doi: 10.3969/j.issn.0490-6756.2017.05.014

基于自适应动量因子的区间神经网络建模方法

陈　实[1], 易　军[1], 李　倩[2], 黄　迪[1], 李太福[1]
(1. 重庆科技学院电气与信息工程学院, 重庆 401331; 2. 重庆大学数学与统计学院, 重庆 401331)

摘　要: 区间神经网络建模是区间控制的核心部分,也是提高系统鲁棒性的重要方法.针对区间神经网络算法收敛速度慢的问题,提出一种自适应动量因子算法.算法利用区间运算建立输入与输出数据的映射模型,通过引入具有自适应特性的动量项,使用最速下降法对动量项进行自适应更新,在加快系统收敛速度的同时,克服系统稳态误差大和容易陷入局部最小值的弊端.典型算例实验表明:区间神经网络能够较为精确地建立区间网络模型,自适应动量因子算法提高了区间神经网络整体性能.

关键词: 区间控制; 区间神经网络; 动量因子; 自适应
中图分类号: TP183　　**文献标识码**: A　　**文章编号**: 0490-6756(2017)05-0978-07

Interval neural network modeling method based on adaptive momentum factor

CHEN Shi[1], *YI Jun*[1], *LI Qian*[2], *HUANG Di*[1], *LI Tai-Fu*[1]
(1. School of Electrical and Information Engineering, Chongqing University of Science and Technology, Chongqing 4011331, China;
2. School of Mathematics and Statistics, Chongqing University, Chongqing 401331, China)

Abstract: The modeling of interval neural network is not only a component of interval control, but also an important role to improve the robust of systems. An adaptive algorithm of momentum factor is proposed to solve the problem of slow convergence speed on the interval neural network. In this paper, interval calculation method is used to establish the mapping model of input and output variables. By introducing a momentum term with adaptive characteristics, the steepest descent algorithm is applied to update the adaptive momentum factor. Compared with the traditional method, this method not only accelerates the convergence speed, but also overcome the disadvantages of the system steady state error and easily to fall into local minimum. According to the nonlinear experiments, interval neural networks are able to establish the zone models, and the algorithm of adaptive momentum factor increase the overall performance of the network. Classic bench mark experiments show that our work can more accurate to establish interval network model, while introducing of adaptive momentum factor algorithm also can improve the overall performance of the interval neural network.

Keywords: Interval control; Interval neural network; Momentum factor; Modeling

收稿日期: 2016-07-07
基金项目: 国家自然科学基金(51375520, 51374268, 51404051, 61174015); 重庆市重大应用技术开发项目(cstc2013yykfC0034); 重庆市优秀人才科技训练计划(cstc2013kjrc-qnrc40008); 重庆市高校创新团队项目(KJTD201324); 重庆市高校优秀成果转化项目(KJZH14218); 重庆市基础科学与前沿技术研究 (cstc2015jcyjBX0099);重庆科技学院研究生科技创新计划项目基金(YKJCX1620402)
作者简介: 陈实(1992－), 男, 硕士生, 研究方向为神经网络、过程控制等. E-mail: cqustcs@sina.com

1 引　言

工业生产中的各种数据往往是不确定的，受传感器精度和环境的影响，测量值总会在一个区间范围内波动，若控制器随着这些波动频繁动作，系统的稳定性就难以得到保障[1,2]. 区间控制是一种允许被控变量小范围波动的控制方法[3,4]，只有当系统波动超出范围后才做出相应动作. 这种忽略系统输出微小变化的控制方法不仅克服了系统所存在的模型误差及严重耦合的问题，也大大提高了系统的稳定性.

在区间控制的建模过程中，如何利用合理的建模方法建立一个客观可靠的系统模型是整个过程的难点所在[5-8]. 文献[9]提出采用不同的缓冲因子算法减小干扰对模型的影响，文献[10]提出采用椭球定界算法对系统误差进行补偿，文献[11]提出采用弹性神经网络结构来自适应设置神经元的个数，但以上方法都是用量化的单一实数表示一个区间数据，其模型无法反映系统真实状况.

区间神经网络是输入、输出变量均为区间数据的神经网络. 该方法建模的最大优势是所有输入、输出变量都是真实的区间数据，避免用单一实数取代不确定值，确保了模型的客观性. 因此，方法在很多领域已得到广泛应用[12-15]. 但由于区间算法的引入，模型的非线性程度大大提高，使得区间神经网络存在系统收敛速度较慢等弊端.

针对上述问题，本文提出一种区间神经网络自适应动量因子算法（Interval Neural Network Based on Adaptive Momentum Factor，INN-AMF）. 算法首先将权值、阈值的反馈更新中引入附加动量，然后用自适应动量因子替代固定值动量因子，最后结合区间运算规则完成模型的建立. 仿真结果表明，该算法相对于同类算法有以下两个优点：(1) 循环更新初期，较大的动量因子使系统收敛速度大幅提高；(2) 循环更新末期，较小的动量因子可以减小系统稳态误差.

2 区间神经网络模型

2.1 区间运算规则

Moore 在文献[16]中提出区间数学理论，阐述了区间运算的主要规则如下.

定义 1　假设一个闭区间 X 的左右端点分别为 $\underline{x}$，$\overline{x}$，则区间 X 可以表示为 $X=[\underline{x},\overline{x}]$，其中 $\underline{x}$，$\overline{x}$ 分别称作区间下限和区间上限.

定义 2　区间中点 x^c 为区间上限和区间下限的算数平均值，即

$$x^c=\frac{1}{2}(\underline{x}+\overline{x}) \tag{1}$$

定义 3　区间半径 x^r 为区间中点到区间两端的距离，即

$$x^r=\frac{1}{2}(\overline{x}-\underline{x}) \tag{2}$$

相对与实数的运算规则，区间函数的运算包含了区间上限和区间下限两部分，根据区间运算法则有如下性质[8]：

性质 1　定义。为四则运算当中的任意一种，则

$$X\circ Y=\{x\circ y\,|\,x\in X,y\in Y\} \tag{3}$$

性质 2　若 $f(x)$ 为单调递增函数，则

$$f(X)=[f(\underline{x}),f(\overline{x})] \tag{4}$$

若 $f(x)$ 为单调递增函数，则

$$f(X)=[f(\overline{x}),f(\underline{x})] \tag{5}$$

以上性质也可以用区间中点和区间半径的形式表示，两种表示方法完全等价.

2.2 区间神经网络拓扑结构

区间神经网络是一种多层前馈神经网络，其特点是信号前向传递，误差反向传播. 在前向传递过程中，输入信号从输入层经隐含层逐层计算，直至输出层. 如果网络输出不能达到预期目标，则根据预测误差反向调整网络的权值和阈值. 单隐层区间神经网络的拓扑结构如图 1 所示.

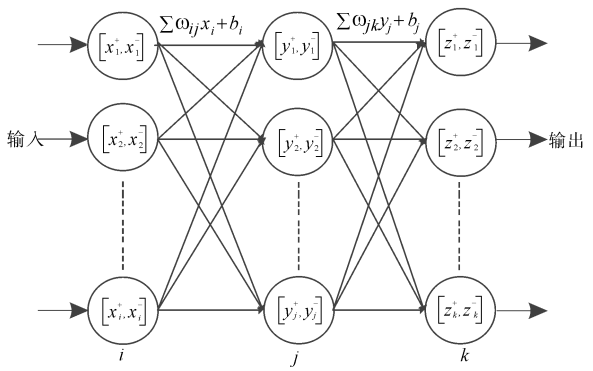


图 1　区间神经网络拓扑结构

Fig. 1　Topological structure of interval neural network

设网络输入为 $X=(X_1,X_2,X_3,\cdots,X_i)$，其中 $X_i=(\underline{x_i},\overline{x_i})$，任意一组 X 对应的输出为 $Z=(z_1,z_2,\cdots,z_k)$，其中 $Z_k=[\underline{z_k},\overline{z_k}]$. 网络的输入、输出神经元的个数由输入变量个数和输出变量个数决定. 隐含层层数和节点数根据实际数据确定. 连接各层的权值 ω 和阈值 b 均为实数.

2.3 区间神经网络前向传递

步骤 1 网络初始化. 根据网络的输入、输出序列,确定输入节点数 i、隐层节点数 j 和输出节点数 k,初始化输入层、隐含层和输出层神经元之间的连接权值 ω_{ij} 和 ω_{jk},初始化隐含层阈值 b_j 和输出层阈值 b_k,给定学习率 η 和误差权重因子 β,选取神经元激励函数,设定循环次数上限和理想误差.

步骤 2 隐含层输入变量计算. 记输入为变量为 $X=<X^c,X^r>$,包含了 X 的中点和半径两部分信息,则隐含层输入的计算公式更新为

$$\begin{cases} y_j^c = \sum \omega_{ij} x_i^c + b_j \\ y_j^r = \sum \omega_{ij} x_i^r + b_j \end{cases} \tag{6}$$

其中,ω_{ij},b_j 分别表示连接 i,j 层的权值和阈值,y_j^c,y_j^r 分别表示隐藏层节点的中点和半径.

步骤 3 隐含层输出变量计算. 为改善模型的线性程度,神经网络在隐含层输入到隐含层输出的计算中,往往会引入激励函数. 以 S 型激励函数为例,如果将 y_j^c 和 y_j^r 直接代入运算,得到的隐含层输出没有实际的数学意义,所以在这一步的计算当中,将隐含层的输入改写为区间上限和区间下限的形式,即 $y_j=[\underline{y_j},\overline{y_j}]$. 由于 S 型函数是一个单调递增的函数,所以将 y_j 的上、下限分别代入计算后,并不会改变区间原有的性质,其计算公式为

$$\begin{cases} \underline{h_j} = f(\underline{y_j}) \\ \overline{h_j} = f(\overline{y_j}) \end{cases} \tag{7}$$

其中,$f(x)$ 是激励函数,$\underline{h_j}$ 和 $\overline{h_j}$ 是隐含层输出.

步骤 4 输出层输出变量计算. 完成隐含层的计算后,需将隐层输出重新转换为中点和半径的表达形式,即 h_j^c 和 h_j^r,则输出层输出的计算公式为:

$$\begin{cases} z_k^c = \sum \omega_{jk} h_j^c + b_k \\ z_k^r = \sum \omega_{jk} h_j^r + b_k \end{cases} \tag{8}$$

其中,z_k^c,z_k^r 分别表示输出层输出的中点和半径,ω_{jk} 表示连接隐含层和输出层之间的权值,b_k 表示输出层阈值.

步骤 5 计算误差. 根据预测输出 z_k 和期望输出 z 的差值,定义二者的欧式距离为

$$d(z,z_k)=\beta(z^c-z_k^c)^2+(1-\beta)(z^r-z_k^r)^2 \tag{9}$$

其中,误差权重 $\beta\in[0,1]$,表示距离函数中两者中点距离和半径距离的重要性程度. 为方便计算,定义误差函数为

$$e=-\frac{1}{2}d(z,z_k)$$

$$=-\frac{1}{2}[\beta(z^c-z_k^c)^2+(1-\beta)(z^r-z_k^r)^2] \tag{10}$$

误差函数充分考虑到在不同函数中中点距离和半径距离对误差的影响程度不同. 当 β 取 0 时,只考虑半径距离对误差的影响;当 β 取 1 时,只考虑中点距离对误差的影响.

2.4 区间神经网络反向更新

步骤 1 权值更新. 根据网络预测误差 e 更新网络连接权值 ω_{ij},ω_{jk}. 误差函数对于 ω_{jk},ω_{ij} 的梯度分别为

$$\begin{cases} \Delta\omega_{ij} = \dfrac{\partial e}{\partial \omega_{ij}} \\ \Delta\omega_{jk} = \dfrac{\partial e}{\partial \omega_{jk}} \end{cases} \tag{11}$$

将(6)~(11)代入,化简可得

$$\begin{cases} \Delta\omega_{ij} = \dfrac{1}{2}\omega_{jk}[\beta(z^c-z_k^c)-(1-\beta)(z^r-z_k^r)] \\ \quad [h_{\min}(1-h_{\min})(x^c-x^r)+ \\ \quad h_{\max}(1-h_{\max})(x^c+x^r)] \\ \Delta\omega_{jk}=\beta(z^c-z_k^c)h^c+(1-\beta)(z^r-z_k^r)h^r \end{cases} \tag{12}$$

根据式(12),网络连接权值 ω_{ij},ω_{jk} 的更新方法为

$$\begin{cases} \omega_{jk}=\omega_{jk}+\eta\Delta\omega_{jk} \\ \omega_{ij}=\omega_{ij}+\eta\Delta\omega_{ij} \end{cases} \tag{13}$$

步骤 2 阈值更新. 阈值更新方式和权值更新一致,都是利用梯度下降的方法. 误差函数对于 b_k,b_j 的梯度为

$$\begin{cases} \Delta\omega_{ij} = \dfrac{\partial e}{\partial \omega_{ij}} \\ \Delta\omega_{jk} = \dfrac{\partial e}{\partial \omega_{jk}} \end{cases} \tag{14}$$

将式(6)~ 式(11)代入,化简可得

$$\begin{cases} \Delta b_j=\omega_{jk}h_{\max}(1-h_{\max})[\beta(z^c-z_k^c)+ \\ \quad (1-\beta)(z^r-z_k^r)] \\ \Delta b_k=\beta(z^c-z_k^c)+(1-\beta)(z^r-z_k^r) \end{cases} \tag{15}$$

根据式(15),网络节点阈值 b_k,b_j 的更新方法为

$$\begin{cases} b_k=b_k+\eta\Delta b_k \\ b_j=b_j+\eta\Delta b_j \end{cases} \tag{16}$$

步骤 3 判断迭代是否结束. 若结束,则输出区间神经网络预测值,反之返回 2.3 中的步骤 2,直至网络输出达到预期要求或循环次数超过设定限.

3　自适应附加动量算法

3.1　固定动量因子的误差更新

区间神经网络采用最速下降法作为权值和阈值的更新方法，该算法是从网络预测误差的负梯度方向修正权值和阈值，其在数值上只与上个时刻的数值和当前时刻输出的误差有关，而忽略了在循环更新当中积累的经验，导致学习过程收敛速度慢，容易陷入局部最小值的问题.

为改善该问题，文献[17]提出一种结合动量项的误差更新方法，以权值更新为例，带动量项的权值更新公式如下.

$$\omega(k+1)=\omega(k)+\eta\Delta\omega(k)+a[\omega(k-1)-\omega(k-2)] \tag{17}$$

其中，$a[\omega(k-1)-\omega(k-2)]$ 表示动量项，$\omega(k)$，$\omega(k-1)$，$\omega(k-2)$ 表示 k，$k-1$，$k-2$ 时刻的权值，a 为动量因子，通常取 [0,1) 之间的实数. 当循环次数较少时，权值更新幅度较大，动量项 $a[\omega(k-1)-\omega(k-2)]$ 的值较大，权值修正量相应增加，从而大幅提升系统的收敛速度；当循环次数较多，权值更新幅度较小，动量项 $a[\omega(k-1)-\omega(k-2)]$ 的值较小，权值修正量相应减小，避免系统振荡或陷入局部最小值. 图 2 为一次区间神经网络建模实验的误差收敛曲线图，当动量因子 a 取不同数值时，收敛曲线如图 2 所示.

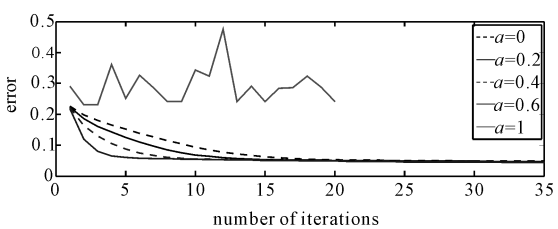


图 2　不同动量因子误差收敛曲线图

Fig. 2　Convergence curves of different momentum factors

其中，$a=0$ 表示不加入动量因子的原始算法. 然而，从图中结果不难看出，加入动量项后的稳态误差也相应增大，且动量因子越大，系统稳态性能越差，当动量因子取值过大时，系统误差不收敛.

3.2　梯度动量因子算法

为解决固定动量因子稳态性能较差的问题，文献[18]提出一种梯度动量因子算法. 该算法将每次循环的系统误差因素纳入到动量因子的更新当中，使系统在循环初期利用较大的动量因子加快系统收敛，而在循环后期利用较小的动量因子改善系统稳态性能. 根据式(13)，梯度动量因子算法可更新为：

$$\omega(k+1)=\omega(k)+\eta\Delta\omega(k)+b(k)[\omega(k-1)-\omega(k-2)] \tag{18}$$

其中，$b(k)=\exp(-\lambda-\|\Delta\omega\|)$；$\lambda$ 为正常数，用于控制动量因子的大小. 动量因子始终介于 0 到 1 之间，且不同网络层的动量因子不同，有利加快网络的训练速度.

但是，梯度动量因子算法在更新过程中只考虑了误差对动量项的影响，并没有结合动量项在循环过程中自身累积的经验，这导致了动量因子的大小过分依赖于误差值，使得系统的收敛速度和稳态性能始终无法达到最优.

3.3　自适应动量因子算法

为进一步提升系统收敛速度并减小稳态误差，本节充分考虑误差梯度和历史动量因子的双重影响，提出一种自适应动量因子算法. 根据式(13)和式(17)，自适应动量因子的算法可更新为

$$\omega(k+1)=\omega(k)+\eta\Delta\omega(k)+a(k)[\omega(k-1)-\omega(k-2)] \tag{19}$$

对于式(19)中自适应动量项 $a(k)$ 提出采用最速下降法进行自适应更新，即沿误差函数的负梯度方向对数值进行在线调整. 以权值更新为例，具体方法如下.

$$a(k)=a(k-1)-\mu\frac{\partial e(k)}{\partial\omega(k)}\frac{\partial\omega(k)}{\partial a(k)} \tag{20}$$

其中，μ 为常数；$e(k)$ 为式(10)中的误差函数. 由式(19)可知：

$$\frac{\partial\omega(k)}{\partial a(k)}=\omega(k-1)-\omega(k-2) \tag{21}$$

将自适应动量因子算法应用到所有的权值、阈值当中，可得到 ω_{ij}，ω_{jk}，b_j，b_k 所对应的 $a_1(k)$，$a_2(k)$，$a_3(k)$，$a_4(k)$ 的表达式分别为

$$a_1(k)=a+\frac{1}{2}\mu\omega_{jk}[\beta(z^c-z_k^c)-(1-\beta)(z^r-z_k^r)][h_{\min}(1-h_{\min})(x^c-x^r)+h_{\max}(1-h_{\max})(x^c+x^r)](a_1(k-1)-a_1(k-2)) \tag{22}$$

$$a_2(k)=a+\mu[\beta(z^c-z_k^c)h^c+(1-\beta)(z^r-z_k^r)h^r](a_2(k-1)-a_2(k-2)) \tag{23}$$

$$a_3(k)=a+\mu\omega_{jk}h_{\max}(1-h_{\max})[\beta(z^c-z_k^c)+(1-\beta)(z^r-z_k^r)](a_3(k-1)-a_3(k-2)) \tag{24}$$

$$a_4(k)=a+\mu\beta(z^c-z_k^c)+(1-\beta)(z^r-z_k^r)(a_4(k-1)-a_4(k-2)) \tag{25}$$

将式(22)～(25)中的 $a_1(k)\sim a_4(k)$ 分别替换

式(19)中的 $a(k)$ 可得到完整的权值、阈值更新公式. 循环前期($z^c - z_k^c$)和($z^r - z_k^r$)较大,故动量因子较大;循 环后期($z^c - z_k^c$)和($z^r - z_k^r$)较小,故动量因子较小,与前面结论一致.

3.4 算法收敛性分析

我们设误差函数序列为$\{e(\omega(k))\}_{k=1}^{\infty}$,根据式(10)中误差函数的定义,对任意时刻的 $e(\omega(k))$ 均有 $e(\omega(k)) \leqslant 0$. 由文献[19]可知,当固定动量因子取 $[0,1)$ 范围时算法收敛,故令自适应动量因子 $a(k) \in [0,1), k = 1,2,\cdots,\eta$ 为 $(0,1)$ 的固定值. 根据以上条件,联立式(10)和式(19),误差函数序列$\{e(\omega(k))\}_{k=1}^{\infty}$在 $\omega(k)$ 处的 Taylor 展开式为

$$\begin{aligned} e(\omega(k)) &= e(\omega(k)) + \\ & e_{\omega}(\omega(k))(\omega(k+1) - \omega(k)) = e(\omega(k)) + \\ & e_{\omega}(\omega(k))(\eta \Delta\omega(k) + a(k)\Delta\omega(k-1)) = \\ & e(\omega(k)) + e_{\omega}(\omega(k))\Delta\omega(k) + \\ & a(k) e_{\omega}(\omega(k))\omega(k-1) = \\ & e(\omega(k)) + \eta \| e_{\omega}(\omega(k)) \|^2 + \\ & a(k)\Delta\omega(k)\Delta\omega(k-1) \end{aligned} \tag{26}$$

其中,$\| e_{\omega}(\omega(k)) \|^2 = \langle e_{\omega}(\omega(k)), e_{\omega}(\omega(k)) \rangle$;$\langle \cdot \rangle$表示内积过程;$e_{\omega}(\omega(k)) = \dfrac{\partial e}{\partial \omega}\Big|_{\omega=\omega(k)}$.

由式(26)可作以下两点分析:

1) 当 $\Delta\omega(k)$ 在连续两次迭代中符号相同时,$\eta \| e_{\omega}(\omega(k)) \|^2$ 与 $a(k)\Delta\omega(k)\Delta\omega(k-1)$ 均为正实数,即 $e(\omega(k+1)) \leqslant e(\omega(k))$,又 $e(\omega(k)) < 0$,所以 $e(\omega(k))$ 单调递减,且 $\omega(k+1)$ 能在动量项的影响下大幅调整.

2) 当 $\Delta\omega(k)$ 在连续两次迭代中符号相反时,说明在迭代过程中系统振荡,指数加权和 $e(\omega(k))$ 减小,$\omega(k+1)$ 小幅调整,自适应动量项可以起到减小振荡的作用.

综上所述,当 $\Delta\omega(k)$ 满足上述分析(1)中条件时,误差函数序列$\{e(\omega(k))\}_{k=1}^{\infty}$单调递增,且由式(10)可知 $e(\omega(k)) \leqslant 0$,故 $e(\omega^k)$收敛.

4 实验与结果分析

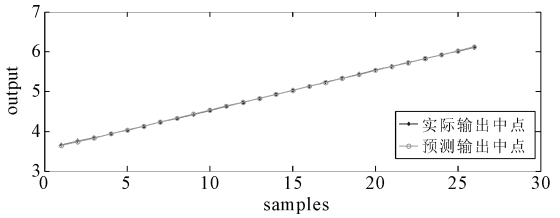
实验采用 1 个线性和 3 个非线性典型算例,每一组算例分别使用区间神经网络算法(Interval Neural Network,INN)[12]、固定动量因子的区间神经网络算法(INN-MF)[17]、梯度动量因子的区间神经网络算法(INN-GMF)[18]和自适应动量因子的区间神经网络算法(INN-AMF)进行性能对比. 所有算例的初始值设定如表 1 所示.

表 1 不同算例初始值设定

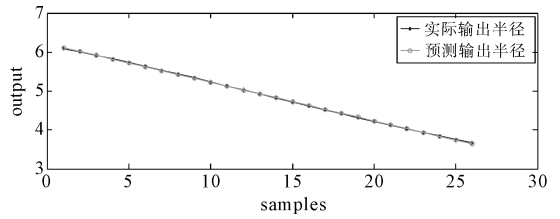
Tab. 1 Initial value of different samples

算例	a	η	取值范围	β	循环次数
$y = 3x + 1$	0.5	0.5	$x^c = 1:0.5:12$	0.5	1000
$y = \sin(x)^2 - \cos(x)^3$	5	0.5	[0,1]之间随机数	0.5	1000
$y = \dfrac{x^2}{e^x} - 1$	0.8	1.8	[0,1]之间随机数	0.5	500
$y = x^2 + x^3$	0.3	0.9	[0,1]之间随机数	0.5	500

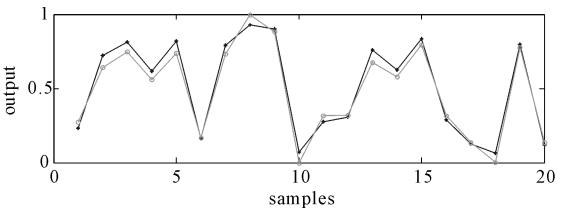
利用区间神经网络的预测跟踪图像如图 3(a)~图 3(h)所示.



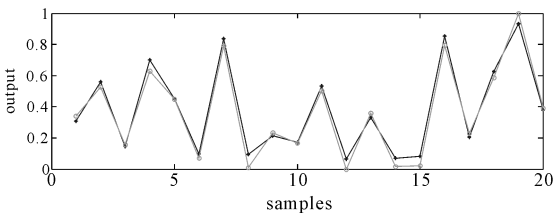
(a)



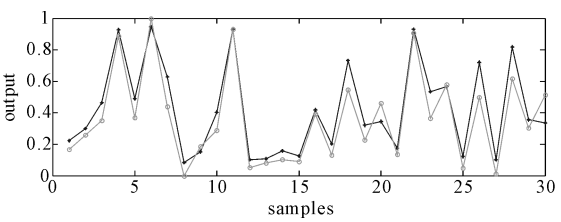
(b)



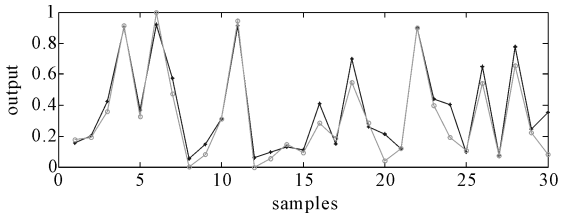
(c)



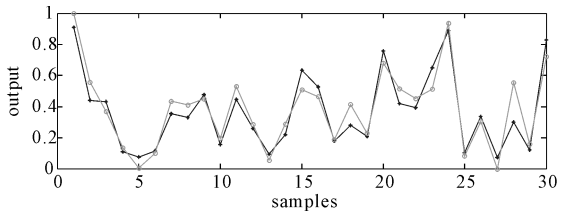
(d)



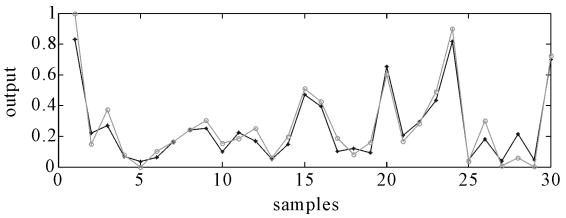
(e)



(f)



(g)



(h)

图 3　不同算例结果跟踪图像

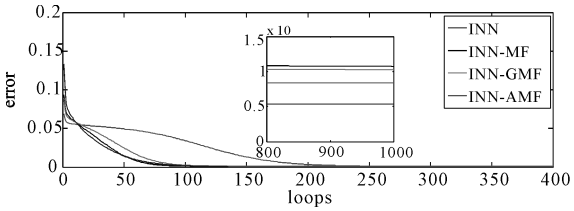
(a) 算例 1 中点跟踪；(b) 算例 1 半径跟踪；(c) 算例 2 中点跟踪；(d) 算例 2 半径跟踪；(e) 算例 3 中点跟踪；(f) 算例 3 半径跟踪；(g) 算例 4 中点跟踪；(h) 算例 4 半径跟踪

Fig. 3 Results tracking of different samples

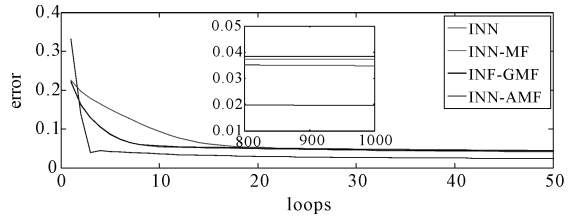
(a) Midpoint tracking of sample 1; (b) Radius tracking of sample 1; (c) Midpoint tracking of sample 2; (d) Radius tracking of sample 2; (e) Midpoint tracking of sample 3; (f) Radius tracking of sample 3; (g) Midpoint tracking of sample 4; (h) Radius tracking of sample 4

其中，第 1 组实验采用连续输入值，即输入变量的中点和半径均为等距的连续数据. 为验证区间神经网络模型对随机数据的适用性，第 2～4 组实验采用随机离散输入值，即输入变量的中点和半径都是 [0,1] 间的随机数据. 由于区间网络算法复杂度高于其他神经网络，所以从图 3(a)～图 3(h)不难看出，线性算例的建模精度明显优于非线性算例. 通过 100 次蒙特卡洛实验显示 4 个算例的所消耗的 CPU 平均时间分别为 3.4883 s，6.1912 s，6.4327 s，4.5223 s.

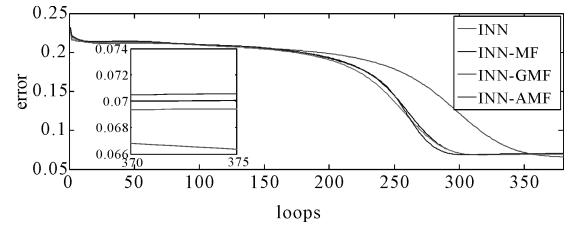
在面向实际的应用时，建模时间是一个重要的技术指标. 图 4(a)～4(d)给出 INN、INN-MF、INN-GMF、INN-AMF 算法在 4 种算例中的收敛曲线.



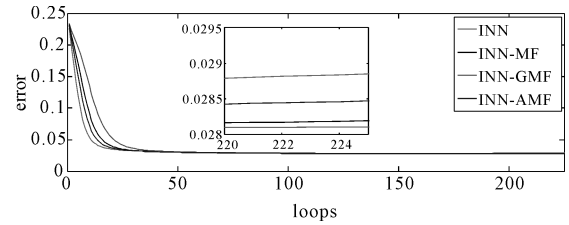
(a)



(b)



(c)



(d)

图 4　不同算法在 4 种算例中收敛曲线

(a) 算例 1 收敛曲线；(b) 算例 2 收敛曲线；(c) 算例 3 收敛曲线；(d) 算例 4 收敛曲线

Fig. 4　Convergence curves of different algorithm in 4 samples

(a) Convergence curve in sample 1; (b) Convergence curve in sample 2; (c) Convergence curve in sample 3; (d) Convergence Curve in Sample 4

实验结果表明，INN-AMF 算法的收敛速度相比 INN 有明显提高，而相比于 INN-MF、INN-GMF 算法稳态性能有较高的提升. 以实验算例 1 为例，INN、INN-MF、INN-GMF、INN-AMF4 种算法在平稳环境下的各种设置相同，达到设定误差值所消耗的 CPU 时间分别 3.4883 s，2.5680 s，2.4001 s，2.3172 s，且 INN-AMF 算法的稳态性能优于没有动量项 INN 算法. 对此可做如下分析：(1) INN-AMF 算法融入了动量项，积累了在循环更新过程中的经验，使得收敛速度大幅度提升；(2) INN-AMF 算法采用了自适应动量因子，根据最速梯度下降的原理，使得动量因子在循环后期接近于

0,因此减小了因引入动量项而带来的稳态误差.

5 结 论

为提高工业建模的客观性、鲁棒性,减小系统波动,本文提出一种自适应动量因子区间神经网络算法.该算法利用最速梯度下降的方法对动量因子进行自适应更新,给出一种有效加速系统收敛的可行方案.仿真实例表明,INN-AMF 算法可以大幅提升系统的收敛的速度,且稳态误差接近甚至优于 INN 算法.文中建立的模型和方法也可以推广到其它线性或非线性系统中,为利用神经网络的建立工业模型提供了一种新的方法.

参考文献:

[1] 张淑宁,王福利,尤富强,等. 基于鲁棒学习的最小二乘支持向量机及其应用[J]. 控制与决策,2010,25:1169.

[2] 柴天佑,丁进良,王宏,等.复杂工业过程运行的混合智能优化控制方法[J]. 自动化学报,2008,34:505.

[3] 孙超,周湛鹏,郝晓辰,等. 基于区间特性和变量软约束的模型预测控制算法[J]. 控制与决策,2015(10):1879.

[4] 邹涛,王丁丁,潘昊,等. 从区间模型预测控制到双层结构模型预测控制[J]. 化工学报,2013,64:4474.

[5] Yi J, Huang D, Fu S, *et al*. Multi-objective bacterial foraging optimization algorithm based on parallel cell entropy for aluminum electrolysis production process[J]. IEEE Trans Ind Electron, 2015, 63:1.

[6] Yi J, Huang D, Fu S, *et al*. Optimized relative transformation matrix using bacterial foraging algorithm for process fault detection[J]. IEEE Trans Ind Electr, 2016, 63:1.

[7] 付涛,弓清忠,王大镇,等. 基于 ANN 模型和 HPSO 算法的数控机床可靠性分布模型研究[J]. 四川大学学报:自然科学版,2015,52:262.

[8] 李静,方正. 不确定分数阶非线性多智能体系统的鲁棒一致性[J]. 四川大学学报:自然科学版,2016,53:67.

[9] 韩红桂,乔俊飞,薄迎春. 基于信息强度的 RBF 神经网络结构设计研究[J]. 自动化学报,2012,38:1083.

[10] 戴文战,苏勇. 缓冲算子调节度与光滑度的关系[J]. 控制与决策,2014,29:158.

[11] 王魏,邓长辉,赵立杰. 椭球定界算法在混合建模中的应用研究[J]. 自动化学报,2014,40:1875.

[12] Li X, Shen J. LMI approach for stationary oscillation of interval neural networks with discrete and distributed time-varying delays under impulsive perturbations.[J]. IEEE Trans Neural Netw, 2010, 21:1555.

[13] Ak R, Vitelli V, Zio E. An interval-valued neural network approach for uncertainty quantification in short-term wind speed prediction[J]. IEEE Trans Neural Netw Learn Syst, 2015, 26:2787.

[14] Khosravi A, Nahavandi S, Creighton D, *et al*. Lower upper bound estimation method for construction of neural network-based prediction intervals.[J]. IEEE Trans Neural Netw, 2011, 22:337.

[15] Quan H, Srinivasan D, Khosravi A. Incorporating wind power forecast uncertainties into stochastic unit commitment using neural network-based prediction intervals[J]. IEEE Trans Neural Netw Learn Syst, 2014, 26:2123.

[16] Moore R E. Interval analysis [M]. Prentice-Hall: Englewood Cliffs, 1966.

[17] 田学民,王强,邓晓刚. 一种引入动量项的小波神经网络软测量建模方法[J]. 化工学报,2011,62:2238.

[18] 何林帮,赵建虎,张红梅,等. 基于 MB 栅格回波强度和改进 BPNN 的底质分类[J]. 中国矿业大学学报,2014,43:956.

[19] 欧世峰,高颖,赵晓晖. 基于随机梯度的变动量因子自适应白化算法[J]. 自动化学报,2012,38:1370.