文章编号: 1001-0920(2019)01-0213-06 DOI: 10.13195/j.kzyjc.2017.0543

基于核映射极限学习机的入口氮氧化物预测

金秀章, 张少康†

(华北电力大学 控制与计算机工程学院，河北 保定 071003)

摘 要: 针对在线贯序极限学习机(OS-ELM)算法隐含层输出不稳定、易产生奇异矩阵和在线贯序更新时没有考虑训练样本时效性的问题,提出一种基于核函数映射的正则化自适应遗忘因子(FFOS-RKELM)算法. 该算法利用核函数代替隐含层,能够产生稳定的输出结果. 在初始阶段加入正则化方法,通过构造非奇异矩阵提高模型的泛化能力;在贯序更新阶段,通过新到的数据自动更新遗忘因子. 将FFOS-RKELM算法应用到混沌时间序列预测和入口氮氧化物时间序列预测中,相比于OS-ELM、FFOS-RELM、OS-RKELM算法,可有效地提高预测精度和泛化能力.

关键词: 极限学习机；核函数；遗忘因子；正则化；时间序列；入口氮氧化物

中图分类号: TP183 **文献标志码:** A

Prediction of inlet NOx based on extreme learning machine of kernel mapping

JIN Xiu-zhang, ZHANG Shao-kang†

(School of Control and Computer Engineering，North China Electric Power University，Baoding 071003，China)

Abstract: To solve the problem that the hidden layer output of an online sequential extreme learning machine(OS-ELM) algorithm is not stable, the singular matrix is easy to produce, and the OS-ELM has no consideration about the training sample timeliness during the sequential updating process, an improved OS-ELM algorithm online sequential extreme learning machine based on adaptive forgetting factor of kernel function mapping(FFOS-RKELM) is presented based on the regularization and adaptive forgetting factor of kernel function mapping. In the FFOS-RKELM algorithm, the kernel function replaces the hidden layer to produce the stable output results. In the initialization phase, the regularization method can improve the generalization ability of the model by constructing a nonsingular matrix. During the sequential updating phase, the forgetting factor can be adjusted automatically according to new data. The FFOS-RKELM algorithm is applied to the prediction of the chaotic time series and the time series of Inlet NOx. Compared with the OS-ELM algoyithm, the FFOS-RELM algorithm and the OS-RKELM algorithm, the proposed algorithm can improve the prediction accuracy and generalization ability more effectively.

Keywords: extreme learning machine；kernel function；forgetting factor；regularization；time series；inlet nitrogen oxides

0 引 言

人工神经网络得到了广泛的研究,并已成功地应用于时间序列预测,但却很少应用在工业中,主要是因为其网络需要大量的训练时间和数据才能得到好的效果[1]. 极限学习机(ELM)[2]作为一种新的单隐层前馈神经网络被提出,ELM算法只更新输出层的权值,使得ELM具有学习速度快、泛化能力强的特点. 但是ELM是一种离线的算法,不能在线更新,因此,Liang等[3]提出了在线贯序极限学习机(OS-ELM)算法,根据上一步输出层的权值和新到的数据递推更新网络权值,实现在线学习.

OS-ELM算法在线的更新输出权值具有较快的收敛速度,比ELM更加准确,但是存在输出不稳定和容易使输出矩阵出现奇异的问题. 新旧数据赋予相同的权值,对于时变系统而言,会大大降低预测的精度. Huang等[4]提出了核极限学习机(KELM)算法,将核函数引入ELM中,利用核映射代替原有的随机映射,产生稳定的输出结果. Huynh等[5]提出了一种正

收稿日期: 2017-05-04；**修回日期:** 2018-05-19.

责任编委: 赵珺.

作者简介: 金秀章(1969−), 男, 副教授, 博士, 从事大型发电机组先进控制策略等研究；张少康(19−), 男, 硕士生, 从事信号分析与信号处理的研究.

†通讯作者. E-mail：15530204649@163.com.

则化的在线贯序极限学习机(OS-RELM)算法,通过引入正则化方法避免了输出矩阵出现奇异的情况,提高了模型的泛化能力. Deng等[6]对上述两种方法进行整合,提出了带有正则化方法的在线贯序核极限学习机(OS-RKELM),既能使输出结果稳定,又避免了输出矩阵出现奇异的情况. Soares等[7]提出了带有遗忘因子的在线贯序极限学习机(FFOS-RELM),根据新到的数据更新遗忘因子,使遗忘因子自动更新,更加适用于时变系统.

本文在上述工作的基础上,将自适应遗忘因子、正则化方法、核函数映射一起加入到OS-ELM算法中,推导基于核函数映射的正则化自适应遗忘因子(FFOS-RKELM)递推公式. 将核函数映射代替隐含层映射,能够产生稳定的输出结果. 加入正则化方法以防止输出矩阵出现奇异的情况,降低对核函数个数的依赖性. 加入自适应遗忘因子,可有效提高新数据占比,非常适用于时变系统. 将FFOS-RKELM算法应用于混沌时间序列和脱硝系统的入口氮氧化物时间序列中,并与OS-ELM、OS-RKELM、FFOS-RELM算法进行比较. 仿真结果表明,该算法的准确性和泛化能力都得到了提高.

1 OS-ELM算法

OS-ELM的SLFN回归模型为

$$f_L(\boldsymbol{x})=\sum_{i=1}^{L}\boldsymbol{\beta}_i g(\boldsymbol{a}_i,b_i,\boldsymbol{x}). \tag{1}$$

其中:$g(\cdot)$为激活函数,$\boldsymbol{a}_i$和b_i为隐层的参数,$\boldsymbol{\beta}_i$为隐含层第i个节点到输出层之间的权值,L为隐层节点数.

OS-ELM学习方法分为两个阶段:初始化阶段和贯序更新阶段[8-9]. 训练数据为$\boldsymbol{D}=\{(\boldsymbol{x}_t,\boldsymbol{y}_t)\}_{t=1}^{T}$,其中$\boldsymbol{x}_t\in\boldsymbol{R}^n,\boldsymbol{y}_t\in\boldsymbol{R}^m$,初始阶段的数据从$\boldsymbol{D}$中选取,即$\boldsymbol{D}_0=\{(\boldsymbol{x}_t,\boldsymbol{y}_t)\}_{t=1}^{T_0}$,满足$T_0<T$,贯序更新阶段的第$k+1$次的数据块为

$$\boldsymbol{D}_{k+1}=\{(\boldsymbol{x}_t,\boldsymbol{y}_t)\}_{t=\left(\sum\limits_{l=0}^{k}T_l\right)+1}^{t=\sum\limits_{l=0}^{k+1}T_l}. \tag{2}$$

其中:$k\geqslant 0,\boldsymbol{T}_{k+1}$为第$k+1$次数据块样本的个数. 初始阶段为一般的ELM学习算法,输出权重为

$$\boldsymbol{\beta}_0=\boldsymbol{H}_0^{+}\boldsymbol{y}_0=(\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0)^{-1}\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{y}_0. \tag{3}$$

其中:$\boldsymbol{H}_0^{+}$为$\boldsymbol{H}_0$的广义逆矩阵,$\boldsymbol{y}_0=[y_1\ \cdots\ y_{T_0}]^{\mathrm{T}}$为$\boldsymbol{D}_0$中的输出,$\boldsymbol{H}_0$为初始隐含层的输出矩阵,即

$$\boldsymbol{H}_0=\begin{bmatrix} g(\boldsymbol{a}_1,b_1,\boldsymbol{x}_1) & \cdots & g(\boldsymbol{a}_L,b_L,\boldsymbol{x}_1)\\ \vdots & \ddots & \vdots\\ g(\boldsymbol{a}_1,b_1,\boldsymbol{x}_{T_0}) & \cdots & g(\boldsymbol{a}_L,b_L,\boldsymbol{x}_{T_0})\end{bmatrix}_{T_0\times L}, \tag{4}$$

$$\boldsymbol{\beta}_0=[\boldsymbol{\beta}_1\ \cdots\ \boldsymbol{\beta}_L]^{\mathrm{T}}. \tag{5}$$

设$\boldsymbol{L}_0=\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0$,则式(3)可以写为

$$\boldsymbol{\beta}_0=\boldsymbol{L}_0^{-1}\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{y}_0. \tag{6}$$

当第$k+1$次的数据过来后,可以得到如下的迭代公式:

$$\boldsymbol{L}_{k+1}=\boldsymbol{L}_k+\boldsymbol{H}_{k+1}^{\mathrm{T}}\boldsymbol{H}_{k+1}, \tag{7}$$

$$\boldsymbol{\beta}_{k+1}=\boldsymbol{\beta}_k+\boldsymbol{L}_{k+1}^{-1}\boldsymbol{H}_{k+1}^{\mathrm{T}}(\boldsymbol{y}_{k+1}-\boldsymbol{H}_{k+1}\boldsymbol{\beta}_k). \tag{8}$$

其中

$$\boldsymbol{H}_{k+1}=\begin{bmatrix} g(\boldsymbol{a}_1,b_1,\boldsymbol{x}_{\left(\sum\limits_{l=0}^{k}T_l\right)+1}) & \cdots & g(\boldsymbol{a}_L,b_L,\boldsymbol{x}_{\left(\sum\limits_{l=0}^{k}T_l\right)+1})\\ \vdots & \ddots & \vdots\\ g(\boldsymbol{a}_1,b_1,\boldsymbol{x}_{\left(\sum\limits_{l=0}^{k+1}T_l\right)}) & \cdots & g(\boldsymbol{a}_L,b_L,\boldsymbol{x}_{\left(\sum\limits_{l=0}^{k+1}T_l\right)})\end{bmatrix}_{T_{k+1}\times L}, \tag{9}$$

$$\boldsymbol{y}_{k+1}=[y_{\left(\sum\limits_{l=0}^{k}T_l\right)+1}\ \cdots\ y_{\left(\sum\limits_{l=0}^{k+1}T_l\right)}]^{\mathrm{T}}. \tag{10}$$

OS-ELM算法的步骤如下:

1) 初始化阶段. 训练数据为$\boldsymbol{D}=\{(\boldsymbol{x}_t,\boldsymbol{y}_t)\}_{t=1}^{T}$,隐含层节点数$L(L<T_0<T)$.

Step 1: 建立一个初始训练数据$\boldsymbol{D}_0$;

Step 2: 选取激活函数$g(\cdot)$,随机产生隐含层参数$\boldsymbol{a}_i$和$b_i,i=1,2,\cdots,L$;

Step 3: 由式(4)和$\boldsymbol{D}_0$计算$\boldsymbol{H}_0$;

Step 4: 由式(6)计算初始的输出权重$\boldsymbol{\beta}_0$,其中$\boldsymbol{L}_0=\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0$;

Step 5: 令$k=0$.

2) 贯序更新阶段.

Step 6: 由式(2)得到第$k+1$次的数据$\boldsymbol{D}_{k+1}$;

Step 7: 由$\boldsymbol{D}_{k+1}$和式(9)得到$\boldsymbol{H}_{k+1}$,由式(10)得到$\boldsymbol{y}_{k+1}$;

Step 8: 由式(7)和(8)得到$\boldsymbol{L}_{k+1}$和$\boldsymbol{\beta}_{k+1}$;

Step 9: 令$k=k+1$,返回2)中的Step 6继续执行.

2 FFOS-RKELM算法

2.1 OS-RKELM算法

OS-ELM算法中需要满足$L<T_0$,并且隐含层的参数是随机给定的,可能出现输出不稳定的情况. 求输出矩阵$\boldsymbol{\beta}$时,应保证矩阵$\boldsymbol{H}_0^{\mathrm{T}}\boldsymbol{H}_0$必须是奇异矩阵. 当两组数据接近时,可能导致矩阵非奇异,从而出现过拟合现象.

针对上述问题,采用核函数映射代替隐含层的随机映射[10],模型如下:

$$\boldsymbol{y}=\sum_{i=1}^{L}\boldsymbol{\beta}_i\boldsymbol{K}(x,x_i)=\boldsymbol{K}_L\boldsymbol{\beta}. \tag{11}$$

其中: $\boldsymbol{K}(x,x_i)$是隐层的神经元,核函数为径向基函数$K(u,v)=\exp[-(\|u-v\|^2/\mu)]$,$\mu$为核参数. 映射样本$\boldsymbol{X}_L=[x_1\ \cdots\ x_L]$随机从$\boldsymbol{X}$中挑选.

为了保证矩阵$\boldsymbol{K}_0^{\mathrm{T}}\boldsymbol{K}_0$是非奇异矩阵,加入正则化方法[11],式(11)中$\boldsymbol{\beta}$的求解转化为以下寻优问题:

$$\min_{\boldsymbol{\beta}}\{\|\boldsymbol{K}\boldsymbol{\beta}-\boldsymbol{y}\|^2+\|\boldsymbol{\beta}\|^2/C\},\tag{12}$$

其中C为正则化系数. 式(12)中的最优解可由下式计算[5]:

$$\boldsymbol{\beta}=(\boldsymbol{K}^{\mathrm{T}}\boldsymbol{K}+\boldsymbol{I}/C)^{-1}\boldsymbol{K}^{\mathrm{T}}\boldsymbol{y}.\tag{13}$$

OS-RKELM算法的推导过程如下:

给定初始的训练数据$\boldsymbol{D}_0=\{(\boldsymbol{x}_t,\boldsymbol{y}_t)\}_{t=1}^{T_0}$,映射样本$\boldsymbol{x}_L$随机从$\boldsymbol{D}_0$中挑选. 由式(13)可知,初始的输出权重为

$$\boldsymbol{\beta}_0=\boldsymbol{Z}_0^{-1}\boldsymbol{K}_0^{\mathrm{T}}\boldsymbol{y}_0,\tag{14}$$

$$\boldsymbol{Z}_0=\boldsymbol{K}_0^{\mathrm{T}}\boldsymbol{K}_0+\boldsymbol{I}/C,\tag{15}$$

$$\boldsymbol{K}_0=\boldsymbol{K}(\boldsymbol{X}_0,\boldsymbol{X}_L).\tag{16}$$

当新数据$\boldsymbol{D}_1=\{(\boldsymbol{x}_i,\boldsymbol{y}_i)\}_{i=T_0+1}^{T_0+T_1}$到来时,$T_1$为新样本的个数,输出权重为

$$\boldsymbol{\beta}_1=\boldsymbol{Z}_1^{-1}\begin{bmatrix}\boldsymbol{K}_0\\ \boldsymbol{K}_1\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{y}_0\\ \boldsymbol{y}_1\end{bmatrix}=\boldsymbol{Z}_1^{-1}\boldsymbol{P}_1.\tag{17}$$

式(17)中$\boldsymbol{Z}_1$和$\boldsymbol{Z}_0$的关系如下所示:

$$\boldsymbol{Z}_1=\boldsymbol{I}/C+\begin{bmatrix}\boldsymbol{K}_0\\ \boldsymbol{K}_1\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{K}_0\\ \boldsymbol{K}_1\end{bmatrix}=$$

$$\boldsymbol{I}/C+\boldsymbol{K}_0^{\mathrm{T}}\boldsymbol{K}_0+\boldsymbol{K}_1^{\mathrm{T}}\boldsymbol{K}_1=\boldsymbol{Z}_0+\boldsymbol{K}_1^{\mathrm{T}}\boldsymbol{K}_1,\tag{18}$$

$$\boldsymbol{K}_1=\boldsymbol{K}(\boldsymbol{X}_1,\boldsymbol{X}_L).\tag{19}$$

结合式(18)对$\boldsymbol{P}_1$进行如下变换:

$$\begin{bmatrix}\boldsymbol{K}_0\\ \boldsymbol{K}_1\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{y}_0\\ \boldsymbol{y}_1\end{bmatrix}=\boldsymbol{K}_0^{\mathrm{T}}\boldsymbol{y}_0+\boldsymbol{K}_1^{\mathrm{T}}\boldsymbol{y}_1=$$

$$\boldsymbol{Z}_0\boldsymbol{Z}_0^{-1}\boldsymbol{K}_0^{\mathrm{T}}\boldsymbol{y}_0+\boldsymbol{K}_1^{\mathrm{T}}\boldsymbol{y}_1=\boldsymbol{Z}_0\boldsymbol{\beta}_0+\boldsymbol{K}_1^{\mathrm{T}}\boldsymbol{y}_1=$$

$$(\boldsymbol{Z}_1-\boldsymbol{K}_1^{\mathrm{T}}\boldsymbol{K}_1)\boldsymbol{\beta}_0+\boldsymbol{K}_1^{\mathrm{T}}\boldsymbol{y}_1=$$

$$\boldsymbol{Z}_1\boldsymbol{\beta}_0-\boldsymbol{K}_1^{\mathrm{T}}\boldsymbol{K}_1\boldsymbol{\beta}_0+\boldsymbol{K}_1^{\mathrm{T}}\boldsymbol{y}_1.\tag{20}$$

将式(20)代入(17),可得

$$\boldsymbol{\beta}_1=\boldsymbol{\beta}_0+\boldsymbol{Z}_1^{-1}\boldsymbol{K}_1^{\mathrm{T}}(\boldsymbol{y}_1-\boldsymbol{K}_1\boldsymbol{\beta}_0).\tag{21}$$

依次类推,可得递推公式如下:

$$\boldsymbol{Z}_{k+1}^{-1}=(\boldsymbol{Z}_k+\boldsymbol{K}_{k+1}^{\mathrm{T}}\boldsymbol{K}_{k+1})^{-1},\tag{22}$$

$$\boldsymbol{\beta}_{k+1}=\boldsymbol{\beta}_k+\boldsymbol{Z}_{k+1}^{-1}\boldsymbol{K}_{k+1}^{\mathrm{T}}(\boldsymbol{y}_{k+1}-\boldsymbol{K}_{k+1}\boldsymbol{\beta}_k),\tag{23}$$

$$\boldsymbol{K}_{k+1}=\boldsymbol{K}(\boldsymbol{X}_{k+1},\boldsymbol{X}_L).\tag{24}$$

为了避免每次求逆矩阵$\boldsymbol{Z}_{k+1}^{-1}$,利用Sherman-Morrison矩阵求逆定理[12]化简式(22),得到如下公式:

$$\boldsymbol{Z}_{k+1}^{-1}=(\boldsymbol{Z}_k+\boldsymbol{K}_{k+1}^{\mathrm{T}}\boldsymbol{K}_{k+1})^{-1}=\boldsymbol{Z}_k^{-1}-\frac{\boldsymbol{Z}_k^{-1}\boldsymbol{K}_{k+1}^{\mathrm{T}}\boldsymbol{K}_{k+1}\boldsymbol{Z}_k^{-1}}{\boldsymbol{I}+\boldsymbol{K}_{k+1}\boldsymbol{Z}_k^{-1}\boldsymbol{K}_{k+1}^{\mathrm{T}}}.\tag{25}$$

令$\boldsymbol{Z}_{k+1}^{-1}=\boldsymbol{P}_k$,则式(22)和(23)变为

$$\boldsymbol{P}_{k+1}=\boldsymbol{P}_k-\frac{\boldsymbol{P}_k\boldsymbol{K}_{k+1}^{\mathrm{T}}\boldsymbol{K}_{k+1}\boldsymbol{P}_k}{1+\boldsymbol{K}_{k+1}\boldsymbol{P}_k\boldsymbol{K}_{k+1}^{\mathrm{T}}},\tag{26}$$

$$\boldsymbol{\beta}_{k+1}=\boldsymbol{\beta}_k+\boldsymbol{P}_{k+1}\boldsymbol{K}_{k+1}^{\mathrm{T}}(\boldsymbol{y}_{k+1}-\boldsymbol{K}_{k+1}\boldsymbol{\beta}_k).\tag{27}$$

OS-RKELM算法的具体步骤如下:

1) 初始化阶段. 初始的训练数据$\boldsymbol{D}_0=\{(\boldsymbol{x}_t,\boldsymbol{y}_t)\}_{t=1}^{T_0}$从数据$\boldsymbol{D}=\{(\boldsymbol{x}_t,\boldsymbol{y}_t)\}_{t=1}^{T}$中挑选,初始化参数$C,\mu$.

Step 1: $\boldsymbol{X}_L$随机从$\boldsymbol{D}_0$中选择,作为映射样本;

Step 2: 由式(16)、$\boldsymbol{D}_0$和$\boldsymbol{X}_L$计算初始的核矩阵$\boldsymbol{K}_0=\boldsymbol{K}(\boldsymbol{X}_0,\boldsymbol{X}_L)$;

Step 3: 由式(14)获得初始的输出权重$\boldsymbol{\beta}_0$,其中$\boldsymbol{Z}_0=(\boldsymbol{K}_0^{\mathrm{T}}\boldsymbol{K}_0+\boldsymbol{I}/C)^{-1}$,$\boldsymbol{y}_0=[y_1\ \cdots\ y_{T_0}]^{\mathrm{T}}$;

Step 4: 令$k=0$.

2) 贯序更新阶段.

Step 5: 当第$k+1$次数据到来时,由式(24)和(10)求得$\boldsymbol{K}_{k+1}$和$\boldsymbol{y}_{k+1}$;

Step 6: 由式(26)和(27)计算$\boldsymbol{P}_{k+1}$和$\boldsymbol{\beta}_{k+1}$;

Step 7: 令$k=k+1$,返回2)中的Step 5继续执行.

2.2 FFOS-RKELM算法

OS-RKELM算法在更新时对新旧样本赋予了相同的权值,对于时变系统而言,新数据更能体现当前的情况. 这里加入自适应遗忘因子进行调整. 遗忘因子根据新样本的信息进行更新,对新数据特别敏感,可以避免饱和现象,适应于时变系统[13]. 令新来的数据个数$T_{k+1}=1$,在贯序更新阶段,把式(27)变为如下形式:

$$\boldsymbol{\beta}_{k+1}=\boldsymbol{\beta}_k+\frac{\boldsymbol{P}_k\boldsymbol{K}_{k+1}^{\mathrm{T}}}{1+\xi_{k+1}}\hat{e}_{k+1},\tag{28}$$

$$\hat{e}_{k+1}=y_{k+1}-\boldsymbol{K}_{k+1}\boldsymbol{\beta}_k,\tag{29}$$

$$\xi_{k+1}=\boldsymbol{K}_{k+1}\boldsymbol{P}_k\boldsymbol{K}_{k+1}^{\mathrm{T}}.\tag{30}$$

其中: $\hat{e}_{k+1}$为在$\boldsymbol{\beta}_k$下的新样本预测误差,ξ_{k+1}为辅助标量. 如果$\xi_{k+1}=0$,则令$\boldsymbol{P}_{k+1}=\boldsymbol{P}_k$;如果$\xi_{k+1}>0$,则把式(26)变为以下形式:

$$\boldsymbol{P}_{k+1}=\boldsymbol{P}_k-\frac{\boldsymbol{P}_k\boldsymbol{K}_{k+1}^{\mathrm{T}}\boldsymbol{K}_{k+1}\boldsymbol{P}_k}{\varepsilon_{k+1}^{-1}+\xi_{k+1}},\tag{31}$$

$$\varepsilon_{k+1}=\lambda_k-\frac{1-\lambda_k}{\xi_{k+1}}.\tag{32}$$

其中: ε_{k+1}为中间变量; λ_k为第k次的遗忘因子,并且满足$0<\lambda_k\leqslant 1$. λ_k越小,对旧数据的影响越小[7]. 第$k+1$次的遗忘因子公式为

$$\lambda_{k+1}=\Big\{1+(1+\rho)\Big[\ln(1+\xi_{k+1})+\Big(\frac{(\nu_{k+1}+1)\eta_{k+1}}{1+\xi_{k+1}+\eta_{k+1}}-1\Big)\frac{\xi_{k+1}}{1+\xi_{k+1}}\Big]\Big\}^{-1},\quad(33)$$

$$\eta_{k+1}=\hat{e}_{k+1}^2/\gamma_{k+1},\quad(34)$$

$$\gamma_{k+1}=\lambda_k\Big(\gamma_k+\frac{\hat{e}_{k+1}^2}{1+\xi_{k+1}}\Big),\quad(35)$$

$$\nu_{k+1}=\lambda_k(\nu_k+1).\quad(36)$$

其中: ρ是一个固定的常数,初始的γ和ν是0～1之间的数.

FFOS-RKELM算法具体步骤如下:

1) 初始化阶段. 初始的训练数据$\boldsymbol{D}_0=\{(\boldsymbol{x}_t,\boldsymbol{y}_t)\}_{t=1}^{T_0}$从数据$\boldsymbol{D}=\{(\boldsymbol{x}_t,\boldsymbol{y}_t)\}_{t=1}^{T}$中挑选,初始化参数$\lambda_0$, γ_0, ν_0, ρ, C, μ.

Step 1: $\boldsymbol{X}_L$随机从$\boldsymbol{D}_0$中选择,作为映射样本;

Step 2: 由式(16)、$\boldsymbol{D}_0$和$\boldsymbol{X}_L$计算初始的核矩阵

$$\boldsymbol{K}_0=\boldsymbol{K}(\boldsymbol{X}_0,\boldsymbol{X}_L);$$

Step 3: 由式(14)获得初始的输出权重$\boldsymbol{\beta}_0$,其中$\boldsymbol{Z}_0=(\boldsymbol{K}_0^{\mathrm{T}}\boldsymbol{K}_0+\boldsymbol{I}/C)^{-1}$, $\boldsymbol{y}_0=[y_1\ \cdots\ y_{T_0}]^{\mathrm{T}}$;

Step 4: 令$k=0$.

2) 贯序更新阶段.

Step 5: 当第$k+1$次数据到来时,由式(24)求得$\boldsymbol{K}_{k+1}$;

Step 6: 由式(28)～(30)得到$\boldsymbol{\beta}_{k+1}$、$\hat{e}_{k+1}$、ξ_{k+1};

Step 7: 如果$\xi_{k+1}=0$,则令$\boldsymbol{P}_{k+1}=\boldsymbol{P}_k$,如果$\xi_{k+1}>0$,则由式(31)和(32)求得$\boldsymbol{P}_{k+1}$;

Step 8: 由式(34)～(36)计算η_{k+1}、γ_{k+1}、ν_{k+1};

Step 9: 由式(33)计算下一步的遗忘因子λ_{k+1},返回2)中Step 5继续执行.

3 混沌时间序列的在线预测

混沌系统常用来检验非线性系统的性能[14-15]. 以典型的混沌时间序列Mackey-Glass和Henon为例进行研究. 这里选取Mackey-Glass混沌时间序列的$\tau=17$,利用四阶龙格-库塔法寻找方程的解. 初始条件$y(0)=1$, $\tau=17$,选取1 000个数据样本,方式如下:

$$[y(t-18),y(t-12),y(t-6),y(t);y(t+6)].$$

其中: $t\in(19,1\,018)$,前4项作为模型的输入,最后一项为模型输出,进行6步预测. Henon混沌时间序列的嵌入维数为4,数据个数为1 000. 选取的初始样本数据$T_0=150$,采用交叉验证的方法得到正则化系数和核参数. 当应用于Mackey-Glass混沌时间序列时, $C=1\,000$, $\mu=0.1$;当应用于Henon混沌时间序列时, $C=100\,000$, $\mu=1$. 初始的遗忘因子$\lambda_0=1$, γ_0、$\boldsymbol{v}_0$为0～1之间的随机数,固定常数$\rho=3.8$. 隐含层的个数对ELM模型的预测有很大影响,分析不同隐含层个数下的预测精度,以平均绝对值误差(MAE)和时间为标准,MAE的公式如下所示:

$$E_{\mathrm{MAE}}=\frac{1}{N}\sum_{j=1}^{N}|y(j)-\overline{y}(j)|.$$

其中: $y(j)$为实际输出, $\overline{y}(j)$为预测输出, N为样本个数. 与OS-ELM、OS-RKELM、FFOS-RELM算法进行比较,结果如表1所示.

以典型的混沌时间序列Mackey-Glass和Henon为例进行研究. 通过对两组混沌时间序列的实验结果对比可知: 隐含层节点数对FFOS-RELM、OS-RKELM和FFOS-RKELM算法的精度影响较小; FFOS-RKELM算法较OS-ELM、FFOS-RELM和OS-

表 1 混沌时间序列预测的MAE和时间

隐含层节点数	Mackey-Glass								Henon							
	FFOS-RKELM		OS-RKELM		FFOS-RELM		OS-ELM		EFOS-RKELM		OS-RKELM		FFOS-RELM		OS-ELM	
	MAE	time/s	MAE	time/s	MAE	time/s	MAE	time/s	MAE	time	MAE	time/s	MAE	time/s	MAE	time/s
50	0.006 9	0.51	0.010 4	0.49	0.009 1	1.11	0.014 9	0.91	0.012 5	0.46	0.017 9	0.42	0.020 4	1.09	0.030 1	0.88
60	0.006 0	0.79	0.010 8	1.07	0.008 6	1.58	0.029 9	1.92	0.009 4	0.75	0.011 2	0.77	0.016 0	1.55	0.081 2	1.40
70	0.005 6	0.91	0.010 0	1.13	0.008 0	1.91	0.048 9	2.11	0.008 1	0.91	0.009 1	0.87	0.015 3	1.99	0.149 5	1.94
100	0.005 6	1.53	0.010 9	1.62	0.007 4	3.12	0.078 3	3.12	0.005 8	1.51	0.007 4	1.46	0.014 1	2.91	0.418 4	2.98

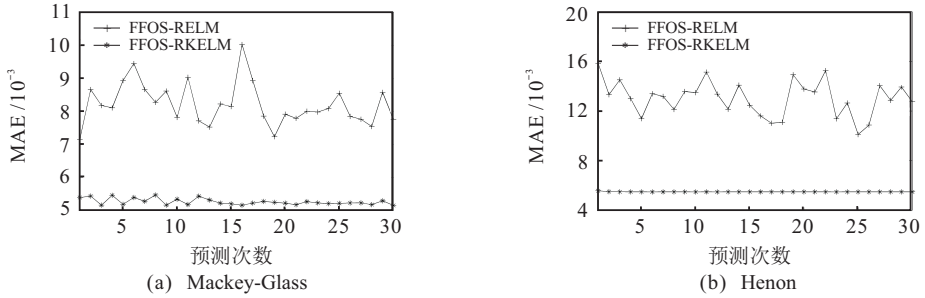
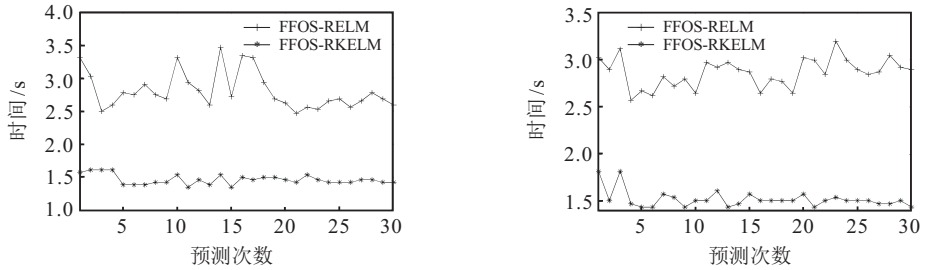


图 1 FFOS-RELM和FFOS-RKELM算法预测结果



(a) Mackey-Glass　(b) Henon

图 2　FFOS-RELM和FFOS-RKELM算法预测时间

RKELM 算法的精度得到显著提高,结果如表 1 所示. 另外,为了进一步验证核函数映射能够产生稳定的输出结果,对 1 000 组数据重复进行 30 次预测,结果如图1和图2所示. 由图1可知,FFOS-RKELM算法的稳定性更佳;由图2可知,FFOS-RKELM 算法的计算时间也较短. 分析表1、 图1和图2的结果可知,FFOS-RKELM 算法的稳定性和精度都优于FFOS-RELM、OS-RKELM、OS-ELM算法.

4　入口氮氧化物时间序列的预测

本实验选取的数据来自某660 MW机组的脱硝系统. 数据每10 s采集一个点,选取不同负荷下的290个数据点作为实验数据,然后把数据进行归一化. 选取初始的训练数据为$T_0 = 150$,映射样本的个数$T_L = 100$,初始遗忘因子$\lambda_0 = 1$,γ_0和v_0为0～1之间的随机数,固定常数$\rho = 3.8$. 入口氮氧化物时间序列的预测模型为

$$y(t+D) = f(y(t-1), y(t-2), \cdots, y(t-\Delta)).$$

其中:D为预测步长;Δ为嵌入维数,这里选取$\Delta = 4$. 分别对入口氮氧化物时间序列进行1步、2步和3步预测,并与OS-ELM、OS-RKELM、FFOS-RELM算法进行比较. 在优选的核参数C、正则化系数μ下计算平均绝对值误差、均方根误差、预测时间. RMSE公式如下:

$$E_{\mathrm{RMSE}} = \sqrt{\frac{1}{N}\sum_{j=1}^{N}[y(j) - \overline{y}(j)]^2}.$$

结果如表2～表4所示.

表 2　1步预测结果

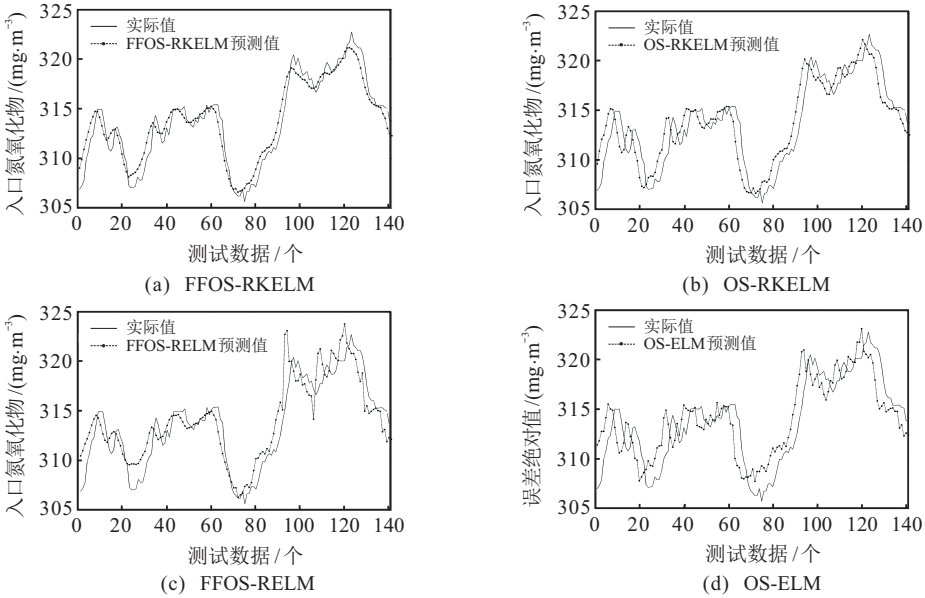
算法	MAE	RMSE	Time/s
OS-ELM	2.024 8	2.483 6	0.31
OS-RKELM	1.567 6	1.930 5	0.15
FFOS-RELM	1.605 2	2.093 0	0.29
FFOS-RKELM	0.960 5	1.173 5	0.11

表 3　2步预测结果

算法	MAE	RMSE	Time/s
OS-ELM	2.819 5	3.328 5	0.30
OS-RKELM	1.717 3	2.083 6	0.17
FFOS-RELM	1.892 9	2.554 5	0.25
FFOS-RKELM	1.360 9	1.712 8	0.14

表 4　3步预测结果

算法	MAE	RMSE	Time/s
OS-ELM	3.400 2	4.073 9	0.32
OS-RKELM	1.934 9	2.321 2	0.15
FFOS-RELM	1.947 8	2.601 1	0.31
FFOS-RKELM	1.638 0	2.232 8	0.13



(a) FFOS-RKELM　(b) OS-RKELM

(c) FFOS-RELM　(d) OS-ELM

图 3　入口氮氧化物1步预测值

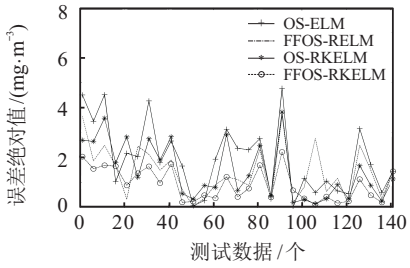


图 4　入口氮氧化物1步预测绝对误差

由表2～表4可以看出,随着预测步数的增加,FFOS-RKELM算法的预测精度更高,运算时间与OS-RKELM算法相差不大. 为更方便观察,以1步预测为例,分别绘制了不同算法下真实值和预测值的曲线以及误差绝对值曲线,如图3和图4所示. 由图3和图4可知,4种算法中FFOS-RKELM算法拟合效果最佳,误差绝对值也最小.

遗忘因子越小,表示旧样本数据占比越小,图5为FFOS-RKELM算法的遗忘因子曲线. 由图5可知:开始时旧数据占比较大,遗忘因子较大;随着数据的增加,旧数据占比越来越小,新数据占比增加,遗忘因子变小,更加适用于时变系统.

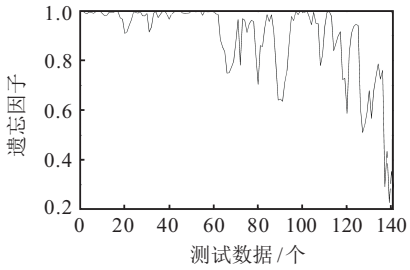


图 5　FFOS-RKELM的遗忘因子

5 结　论

本文把核函数映射、正则化方法以及自适应遗忘因子一起加入到OS-ELM算法中,结合它们的优点得到FFOS-RKELM算法,利用核函数代替隐含层,能够产生稳定的输出结果,使预测更加准确. 加入正则化方法以防止输出矩阵出现奇异矩阵,降低对隐层个数的依赖,从而提高模型的泛化能力. 加入自适应遗忘因子,可有效提高新数据占比,更适用于时变系统. 对Mackey-Glass混沌时间序列、Henon混沌时间序列、入口氮氧化物时间序列进行仿真,结果表明,FFOS-RKELM算法的预测精度和泛化能力都优于其他3种算法.

参考文献(References)

[1] Siniscalchi S M, Svendsen T, Lee C H. An artificial neural network approach to automatic speech processing[J]. Neurocomputing, 2014, 140: 326-338.

[2] Huang G B, Zhu Q Y, Siew C K, et al. Extreme learning machine: Theory and applications[J]. Neurocomputing, 2006, 70(1): 489-501.

[3] Liang N Y, Huang G B, Saratchandran P, et al. A fast and accurate online sequential learning algorithm for feedforward networks[J]. IEEE Trans on Neural Networks, 2006, 17(6): 1411-1423.

[4] Huang G B, Zhou H M, Ding X J, et al. Extreme learning machine for regression and multiclass classification[J]. IEEE Trans on Systems, Man, and Cybernetics, Part B: Cybernetics, 2012, 42(2): 513-529.

[5] Huynh H T, Won Y. Regularized online sequential learning algorithm for singlehidden layer feedforward neural networks[J]. Pattern Recognition Letters, 2011, 32(14): 1930-1935.

[6] Deng W Y, Ong Y S, Tan P S, et al. Online sequential reduced kernel extreme learning machine[J]. Neurocomputing, 2016, 174: 72-84.

[7] Soares S G, Araujo R. An adaptive ensemble of online extreme learning machines with variable forgetting factor for dynamic system prediction[J]. Neurocomputing, 2016, 171: 693-707.

[8] Javed K, Gouriveau R, Zerhouni N. SWELM: Asummation wavelet extreme learning machine algorithm with a priori parameter initialization[J]. Neurocomputing, 2014, 123: 299-307.

[9] 杜占龙, 李小民, 郑宗贵, 等. 基于正则化与遗忘因子的极限学习机及其在故障预测中的应用[J]. 仪器仪表学报, 2015, 36(7): 1546-1553.
(Du Z L, Li X M, Zheng Z G, et al. Extreme learning machine based on regularization and forgetting factor and its application in fault prediction[J]. Chinese J of Scientific Instrument, 2015, 36(7): 1546-1553.)

[10] You C X, Huang J Q, Lu F. Recursive reduced kernel based extreme learning machine for aeroengine fault pattern recognition[J]. Neurocomputing, 2016, 214: 1038-1045.

[11] Shao Z F, Meng Joo Er. An online sequential learning algorithm for regularized extreme learning machine[J]. Neurocomputing, 2016, 173: 778-788.

[12] Lan Y, Soh Y C, Huang G B. Ensemble of online sequential extreme learning machine[J]. Neurocomputing, 2009, 72: 3391-3395.

[13] Jerome M, Rui A, Francisco S. Adaptive fuzzy identification and predictive control for industrial processes[J]. Expert Systems with Applications, 2013, 40(17): 6964-6975.

[14] 李军, 李大超. 基于优化核极限学习机的风电功率时间序列预测[J]. 物理学报, 2016, 65(13): 39-48.
(Li J, Li D C. Wind power time series prediction based onoptimized online kernal extreme learning machine[J]. Physics J, 2016, 65(13): 39-48.)

[15] 刘伟, 王科俊, 邵克勇. 混沌时间序列的混合粒子群优化预测[J]. 控制与决策, 2007, 22(5): 562-565.
(Liu W, Wang K J, Shao K Y. Predicting chaotic time series using hybrid particle swarmoptimization algorithm[J]. Control and Decision, 2007, 22(5): 562-565.)

(责任编辑：李君玲)