.4]; 本文算法只有一个输入变量 $u(k)$, 而文献[1]的输入变量为 4 个($y(k)$, $y(k-1)$, $u(k)$ 和 $u(k-1)$)。



(a) 期望输出与网络输出



(b) 期望输出与网络输出误差曲线

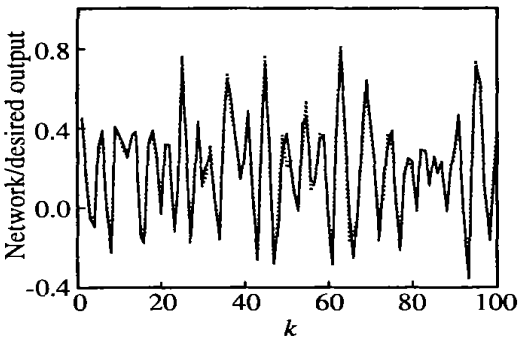
图 2 例 1 仿真曲线

例 2 考虑如下时变 NARMAX 模型[11]

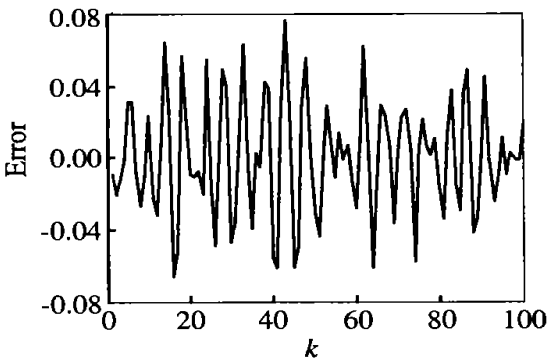
$$\begin{aligned} y(k+1) = {} & 0.3y(k) + 0.5u(k) - \\ & 0.8y(k-1)\exp(-y(k))(1 + \\ & 0.5\sin(0.002\pi k))) + \\ & 0.1u(k)u(k-1) + v(k) \end{aligned}$$

其中, $v(k)$ 是零均值标准差为 0.01 的高斯白噪声, $u(k)$ 是独立的[0, 1] 之间均匀分布的序列。连接权值均初始化为[0, 1] 的均匀分布随机数, 误差协方差矩阵初始化为 $P(0) = \mathrm{diag}(1, 1, \cdots, 1)$, R 阵取为 0.01 的对角矩阵。网络模型为 1–12–1, 输入单元为 $u(k)$, 共有 180 个连接权系数。经启发式算法求得 $Q = \mathrm{diag}(0.006, 0.006, \cdots, 0.006)$。

仿真结果如图 3 所示。图 3(a) 中实线表示系统的期望输出, 虚线表示网络的实际输出; 图 3(b) 表示网络输出与系统的期望输出误差曲线。可以看出, 此系统依赖于过去的全部信息。文献[1] 输入变量为 $y(k)$, $y(k-1)$, $u(k)$ 和 $u(k-1)$, 误差范围为 [−0.5, 0.5]; 而回归网络算法只需要输入变量 $u(k)$, 误差范围为[−0.08, 0.08]。显然, 回归网络的精度远高于前馈网络。



(a)　期望输出与网络输出



(b)　期望输出与网络输出误差曲线

图 3　例 2 仿真曲线

5　结　　论

本文给出了基于回归神经网络的用于非线性时变系统辨识的全局非线性时变系统辨识算法和局部化算法, 其中局部化算法使存储量和计算量大大降低。本文算法与现有的其他同类神经网络模型相比, 不用预先估计系统的时滞阶数, 而且精度更高, 是一种实用的非线性时变系统辨识算法。

参考文献(References):

[1] Wang Z O, Zhao C H. Identification of nonlinear time varying system using feed-forward neural networks[J]. *J of Tianjin University*, 2000, 1(1): 8–13.

[2] Thomas F J, Heinz U. On-line identification of nonlinear time variant systems using structural adaptive radial basis function networks[A]. *Proc of the Americal Control Conf* [C]. Albuquerque, 1997. 1037–1041.

[3] Yingwei L, Sundararajan N, Saratchandran P. Identification of time-varying nonlinear systems using minimal radial basis function neural networks[J]. *IEE Proc-Control Theory and Application*, 1997, 144(2): 202–208.

[4] Amir F A, Alexander G P. New results on recurrent networks training: Unifying the algorithms and accelerating convergence[J]. *IEEE Trans on Neural Networks*, 2000, 11(8): 697–709.

[5] Barak A P. Gradient calculations for dynamic recurrent neural networks: A survey[J]. *IEEE Trans on Neural Networks*, 1995, 6(5): 1212–1228.

[6] 韦巍. 一种回归神经网络的快速在线学习算法[J]. 自动化学报, 1998, 24(5): 616-621.
(Wei Wei. A new on-line recursive learning algorithm for recurrent neural networks[J]. *ACTA Automation Sinica*, 1998, 24(5): 616-621.)

[7] Kuan C M, Hornik K, White H. A convergence result for learning in recurrent neural networks[J]. *Neural Computation*, 1994, 6(2): 420–440.

[8] 刘豹, 王正欧. 系统辨识[M]. 北京: 机械工业出版社, 1993.

[9] Liu B, Wang Z O. Estimation of parameters of time varying systems with small samples[A]. *Proc of the 7th IFAC/IFORS Symposium on Identification and System Parameter* [C]. York: Pergamon Press, 1985. 1149–1154.

[10] Puskorious G V, Feldkamp L A. Decoupled extended Kalman filter training of feedforward layered networks [A]. *Int Joint Conf Neural Networks*[C]. Int Neural Network Soc Published by IEEE, 1991. 771–777.

[11] Shan S, Palmieri F. MEKA —— A fast, local algorithm for training feedforword neural networks[A]. *Int Joint Conf Neural Networks*[C]. Int Neural Network Soc Published by IEEE, 1990. 41-46.