文章编号: 1001-0920 (2014) 07-1274-05 DOI: 10.13195/j.kzyjc.2013.0518

非线性系统RBF神经网络多步预测控制

樊兆峰[1,2], 马小平[1], 邵晓根[2]
(1. 中国矿业大学 信息与电气工程学院，江苏 徐州 221008；2. 徐州工程学院 信电工程学院，江苏 徐州 221111)

摘 要: 针对较强非线性的控制问题, 提出一种以RBF神经网络为模型的多步预测控制方法. 构建多步预测模型, 并给出预测误差关于控制序列的雅可比矩阵的计算方法. 利用Levenberg-Marquardt(L-M)算法设计滚动优化策略, 通过误差修正参考输入的方法实现了反馈校正, 证明了控制系统的稳定性. 仿真结果表明所提出的控制方法效果较好.
关键词: 预测控制；RBF神经网络；L-M算法；滚动优化
中图分类号: TP273 **文献标志码:** A

RBF neural network multi-step predictive control for nonlinear systems

FAN Zhao-feng[1,2], *MA Xiao-ping*[1], *SHAO Xiao-gen*[2]
(1. School of Information and Electrical Engineering，China University of Mining and Technology，Xuzhou 221008, China；2. College of Information and Electrical Engineering，Xuzhou Institute of Technology，Xuzhou 221111，China. Correspondent：MA Xiao-ping，E-mail：xpma@cumt.edu.cn)

Abstract: Aim at solving the strong nonlinear control problem, a multi-step predictive control method is proposed, which uses a RBF neural network as a model. A multi-step predictive model is constructed, a Jacobian matrix computing method for predictive error about control sequence is given, a receding horizon optimization policy is designed by using L-M algorithm, feedback correction is achieved by modifying reference input according the error, and the stability of the system is proved. Simulation results of the control method validate desirable performances.
Key words: predictive control；RBF neural network；L-M algorithm；receding horizon optimization

0 引 言

预测控制是来源于工业实践的一类新型计算机优化控制算法, 其本质是基于模型的有限时域优化算法. 预测控制一般建立在3项基本原理基础上: 预测模型、滚动优化和反馈校正[1]. 典型的控制算法(如动态矩阵控制、广义预测控制等) 使用线性模型预测, 对强非线性的对象难以应用, 而非线性系统的数学模型 (如块联模型、基于各种核函数描述的模型) 又存在结构特定、辨识困难、处理复杂等问题, 实际中也很少使用. 由于神经网络能够充分逼近复杂的非线性关系, 具有学习和适应不确定系统的动态特性、较强的鲁棒性、容错性等特点, 成为预测控制的有力工具[2].

部分文献选取BP网络为预测模型[3-4], 虽然BP网络是使用最为成熟的网络, 但也存在局部最优、训练速度慢等缺点, 而RBF神经网络在一定程度上克服了BP网络的问题, 因此, 近年来已有研究者将RBF神经网络用于预测控制[5-7]. 已有文献中, 有的通过求解非线性方程获得最优控制量 (实质是一步预测控制), 有的利用附加的RBF神经网络进行优化. 目前使用L-M算法实现多步预测控制的文献较少, 该方法相比于前者既能实现真正的多步预测控制, 又能提高响应速度, 减少超调量; 相比于后者可以减少一个附加的优化网络. 因此, 本文以L-M算法为基础设计非线性系统RBF神经网络多步预测控制器.

1 系统控制原理

控制系统如图1所示. RBF神经网络使用前先离线学习并以一定精度逼近非线性被控对象, 控制前先将控制量u作用于RBF神经网络预测模型, 得到预测输出值y_m. 优化控制器根据参考输入r和y_m求得有限时域(即预测时域) 内的局部最优控制量u^*, 用u^*实现控制. 为了减小预测误差的影响, 在每个采样周期内计算系统实际输出y与预测值y_m的误差值,

收稿日期: 2013-04-26；**修回日期:** 2013-07-22.
基金项目: 国家自然科学基金项目(60974126)；建设部科技计划基金项目(2013-K8-32).
作者简介: 樊兆峰(1972−), 男, 副教授, 博士生, 从事神经网络预测控制、机器人控制的研究；马小平(1961−), 男, 教授, 博士生导师, 从事计算机控制理论及应用等研究.

用于修正参考输入值, 即可实现闭环反馈校正.

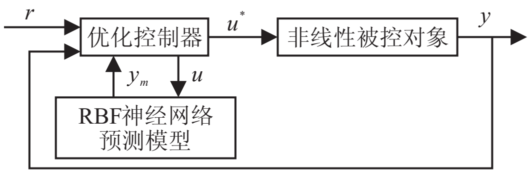

图 1　控制系统结构

2　预测模型

2.1　非线性系统模型表示

由于预测控制要依赖于计算机实现, 属于离散控制系统, 本文采用非线性离散系统的数学模型. 考虑如下单输入单输出 (SISO) 被控系统:

$$\begin{aligned}&y(k+1)=\\&f[y(k),y(k-1),\cdots,y(k-n_y+1),\\&u(k),u(k-1),\cdots,u(k-n_u+1)].\end{aligned}\qquad(1)$$

其中: u 为控制输入值; y 为非线性系统的输出值; f 为系统的非线性映射关系; n_y, n_u 分别为输出时间序列和控制时间序列的阶次, 可以通过模型阶次辨识的方法得到[8]. 如果将模型 (1) 中的历史 (k 时刻和之前时刻) 输出信息 $y(k), y(k-1), \cdots, y(k-n_y+1)$ 和历史输入信息 $u(k), u(k-1), \cdots, u(k-n_u+1)$ 作为 RBF 神经网络的输入, 模型下一时刻的输出 $y(k+1)$ 作为 RBF 神经网络的输出, 通过输入输出样本数据训练神经网络, 使 RBF 神经网络以一定精度逼近非线性映射关系 f, 则 RBF 网络即可成为非线性系统的数学模型.

2.2　一步预测模型

系统一步预测模型如图 2 所示. 图 2 中: $\boldsymbol{p}$ 为输入向量, 维数为 $R\times 1$, $R=n_y+n_u$, 其结构形式为 $[y(k), y(k-1), \cdots, y(k-n_y+1), u(k), u(k-1), \cdots, u(k-n_u+1)]^{\mathrm{T}}$; S_1 为径向基神经元的个数, S_2 为线性输出神经元的个数, 对 SISO 系统而言有 $S_2=1$; $\|\cdot\|$ 为计算向量间的欧氏距离; $\mathbf{IW}$ 为输入权值矩阵, 维数为 $S_1\times R$, $\mathbf{LW}$ 为输出权值矩阵, 维数为 $S_2\times S_1$; $\boldsymbol{b}_1$ 为输入偏置向量, 维数为 $S_1\times 1$, $\boldsymbol{b}_2$ 为输出偏置向量, 维数为 $S_2\times 1$; $\boldsymbol{n}_1$ 为径向基神经元的输入向量, 维数为 $S_1\times 1$, $\boldsymbol{n}_2$ 为线性输出神经元的输入向量, 维数为 $S_2\times 1$; $\boldsymbol{a}$ 为径向基神经元的输出, 维数为 $S_1\times 1$; $y_m(k+1)$ 为 RBF 神经网络的输出. 由图 2 可以得到

$$\boldsymbol{n}_1=\|\mathbf{IW}-\boldsymbol{p}\|.*\boldsymbol{b}_1,\qquad(2)$$

$$\boldsymbol{a}=\exp(-\boldsymbol{n}_1.^2),\qquad(3)$$

$$y_m(k+1)=\mathbf{LW}\times\boldsymbol{a}+\boldsymbol{b}_2.\qquad(4)$$

其中: “$.*$”为两个向量的点乘; “$.^2$”为向量点乘方; $\|\mathbf{IW}-\boldsymbol{p}\|$ 为 S_1 维欧氏距离列向量, 且有

$$\|\mathbf{IW}-\boldsymbol{p}\|=\\\left[\sqrt{\sum_{l=1}^{R}(\mathbf{IW}_{1,l}-p_l)^2}\ \cdots\ \sqrt{\sum_{l=1}^{R}(\mathbf{IW}_{S_1,l}-p_l)^2}\right]^{\mathrm{T}},\qquad(5)$$

$\mathbf{IW}_{1,l}$ 为矩阵 $\mathbf{IW}$ 的第 1 行第 l 列元素, p_l 为向量 $\boldsymbol{p}$ 的第 l 个元素.

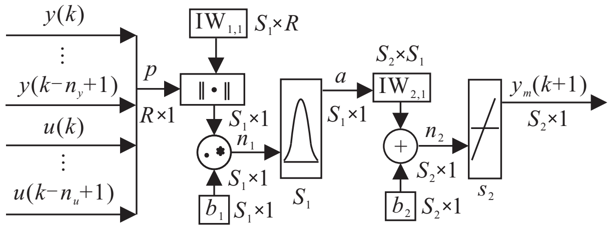

图 2　一步预测模型结构

由式 (2) ∼ (4) 可得 RBF 网络输入输出映射关系的数学表达式为

$$\begin{aligned}&y_m(k+1)=\\&\mathbf{LW}\times(\exp(-\|\mathbf{IW}-\boldsymbol{p}\|.^2).*(\boldsymbol{b}_1.^2))+\boldsymbol{b}_2.\end{aligned}\qquad(6)$$

将系统历史信息 $\boldsymbol{p}$ 代入式 (6) 可计算模型的下一步输出 $y_m(k+1)$, 由于经过训练后 RBF 网络能够充分逼近非线性映射关系 f, 可用 $y_m(k+1)$ 代替被控对象的下一步输出 $y(k+1)$, 从而实现一步预测.

2.3　多步预测模型

在一步预测模型的基础上, 用一步预测得到的输出值 $y_m(k+1)$ 和控制输入值 $u(k+1)$ 更新输入向量 $\boldsymbol{p}$ 为 $[y_m(k+1)\ \ y_m(k)\ \ \cdots\ \ y(k-n_y+2)\ \ u(k+1) \rightarrow \leftarrow u(k)\ \ \cdots\ \ u(k-n_u+2)]^{\mathrm{T}}$, 调用一步预测模型 (代入式 (6)) 即可预测再下一时刻的输出值 $y_m(k+2)$, 如此不断更新输入向量, 递归调用一步预测模型, 理论上便可预测未来任何时刻的输出值. 设优化时域等于控制时域等于 d, 当前时刻为 k, 由上述方法可得到系统的 d 步预测模型如图 3 所示.

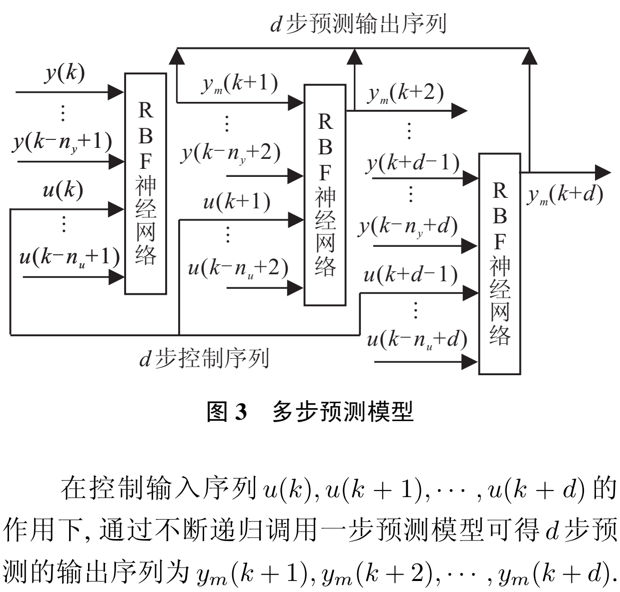

图 3　多步预测模型

在控制输入序列 $u(k), u(k+1), \cdots, u(k+d)$ 的作用下, 通过不断递归调用一步预测模型可得 d 步预测的输出序列为 $y_m(k+1), y_m(k+2), \cdots, y_m(k+d)$.

令

$$\boldsymbol{u}=[u(k)\ \ u(k+1)\ \ \cdots\ \ u(k+d)]^{\mathrm{T}}, \tag{7}$$

$$\boldsymbol{y}_m=[y_m(k+1)\ \ y_m(k+2)\ \ \cdots\ \ y_m(k+d)]^{\mathrm{T}}, \tag{8}$$

$$\boldsymbol{y}_e=[y_e(k+1)\ \ y_e(k+2)\ \ \cdots\ \ y_e(k+d)]^{\mathrm{T}}. \tag{9}$$

其中: 控制向量$\boldsymbol{u}$、模型预测输出向量$\boldsymbol{y}_m$、期望输出向量$\boldsymbol{y}_e$的维数均为d, $\boldsymbol{y}_e$为在k时刻控制器根据参考输入r换算得到的d步期望输出序列.

3 滚动优化与反馈校正

对于多步(d步)预测控制而言, 优化就是选择一个d维控制向量$\boldsymbol{u}$, 在该控制向量的作用下, 系统的d步预测输出$\boldsymbol{y}_m$能充分逼近参考输出向量$\boldsymbol{y}_e$. 用$\boldsymbol{u}^*$表示该控制向量, 以示区别. 预测控制器在k时刻只将向量$\boldsymbol{u}^*$中的第1个元素即$u^*(k)$用于对系统的控制, 在下一个时刻$k+1$再重复k时刻的预测优化过程, 并将重新求得的控制向量中的第1个元素用于对系统的控制, 如此不断循环实现滚动优化.

3.1 目标性能函数

为了求得优化控制$\boldsymbol{u}^*$, 取如下二次型函数为目标性能函数:

$$\boldsymbol{J}=\sum_{j=1}^{d}[y_e(k+j)-y_m(k+j)]^2+ \lambda^2\sum_{j=1}^{d}[u(k+j)-u(k+j-1)]^2, \tag{10}$$

其中λ为优化权重系数. 定义d维偏差向量

$$\boldsymbol{e}=\boldsymbol{y}_e-\boldsymbol{y}_m= [e(k+1)\ \ e(k+2)\ \ \cdots\ \ e(k+d)]^{\mathrm{T}}. \tag{11}$$

定义d维控制增量

$$\Delta\boldsymbol{u}=[u(k+1)-u(k)\ \ u(k+2)-u(k+1)\ \ \cdots\to \leftarrow u(k+d)-u(k+d-1)]^{\mathrm{T}}. \tag{12}$$

$2d$维向量

$$\boldsymbol{f}=[\boldsymbol{e}^{\mathrm{T}}\ \ \lambda\Delta\boldsymbol{u}^{\mathrm{T}}]^{\mathrm{T}}. \tag{13}$$

则式(10)可写作向量表示的最小二乘形式

$$\min\ \boldsymbol{J}=\boldsymbol{f}^{\mathrm{T}}\boldsymbol{f}. \tag{14}$$

3.2 L-M 优化算法

式(14)经典的L-M迭代算法公式为

$$\boldsymbol{u}^{i+1}=\boldsymbol{u}^i-(A_i^{\mathrm{T}}A_i+\alpha_iI)^{-1}A_i^{\mathrm{T}}\boldsymbol{f}^i. \tag{15}$$

其中: α_i为一正系数, I为d维单位矩阵, $\boldsymbol{u}^i$为第i次迭代的控制输入序列, $\boldsymbol{f}^i$为第i次迭代计算的向量$\boldsymbol{f}$值, $2d\times d$维矩阵A_i表示为

$$A_i=\frac{\partial\boldsymbol{f}^i}{\partial\boldsymbol{u}}=\Big[\Big(\frac{\partial\boldsymbol{e}^i}{\partial\boldsymbol{u}}\Big)^{\mathrm{T}}\ \ \Big(\frac{\partial\lambda\Delta\boldsymbol{u}^i}{\partial\boldsymbol{u}}\Big)^{\mathrm{T}}\Big]^{\mathrm{T}}=[G^{\mathrm{T}}\ \ D^{\mathrm{T}}]^{\mathrm{T}}. \tag{16}$$

显然A_i由偏差和控制增量对控制序列的两个雅可比矩阵G和D组成, 由于历史和当前时刻的状态与未来的控制作用无关, 根据式(7)$\sim$(9)和(11)有

$$G=\frac{\partial\boldsymbol{e}^i}{\partial\boldsymbol{u}}= \begin{bmatrix} \dfrac{\partial e(k+1)^i}{\partial u(k)} & 0 & \cdots & 0\\ \dfrac{\partial e(k+2)^i}{\partial u(k)} & \dfrac{\partial e(k+2)^i}{\partial u(k+1)} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ \dfrac{\partial e(k+d)^i}{\partial u(k)} & \dfrac{\partial e(k+d)^i}{\partial u(k+1)} & \cdots & \dfrac{\partial e(k+d)^i}{\partial u(k+d-1)} \end{bmatrix}. \tag{17}$$

雅可比矩阵G是下三角矩阵, 由式(6)和(11)得到第1行第1列元素为

$$\frac{\partial e(k+1)^i}{\partial u(k)}=-\frac{\partial y_m(k+1)^i}{\partial u(k)}= -2\mathbf{LW}\times(a^i(k+1).*\boldsymbol{b}_1.^2.*(\mathbf{IW}_{:,n_y+1}-u(k))). \tag{18}$$

其中: $a^i(k+1)$为$k+1$时刻径向基第i次输出向量值; $(\mathbf{IW}_{:,n_y+1}-u(k))$为列向量$[\mathbf{IW}_{1,n_y+1}-u(k)\to\leftarrow\mathbf{IW}_{2,n_y+1}-u(k)\ \ \cdots\ \ \mathbf{IW}_{S_1,n_y+1}-u(k)]^{\mathrm{T}}$.

定义函数

$$h(x)=\begin{cases}0,\ x\leqslant 0;\\ 1,\ x>0.\end{cases}$$

设前$s(1\leqslant s\leqslant d-1)$行元素已求出, 下面求解第$s+1$行第$q$列不为零的元素为

$$\frac{\partial e(k+s+1)^i}{\partial u(k+q-1)}=-\frac{\partial y_m(k+s+1)^i}{\partial u(k+q-1)}= -2\mathbf{LW}\Big(a^i(k+s+1).*\boldsymbol{b}_1.^2.*\Big((\mathbf{IW}_{:,n_y+1+s-q}- u(k+q-1))h(n_u-s+q)+\sum_{l=q}^{s}h(n_y-s+ l)(\mathbf{IW}_{:,l}-y_m(k+l))\frac{\partial y_m(k+l)^i}{\partial u(k+l-1)}\Big)\Big). \tag{19}$$

当q遍历$1\sim s+1$的每一个值时, 根据式(19)可以求出第$s+1$行的每一列值, 当s遍历$1\sim d-1$的每一个值时, 根据式(18)和(19)可以求出整个矩阵G.

对于第2个雅可比矩阵D, 由式(7)、(12)和(16)可得

$$D=\frac{\partial\lambda\Delta\boldsymbol{u}^i}{\partial\boldsymbol{u}}= \lambda\begin{bmatrix} -1 & 1 & 0 & \cdots & 0\\ 0 & -1 & 1 & \cdots & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & 0 & -1 \end{bmatrix}=\lambda B. \tag{20}$$

根据式(17)$\sim$(20)可计算出完整的矩阵A_i, 至此可以设计L-M算法如下.

Step 1: 初始化参数. 取$\alpha_i>0$和增长因子$\beta>1$,

初始点 $\boldsymbol{u}^1$ 和充分小的允许误差 $\epsilon > 0$, 最大迭代次数 i_m, 置迭代次数 $i = 1$.

Step 2: 若迭代次数 $i > i_m$, 则停止计算, 得到解 $\boldsymbol{u}^* = \boldsymbol{u}^i$, 否则计算矩阵 A_i 和向量 f^i.

Step 3: 由式 (15) 计算第 $i+1$ 次控制向量 $\boldsymbol{u}^{i+1}$, 由式 (6) 和 (10) 计算目标性能函数值 $\boldsymbol{J}^{i+1}$.

Step 4: 若 $\boldsymbol{J}^{i+1} < \boldsymbol{J}^i$, 则转至 Step 6.

Step 5: 若 $\boldsymbol{e}^{\mathrm{T}}\boldsymbol{e} \leqslant \epsilon$, 则停止计算, 得到解 $\boldsymbol{u}^* = \boldsymbol{u}^i$, 否则置 $\alpha_i \leftarrow \beta\alpha_i$, 转至 Step 3.

Step 6: 若 $\boldsymbol{e}^{\mathrm{T}}\boldsymbol{e} \leqslant \epsilon$, 则停止计算, 得到解 $\boldsymbol{u}^* = \boldsymbol{u}^{i+1}$, 否则置 $\alpha_i \leftarrow \alpha_i/\beta$, $i \leftarrow i+1$, $u_i \leftarrow u_{i+1}$, 转至 Step 2.

3.3 反馈校正

在第 k 次采样周期内, 设检测的系统实际输出为 $y(k)$, 上一周期时预测的本次系统输出为 $y_m(k)$, 则预测误差为 $\delta = y(k) - y_m(k)$, 在根据参考输入确定本次期望输出序列向量时作以下修正:

$$\boldsymbol{y}_e = [y_e(k+1)+\delta \;\; y_e(k+2)+\delta \;\; \cdots \;\; y_e(k+d)+\delta]^{\mathrm{T}}. \tag{21}$$

然后进行本次预测优化控制即可实现反馈校正.

4 稳定性分析

以 $\boldsymbol{J}_k$ 表示第 k 次的目标性能函数值, 由式 (10) 可知 $\boldsymbol{J}_k \geqslant 0$, 如果 $k \to \infty$ 时有 $\boldsymbol{J}_k = \boldsymbol{J}_\infty = 0$, 则系统必然是稳定的. 这是因为当 $\boldsymbol{J}_\infty = 0$ 时, 由式 (10) 可得 $y_m(\infty) = y_e(\infty)$, 即系统输出等于期望输出, 系统显然是渐近稳定的, 下面只需证明当 $k \to \infty$ 时有 $\boldsymbol{J}_\infty = 0$ 成立.

采用反证法[9], 假设 $k \to \infty$ 时 $\boldsymbol{J}_\infty \neq 0$. 由 L-M 算法的 Step 4 可知, 目标性能函数 $\boldsymbol{J}_k$ 是一个单调递减的函数, 对于任意时刻 k 有

$$\Delta\boldsymbol{J}_k = \boldsymbol{J}_{k+1} - \boldsymbol{J}_k < 0,\; \boldsymbol{J}_\infty \leqslant \boldsymbol{J}_k \leqslant \boldsymbol{J}_0.$$

其中 $\boldsymbol{J}_0$ 为初始时刻的目标性能函数值, 为一常数. 显然 $\boldsymbol{J}_k$ 属于有界闭集 (紧集), 由 Weierstrass 定理可知 $\Delta\boldsymbol{J}k$ 存在最大值. 设最大值为 $-\Delta_{\max}$, 由题设 $\boldsymbol{J}_\infty \neq 0$, 又因为目标性能函数非负, 所以 $-\Delta_{\max} \neq 0$. 将 $\boldsymbol{J}_k$ 表示为

$$\boldsymbol{J}_k = \boldsymbol{J}_0 + \sum_{i=0}^{k-1}\Delta\boldsymbol{J}_i \leqslant \boldsymbol{J}_0 - k\Delta_{\max}. \tag{22}$$

因为 $-\Delta_{\max}$ 是一个小于 0 的常数, 所以式 (22) 表明当 $k \to \infty$ 时 $\boldsymbol{J}_\infty \to -\infty$, 显然这与已知目标性能函数非负矛盾, $\boldsymbol{J}_\infty = 0$.

综上可知, 本文设计的控制系统是渐近稳定的.

5 仿真分析

考虑如下离散非线性系统:

$$y_{k+1} = u_k^3 + \frac{y_k}{1+y_k^2}. \tag{23}$$

由式 (23) 可知, 系统的输入维数 $R = n_y + n_u = 2$, 输出为 1 维, 因此需要构造一个 2 输入 1 输出的 RBF 神经网络. 通过以下方式产生样本: 在 $[-1 \;\; 1]$ 范围内随机生成 1 000 个数据为输入 u, 根据式 (23) 计算输出数据 y, 将 1 000 个输入输出数据对作为网络训练的样本. 仿真在 Matlab 2010 b 环境下编程实现, RBF 神经网络的训练利用 Matlab 的神经网络工具箱中的函数 newrb, 训练精度取 1×10^{-4}.

5.1 本文控制方法仿真

以正弦信号为输入进行仿真, 参数设置如下: $d = 3$, 即采用 3 步预测控制, $\lambda = 0.01, \alpha_i = 10, \beta = 3$, $\epsilon = 1 \times 10^{-5}$, 采样周期为 1 s, 初始点 $u_1 = [1 \;\; 1 \;\; 1]$. 最大迭代次数取 100, 得到仿真结果如图 4 所示, 实线为实际输出, 虚线为参考输出. 由图 4 可见, 本文设计的控制方法响应速度快, 基本无超调, 对正弦信号能很好地跟踪.

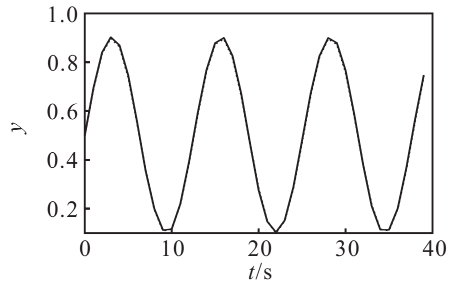

图 4　正弦信号跟踪曲线

5.2 本文控制方法实时性

对预测控制而言, 一个重要的问题是优化时间, 即实时性问题, 这是能否实用的关键, 所以要对本文的控制优化时间进行测定. 系统硬件条件为 CPU 1.79 GHz, 内存为 0.99 GB, 操作系统为 windows XP, 测定工具采用 Matlab 中测试程序运行时间的指令 tic, toc. 仍以前文的仿真程序为基础, 相应的控制程序和参数设置不变, 只改变网络训练精度, 并由此改变网络规模, 分别对 40 次优化过程进行测试后, 得到结果如表 1 所示. 表 1 中: ε 为 RBF 神经网络的训练精度, $N_{\rm net}$ 为神经网络规模, 即神经元的个数; $t_{\min}$ 为最小优化时间; $t_{\max}$ 为最大优化时间; $t_{\rm avg}$ 为平均优化时间.

表 1　本文控制方法所需优化时间

ε	$N_{\rm net}$	$t_{\min}$/s	$t_{\max}$/s	$t_{\rm avg}$/s
1×10^{-2}	9	5.43×10^{-4}	1.2×10^{-3}	6.83×10^{-4}
1×10^{-3}	15	4.72×10^{-4}	2.16×10^{-2}	1.4×10^{-3}
1×10^{-4}	22	4.83×10^{-4}	6.74×10^{-2}	8.6×10^{-3}
1×10^{-5}	32	5.01×10^{-4}	3.55×10^{-1}	2.27×10^{-2}
1×10^{-6}	45	5.12×10^{-4}	3.46×10^{-1}	3.01×10^{-2}

由表 1 可知, 随着网络规模的增大, 优化时间加长. 在满足以上硬件条件的基础上, 网络规模小于 45 时, 本文设计的控制方法一般可用于采样周期大于 0.5 s 的非线性系统控制.

5.3 与其他控制方法比较

目前, 在非线性系统控制领域, 工业中应用较多的是 PID 控制和滑模变结构控制, 文献 [7] 也提出了一种 RBF 神经网络预测控制方法, 为此将本文设计的方法与这 3 种方法进行比较. 参考信号取多点跃变信号, PID 控制采用增量式, 其参数的整定先用 Ziegler-Nichols 法初步整定, 再用优选法仿真整定为 $K_{\rm P} = 0.184$, $K_{\rm I} = 1.6$, $K_{\rm D} = 0.01$. 滑模变结构控制采用文献 [10] 的新方法, 其中参数设置为: 采样周期 0.1 s, $c = 6$, $\alpha = 8$, $q = 10$, 初始输出 $y_k = 0$. 对于文献 [7], 由于仿真的模型相同, 参数设置保持原文设定不变. 3 种控制方法的比较如图 6 所示. 由图 6 可知: PID 控制比本文方法响应慢, 超调大; 滑模变结构响应也很快, 但超调比本文方法大, 且出现了明显的振荡, 文献 [7] 的方法响应较慢, 且在个别点上超调较大.

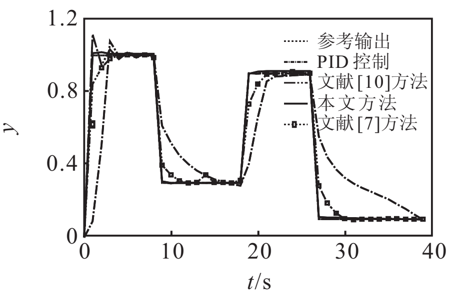

图 5 控制效果比较

6 结 论

针对复杂非线性系统, 本文给出了一种利用 RBF 神经网络构建多步预测模型的方法. 在滚动优化策略上结合 L-M 算法, 通过修正预测值进行反馈校正, 从而实现了完整的非线性系统预测控制. 仿真结果表明, 所设计的方法具有响应速度快、超调量小、稳态误差低的优点. 虽然为了简化, 只分析了 SISO 系统, 但只需对本文方法略做调整 (改变模型的输入输出数) 也适用于多输入多输出 MIMO 系统. 本文的工作为复杂非线性系统的预测控制奠定了基础, 具有一定的理论指导意义和工程应用价值.

另外要特别指出的是: 通过大量仿真实验分析, 发现目标性能函数一般不是控制量的单峰函数, 而 L-M 算法本质上是一个局部优化算法, 因此初始点的选取较为重要. 如何确定初始点以及采用效率高的全局优化算法可能是将来的两个研究方向.

参考文献(References)

[1] 席裕庚. 预测控制[M]. 北京: 国防工业出版社, 1993: 5-9.
(Xi Y G. Predictive control[M]. Beijing: National Defence Industry Press, 1993: 5-9.)

[2] 戴文战, 娄海川, 杨爱萍. 非线性系统神经网络预测控制研究进展[J]. 控制理论与应用, 2009, 26(5): 521-530.
(Dai W Z, Lou H C, Yang A P. An overview of neural network predictive control for nonlinear systems[J]. Control Theory & Applications, 2009, 26(5): 521-530.)

[3] 张日东, 王树青. 基于神经网络的非线性系统多步预测控制[J]. 控制与决策, 2005, 20(3): 332-336.
(Zhang R D, Wang S Q. Neural network based multi-step predictive control for nonlinear systems[J]. Control and Decision, 2005, 20(3): 332-336.)

[4] 张兴会, 陈增强, 袁著祉. 基于神经网络模型的非线性多步预测学习控制器[J]. 控制与决策, 2002, 17(增): 820-822.
(Zhang X H, Chen Z Q, Yuan Z Z. Nonlinear predictive controller based on neural networks[J]. Control and Decision, 2002, 17(S): 820-822.)

[5] Hui P, Jun W, Garba I, et al. Nonlinear system modeling and predictive control using the RBF nets-based quasi-linear ARX model[J]. Control Engineering Practice, 2009, 17(1): 59-66.

[6] Maciej L. Computationally efficient nonlinear predictive control based on RBF neural multi-models[C]. Proc of the 9th Int Conf on Adaptive and Natural Computing Algorithms. Berlin: Springer-Verlag, 2009: 89-98.

[7] 宫赤坤, 闫雪. 基于 RBF 神经网络的预测控制[J]. 上海理工大学学报, 2005, 27(5): 421-424.
(Gong C K, Yan X. Predictive control based on RBF neural network[J]. J of University of Shanghai for Science and Technology, 2005, 27(5): 421-424.)

[8] Alireza R, Scott S. Identification of nonlinear systems using NARMAX model[J]. Nonlinear Analysis, 2009, 71(12): 1198-1202.

[9] Goh B S. Convergence of algorithms in optimization and solutions of nonlinear equations[J]. J of Optimization Theory and Applications, 2010, 144(1): 43-55.

[10] 朱齐丹, 汪瞳. 一种离散时间系统变结构控制的新方法[J]. 控制与决策, 2009, 24(8): 1209-1212.
(Zhu Q D, Wang T. New variable structure control scheme for discrete to time systems[J]. Control and Decision, 2009, 24(8): 1209-1212.)

(责任编辑: 郑晓蕾)