时滞系统的 LM-Smith 神经网络控制器*

吴龙庭[1,2]，曹顺安[3]，胡家元[3]

(1. 长沙理工大学 化学与生物工程学院 电力与交通材料保护湖南省重点实验室，长沙 410004；2. 中南财经政法大学 会计学院，武汉 430073；3. 武汉大学 动力与机械学院，武汉 430072)

摘 要：针对 Smith 控制对时滞系统抗干扰性差的弱点，提出了一种基于神经网络辨识的 LM-Smith 控制器。该控制器在经典 Smith 控制中引入神经元模型，实时辨识时变被控对象，使预估模型能准确跟踪被控对象，实现对时滞环节的完全补偿。仿真结果表明该方法构造简单、准确性好、鲁棒性较强，改善了经典 Smith 控制的控制效果。

关键词：神经网络控制；Smith 控制；时滞系统

中图分类号：TP13 **文献标志码**：A **文章编号**：1001-3695(2014)04-1115-04

doi:10.3969/j.issn.1001-3695.2014.04.038

LM-Smith neural network controller for time-delay system

WU Long-ting[1,2], CAO Shun-an[3], HU Jia-yuan[3]

(1. *Hunan Provincial Key Laboratory of Materials Protection for Electric Power & Transportation, School of Chemistry & Biological Engineering, Changsha University of Science & Technology, Changsha* 410004, *China*; 2. *School of Accounting & Finance, Zhongnan University of Economics & Law, Wuhan* 430073, *China*; 3. *College of Power & Mechanical Engineering, Wuhan University, Wuhan* 430072, *China*)

Abstract: Aiming at Smith controller' weakness of dealing with system disturbance, this paper presented a novel LM-Smith controller based on neural network identification. This algorithm used Levenberg-Marquardt neural network as Smith predictive model. Through online identification, predictive model could follow controlled object real-time, which solved Smith control's inefficiency when control models mismatch. Simulation show that the controlling algorithm get the better of static characters, preferable dynamic quality and strong robustness.

Key words: neural network control; Smith control; time-delay system

在工业过程控制中，大纯时滞现象普遍存在，其产生的原因包括系统传递物料延时和仪器仪表分析样品延时等。时滞现象的存在一方面使得控制器的控制信号不能立刻作用于被控对象，影响系统的快速性和稳定性；另一方面当被控对象发生变化或受到干扰时，闭环控制系统得不到及时的反馈，无法立刻响应，影响系统的动态性能[1]。因此，滞后环节一直被认为是最难控制的动态环节，尤其对大时滞系统(纯滞后时间 τ 与动态时间常数 T 之比大于 0.3)而言，研究时滞系统的控制方法具有重要的现实意义[2]。

1957 年，Smith 提出了一种针对时滞系统的预估控制方案。该方案首先给被控对象建立数学模型(称之为 Smith 预估模型)，然后对系统对象和预估模型同时施加控制信号，分别引出系统和模型各自的反馈信号[3]。如果模型是精确的，则模型输出就是系统输出，模型的反馈信号能及时反映系统响应，不受系统中时滞环节的影响，实现对系统时滞的完全补偿。该控制方案不影响系统的稳定性，只使控制作用在时间上出现一个向后的相移，被称之为 Smith 控制。但在实际工程应用中，由于系统是时变的，同时也不可能给系统建立完全精确的数学模型，因此 Smith 控制的实际控制效果受到影响，在模型偏差较大时，甚至会影响系统的稳定性。

将神经网络控制、模糊免疫控制、内模控制等先进控制策略与 Smith 控制相结合，在一定程度上能克服传统 Smith 控制抗干扰性差的弱点。安连祥等人[4]在传统的模糊 Smith 控制中引入了开关控制和 PID 控制，根据系统偏差的大小决定系统使用何种控制策略，只有当偏差进入模糊控制调节区间时，才使用模糊控制算法调节，提高了系统响应的快速性和鲁棒性；黄越洋等人[5]使用单神经元设计不完全微分 PID 控制器，将该控制器替代传统 Smith 控制中的 PI 控制器，利用 Hebb 学习规则在线整定神经元参数，仿真证明对非线性时滞系统有良好的控制效果；宋申民等人[6]给 Smith 控制加入前馈补偿环节，使用基于粗糙集理论的进化计算在线调整前馈补偿的增益，增强 Smith 控制的鲁棒性。这些方案在一定程度上丰富和发展了 Smith 控制的内容，但同时也使得控制器的设计更加复杂，增加了调试和实现的难度。如何设计简单、鲁棒、稳定的时滞系统控制器，仍是一个重要课题。鉴于此，本文提出一种改进的 LM-Smith 神经网络控制器。

1 LM-Smith 神经网络控制器

常规 Smith 控制器结构如图 1 所示。其中 $G_0(s)e^{-\tau s}$ 是被控对象，$G_c(s)$ 是控制器，$G_m(s)$ 与 $e^{-t_m s}$ 是建立的预估模型。当模型准确时，$G_m(s)=G_0(s)$，$t_m=\tau$，反馈信号为

收稿日期：2013-07-03；**修回日期**：2013-08-16 **基金项目**：长沙理工大学电力与交通材料保护湖南省重点实验室开放基金资助项目(2011CL07)；中南财经政法大学基本科研业务费资助项目(31541111121)；国家社会科学基金资助项目(10CGL010)

作者简介：吴龙庭(1982-)，男，湖北沙市人，讲师，工学博士，主要研究方向为财务软件、决策方法、自动控制(aaron792@163.com)；曹顺安(1962-)，男，安徽怀宁人，教授，工学博士，主要研究方向为水处理及水质控制工程；胡家元(1986-)，男，浙江金华人，工学博士，主要研究方向为水处理技术.

$$H(s)+Y(s)=G_m(s)U(s) \tag{1}$$

相当于直接从 $G_m(s)$ 的输出端引出了反馈信号,绕开了时滞环节 $e^{-t_m s}$ 的影响。此时系统的传递函数为

$$\frac{Y(s)}{R(s)}=\frac{G_C(s)G_0(s)e^{-\tau s}}{1+G_C(s)G_0(s)} \tag{2}$$

特征方程没有时滞项,预估模型实现了对时滞环节的完全补偿。当模型不准确时

$$\frac{Y(s)}{R(s)}=\frac{G_C(s)G_0(s)e^{-\tau s}}{1+G_C(s)G_m(s)+G_C(s)(G_0(s)e^{-\tau s}-G_m(s)e^{-t_m s})} \tag{3}$$

特征方程仍受时滞环节的影响,且模型误差越大,控制性能越差。

为了对模型误差进行补偿,可以将图 1 中控制对象的预估模型看做是两个控制环节 $G_m(s)e^{-t_m s}$ 与 $G_m(s)$ 的并联,给这两个控制环节分别构造一个神经元网络去估计,并使用 Levenberg-Marquardt 算法对神经元网络进行训练,则得到 LM-Smith 神经网络控制器,如图 2 所示。其中 LMNN1(Levenberg-Marquardt neural network 1)是 $G_m(s)e^{-t_m s}$ 的预估模型,LMNN2 是 $G_m(s)$ 的预估模型。在实际控制过程中,通过被控对象和输入输出信号在线训练神经元模型,使其跟踪被控对象,实现对时滞环节的完全补偿,增强 Smith 控制的鲁棒性。

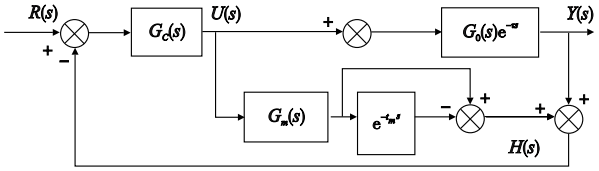

图1 Smith控制结构

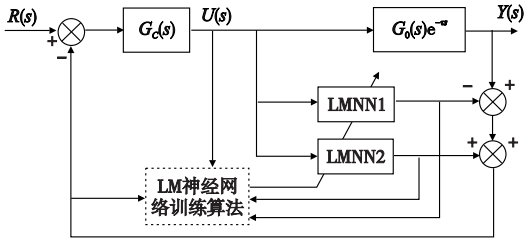

图2 LM-Smith神经网络控制器结构

2 LM 算法描述与分析

人工神经网络的基本结构如图 3 所示,其中圆圈代表神经元,箭头表示信息传导方向,网络包含一个输入层和一个输出层,中间可以有多个隐含层。由于 1989 年,Hornik 等人已经证明只要隐含层包含的神经元足够多,单隐含层的神经网络已经能以任意精度逼近任何多元函数,所以实际应用以单隐含层结构居多,本文的分析也以单隐含层结构为对象,更一般的多层神经网络的分析参见文献[7]。设输入向量为 $\boldsymbol{X}(x_1,x_2,\cdots,x_n)$,输入层节点 x_i 与隐含层节点 net_j 之间的权重为 w_{ji};隐含层节点数为 t,函数为 $f(x)$,节点 net_j 与输出层节点 y_k 之间的权重为 z_{kj};输出层函数为 $g(x)$,输出向量为 $\boldsymbol{Y}(y_1,y_2,\cdots,y_m)$,则隐含层节点的输入 $\text{net}_j=\sum_{i=1}^{n}w_{ji}x_i$,输出 $a_j=f(\text{net}_j)$;输出层节点的输入 $o_k=\sum_{j=1}^{t}v_{kj}a_j$,输出 $y_k=g(o_k)$。使用神经网络逼近多元函数,首先获取多元函数的输入/输出样本。将输入信号输入神经网络,得到网络输出,比较网络输出值与训练样本中期望输出值,得到输出误差,根据输出误差调整网络权重向量;然后重新输入输入信号。重复上述过程,直到网络输出误差可以忽略不计。最常用的神经网络训练方法是 BP 算法[8]。

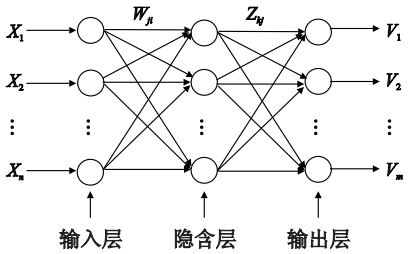

图3 人工神经网络结构

2.1 BP 算法

BP 算法本质上是以网络误差之平方和作为性能指标函数,按梯度法调整权重向量以使指标函数达到最小值的算法[9]。设性能指标函数 $F(x)=E[\sum_{k=1}^{m}e_k^2]=E[\boldsymbol{e}^{\mathrm{T}}\boldsymbol{e}]$,其中 $\boldsymbol{e}$ 是网络输出与期望输出之间的误差向量,e_k 是它的第 k 个分量。将 $F(x)$ 在 x_k 处用一阶泰勒级数展开

$$F(x_{k+1})=F(x_k+\Delta x_k)\approx F(x_k)+g_k^{\mathrm{T}}\Delta x_k \tag{4}$$

令 $\Delta x_k=\alpha p_k$,其中 α 为向量长度,p_k 为方向向量,则欲使 $F(x_{k+1})<F(x_k)$,必须 $g_k^{\mathrm{T}}\Delta x_k=\alpha g_k^{\mathrm{T}}p_k<0$,显然当 $p_k=-g_k$ 时,$F(x_{k+1})$ 减小最快,所以得到 x_k 的迭代式为

$$x_{k+1}=x_k-\alpha g_k \tag{5}$$

其中:α 是学习速率。显然性能指标函数 $F(x)$ 是权重向量 $\boldsymbol{W}$ 和 $\boldsymbol{Z}$ 的函数,应用式(5)得到 $\boldsymbol{W}$ 和 $\boldsymbol{Z}$ 的迭代式为

$$w_{ji}(k+1)=w_{ji}(k)-\alpha\frac{\partial F}{\partial w_{ji}} \tag{6}$$

$$z_{kj}(k+1)=z_{kj}(k)-\alpha\frac{\partial F}{\partial z_{kj}} \tag{7}$$

计算上述迭代式的关键是求出偏导数$\frac{\partial F}{\partial w_{ji}}$和$\frac{\partial F}{\partial z_{kj}}$,根据链式求导法则

$$\frac{\partial F}{\partial o_k}=\frac{\partial F}{\partial y_k}\times\frac{\partial y_k}{\partial o_k} \tag{8}$$

$$\frac{\partial F}{\partial z_{kj}}=\frac{\partial F}{\partial o_k}\times\frac{\partial o_k}{\partial z_{kj}} \tag{9}$$

$$\frac{\partial F}{\partial w_{ji}}=\sum_{k=1}^{m}\frac{\partial F}{\partial o_k}\times\frac{\partial o_k}{\partial a_j}\times\frac{\partial a_j}{\partial \text{net}_j}\times\frac{\partial \text{net}_j}{\partial w_{ji}} \tag{10}$$

根据神经网络的定义求出上述偏导数,然后代入式(6)和(7),即得到权重向量的迭代式。由于计算$\frac{\partial F}{\partial w_{ji}}$,必须用到上一层的$\frac{\partial F}{\partial o_k}$,误差从输出层向隐含层反向传播,所以该算法称之为反传算法。BP 算法结构简单、易于构造、数学形式优美,能为多层神经网络提供可靠的训练方法,因此一经提出便得到广泛重视,成为使用最广的神经网络学习算法[10]。BP 算法也存在不足:a)使用最速下降法,能保证误差不断减小,但减小速度可能过慢;b)使用该算法易陷入局部最小。针对这些缺点,出现了各种改进算法,如共轭梯度法[11]、动量因子法[12]等,其中 Levenberg-Marquardt(LM)算法是近年来使用较广的改进算法。

2.2 LM 算法

LM 算法以牛顿法为基础,使用二阶泰勒级数展开。由于使用更高阶的泰勒级数,所以理论上收敛速度更快,不足之处是不能保证算法一定收敛[13]。将目标性能函数 $F(x)$ 用二阶泰勒级数展开:

$$F(x_{k+1})=F(\boldsymbol{x}_k+\Delta\boldsymbol{x}_k)\approx F(\boldsymbol{x}_k)+g_k^{\mathrm{T}}\Delta\boldsymbol{x}_k+\frac{1}{2}\Delta\boldsymbol{x}_k^{\mathrm{T}}A_k\Delta\boldsymbol{x}_k \tag{11}$$

其中:$\boldsymbol{x}_k$ 是 n 维向量 $\boldsymbol{x}$ 的第 k 次迭代值;g_k、A_k 分别为一阶梯度

$\nabla F(x)$与二阶梯度$\nabla^2 F(x)$的第 k 次迭代值。根据多元函数微分学,$F(x_{k+1})$在偏导数为零处取极小值[14],对式(11)右边取偏导数并令其为零,得到

$$g_k + A_k \Delta x_k = 0 \tag{12}$$

解得

$$x_{k+1} = x_k - A_k^{-1} g_k \tag{13}$$

设 $F(x)$为平方函数之和,即

$$F(x) = \sum_{i=1}^{N} v_i^2(x) = v^{\mathrm{T}}(x) v(x) \tag{14}$$

则一阶梯度第 j 个分量为

$$[\nabla F(x)]_j = \frac{\partial F(x)}{\partial x_j} = 2\sum_{i=1}^{N} v_i(x) \frac{\partial v_i(x)}{\partial x_j} \tag{15}$$

写成矩阵形式

$$\nabla F(x) = 2J^{\mathrm{T}}(x) v(x) \tag{16}$$

其中:$J(x)$为雅可比矩阵。

$$J(x) = \begin{pmatrix} \frac{\partial v_1(x)}{\partial x_1} & \frac{\partial v_1(x)}{\partial x_2} & \cdots & \frac{\partial v_1(x)}{\partial x_n} \\ \frac{\partial v_2(x)}{\partial x_1} & \frac{\partial v_2(x)}{\partial x_2} & \cdots & \frac{\partial v_2(x)}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial v_N(x)}{\partial x_1} & \frac{\partial v_N(x)}{\partial x_2} & \cdots & \frac{\partial v_N(x)}{\partial x_n} \end{pmatrix} \tag{17}$$

接着计算二阶梯度$\nabla^2 F(x)$

$$[\nabla^2 F(x)]_{kj} = \frac{\partial^2 F(x)}{\partial x_k \partial x_j} =$$

$$2\sum_{i=1}^{N}\left[\frac{\partial v_i(x)}{\partial x_k} \times \frac{\partial v_i(x)}{\partial x_j} + v_i(x) \frac{\partial^2 v_i(x)}{\partial x_k \partial x_j}\right] =$$

$$2J^{\mathrm{T}}(x) J(x) + 2S(x) \tag{18}$$

其中:

$$S(x) = \sum_{i=1}^{N} v_i(x) \nabla^2 v_i(x) \tag{19}$$

当 $S(x)$很小时,$\nabla^2 F(x)$可以近似表示为

$$[\nabla^2 F(x)]_{kj} \cong 2J^{\mathrm{T}}(x) J(x) \tag{20}$$

将式(16)(20)代入式(13)得

$$x_{k+1} = x_k - [J^{\mathrm{T}}(x_k) J(x_k)]^{-1} J^{\mathrm{T}}(x_k) v(x_k) \tag{21}$$

由于矩阵 $M = J^{\mathrm{T}}(x_k) J(x_k)$不一定可逆,使用近似矩阵 G 加以改进。

$$G = M + \mu I \tag{22}$$

其中:I 为单位矩阵;$\mu > 0$,当 μ 足够大时,可以保证 G 是正定阵,所以可逆。使用 G 代替 M 代入式(21),则得到 Levenberg-Marquardt 算法

$$x_{k+1} = x_k - [J^{\mathrm{T}}(x_k) J(x_k) + \mu_k I]^{-1} J^{\mathrm{T}}(x_k) v(x_k) \tag{23}$$

可以看到,当 μ_k 极大时,LM 算法还原为最速下降法;随着 μ_k 的减小,算法收敛速度逐步增大。

2.3　LMBP 算法

将 LM 算法应用于 BP 神经网络训练,称之为 LMBP 算法[15]。该算法与 BP 算法类似,也具有误差向后传播的特点。设神经网络训练样本的容量为 Q,输出层节点数为 m,则性能指标函数为

$$F(x) = \sum_{q=1}^{Q} e_q^{\mathrm{T}} e_q = \sum_{q=1}^{Q}\sum_{k=1}^{m} e_{kq}^2 = \sum_{i=1}^{N} v_i^2 \tag{24}$$

其中:e_q 是第 q 个样本的期望输出向量与网络输出向量之差,e_{kq}是 e_q 的第 k 个分量。参数向量为

$$x^{\mathrm{T}} = [w_{11}, \cdots, w_m, z_{11}, \cdots, z_{mt}] \tag{25}$$

则多层网络训练的雅可比矩阵可以写为

$$J(x) = \begin{pmatrix} \frac{\partial e_{11}}{\partial w_{11}} & \frac{\partial e_{11}}{\partial w_{12}} & \cdots & \frac{\partial e_{11}}{\partial z_{mt}} \\ \frac{\partial e_{21}}{\partial w_{11}} & \frac{\partial e_{21}}{\partial w_{12}} & \cdots & \frac{\partial e_{21}}{\partial z_{mt}} \\ \vdots & \vdots & & \vdots \\ \frac{\partial e_{mQ}}{\partial w_{11}} & \frac{\partial e_{mQ}}{\partial w_{12}} & \cdots & \frac{\partial e_{mQ}}{\partial z_{mt}} \end{pmatrix} \tag{26}$$

其中:一般项　$\frac{\partial e_{dq}}{\partial z_{kj}} = \frac{-\partial Y_{dq}}{\partial z_{kj}} = -\frac{\partial g(o_{dq})}{\partial z_{kj}} =$

$$\begin{cases} -g'(o_{dq}) \frac{\partial(\sum_{j=1}^{t} z_{dj} a_{jq})}{\partial z_{kj}} = -g'(o_{dq}) a_{jq} & d = k \\ 0 & d \neq k \end{cases} \tag{27}$$

$$\frac{\partial e_{dq}}{\partial w_{ji}} = \frac{-\partial Y_{dq}}{\partial w_{ji}} = -g'(o_{dq}) \frac{\partial o_{dq}}{\partial w_{ji}} =$$

$$-g'(o_{dq}) \frac{\partial o_{dq}}{\partial a_{jq}} \times \frac{\partial a_{jq}}{\partial \mathrm{net}_{jq}} \times \frac{\partial \mathrm{net}_{jq}}{\partial w_{ji}} = -z_{dj} x_{iq} g'(o_{dq}) f'(\mathrm{net}_{jq}) \tag{28}$$

根据上式计算出雅可比矩阵,然后将式(25)代入式(23)得到权重向量的迭代计算公式。LMBP 算法的迭代过程可以概括如下:

a)确定神经网络结构,设定各项初值,包括各项权重初值、学习速率 μ 和一个小于 1 的正数 θ,准备训练样本。

b)将样本输入网络,计算输出值、误差向量和向量 V。

c)根据式(27)和(28),计算雅可比矩阵式(26)。

d)将式(26)代入式(23),解得 x_{k+1}。

e)用 x_{k+1}重新计算性能指标函数。若函数值减小,返回步骤 b);否则令 $\mu = \mu/\theta$,返回步骤 d)。

当性能指标函数小于给定值时,认为迭代收敛,停止迭代。

3　仿真实验研究

因为大部分工业控制对象都可以用一阶惯性环节表示[16],所以选取被控对象的数学模型 $G_0(s)\mathrm{e}^{-\tau s} = \frac{1}{55s+1}\mathrm{e}^{-122s}$,控制器采用简单 PID 控制,控制方程为

$$G_C(s) = 0.41 + 0.12\frac{1}{s} + 0.69s \tag{29}$$

仿真实验验证 LM-Smith 控制器对时变系统的准确性和可靠性。在进行实际控制仿真之前,首先对控制器中的神经网络进行预训练。设采样时间为 30 s,则 $G_m(s)\mathrm{e}^{-t_m s}$与 $G_m(s)$的离散方程可分别表示为(设预估模型精确)

$$y(k) = 0.5796y(k-1) + 0.399u(k-5) + 0.02146u(k-6) \tag{30}$$

$$y(k) = 0.4724y(k-1) + 0.6332u(k-1) \tag{31}$$

在控制器中,LMNN1 辨识 $G_m(s)\mathrm{e}^{-t_m s}$,LMNN2 辨识 $G_m(s)$。训练神经网络逼近多元函数的方法是先确定神经网络的结构,然后使用训练样本训练神经网络,使其在相同的输入下输出与训练样本一致,最后再用测试样本测试神经网络。若测试误差在可接受范围之内,则表示神经网络训练成功;若误差过大,则重新训练网络或增大训练样本容量后再训练,直到测试误差满意为止。为保证测试结果的真实有效,测试样本不能与训练样本重叠[17]。

LMNN1 与 LMNN2 的预训练结果如图 4 所示。为了充分激励系统,训练样本选取了方波、阶跃、正弦和随机四种信号,可以看出 $G_m(s)\mathrm{e}^{-t_m s}$的响应具有明显的时滞。根据式(30)和(31)以及经验,设定 LMNN1 和 LMNN2 均只包含一个隐含层,隐含层节点数同为 5,激发函数为 $\tan h(x)$。学习速率 μ 越大,网络性能函数下降越快,但过大的学习速率容易导致算法不收

敛,仿真中选择 $\mu=1$,$\theta=0.5$。LMNN1 和 LMNN2 的输入向量分别为 $x=[y(k-1),u(k-5),u(k-6)]'$ 和 $x=[y(k-1),u(k-1)]'$,输出节点数均为 1,激发函数为线性函数。使用 LMBP 算法训练网络,然后使用谐波信号进行测试,可以看到测试结果很理想,这是因为控制对象离散化后都是线性函数,易于用神经网络表示。

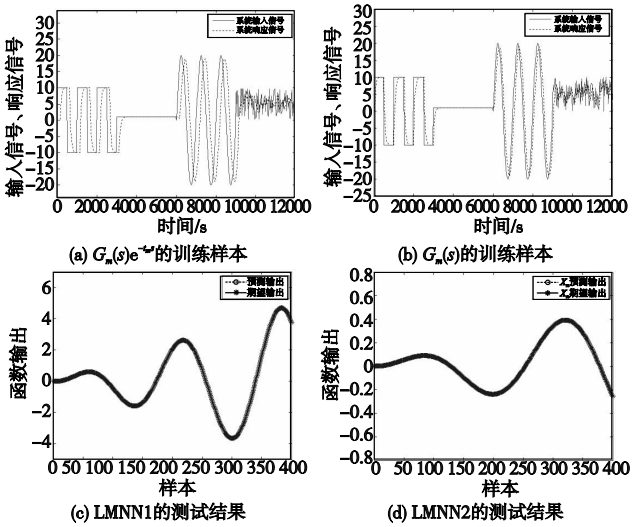

图4 神经网络预训练结果

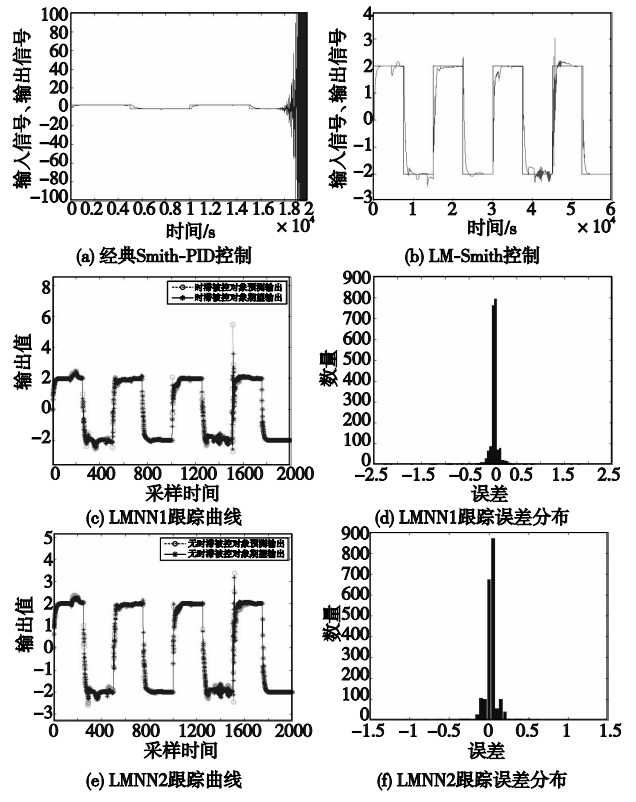

图5 LM-Smith控制器仿真

控制仿真共采样 2 000 次,被控对象 $G_0(s)e^{-\tau s}$ 在第 150 次采样变为 $\frac{0.8}{80s+1}e^{-165s}$;在第 450 次采样变为 $\frac{1}{55s+1}e^{-122s}$;在第 750 次时刻变为 $\frac{1.2}{60s+1}e^{-110s}$;在第 1 050 次采样变为 $\frac{1}{55s+1}e^{-122s}$;在第 1 350 次采样以后变为 $\frac{1.2}{40s+1}e^{-150s}$。每次循环仿真结束之后,使用最新的输入/输出数据更新训练样本,并重新对 LMNN1 和 LMNN2 进行训练。仿真结果如图 5 所示。如果采用经典的 Smith-PID 控制(相同 PID 控制参数),系统最终发散。在控制器中引入 LM 神经网络辨识控制对象,则控制曲线经过震荡后最终收敛,如图 5(b)所示。控制曲线中存在几个小峰,这是由于被控对象发生变化的缘故。图 5(c)~(f)分别是预估模型 LMNN1 和 LMNN2 的跟踪曲线和跟踪误差分布,图中的期望输出是时变被控对象的输出曲线。可以看到无论是 LMNN1 还是 LMNN2,在被控对象变化不大的条件下,均能稳定地跟踪被控对象,保证 Smith 控制对时滞环节的完全补偿。从整个仿真结果来看,LM-Smith 控制器快速性较好,控制精度较高,具有一定的鲁棒性。

4 结束语

针对 Smith 控制对时滞系统抗干扰性差的弱点,本文提出了一种基于神经网络辨识的 LM-Smith 控制器。该控制器在经典 Smith 控制中引入神经元模型,实时辨识时变被控对象,使预估模型能准确跟踪被控对象,实现对时滞环节的完全补偿。仿真结果表明该方法改善了经典 Smith 控制的控制效果,具有构造简单、准确性好、鲁棒性强的优点。

参考文献:

[1] NA Jing, REN Xue-mei, HUANG Hong. Time-delay positive feedback control for nonlinear time-delay systems with neural network compensation[J]. Acta Automatica Sinica, 2008, 34(9): 1197-1204.

[2] 陆平,赵捷,郭鹏. 模糊内模 PID 控制及应用[J]. 自动化仪表,2012,33(3):50-52.

[3] 曹顺安,黄艳,聂鑫. 火电厂锅炉水质调节 IMC-PID 鲁棒自适应控制[J]. 化工自动化及仪表,2008,35(6):30-33.

[4] 安连祥,马华民,刘永刚,等. 基于改进 Smith 预估器的二阶时滞系统[J]. 计算机仿真,2011,28(1):198-200.

[5] 黄越洋,石元博,张茜. 基于神经网络 Smith 预估器的预测控制[J]. 计算机仿真,2009,26(2):187-189.

[6] 宋申民,陈兴林,段广仁. 基于粗糙集理论与进化计算的时滞系统 Smith 控制[J]. 系统仿真学报,2006,18(8):2247-2248.

[7] HAGAN M T, DEMUTH H B, BEALE M H, *et al*. 神经网络设计[M]. 戴葵,译. 北京:机械工业出版社,2002:240-248.

[8] 李艳,许合金. 造纸过程定量水分的小波神经网络 PID 控制[J]. 计算机仿真,2012,29(3):249-253.

[9] 王冬青,王钰,佟河亭,等. 气动人工肌肉手臂的神经网络 Smith 预估控制[J]. 控制工程,2012,19(2): 254-257.

[10] 刘金琨. 先进 PID 控制 MATLAB 仿真[M]. 2 版. 北京:电子工业出版社,2007:301-310.

[11] 徐雪松,欧阳烧. 钢铁炉温不确定时滞系统 Smith 免疫预测控制[J]. 计算机应用,2012,32(10):2956-2959.

[12] 张俊,罗大庸. 网络控制系统的 Smith 预估时滞补偿方法[J]. 哈尔滨理工大学学报,2012,17(5):34-37.

[13] 方彦军,李晓春. 基于仿人分层递阶智能控制的复杂大延时锅炉水磷酸盐处理系统研究[J]. 信息与控制,2002,31(5):477-480.

[14] 吕鹏刚,何承波,刘开培. 基于亚当模块的火电厂自动加药系统[J]. 微计算机信息,2001,17(11): 55-56.

[15] 杨蒲,郭立冬. 连城电厂锅炉炉水磷酸盐协调处理解耦控制系统[J]. 冶金自动化,2004,2(1):40-42.

[16] 张俊,罗大庸. 网络控制系统的 Smith 预估时滞补偿方法[J]. 哈尔滨理工大学学报,2012,17(5):34-37.

[17] 杨智,杨照华,魏列江. 电厂锅炉给水自动加药 PLC 控制系统设计与实现[J]. 自动化仪表,2001,22(8):47-49.