# 基于替代函数及贝叶斯框架的 1 范数 ELM 算法

韩 敏 [1] 李德才 [1]

**摘 要** 针对极端学习机 (Extreme learning machine, ELM) 算法的不适定问题和模型规模控制问题, 本文提出基于 1 范数正则项的改进型 ELM 算法. 通过在二次损失函数基础上引入 1 范数正则项以控制模型规模, 改善 ELM 的泛化能力. 此外, 为简化 1 范数正则化方法的求解过程, 利用边际优化方法, 构建适当的替代函数, 以便于采用贝叶斯方法代替计算复杂的交叉检验方法, 并实现正则化参数的自适应估计. 仿真结果表明, 本文所提算法能够有效简化模型结构, 并保持较高的预测精度.

**关键词** 1 范数正则化, 极端学习机, 替代函数, 贝叶斯方法

**DOI** 10.3724/SP.J.1004.2011.01344

## An Norm 1 Regularization Term ELM Algorithm Based on Surrogate Function and Bayesian Framework

HAN Min[1] LI De-Cai[1]

**Abstract** Focusing on the ill-posed problem and the model scale control of ELM (Extreme learning machine), this paper proposes an improved ELM algorithm based on 1-norm regularization term. This is achieved by involving an 1-norm regularization term into the original square cost function, and it can be used to control the model scale and enhance the generalization capability. Furthermore, to simplify the solving process of the 1-norm regularization method, the bound optimization algorithm is employed and a suitable surrogate function is established. Based on the surrogate function, the Bayesian algorithm can be used to substitute the complicated cross validation method and estimate the regularization parameter adaptively. Simulation results illustrate that the proposed method can effectively simplify the model structure, while remaining acceptable prediction accurate.

**Key words** Norm 1 regularization, extreme learning machine (ELM), surrogate function, Bayesian method

传统的前馈神经网络大多采用基于梯度原理的训练算法, 这些方法普遍存在训练效率低、学习能力不强的问题[1−3]. 作为一种新型的网络结构, 极端学习机 (Extreme learning machine, ELM) 具有传统前馈神经网络所不具有的诸多优点[4]. 该算法对单隐层神经网络的输入权值和隐层节点偏移量进行随机赋值, 并且只通过一步计算即可解析地求出网络的输出权值. ELM 能够极大地提高网络训练速度和泛化能力, 近年来在许多研究领域都得到了广泛的关注[5−6].

对于 ELM, 通常采用伪逆算法进行求解. 如果网络的输出矩阵存在不适定问题, 奇异值的幅值分布较为连续, 没有明显的跳跃, 并且最小奇异值非常接近于零, 将会得到幅值很大的输出权值向量 $\boldsymbol{w}$ (很容易达到 10E+6 的数量级), 从而易出现过拟合现象, 影响 ELM 的建模和预测能力[7]. 另一方面, 在保证模型精度的前提下, 确定最优的模型规模是提高网络泛化能力的有效方法之一. 相对于传统的前馈网络, 由于 ELM 采用较高维数的网络结构, 如何有效控制其模型规模显得尤为重要. 在 ELM 中, 隐层节点数是唯一需要人为确定的参数, 然而对于如何选择最优的隐层节点, 目前仍未见有效的方法. Huang 等[8−9] 尝试采用增量式算法自适应选择网络的隐层节点, 在每一次迭代过程中随机生成增加的隐层节点, 并调整其所对应的输出权值. 这种方法虽实现简单, 但最终确定的网络结构往往规模庞大, 在取得相同测试精度的条件下, 所需的隐层节点数要远远多于通过交叉检验方法所确定的节点.

为提高 ELM 算法的性能, 可以考虑引入正则化方法, 以解决输出矩阵的不适定问题. 其借助正则项消除原线性回归方法中出现的病态问题, 抑制较大输出权值的产生, 提高 ELM 的建模和预测能力[10]. 其中 1 范数正则化方法是一种常用的惩罚算法, 通过选择适当的正则项系数可以有效控制模型部分无关节点的输出权值趋近于零, 从而达到简化模型结构的目的. 然而, 由于 1 范数正则项在原点处不可微, 使算法求解过程变得复杂, 而且, 正则化系数也需要通过交叉检验或 Bootstrap 方法确定, 计算量较大.

收稿日期 2010-08-30 录用日期 2010-12-27
Manuscript received August 30, 2010; accepted December 27, 2010
国家自然科学基金 (61074096) 资助
Supported by National Natural Science Foundation of China (61074096)
1. 大连理工大学电子信息与电气工程学部 大连 116023
1. Faculty of Electronic Information and Electrical Engineering, Dalian University of Technology, Dalian 116023

针对上述问题, 本文提出基于替代函数及贝叶斯框架的 1 范数 ELM 算法 (以下简称为 N1-ELM). 借以简化 1 范数 ELM 算法的求解过程, 提高 ELM 学习方法的效率和预测性能. 其基本思想为: 在二次损失函数基础上引入 1 范数正则项, 替代原有线性回归方法. 并利用边际最优方法, 选择适当的替代函数代替包含 1 范数正则项的损失函数, 使其便于求解. 在此基础上, 通过对损失函数进行等价变换, 并采用贝叶斯方法对正则化参数进行自适应估计. 在此过程中, 由于正则项的抑制作用, 部分对网络影响较小的权值将逐渐趋近于零, 从而实现对 ELM 隐层节点的自适应选择.

## 1　基于 1 范数正则化的极端学习机

以单输出的 ELM 网络为例, 对于任意 $N$ 个不同的训练样本 $(\boldsymbol{x}_i,\ t_i)$, 其中 $\boldsymbol{x}_i=[x_{1i},x_{2i},\cdots,x_{ni}]$, 如果网络的隐层节点数为 $L$, 激活函数为 $g(x)$, 则相应 ELM 的数学模型可以表示为

$$\sum_{j=1}^{L} w_j g\left(\tilde{\boldsymbol{w}}_j^{\mathrm{T}}\boldsymbol{x}_i+b_j\right)=t_i,\quad i=1,\cdots,N \tag{1}$$

其中, $\tilde{\boldsymbol{w}}_j=[\tilde{w}_{j1},\tilde{w}_{j2},\cdots,\tilde{w}_{jn}]^{\mathrm{T}}$ 为输入节点同第 $j$ 个隐含层节点之间的连接权值, $w_j$ 为第 $j$ 个隐层节点同输出节点的权值, $b_j$ 是第 $j$ 个隐含层节点的阈值. 其中 $\tilde{\boldsymbol{w}}_j$ 和 $b_j$ 随机生成, 且在训练过程中保持不变. 因此, 输出权值 $\boldsymbol{w}$ 为 ELM 中唯一需要确定的参数. 对式 (1) 中的 $N$ 个方程式进行整理, 可得

$$\Phi\boldsymbol{w}=\boldsymbol{t} \tag{2}$$

其中,

$$\Phi\left(\tilde{\boldsymbol{w}}_1,\cdots,\tilde{\boldsymbol{w}}_L,b_1,\cdots,b_L,\boldsymbol{x}_1,\cdots,\boldsymbol{x}_N\right)=\begin{bmatrix} g(\tilde{\boldsymbol{w}}_1^{\mathrm{T}}\boldsymbol{x}_1+b_1) & \cdots & g(\tilde{\boldsymbol{w}}_L^{\mathrm{T}}\boldsymbol{x}_1+b_L)\\ \vdots & \ddots & \vdots\\ g(\tilde{\boldsymbol{w}}_1^{\mathrm{T}}\boldsymbol{x}_N+b_1) & \cdots & g(\tilde{\boldsymbol{w}}_L^{\mathrm{T}}\boldsymbol{x}_N+b_L)\end{bmatrix}_{N\times L} \tag{3}$$

$$\boldsymbol{w}=\begin{bmatrix} w_1^{\mathrm{T}}\\ \vdots\\ w_L^{\mathrm{T}}\end{bmatrix}_{L\times 1}\qquad \boldsymbol{t}=\begin{bmatrix} t_1^{\mathrm{T}}\\ \vdots\\ t_N^{\mathrm{T}}\end{bmatrix}_{N\times 1} \tag{4}$$

通常可以采用伪逆算法计算输出权值, 即

$$\boldsymbol{w}=\Phi^{\dagger}\boldsymbol{t} \tag{5}$$

伪逆方法实现简单, 使 ELM 在计算效率方面较常规的前馈神经网络具有较明显的优势. 然而, 如果网络的输出矩阵 $\Phi$ 存在不适定问题, 则易导致较大幅值的输出权值出现以及过拟合现象发生, 影响网络的泛化能力.

根据传统神经网络的建模方法, 在目标函数中引入正则项可以有效防止过拟合现象的发生. 相应的目标函数 $L(\boldsymbol{w})$ 可以表示为

$$L\left(\boldsymbol{w}\right)=E_D\left(\boldsymbol{w}\right)+\lambda E_W\left(\boldsymbol{w}\right) \tag{6}$$

其中 $\lambda$ 为正则项系数, 控制着误差项 $E_D(\boldsymbol{w})$ 和正则化项 $E_W(\boldsymbol{w})$ 的相对重要程度. 通常, 误差项 $E_D(\boldsymbol{w})$ 表示为

$$E_D\left(\boldsymbol{w}\right)=\frac{1}{2}||\boldsymbol{t}-\Phi\boldsymbol{w}||^2 \tag{7}$$

正则项根据需要处理的问题不同, 通常可分别选择 1 范数正则化方法或 2 范数正则化方法. 其中 1 范数正则化方法, 又称为 LASSO, 是一种较为常用的正则化方法:

$$E_W\left(\boldsymbol{w}\right)=\frac{1}{2}||\boldsymbol{w}||_1 \tag{8}$$

如果正则项系数 $\lambda$ 足够大, 1 范数正则化方法可以有效控制模型部分无关节点的输出权值趋近于零, 从而达到简化模型结构的目的[11].

因此, 同时考虑输出矩阵的不适定性和模型复杂度控制两方面的问题, 本文采用 1 范数正则化方法尝试对原 ELM 算法进行优化.

## 2　基于替代函数的模型转化

对于 1 范数正则化方法, 由于绝对值的存在, 通常难以得到其解析解, 为解决此问题可以引入边际优化方法[12], 通过选择适当的替代函数, 将其转化为易于求解的形式.

根据边际优化方法, 最大化目标函数 $L(\boldsymbol{w})$ 可以等价于迭代最大化相应的替代函数 $Q$, 即

$$\hat{\boldsymbol{w}}^{(t+1)}=\arg\max_{\boldsymbol{w}} Q\left(\boldsymbol{w}|\hat{\boldsymbol{w}}^{(t)}\right) \tag{9}$$

如果替代函数满足: $L\left(\boldsymbol{w}\right)-Q\left(\boldsymbol{w}|\hat{\boldsymbol{w}}^{(t)}\right)$ 在 $\boldsymbol{w}=\hat{\boldsymbol{w}}^{(t)}$ 处取得最小值, 则可保证目标函数 $L(\boldsymbol{w})$ 在整个迭代过程中单调递增.

为构造合适的替代函数, 首先考虑 1 范数正则项的下边界:

$$-||\boldsymbol{w}||_1\geq-\frac{1}{2}\left(\sum_i\frac{w_i^2}{|\hat{w}_i|}+\sum_i|\hat{w}_i|\right) \tag{10}$$

其中, 当且仅当 $\boldsymbol{w}=\hat{\boldsymbol{w}}$ 时, 不等式左右两边相等.

$L\left(\boldsymbol{w}\right)=-E_D\left(\boldsymbol{w}\right)-||\boldsymbol{w}||_1\geq$

$$-E_D(\boldsymbol{w})-\frac{1}{2}\left(\sum_i\frac{w_i^2}{|\hat{w}_i|}+\sum_i|\hat{w}_i|\right) \tag{11}$$

令不等式右端项为 $Q(\boldsymbol{w}|\hat{\boldsymbol{w}})$, 有

$$L(\boldsymbol{w})-Q(\boldsymbol{w}|\hat{\boldsymbol{w}})\geq 0 \tag{12}$$

其中, 当且仅当 $\boldsymbol{w}=\hat{\boldsymbol{w}}$ 时, 不等式左右两边相等. 因此, $Q(\boldsymbol{w}|\hat{\boldsymbol{w}})$ 满足替代函数条件, 可以通过最大化替代函数等价得到原目标函数的最优解. 鉴于此, 舍弃 $Q(\boldsymbol{w}|\hat{\boldsymbol{w}})$ 中与输出权值 $\boldsymbol{w}$ 无关的项, 得到替代函数形式为

$$Q\left(\boldsymbol{w}|\hat{\boldsymbol{w}}^{(t)}\right)=-\frac{1}{2}\sum_{n=1}^{N}||\boldsymbol{t}-\Phi\boldsymbol{w}||^2-\frac{\lambda}{2}\boldsymbol{w}^{\mathrm{T}}\Lambda^{(t)}\boldsymbol{w} \tag{13}$$

其中

$$\Lambda^{(t)}=\mathrm{diag}\left\{|\hat{w}_1^{(t)}|^{-1},|\hat{w}_2^{(t)}|^{-1},\cdots,|\hat{w}_L^{(t)}|^{-1}\right\} \tag{14}$$

对比 $L$ 和 $Q$ 可以看出, 替代函数 $Q$ 等价于 2 范数正则化问题, 相比于原目标函数 $L$ 的 1 范数正则项更易求解. 本文中将采用贝叶斯方法实现对模型输出权值 $\boldsymbol{w}$ 的自适应估计.

另一方面, 最大化原目标函数 $L$ 等价于迭代优化替代函数 $Q$, 且在迭代过程中 $Q(\boldsymbol{w}|\hat{\boldsymbol{w}})$ 将保持单调递增. 此外, 由于式 (13) 为凹函数, 必存在唯一极值. 因此, 在迭代过程中, 输出权值向量 $\hat{\boldsymbol{w}}^{(t+1)}$ 的初值 $\hat{\boldsymbol{w}}^{(0)}$ 可以随机选取.

## 3 贝叶斯框架下的 1范数正则项求解方法

### 3.1 等价变换

如前所述, 正则化方法能够有效解决输出矩阵的不适定问题, 改善模型求解的性质. 其中正则化参数控制着模型拟合精度和结构复杂度的相对重要性, 因此, 正则化参数的正确选择在一定意义上决定着正则化方法的有效性.

对于正则化系数, 通常需要通过交叉检验或者 Bootstrap 方法确定, 计算量较大. 贝叶斯方法是一种基于统计理论的有效参数估计方法, 其在证据框架下, 通过最大化目标似然函数实现模型参数的自适应估计. 然而, 对于 1 范数正则项下的目标函数, 直接采用贝叶斯方法进行求解比较困难. 从式 (13) 中可以看出, 通过替代函数的作用, 可以将 1 范数正则项转化为易于求解的 2 范数形式. 由边际最优理论可知, 对替代函数的优化问题等价于对原始目标函数的优化, 因此, 可以利用贝叶斯方法求解式 (13), 进而自适应确定正则化参数 $\lambda$ 以及对应的输出权值 $\boldsymbol{w}$.

为采用贝叶斯方法对式 (13) 进行求解, 需要首先进行如下变换. 令

$$\Lambda'=\left(\Lambda^{(t)}\right)^{\frac{1}{2}} \tag{15}$$

式 (13) 可以等价表示为

$$Q\left(\boldsymbol{w}|\hat{\boldsymbol{w}}^{(t)}\right)=-\frac{1}{2}\sum_{n=1}^{N}||\boldsymbol{t}-\Phi\Lambda'^{-1}\Lambda'\boldsymbol{w}||^2-\frac{\lambda}{2}\boldsymbol{w}^{\mathrm{T}}\Lambda'\Lambda'\boldsymbol{w} \tag{16}$$

再令

$$\boldsymbol{w}'=\Lambda'\boldsymbol{w} \tag{17}$$

$$\Phi'=\Phi\Lambda'^{-1} \tag{18}$$

将式 (17) 和式 (18) 代入式 (16), 可得

$$Q\left(\boldsymbol{w}'|\boldsymbol{w}'^{(t)}\right)=-\frac{1}{2}\sum_{n=1}^{N}||\boldsymbol{t}-\Phi'\boldsymbol{w}'||^2-\frac{\lambda}{2}\boldsymbol{w}'^{\mathrm{T}}\boldsymbol{w}' \tag{19}$$

对于式 (19), 可根据训练样本, 由贝叶斯方法迭代估计输出权值 $\boldsymbol{w}$ 以及正则化参数 $\lambda$.

### 3.2 参数求解

假设模型输出 $\boldsymbol{t}$ 由某种包含零均值高斯噪声的函数产生, 且各数据满足独立同分布, 则其先验分布可以定义为

$$p(\boldsymbol{t}|\boldsymbol{w}',\beta)=\left(\frac{\beta}{2\pi}\right)^{\frac{N}{2}}\exp\left\{-\frac{\beta}{2}||\boldsymbol{t}-\Phi'\boldsymbol{w}'||^2\right\} \tag{20}$$

对于网络的输出权值 $\boldsymbol{w}'$, 在没有先验知识的情况下, 可以假设 $p(\boldsymbol{w}')$ 服从指数型分布, 而其中最为常见的是高斯分布, 因此可以定义输出权值的先验分布为

$$p(\boldsymbol{w}'|\alpha)=\left(\frac{\alpha}{2\pi}\right)^{\frac{L}{2}}\exp\left\{-\frac{\alpha}{2}\boldsymbol{w}'^{\mathrm{T}}\boldsymbol{w}'\right\} \tag{21}$$

其中, $\alpha$ 和 $\beta$ 为超参数, 控制对应分布的方差.

对于输出权值 $\boldsymbol{w}'$, 其后验概率可以表示为

$$p(\boldsymbol{w}'|\boldsymbol{t})=\frac{p(\boldsymbol{t}|\boldsymbol{w}',\beta)\,p(\boldsymbol{w}'|\alpha)}{p(\boldsymbol{t})} \tag{22}$$

其中, $p(\boldsymbol{t})=\int p(\boldsymbol{t}|\boldsymbol{w}',\beta)\,p(\boldsymbol{w}'|\alpha)\mathrm{d}\boldsymbol{w}'$

将式 (20) 和式 (21) 代入式 (22), 可得

$$p(\boldsymbol{w}'|\boldsymbol{t})=\frac{1}{Z_L}\exp\left(-M(\boldsymbol{w}')\right) \tag{23}$$

其中

$$M(\boldsymbol{w}')=\frac{\beta}{2}||\boldsymbol{t}-\Phi'\boldsymbol{w}'||^2+\frac{\alpha}{2}\boldsymbol{w}'^{\mathrm{T}}\boldsymbol{w}' \tag{24}$$

$$Z_L = \int \exp\left\{-M\left(\boldsymbol{w}'\right)\right\}\mathrm{d}\boldsymbol{w}' \tag{25}$$

对式 (23) 取对数, 可得 $p(\boldsymbol{w}|\boldsymbol{t})$ 的对数似然函数为

$$\ln p\left(\boldsymbol{w}'|\boldsymbol{t}\right) = -\frac{\beta}{2}||\boldsymbol{t} - \Phi'\boldsymbol{w}'||^2 - \frac{\alpha}{2}\boldsymbol{w}'^{\mathrm{T}}\boldsymbol{w}' + \mathrm{const} \tag{26}$$

固定参数 $\alpha$ 和 $\beta$, 最大化 $\ln p\left(\boldsymbol{w}'|\boldsymbol{t}\right)$ 可得到输出权值 $\boldsymbol{w}'$ 为

$$\boldsymbol{w}' = \beta\left(\alpha\boldsymbol{I} + \beta\Phi'^{\mathrm{T}}\Phi'\right)^{-1}\Phi'^{\mathrm{T}}\boldsymbol{t} \tag{27}$$

比较式 (19) 和式 (26) 可知, 最大化似然函数 (26) 等价于最大化相应的替代函数 $Q\left(\boldsymbol{w}'|\hat{\boldsymbol{w}}'^{(t)}\right)$, 如式 (19) 所示, 其中 $\lambda$=$\alpha/\beta$. 因此, 原目标函数中权值向量 $\boldsymbol{w}$ 的最优解可以表示为

$$\hat{\boldsymbol{w}}^{(t+1)} = \left(\Lambda'\right)^{-1}\boldsymbol{w}' \tag{28}$$

对于参数 $\alpha$ 和 $\beta$, 可以根据证据函数方法确定. 通过对输出权值 $\boldsymbol{w}$ 积分, 可得 $p\left(\boldsymbol{t}|\alpha,\beta\right)$ 的边际似然函数为

$$p\left(\boldsymbol{t}|\alpha,\beta\right) = \int p\left(\boldsymbol{t}|\boldsymbol{w}',\beta\right)p\left(\boldsymbol{w}'|\alpha\right)\mathrm{d}\boldsymbol{w}' \tag{29}$$

代入式 (20) 和式 (21), 进而可以得到边际似然函数 $p\left(\boldsymbol{t}|\alpha,\beta\right)$ 的对数表达式为

$$\ln p\left(\boldsymbol{t}|\alpha,\beta\right) = \frac{L}{2}\ln\alpha + \frac{N}{2}\ln\beta - \frac{\beta}{2}||\boldsymbol{t} - \Phi'\boldsymbol{w}'||^2 - \frac{\alpha}{2}\boldsymbol{w}'^{\mathrm{T}}\boldsymbol{w}' - \frac{1}{2}\ln|\boldsymbol{A}| - \frac{N}{2}\ln\left(2\pi\right) \tag{30}$$

其中

$$\boldsymbol{A} = \alpha\boldsymbol{I} + \beta\Phi'^{\mathrm{T}}\Phi' \tag{31}$$

令对数似然函数最大化, 可以得到参数 $\alpha$ 和 $\beta$ 的估计式分别为

$$\alpha = \frac{\gamma}{\boldsymbol{w}'^{\mathrm{T}}\boldsymbol{w}'} \tag{32}$$

$$\beta = \frac{N - \gamma}{||\boldsymbol{t} - \Phi'\boldsymbol{w}'||^2} \tag{33}$$

其中,

$$\gamma = \sum_{i=1}^{L}\frac{\lambda_i}{\alpha + \lambda_i} \tag{34}$$

$\lambda$ 为矩阵 $\beta\Phi'^{\mathrm{T}}\Phi'$ 的特征值. 由于式 (32) 和式 (33) 分别为参数 $\alpha$ 和 $\beta$ 的隐式解, 需要利用迭代方式进行求解. 即随机给出 $\alpha$ 和 $\beta$ 的一组初始值, 根据式 (27) 和式 (34) 分别得到变量 $\boldsymbol{w}'$ 和 $\gamma$. 再由式 (32) 和式 (33) 计算 $\beta\Phi'^{\mathrm{T}}\Phi'$ 的修正值, 如此反复迭代, 直至参数 $\beta\Phi'^{\mathrm{T}}\Phi'$ 收敛为止. 而在每一次迭代过程中, 变量 $\boldsymbol{w}'$ 和 $\gamma$ 的估计式中所对应的超参数 $\beta\Phi'^{\mathrm{T}}\Phi'$ 均为前一次迭代结束时所得到的估计值.

如果定义变量 $k$ 表示迭代次数, 而 $\alpha^k$, $\beta^k$ 和 $\alpha^{k-1}$, $\beta^{k-1}$ 分别为第 $k$ 次和 $k-1$ 次迭代过程中超参数 $\alpha$ 和 $\beta$ 的估计值, 则前述 $\alpha$ 和 $\beta$ 的计算过程可以具体表示为:

**步骤 1.** 令 $k = 1$, 并随机给出超参数 $\alpha$ 和 $\beta$ 的一组初始值 $\beta^0$ 和 $\beta^0$;

**步骤 2.** 根据式 (27) 和式 (34) 分别计算变量 $\boldsymbol{w}'$ 和 $\lambda$ 的估计值;

**步骤 3.** 由式 (32) 和式 (33) 对超参数 $\alpha$ 和 $\beta$ 的取值进行更新;

**步骤 4.** 若达到最大迭代次数或者超参数 $\alpha$ 和 $\beta$ 收敛, 则停止迭代操作, 否则 $k = k + 1$, 转到步骤 2, 并进入下一次迭代过程.

### 3.3 N1-ELM 方法的实现步骤

综上所述, 基于替代函数和贝叶斯框架的改进型极端学习机的实现步骤可以表示如下:

**步骤 1.** 根据式 (10) ∼ (13), 将原目标函数 $L$ 转化为替代函数 $Q$. 令 $t = 0$, 随机选取模型输出权值 $\boldsymbol{w}$ 在迭代过程中的初始值 $\hat{\boldsymbol{w}}^{(t)}$, 相应有:

$$\Lambda^{(t)} = \mathrm{diag}\left\{|\hat{w}_1^{(t)}|, |\hat{w}_2^{(t)}|, \cdots, |\hat{w}_L^{(t)}|\right\} \tag{35}$$

**步骤 2.** 由式 (15) ∼ (18), 将替代函数 $Q$ 等价变换为标准的 2 范数正则化问题, 如式 (19) 所示.

**步骤 3.** 采用贝叶斯方法优化替代函数, 并初始化参数 $\alpha$ 和 $\beta$.

**步骤 4.** 分别根据式 (27) 和式 (34) 计算变量 $\boldsymbol{w}'$ 及 $\gamma$.

**步骤 5.** 由式 (32) 和式 (33) 更新参数 $\alpha$ 和 $\beta$, 重复步骤 4 和 5, 至 $\alpha$ 和 $\beta$ 收敛 (如参数 $\alpha$ 和 $\beta$ 的变化量小于某一给定阈值或达到最大迭代次数).

**步骤 6.** 由式 (28) 和式 (14) 更新模型输出权值向量 $\hat{\boldsymbol{w}}^{(t+1)}$、对角阵 $\Lambda^{(t+1)}$ 及其对应的替代函数, $t = t + 1$. 如果输出权值向量 $\boldsymbol{w}$ 收敛 (如 $\boldsymbol{w}$ 的变化量小于某一给定阈值或达到最大迭代次数), 转至步骤 7, 否则转至步骤 3.

**步骤 7.** 如果 $\boldsymbol{w}_i'^{(t+1)}$, $i = 1, 2, \cdots, L$, 小于某一给定阈值如 10E−3, 则认为对应的节点对模型贡献较小, 可以将其移除.

## 4 仿真实例

### 4.1 SinC 函数

采用改进的极端学习机方法拟合如下方程:

$$y(x) = \begin{cases} \dfrac{\sin(x)}{x}, & x \neq 0 \\ 1, & x = 0 \end{cases} \tag{36}$$

其中, 输入变量 $x$ 服从 $(-10, 10)$ 的均匀分布. 由式 (36) 采集 400 组样本, 前 200 组样本用于训练, 后 200 组样本用于模型检验. 为体现实验条件的真实性, 在训练样本上叠加均值为 0、标准差为 0.2 的高斯噪声.

初始选择隐层节点数为 50, 根据第 3.3 节中步骤 1 ∼ 6 对 ELM 网络进行训练, 经过 25 次迭代之后输出权值 $\boldsymbol{w}$ 收敛, 图 1 为对 SinC 函数的逼近效果, 相应各输出权值的分布情况如图 2 所示. 从图 2 中可以看出, 由于 1 范数正则项的作用, 大部分输出权值接近于零, 如果以 10E−3 作为阈值, 则最终有效的隐层节点个数为 24, 在保证模型精度的条件下, 有效简化了 ELM 网络的结构. 此外, 正则项的存在改善了输出矩阵的自身特性, 有效抑制了较大输出权值的出现, 提高了网络的建模和预测能力. 在隐层节点数同为 50 的情况下, ELM 算法及本文所提 N1-ELM 算法的性能比较结果如表 1 所示.

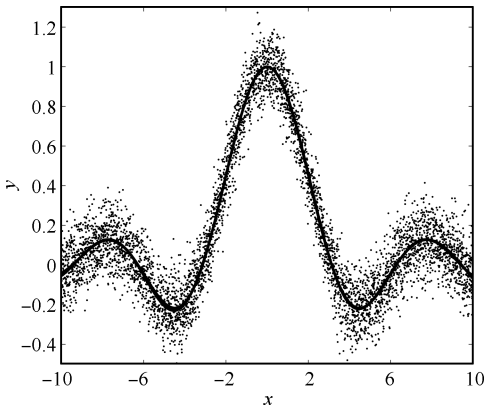



图 1 N1-ELM 算法对 SinC 函数的逼近结果

Fig. 1 Approximation result of N1-ELM to SinC function

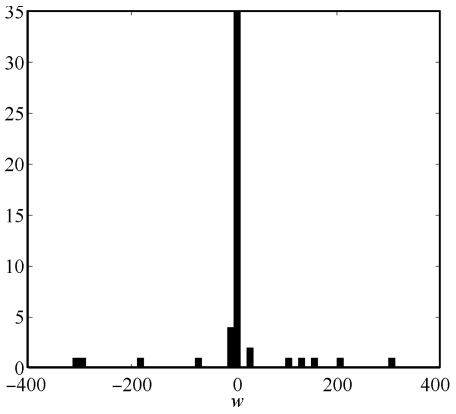



图 2 N1-ELM 算法中各输出权值的分布情况

Fig. 2 Distribution of output weights of N1-ELM

为进一步验证所提算法的有效性, 分别对文献 [4] 所采用的 ELM 算法、BP 算法以及支持向量回归 (Support vector regression, SVR) 进行仿真, 并分别从训练时间、训练误差 (Mean square error, MSE)、测试误差以及模型规模等方面进行比较. 其中每组实验进行 50 次, 最终结果为 50 次实验的平均值, 相应的仿真结果如表 2 所示.

表 1 ELM 算法同 N1-ELM 算法的比较结果

Table 1 Comparison of ELM and N1-ELM

| 方法 | $\|\|\boldsymbol{w}\|\|_2$ | 训练误差 | 测试误差 |
|---|---|---|---|
| ELM | $2.94 \times 10^6$ | 0.1148 | 0.0106 |
| N1-ELM | $6.48 \times 10^2$ | 0.0996 | 0.0057 |

表 2 ELM, BP 及 SVR 算法同 N1-ELM 算法的比较结果

Table 2 Comparison of ELM, BP, SVR, and N1-ELM

| 方法 | 时间 | 训练误差 | 测试误差 | 模型规模 |
|---|---|---|---|---|
| ELM | 0.125 | 0.1148 (0.0037) | 0.0097 (0.0028) | 20 |
| BP | 21.26 | 0.1196 (0.0042) | 0.0159 (0.0041) | 20 |
| SVR | 1 273.4 | 0.1149 (0.0007) | 0.0130 (0.0012) | 2 499.9 |
| N1-ELM | 2.5031 | 0.0994 (0.0010) | 0.0059 (0.0011) | 25.65 |

从表 2 中可以看出, 由于正则化方法消除了原线性回归中的病态问题, 因此, 较 ELM 算法, 本文提出的 N1-ELM 算法在泛化能力上具有明显提升. 同其他三种方法相比较, N1-ELM 算法具有更小的训练和测试误差, 且结果更为稳定. 在模型规模方面, 虽然 N1-ELM 算法初始化隐层节点数为 50, 但在 1 范数正则项的作用下, 模型最终包含的隐层节点个数平均为 25.65, 接近于 ELM 算法通过交叉检验确定的最优隐层节点数, 从而说明本文所提方法在提升网络性能的同时, 能够有效简化模型结构. 在训练时间上, 同 BP 算法及 SVR 相比, N1-ELM 算法具有较为明显的优势.

虽然较 ELM, N1-ELM 算法更为耗时, 但模型具有更好的泛化效果, 而且由于模型规模可通过贝叶斯方法自适应确定, 无需引入计算量较大的 Bootstrap 或交叉检验方法进行确定, 因此, 在计算效率方面仍具有一定的竞争力.

### 4.2 标杆数据

为验证所提 N1-ELM 算法的有效性, 分别与 EI-ELM[8] 及 I-ELM[9] 两种 ELM 的改进算法进行比较, 两者采用增量式方法选择 ELM 网络的隐层节点. 为保证比较结果的可靠性, 选择 11 组标杆数据作为仿真对象, 其具体描述如表 3 所示.

根据文献 [8], 将输入数据归一化到区间 $[-1, 1]$, 相应的输出数据归一化至区间 $[0, 1]$. 在三种方法中, ELM 网络均选择 Sigmoid 函数作为激活函数. 每组实验进行 20 次, 最终结果为 20 次实验的平均值. 每组实验中, 训练样本和测试样本根据表 3 随机

划分, 测试误差采用均方根误差 (Root mean square error, RMSE) 进行度量.

表 3 各组标杆数据的具体描述

Table 3 Detail description of the datasets

| 数据集 | 训练样本数 | 测试样本数 | 输入特征数 |
|---|---|---|---|
| Abalone | 2 000 | 2 177 | 8 |
| Ailerons | 7 154 | 6 596 | 39 |
| Bank | 4 500 | 3 691 | 8 |
| California | 8 000 | 12 640 | 8 |
| Computer activity | 4 000 | 4 192 | 12 |
| Census (8 L) | 10 000 | 12 784 | 8 |
| Delta ailerons | 3 000 | 4 129 | 5 |
| Delta elevators | 4 000 | 5 517 | 6 |
| Kinematics | 4 000 | 4 129 | 8 |
| Puma | 4 500 | 3 692 | 8 |
| Pyrim | 80 | 87 | 4 |

表 4 EI-ELM 及 N1-ELM 算法在标杆各数据上的比较结果

Table 4 Comparison of EI-ELM and N1-ELM

| 数据集 | EI-ELM ($k = 20$) | | N1-ELM | |
|---|---|---|---|---|
| | 测试误差 (标准差) | 训练时间 (s) | 测试误差 (标准差) | 训练时间 (s) |
| Abalone | 0.0876 (0.0015) | 1.5785 | 0.0770 (0.0014) | 1.1594 |
| Ailerons | 0.0571 (0.0022) | 6.2519 | 0.0461 (0.0006) | 3.0484 |
| Bank | 0.0896 (0.0036) | 3.1058 | 0.0504 (0.0020) | 1.7344 |
| California | 0.1548 (0.0033) | 4.9486 | 0.1333 (0.0020) | 3.4063 |
| Computer activity | 0.0991 (0.0036) | 2.3311 | 0.0463 (0.0046) | 1.8125 |
| Census (8 L) | 0.0865 (0.0011) | 6.1100 | 0.0705 (0.0020) | 4.2962 |
| Delta ailerons | 0.0467 (0.0042) | 1.4570 | 0.0390 (0.0008) | 1.8938 |
| Delta elevators | 0.0586 (0.0038) | 2.5385 | 0.0534 (0.0004) | 2.5562 |
| Kinematics | 0.1416 (0.0019) | 2.9017 | 0.1266 (0.0029) | 1.5516 |
| Puma | 0.1827 (0.0017) | 2.7264 | 0.1790 (0.0020) | 2.6102 |
| Pyrim | 0.1300 (0.0405) | 0.1533 | 0.1293 (0.0271) | 0.1383 |

表 4 和表 5 分别比较了 EI-ELM, I-ELM 以及 N1-ELM 三种算法在 11 组标杆数据上的仿真结果. 根据文献 [8] 的建议, 在 EI-ELM 算法中, 每次迭代所参考的候选节点个数 $k$ 选为 20, 相应各算法对应的模型规模如表 6 所示.

表 5 I-ELM 及 N1-ELM 算法在标杆各数据上的比较结果

Table 5 Comparison of I-ELM and N1-ELM

| 数据集 | I-ELM | | N1-ELM | |
|---|---|---|---|---|
| | 测试误差 (标准差) | 训练时间 (s) | 测试误差 (标准差) | 训练时间 (s) |
| Abalone | 0.0876 (0.0033) | 0.7695 | 0.0770 (0.0014) | 1.1594 |
| Ailerons | 0.0824 (0.0232) | 1.8810 | 0.0461 (0.0006) | 3.0484 |
| Bank | 0.0757 (0.0032) | 0.7914 | 0.0504 (0.0020) | 1.7344 |
| California | 0.1543 (0.0019) | 1.5665 | 0.1333 (0.0020) | 3.4063 |
| Computer activity | 0.1057 (0.0078) | 0.7185 | 0.0463 (0.0046) | 1.8125 |
| Census (8 L) | 0.0871 (0.0018) | 2.1199 | 0.0705 (0.0020) | 4.2962 |
| Delta ailerons | 0.0468 (0.0052) | 0.6340 | 0.0390 (0.0008) | 1.8938 |
| Delta elevators | 0.0640 (0.0055) | 0.6516 | 0.0534 (0.0004) | 2.5562 |
| Kinematics | 0.1406 (0.0014) | 0.7117 | 0.1266 (0.0029) | 1.5516 |
| Puma | 0.1856 (0.0039) | 0.7983 | 0.1790 (0.0020) | 2.6102 |
| Pyrim | 0.1712 (0.0626) | 0.0810 | 0.1293 (0.0271) | 0.1383 |

从表 4 和表 5 中可以看出, 与 EI-ELM 及 I-ELM 算法相比, 本文所提 N1-ELM 算法有效提升了 ELM 算法的性能, 具有较好的泛化能力. 另一方面, 为得到相近的预测精度, EI-ELM 及 I-ELM 算法所需网络隐层节点数要明显多于 N1-ELM 算法. 以 Delta elevators 数据为例, 虽然 N1-ELM 算法平均采用 20.25 个隐层节点构建网络模型, 但其测试误差却要低于 EI-ELM 及 I-ELM 算法分别为 50 和 500 个隐层节点时所对应的模型结构. 因此, 本文所提方法能够在保证测试精度的基础上, 有效简化模型结构. 在训练时间方面, N1-ELM 算法同 EI-ELM 算法相近, 而略高于 I-ELM 算法, 其运算时间主要消耗在模型规模的简化, 即输出权值向量 $\boldsymbol{w}$ 的迭代估计上. 但若仅考虑预测精度, 在训练时间上, N1-ELM 算法同 I-ELM 算法仍具有可比性. 以 Abalone 数据为例, 当输出权值向量 $\boldsymbol{w}$ 的迭代次数由 15 减为 3 时, 相应的模型隐层节点数由均值 33.45 提高到 43.7, 测试误差为 0.0772, 几乎保持不变, 但训练时间由 1.1594 s 减低到 0.7406 s, 接近于 I-ELM 算法.

表 6 EI-ELM, I-ELM 及 N1-ELM 在模型规模上的比较结果

Table 6 Comparison of EI-ELM, I-ELM, and N1-ELM on model scale

| 数据集 | EI-ELM ($k = 20$) 模型规模 | I-ELM 模型规模 | N1-ELM 模型规模 |
|---|---|---|---|
| Abalone | 50 | 500 | 33.45 |
| Ailerons | 50 | 500 | 45.20 |
| Bank | 50 | 500 | 49.45 |
| California | 50 | 500 | 49.95 |
| Computer activity | 50 | 500 | 49.90 |
| Census (8 L) | 50 | 500 | 50.00 |
| Delta ailerons | 50 | 500 | 39.95 |
| Delta elevators | 50 | 500 | 20.25 |
| Kinematics | 50 | 500 | 48.95 |
| Puma | 50 | 500 | 39.60 |
| Pyrim | 50 | 500 | 18.05 |

## 5 结论

针对 ELM 算法的不适定性问题和模型规模控制问题, 本文提出基于 1 范数正则化的改进型 ELM 算法, 并通过 SinC 函数及 11 组标杆数据验证了所提方法的有效性. 仿真结果表明, 1 范数正则项的引入有效抑制了较大输出权值的出现以及过拟合现象的发生, 并在保持较高预测精度的同时, 能够有效简化模型结构. 另一方面, 通过边际优化方法, 可以简化 1 范数正则项的求解形式, 使贝叶斯方法能够应用于对模型参数的自适应估计, 从而避免了计算量较大的 Bootstrap 或交叉检验过程.

## References


1 Meng Zu-Qiang, Cai Zi-Xing. Identification method of nonlinear systems based on parallel genetic algorithm. *Control and Decision*, 2003, **18**(3): 367−370
(蒙祖强, 蔡自兴. 一种基于并行遗传算法的非线性系统辨识方法. 控制与决策, 2003, **18**(3): 367−370)

2 Qiao Jun-Fei, Han Hong-Gui. Optimal structure design for RBFNN structure. *Acta Automatica Sinica*, 2010, **36**(6): 865−872
(乔俊飞, 韩红桂. RBF 神经网络的结构动态优化设计. 自动化学报, 2010, **36**(6): 865−872)

3 Ye Jian, Ge Lin-Dong, Wu Yue-Xian. An application of improved RBF neural network in modulation recognition. *Acta Automatica Sinica*, 2007, **33**(6): 652−654
(叶健, 葛临东, 吴月娴. 一种优化的 RBF 神经网络在调制识别中的应用. 自动化学报, 2007, **33**(6): 652−654)

4 Huang G B, Zhu Q Y, Siew C K. Extreme learning machine: theory and applications. *Neurocomputing*, 2006, **70**(1−3): 489−501

5 Malathi V, Marimuthu N S, Baskar S. Intelligent approaches using support vector machine and extreme learning machine for transmission line protection. *Neurocomputing*, 2010, **73**(10−12): 2160−2167

6 Minhas R, Baradarani A, Seifzadeh S, Wu Q M J. Human action recognition using extreme learning machine based on visual vocabularies. *Neurocomputing*, 2010, **73**(10−12): 1906−1917

7 Tang X L, Han M. Partial Lanczos extreme learning machine for single output regression problems. *Neurocomputing*, 2009, **72**(13−15): 3066−3076

8 Huang G B, Chen L. Enhanced random search based incremental extreme learning machine. *Neurocomputing*, 2008, **71**(16−18): 3460−3468

9 Huang G B, Chen L, Siew C K. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Transactions on Neural Network*, 2006, **17**(4): 879−892

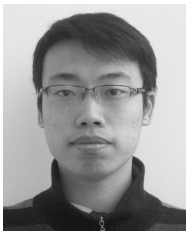
10 Hansen P C. *Rank-deficient and Discrete Ill–posed Problems: Numerical Aspects of Linear Inversion*. Philadelphia: SIAM, 1998. 45−68

11 Kaban A. On Bayesian classification with Laplace priors. *Pattern Recognition Letters*, 2007, **28**(10): 1271−1282

12 Krishnapuram B, Carin L, Figueiredo M A T, Hartemink A J. Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005, **27**(6): 957−968



**韩 敏** 大连理工大学教授. 主要研究方向为神经网络理论及其应用, 复杂系统建模及自适应控制. 本文通信作者.
E-mail: minhan@dlut.edu.cn
(**HAN Min** Professor at Dalian University of Technology. Her research interest covers neural networks theory and application, complex systems modeling, and adaptive control. Corresponding author of this paper.)


**李德才** 大连理工大学博士研究生. 主要研究方向为基于多变量时间序列的复杂系统建模及预测研究.
E-mail: ldcai@mail.clut.edu.cn
(**LI De-Cai** Ph. D. candidate at Dalian University of Technology. His research interest covers complex system modeling and forecasting based on multivariate time series.)