文章编号: 1001-0920(2017)07-1247-06 DOI: 10.13195/j.kzyjc.2016.0887

基于LM算法的在线自适应RBF网结构优化算法

张昭昭[1†], 乔俊飞[2], 余 文[3]

(1. 辽宁工程技术大学 电子与信息工程学院,辽宁 葫芦岛, 125105; 2. 北京工业大学 电子信息与控制工程学院, 北京 100124; 3. 墨西哥国立理工大学 高级研究中心自控中心, 墨西哥城 07360)

摘 要: 针对LM算法不能在线训练RBF网络以及RBF网络结构设计算法中存在的问题,提出一种基于LM算法的在线自适应RBF网络结构优化算法. 该算法引入滑动窗口和在线优化网络结构的思想,滑动窗口的引入既使得LM算法能够在线训练RBF网络,又使得网络对学习参数的变化具有更好的鲁棒性,并且易于收敛. 在线优化网络结构使得网络在学习过程中能够根据训练样本的训练误差和隐节点的相关信息,在线自适应调整网络结构,跟踪非线性时变系统的变化,使网络维持最为紧凑的结构,以保证网络的泛化性能. 最后通过仿真实验验证了所提出算法的性能.

关键词: LM算法; RBF网络; 在线自适应; 滑动窗口; 泛化性能; 时变系统

中图分类号: TP183 **文献标志码:** A

Online self-adaptive optimal algorithm for RBF network based on Levenberg-Marquardt algorithm

ZHANG Zhao-zhao[1†], *QIAO Jun-fei*[2], *YU Wen*[3]

(1. Institute of Electronic and Information Engineering, Liaoning Technical University, Huludao 125105, China; 2. College of Electronic and Control Engineering, Beijing University of Technology, Beijing 100124, China; 3. Department of Control Automatic, Mexico National University of Science and Technology, Mexico City, D.F.07360, Mexico)

Abstract: Aming at the problem that the Levenberg-Marquardt(LM) algorithm can not online train RBF network and the problem in RBF network structure design methods, this paper presents an online self-adaptive RBF network structure design method based on the LM algorithm. The ideal of sliding window and online structure optimization are introduced in this algorithm, the introduction of sliding window enables the RBF network to be trained online by the LM algorithm, and makes the RBF network more robust to the changes of the learning parameters and is easy to converge. The online structure optimization can online self-adaptive adjust the structure of RBF network based on the information of training errors and hidden unites to track the time-varying systems, which helps to maintain a compact netowrk and satisfactory generation. Finally, the experiment results show the performance of the proposed algorithm.

Keywords: Levenberg-Marquardt algorithm; RBF network; online self-adaptive; sliding window; generalization ability; time-varying system

0 引 言

RBF网络以其强大的非线性拟合能力,以及收敛速度快、不易陷入局部极小点、鲁棒性强等优点,已经广泛应用于工业控制、模式分类、信号处理和非线性系统建模领域[1-4]. RBF网络结构设计和参数学习方法是其成功应用的关键[5]. 所谓结构设计就是采用什么策略来确定RBF网络中隐节点的数量;所谓参数学习就是如何调整RBF网络中的3个参数[6].

就RBF网结构优化算法而言,最为经典的在线设计算法是RAN、MRBF和GGAP-RBF算法. 上述算法的设计原则都是在线设计既能满足精度要求又能保持紧凑结构的网络,进而保证网络的泛化性能. RAN[7]的设计思想是在在线学习过程中不断检测每个训练样本,当新训练样本满足‘‘新性’’时,就为该训练样本分配新节点,否则用梯度下降法对网络参数进行修正. RAN算法能够充分利用RBF网络核

收稿日期: 2016-07-11; **修回日期:** 2017-01-23.

基金项目: 国家自然科学基金项目(61440059); 辽宁省自然科学基金项目(201602363); 国家留学基金委项目(201508210045).

作者简介: 张昭昭(1973−), 男, 副教授, 博士, 从事智能信息处理、神经网络结构优化设计等研究; 乔俊飞(1968−), 男, 教授, 博士生导师, 从事计算智能、智能特征建模和智能优化控制等研究.

†通讯作者. E-mail: zzzhao123@126.com

函数局部响应特性,参数修正时只修正新增加的隐节点,加快了网络的学习速度. 然而,在RAN算法中隐节点一旦添加便不再删除,因此对于复杂的在线学习任务,网络中必然会出现冗余的隐节点,从而影响网络的泛化性能. MRAN[8]在RAN的基础上进行了改进,其网络结构优化策略是除了满足误差准则和距离准则外,若当前网络对于连续多个训练样本的偏差都过大则增加一个新隐节点,若连续多个训练样本都不能激活某个隐节点则删除该隐节点[9]. 然而, MRAN增加隐节点时具有一定的盲目性,新增加的隐节点的核函数中心随机确定,且这两个算法都对学习参数变化的鲁棒性比较差,网络的泛化性能不能保证[10]; GGAP-RBF[11]算法结合网络学习精度判断隐节点的重要性,从而实现对网络结构的优化. 但是, GGAP-RBF需要根据全局样本设定初始值,对于一个在线RBF网结构优化算法,预先获得全局样本集是不可能实现的[6].

就网络参数学习算法而言,上述RBF网络结构设计算法均采用梯度下降法. 梯度下降法属于一阶算法,一阶算法在实际应用时存在的主要问题是收敛速度缓慢且易陷入误差曲面的局部极小. LM[12]算法属于二阶算法,当误差曲面的梯度较小时,LM算法类似于最速梯度法;当误差曲面的梯度较大时,LM算法类似于高斯-牛顿法. 由于LM算法能够根据Hessian矩阵估计误差曲面各个梯度方向上的学习率,与一阶算法相比, LM算法是目前训练神经网络最有效的算法[13]. ErrCor(error correction)[14]算法就是利用LM算法训练RBF网络,并在每次迭代后,在误差曲面峰值对应的训练样本处增加一个隐节点来提高RBF网络的学习能力. 该算法鲁棒性较强,也确实能够设计出结构最为紧凑的RBF网络结构. 然而,该算法属于离线式设计,不能应对非线性时变系统.

在前人研究的基础上,针对上述RBF网络结构设计中存在的问题,本文提出一种基于LM算法的在线自适应RBF网结构设计算法. 该算法构建一个滑动窗口,并采用LM算法训练RBF网络,在训练的过程中能够根据训练的误差信息以及每个隐节点相关信息自适应增加、删除或合并隐节点,使得学习过程中RBF网络结构紧凑,从而保证RBF网络的泛化性能. 采用滑动窗口的方法,既能够实现LM算法的在线应用,也能够使得RBF网络对学习参数变化具有更好的鲁棒性,并更易收敛. 最后通过仿真实验验证了所提出算法的性能.

1 RBF网络

RBF网络是一个包含输入层、隐含层和输出层的三层前馈网络,输入层到隐含层的权值固定为1. 不失一般性,设RBF网络结构为I-H-1,即I个输入节点,H个隐节点和1个输出节点,其结构如图1所示.

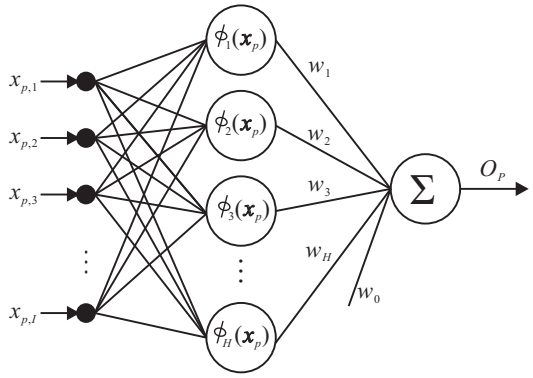

图1 RBF网络结构

设RBF网络第p个I维样本为$\boldsymbol{x}_p = [x_{p,1}, x_{p,2}, \cdots, x_{p,I}]$,则第$h$个隐节点的输出为

$$\phi_h(\boldsymbol{x}_p) = \exp\Big(-\frac{\|\boldsymbol{x}_p - \boldsymbol{c}_h\|^2}{\sigma_h}\Big). \tag{1}$$

其中: $\boldsymbol{c}_h$和σ_h分别表示第h个隐节点的核函数中心和宽度, $\|\cdot\|$表示欧氏距离.

网络对第P个样本的输出为

$$O_P = \sum_{h=1}^{H} w_h \varphi_h(\boldsymbol{X}_P) + w_0. \tag{2}$$

其中: w_h表示第h个隐节点与输出节点的连接权值, w_0为偏置.

2 在线自适应RBF网络结构优化

2.1 LM算法

LM算法是一种二阶算法,已成功地应用于BP网络训练. 在LM算法训练RBF网络方面,文献[13]对LM算法进行了改进. 在此基础上,文献[14]提出了一种根据训练误差峰值构造RBF网络结构的ErrCor算法. 该算法属于增长算法,能够设计出结构最为紧凑的RBF网络结构.

LM算法训练RBF网络时,参数更新规则为

$$\boldsymbol{\Delta}_{k+1} = \boldsymbol{\Delta}_k - (\boldsymbol{Q}_k + \mu_k \boldsymbol{I})^{-1}\boldsymbol{g}_k. \tag{3}$$

其中: $\boldsymbol{\Delta}$表示RBF网络中的可调参数(包括RBF隐节点核函数的中心$\boldsymbol{c}$、宽度σ, 以及RBF网络隐节点与输出节点的连接权值w); $\boldsymbol{Q}$表示Quasi-Hessian矩阵; $\boldsymbol{I}$为单位矩阵; μ为组合系数; $\boldsymbol{g}$为梯度向量.

Quasi-Hessian矩阵$\boldsymbol{Q}$可通过Subquasi-Hessian矩阵求和得到,即

$$\boldsymbol{Q} = \sum_{p=1}^{P} \boldsymbol{q}_p,\ \boldsymbol{q}_p = \boldsymbol{j}_p^{\mathrm{T}}\boldsymbol{j}_p. \tag{4}$$

梯度向量$\boldsymbol{g}$可通过子梯度向量$\boldsymbol{\eta}_p$求和得到,即

$$\boldsymbol{g}=\sum_{p=1}^{P}\boldsymbol{\eta}_p,\ \boldsymbol{\eta}_p=\boldsymbol{j}_p^{\mathrm{T}}e_p. \tag{5}$$

RBF网络训练误差为

$$e_p=y_p-o_p. \tag{6}$$

其中: y_p为RBF网络期望输出, o_p为RBF网络实际输出.

Jacobian矩阵第n行元素$\boldsymbol{j}_{p,n}$可根据下式计算:

$$\boldsymbol{j}_{p,n}=\frac{\partial e_p}{\partial \Delta_n}, \tag{7}$$

其中n表示RBF网络中的3个可调参数.

对于给定的p个训练样本,考虑RBF网络中可调参数w_h、$\boldsymbol{c}_{h,i}$和σ_h, 则Jacobian矩阵中的行元素值为

$$\begin{aligned}\boldsymbol{j}_p=&\Big[\frac{\partial e_p}{\partial w_0},\frac{\partial e_p}{\partial w_1}\cdots\frac{\partial e_p}{\partial w_h}\cdots\frac{\partial e_p}{\partial w_H},\frac{\partial e_p}{\partial c_{1,1}}\cdots\frac{\partial e_p}{\partial c_{1,i}}\cdots\\&\frac{\partial e_p}{\partial c_{1,I}}\cdots\frac{\partial e_p}{\partial c_{h,1}}\cdots\frac{\partial e_p}{\partial c_{h,i}}\cdots\frac{\partial e_p}{\partial c_{h,I}}\cdots\frac{\partial e_p}{\partial c_{H,1}}\cdots\\&\frac{\partial e_p}{\partial c_{H,i}}\cdots\frac{\partial e_p}{\partial c_{H,I}},\frac{\partial e_p}{\partial \sigma_1}\cdots\frac{\partial e_p}{\partial \sigma_h}\cdots\frac{\partial e_p}{\partial \sigma_H}\Big].\end{aligned} \tag{8}$$

结合式(1)、(2)和(6), 根据微分链式法则,式(8)对第p个学习样本时, Jacobian矩阵中行元素的值为

$$\frac{\partial e_p}{\partial w_h}=-\varphi_h(\boldsymbol{X}_p),\ \frac{\partial e_p}{\partial w_0}=-1, \tag{9}$$

$$\frac{\partial e_p}{\partial c_{h,i}}=-\frac{2w_h\varphi_h(\boldsymbol{X}_p)(x_{p,i}-c_{h,i})}{\sigma_h}, \tag{10}$$

$$\frac{\partial e_p}{\partial \sigma_h}=-\frac{w_h\varphi_h(\boldsymbol{X}_p)\|\boldsymbol{X}_p-\boldsymbol{c}_h\|^2}{\sigma_h^2}. \tag{11}$$

依据式(9)～(11), 可求得Jacobian矩阵$\boldsymbol{j}_p$行的所有行元素. 对于所有学习样本,可求得Jacobian矩阵中的所有元素. 随后,可依据式(4)和(5)分别求得quasi-Hiessian矩阵$\boldsymbol{Q}$和梯度向量$\boldsymbol{g}$, 进而依据式(3)对RBF网络中3个参数进行调整.

2.2　在线自适应RBF网络结构优化

虽然LM算法是目前训练神经网络最有效的算法,但LM算法训练RBF网络时只能采取批处理的方式进行,所以ErrCor算法属于离线式设计RBF网络结构的算法,不能在线进行,因此难以应对非线性时变系统. 另外,由于RAN、MRAN和GGAP-RBF在线算法均利用最新的单个样本训练RBF网络,容易陷入局部最优,而且抗劣质样本能力较差. 对于在线建模,最合理的方法是在学习的过程中尽量利用最新的多个样本动态调节网络参数[10].

本文采用滑动窗口的方法解决上述问题. 滑动窗口实质上是一个长度固定的“先进先出”队列,队列中的元素为在线输入样本按照进入窗口的时间顺序排列. 设在线输入样本为(x_n,y_n), 则长度为L的滑动窗口中的元素可表示为$[(x_i,y_i),(x_{i+1},y_{i+1}),\cdots,(x_{i+L-1},y_{i+L-1})]$. 当一个新样本产生后,窗口中最前边的一个样本便滑出窗口,而最新的样本将进入窗口. 滑动窗口中所有样本都是RBF网络训练样本.

根据上述,当采用滑动窗口方法,用LM算法训练RBF网络时, RBF网络学习的目标函数为

$$e_L=\sum_{i=1}^{L}\beta_i(y_i-o_i)^2. \tag{12}$$

其中: L为滑动窗口的长度; y_i和o_i分别表示窗口中第i个样本的期望输出和实际输出; β_i表示遗忘因子,表示为

$$\beta_i=\frac{2i}{L(L+1)},\ \sum_{i}^{L}\beta_i=1. \tag{13}$$

由式(12)和(13)可知,在线学习时,最新的样本中含有的信息量大,因此滑动窗口中最新样本的加权系数较大,而旧的样本加权系数较小.

本文提出的在线自适应RBF网络结构设计算法包括3个操作,分别是网络隐节点的增加、删除和合并操作. 隐节点的增加是在RBF网络在线学习过程中,一直检测并记录每个滑动窗口中的训练样本在学习过程中的最大误差,当RBF网络训练到一定步数且滑动窗口中训练样本的均方差未达到设定的目标值时,在RBF网络隐层增加一个隐节点,并设该新增加隐节点核函数中心为当前所记录的最大训练误差所对应的训练样本. 滑动窗口训练样本的均方差为

$$e_{\mathrm{rmse}}=\sqrt{\frac{\sum_{i=1}^{L}\beta_i(y_i-o_i)^2}{L}}. \tag{14}$$

隐节点删除操作. 对于在训练过程中删除连续多个训练样本都不能被激活的隐节点,若某个隐节点对连续多个训练样本都不能被激活,表明该隐节点随着学习对象的变化已经没有学习能力,则删除该隐节点. 判断隐节点是否被激活的准则为[8]

$$\phi_h^k(\boldsymbol{x}_k)=w_h^k\exp\Big(-\frac{\|\boldsymbol{x}_k-\boldsymbol{c}_h\|^2}{\sigma_h}\Big),\ r_h^k=\Big\|\frac{\phi_h^k}{\phi_{\max}^k}\Big\|. \tag{15}$$

其中: ϕ_h^k表示k时刻第h个隐节点的输出, w_h^k表示k时刻第h个隐节点与输出节点的连接权值, r_h^k表示k时刻第h个隐节点的归一化输出, $\phi_{\max}^k$表示k时刻所有隐节点输出中绝对值最大的值. 对于连续的多个训练样本,如果某个隐节点的输出ϕ_h^k都小于给定的阈值,则删除该隐节点.

隐节点的合并操作. 在学习的过程中,如果当前RBF网络中两个隐节点的核函数中心距离很近且宽

度也非常接近,则由RBF网络隐节点局部响应特性的特点可知,这两个隐节点功能几乎相同,因此可以将这两个隐节点合并成为一个隐节点,在精简RBF网络结构的同时不会影响当前网络的学习性能. 合并隐节点后相关参数设置为

$$\begin{cases} w_i = w_i + w_j, \\ c_i = (c_i + c_j)/2, \\ \sigma_i = \max(\sigma_i, \sigma_j). \end{cases} \tag{16}$$

采用滑动窗口的方法既能解决LM算法不能在线应用的问题,又能使得算法具有较好的鲁棒性,而且算法参数容易整定. 所提出的网络结构设计算法既综合了RAN、MRAN和GGAP-RBF算法的优点,又都克服了各自的缺点. 其中增加隐节点时直接在当前训练误差最大的样本处增加一个隐节点,既能快速有效抑制当前RBF网络学习误差,又能避免增加隐节点的盲目性.

基于上述考虑,可获得自适应RBF网络结构优化算法,描述如下.

Step1: 开始训练时,网络中只有1个隐节点. 初始化滑动窗口长度,用第1个样本充满滑动窗口.

Step2: 对于每一个在线输入样本$(\boldsymbol{x}_k, y_k)$,将该训练样本移入滑动窗口,同时移除最前面的样本.

Step3: 用式(14)计算当前滑动窗口中训练样本的均方差$e_{\rm rmse}$,并记录到目前为止训练误差绝对值最大值所对应的训练样本.

如果达到设定的训练步数且$e_{\rm rmse} > e_{\rm obj}$($e_{\rm obj}$为网络训练误差目标值),则增加一个隐节点. 新增加隐节点的核函数中心为到目前为止所记录的训练误差绝对值最大值所对应的训练样本,新增加隐节点宽度和连接权值均为1.

用式(8)~(11)计算$\boldsymbol{j}_p$; 并计算$\boldsymbol{q}_p = \boldsymbol{j}_p^{\rm T}\boldsymbol{j}_p$和子梯度向量$\boldsymbol{\eta}_p = \boldsymbol{j}_p^{\rm T}e_p$; 用式(4)计算quasi-Hiessian矩阵$\boldsymbol{Q}$; 用式(5)计算梯度向量$\boldsymbol{g}$; 用式(3)调整当前网络中的所有参数; 用式(15)计算每个隐节点的输出,判断并标记每个隐节点的激活状态.

如果当前滑动窗口训练$e_{\rm rmse}$相对于上一时刻没有下降,则对式(3)中的组合参数μ_k进行调整[15].

Step4: 如果训练达到设定的训练步数,则检查当前隐节点中是否存在连续多个训练样本都不能被激活的隐节点;若存在,则删除该隐节点.

Step5: 检查网络中是否有冗余的隐节点,如果某两个隐节点中心间的欧氏距离小于预先设定的阈值,则合并这两个隐节点,并用式(16)为合并的隐节点赋值.

Step6: 如果不是最后一个训练样本,则$k = k + 1$, 转Step2.

3 仿真分析

3.1 非线性函数逼近

为检验本文所提出的算法的性能,选取典型非线性函数进行逼近. 该实验的目的在于检验采用本文所提出的算法是否能够设计出与实际一致的最小RBF网络.

非线性函数表达式如下[16]:

$$\begin{aligned} y(x) = {} & \exp\left[-\frac{(x_1-0.3)^2+(x_2-0.2)^2}{0.01}\right] + \\ & \exp\left[-\frac{(x_1-0.7)^2+(x_2-0.2)^2}{0.01}\right] + \\ & \exp\left[-\frac{(x_1-0.1)^2+(x_2-0.5)^2}{0.02}\right] + \\ & \exp\left[-\frac{(x_1-0.9)^2+(x_2-0.5)^2}{0.01}\right] + \\ & \exp\left[-\frac{(x_1-0.3)^2+(x_2-0.8)^2}{0.01}\right] + \\ & \exp\left[-\frac{(x_1-0.7)^2+(x_2-0.8)^2}{0.01}\right]. \end{aligned} \tag{17}$$

由式(17)可见,该非线性目标函数可由有6个隐节点的RBF网络实现. 实验时随机产生2 000个训练样本$((x_1,x_2),y)$,(x_1,x_2)为输入,y为输出,$x_i \in (0,1), i \in \{1,2\}$. 实验时设置滑动窗口长度$L = 20$.

由于训练样本随机产生,训练结果具有一定的随机性. 本文将该实验独立运行20次,其中有16次训练结果为6个隐节点,4次训练结果为7个隐节点.

表1所示为16次训练结果核函数中心、核函数宽度和隐节点与输出节点连接权值的平均值. 从表1可以看出,训练结束后,RBF网络的隐节点中心、扩展宽度及输出权值与目标函数的实际值非常接近,表明本文算法能够较好地实现给定的最简RBF网络,最终训练结果也远优于MRAN的结果[16].

表1 目标函数实际值与训练结果对比

实际中心	(0.3, 0.2)	(0.7, 0.2)	(0.1, 0.5)	(0.9, 0.5)	(0.3, 0.8)	(0.7, 0.8)
估计中心	(0.311 0, 0.200 0)	(0.679 8, 0.199 3)	(0.098 8, 0.500 0)	(0.879 9, 0.510 3)	(0.300 0, 0.799 4)	(0.698 1, 0.802 2)
实际宽度	0.1	0.1	0.144	0.144	0.1	0.1
估计宽度	0.099 8	0.099 9	0.142 5	0.149 6	0.099 2	0.099 5
实际权值	1	1	1	1	1	1
估计权值	1.029 9	1.007 3	1.020 0	0.988 7	1.002 1	1.000 3

3.2　非线性时变系统辨识

本实验的目的是检验本文所提出的算法辨识非线性时变系统的能力. 选择非线性系统预测的一个基准问题Mackey-Glass(MG)进行实验, MG序列由下面的时延微分方程产生:

$$\frac{\mathrm{d}x}{\mathrm{d}t}=\frac{0.2x(t-\tau)}{1+x^{10}(t-\tau)}-0.1x(t). \tag{18}$$

本文以嵌入矢量$[x(t)\ \ x(t-6)\ \ x(t-12)\ \ x(t-18)]$预测$x(t+50)$的值.

式(18)在τ不变的情况下为静态混沌时间序列. 为构造非线性时变系统,分别让τ等于17、30、50、100, 图2所示是$x(t+50)$相对于$x(t)$的混沌特性随着τ增大而增大的情况.

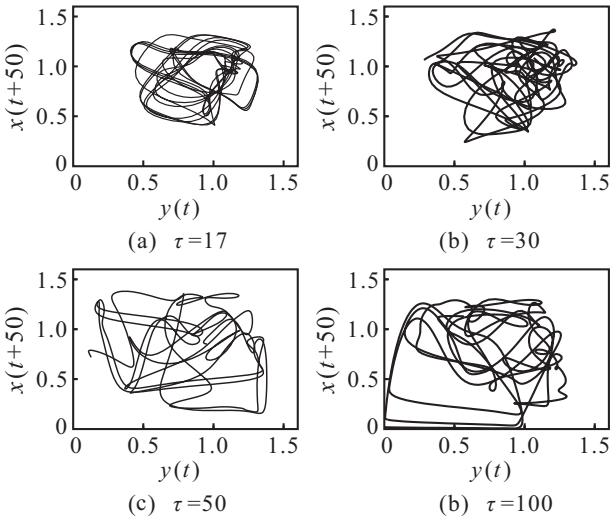

图2　不同τ值时$x(t+50)$的混沌特性

本文通过混合时延系数τ不同的静态MG序列得到时变系统,混合方法如下:

$$f(x_t)=\alpha(t)f_1x(t)+(1-\alpha(t))f_2(x_t), \tag{19}$$

$$\alpha(t)=\exp(-5t/T),\ t=1,2,\cdots,T. \tag{20}$$

通过混合4个静态MG序列,即$\tau=17\rightarrow30$, $\tau=30\rightarrow50$, $\tau=50\rightarrow100$, $\tau=100\rightarrow50$, $\tau=50\rightarrow30$, $\tau=30\rightarrow17$变化的数据各500个,共得到3 000个样本. 用前2 500个样本进行训练,后500个样本进行测试. 实验时设置滑动窗口的长度$L=200$.

在线学习过程中的预测情况如图3所示,图4记录了在线学习过程中的RSME变化情况及网络隐节点变化动态,图5所示为500个测试样本的测试误差. 从图3可以看出,在网络学习的初始阶段,训练误差很大,但随着网络结构的迅速构建,误差很快得到了抑制. 由图4也可以看出:在学习的初始阶段,网络训练RSME较大,并且在网络快速构建的初始阶段训练RSME有小幅震荡,但总体趋势趋于收敛; 在$t=241$时,网络中隐节点数目就达到36个,基本完成了网络的构建; 在此后的学习过程中,网络结构始终保持微调状态,而且能很好地学习在线任务; 在整个学习过程中,最大隐节点数达到41个, 2 500个样本学习结束时隐节点数为39个. 由图5可以看出,最大的绝对误差在0.03以内, 500个测试样本的平均误差为0.007 2. 针对该实验,作者做了20次独立的实验,并且让滑动窗口的长度L分别等于100、150和200, 隐节点数均在39 ~ 45之间, 500个测试样本的平均误差也在0.007 2 ~ 0.013 6之间,这是一个很稳定的实验结果.

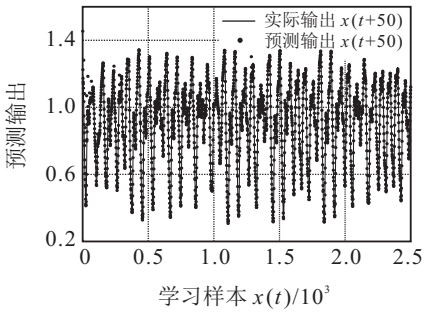

图3　在线学习情况

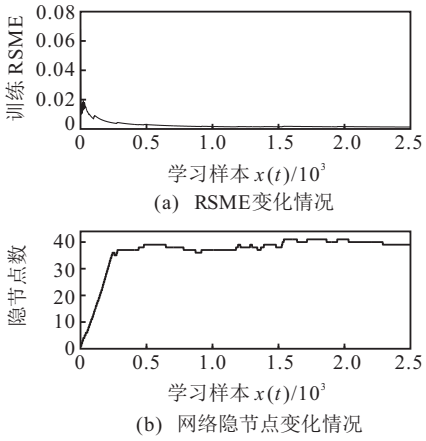

图4　训练RSME和隐节点变化情况

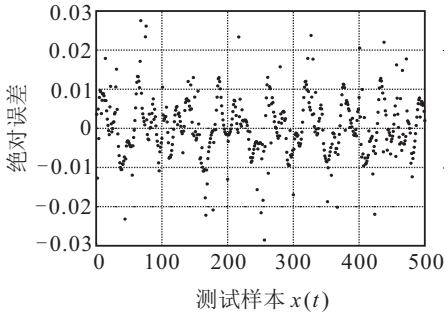

图5　500个测试样本测试误差

表2所示为本文算法与RAN、MRAN算法针对同样问题的学习结果比较.

表2　3种算法学习结果比较

算法	隐节点数	学习RSME	测试误差
RAN	51	0.031 6	0.042 7
MRAN	47	0.027 4	0.031 9
本文算法	39	0.005 7	0.007 2

从表2可以看出,本文算法无论是最终网络结构的紧凑性还是学习RSME和测试误差均优于RAN和MARN. 究其原因,一是本文RBF网络学习算法采用LM算法,该算法的学习效果确实优于梯度法;二是本文RBF网络结构优化算法综合了RAN和MRAN的优点,尤其是增加隐节点时直接在产生最大误差的样本点处增加隐节点,该方法能以最快速度抑制当前网络的训练误差;三是采用滑动窗口的方法,既提高了网络学习的鲁棒性,也提高了网络结构设计方法的稳定性.

4 结 论

本文针对LM算法不能在线训练RBF网络及RBF网络结构设计中存在的问题,提出了一种基于LM算法的在线自适应RBF网络结构优化算法. 该算法的实质是引入了滑动窗口的方法和综合前人在RBF网络结构优化方法的优点. 滑动窗口的引入既使得LM算法能够在线训练RBF网络,又使得网络对学习参数的变化具有更好的鲁棒性并易于收敛. 网络结构优化算法包括隐节点的增加、删除和合并3个操作,其中增加隐节点时直接在当前误差最大的样本点处增加隐节点,能以最快的速度有效抑制当前网络的训练误差. 最后通过对典型非线性函数逼近和时变系统辨识问题的实验,表明了本文算法在保证RBF网络学习性能和泛化性能的基础上,确实能够设计出结构最为紧凑的RBF网络.

参考文献(References)

[1] Lei Xu, Fang Qian, Yaping Li, et al. Resource allocation based on quantum particle swarm optimization and RBF neural network for overlay cognitive[J]. Neurocomputing, 2016, 173(3): 1250-1256.

[2] Tao Xiong, Yukun Bao, Zhongyi Hu, et al. Forecasting interval time series using fully complex valued RBF neural network with DPSO and PSO algorithms[J]. Information Sciences, 2015, 305(1): 77-92.

[3] 刘全, 肖飞, 付启明, 等. 基于自适应归一化RBF网络的Q-V值函数协同逼近模型[J]. 计算机学报, 2015, 38(7): 1386-1396.
(Liu Q, Xiao F, Fu Q M, et al. Commaborative Q-V value function approximation model based on adaptive normalized radial basis function network[J]. Chinese J of Computers, 2015, 38(7): 1386-1396.)

[4] 张琨, 崔胜民, 王剑锋. 基于自适应RBF网络补偿的智能车循迹控制[J]. 控制与决策, 2014, 29(4): 627-637.
(Zhang K, Cui S M, Wang J F. Intelligent vehicle's path tracking control based on self-adaptive RBF network compensation[J]. Control and Decision, 2014, 29(4): 627-637.)

[5] 杨一, 高社生, 胡高歌. 基于敏感度方差重要性的RBF神经网络结构优化算法[J]. 控制与决策, 2015, 30(8): 1393-1398.
(Yang Y, Gao S S, Hu G G. Optimal algorithm for RBF neural network structure based on variance significance in output sensitivity[J]. Control and Decision, 2015, 30(8): 1393-1398.)

[6] 张昭昭, 乔俊飞. 基于在线聚类的RBF神经网络结构设计[J]. 控制与决策, 2012, 27(7): 997-1002.
(Zhang Z Z, Qiao J F. Design RBF neural network architecture based on online substractive clustering[J]. Control and Decision, 2012, 27(7): 997-1002.)

[7] Platt J. A resource-allocating network for function interpolation[J]. Neural Computation, 1991, 3(2): 213-225.

[8] Lu Y W, Sundararajan N, Saratchandran P. A sequential learning scheme for function approximation using minimal radial basis function neural networks[J]. Neural Computation, 1997, 9(2): 461-478.

[9] 魏海坤, 宋文忠, 李奇. 非线性系统RBF网络在线建模的资源优化网络方法[J]. 自动化学报, 2005, 31(6): 970-974.
(Wei H K, Song W Z, Li Q. Resource optimizing networks for nonlinear system online modelling using radial basis function networks[J]. Acta Automatica Sinica, 2005, 31(6): 970-974.)

[10] Jawad Arif, Nilanjan Ray Chaudhuri, Swakshar Ray, et al. Online Levenberg-Marquardt algorithm for neural network based estimation and control of power systems[C]. Proc of Int Joint Conf on Neural Networks. Atlanta, 2009: 199-206.

[11] Huang G B, Saratchandran P, Sundararajan N. A generalized growing and pruning RBF(GGAP-RBF) neural network for function approximation[J]. IEEE Trans on Neural Networks, 2005, 16(1): 57-67.

[12] Levenberg K. A method for the solution of certain problems in least squares[J]. Quarterly of Applied Mathematics, 1994, 2: 164-168.

[13] Bogdan M Wilamowski, Hao Yu. Improved computation for Levenberg-Marquardt training[J]. IEEE Trans on Neural Networks, 2010, 21(6): 930-937.

[14] Hao Yu, Philip D Reiner, TianTian Xie, et al. An incremental design of radial basis function networkworks[J]. IEEE Trans on Neural Networks and Learning Systems, 2014, 25(10): 1793-1803.

[15] Hagan M T, Menhaj M B. Training feedforward networks with the Marquardt algorithm[J]. IEEE Trans on Neural Networks, 1994, 5(6): 989-993.

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

（责任编辑：曹洪式）