# 基于反传混沌粒子群训练的前馈神经网络研究*

朱群雄，董春岩，林晓勇†
（北京化工大学 信息科学与技术学院，北京 100029）

**摘 要**：为了解决前馈神经网络训练收敛速度慢、易陷入局部极值及对初始权值依赖性强等缺点，提出了一种基于反传的无限折叠迭代混沌粒子群优化（ICMICPSO）算法训练前馈神经网络（FNNs）参数。该方法在充分利用 BP 算法的误差反传信息和梯度信息的基础上，引入了 ICMIC 混沌粒子群的概念，将 ICMIC 粒子群（ICMICPS）作为全局搜索器，梯度下降信息作为局部搜索器来调整网络的权值和阈值，使得粒子能够在全局寻优的基础上对整个空间进行搜索。通过仿真实验与多种算法进行对比，结果表明在训练和泛化能力上 ICMICPSO-BPNN 方法明显优于其他算法。

**关键词**：前馈神经网络；BP 网络；粒子群优化；混沌映射
**中图分类号**：TP301.6 **文献标志码**：A **文章编号**：1001-3695(2014)01-0120-04
doi:10.3969/j.issn.1001-3695.2014.01.028

# Research of training feedforward neural networks based on hybrid chaos particle swarm optimization-back-propagation

ZHU Qun-xiong, DONG Chun-yan, LIN Xiao-yong†
(*College of Information Science & Technology, Beijing University of Chemical Technology, Beijing* 100029, *China*)

**Abstract**: In order to overcome the shortcomings of feedforward neural network's slow convergence, involving the local optimal and depending on the initial weights, this paper proposed a new method to train feedforward neural networks (FNNs) parameters based on the iterative chaotic map with infinite collapses particle swarm optimization (ICMICPSO) algorithm. This algorithm made full use of the information of BP's error back propagation and gradient. It used ICMICPS as the global optimizer to adjust the neural networks' weights and thresholds, when network parameters converge around global optimum. And it used gradient information as a local optimizer to accelerate the modification at a local scale. Compared with other algorithms, results show that the performance of the ICMICPSO-BPNN method is superior to the contrast methods in training and generalization ability.
**Key words**: feedforward neural networks; back-propagation neural networks; particle swarm optimization; chaos map

## 0 引言

近年来，神经网络广泛应用于模式识别、数据挖掘和智能控制等领域。应用较多的三层前馈神经网络主要采用基于误差反传的梯度下降（BP）方法进行训练，是当前前馈神经网络（FNNs）中研究最为成熟且应用最为广泛的一种有导师学习算法[1]。然而，这种方法对初始权值选择敏感，并且容易陷入局部极值，使得神经网络的训练效果变差，从而影响了 BP 神经网络的性能。近几年来，逐渐有很多学者利用粒子群、蚁群、遗传算法等人工智能方法训练神经网络，取得了良好效果。

粒子群优化（particle swarm optimization, PSO）算法是由 Kennedy 等人[2]在 1995 年提出的一种群智能优化算法。该算法具有收敛速度快、建模简单且易于实现等优点，在神经网络训练、模式分类、模糊系统控制等传统优化问题上取得了良好效果。但 PSO 算法也存在一些问题，如算法在搜索前期收敛速度快，而在后期容易陷入局部最优值及鲁棒性较差等缺陷。针对这些问题，研究人员提出了很多改进算法。如 Shi 等人[3]提出了惯性权重线性递减的 PSO 优化算法。Van Den Bergh 等人[4]提出了 Multi-start PSO 算法，迭代若干次后都保留历史上的最优例子，并提出了协同粒子群优化因子[5]。Ho 等人[6]提出了一种动态惯性权重的粒子群优化算法，惯性权重随着迭代次数的增加而降低，开始阶段具有良好的全局搜索能力，在后期能够以较小的惯性权重收敛到最优解。混沌（chaos）是存在于非线性系统当中的一种较为普遍的现象，是由确定性方程得到的具有随机性的运动状态。混沌运动具有随机性、普遍性、规律性等特性，能在一定的范围内按照自身的规律不重复遍历所有状态。有很多学者将混沌与粒子群结合，以改善粒子群的全局寻优能力，避免种群早熟。如 Xie 等人[7]在粒子群中引入混沌系统，以一定概率随机初始化种群中粒子位置。Liu 等人[8]将混沌概念引入到自适应惯性权重粒子群优化算法中，改善粒子的全局寻优性能。然而，这些算法都集中利用 Logistic 传统映射方法产生混沌变量，改进算法性能受限。

本文提出了一种 ICMIC 粒子群（ICMICPSO）方法。将 ICMICPSO 与梯度下降法相结合，形成混合 ICMICPSO-BP 算法训练 FNNs。算法在初始阶段利用 PSO 进行全局搜索，当出现早熟收敛时，利用混沌搜索带领粒子逃离局部最优解，并利用梯

**收稿日期**：2013-04-10；**修回日期**：2013-05-27 **基金项目**：国家自然科学基金资助项目（61074153）
**作者简介**：朱群雄（1960-），男，江苏无锡人，教授，博导，主要研究方向为智能系统、数据挖掘、可拓工程论与过程工业模拟、优化及故障诊断；董春岩（1987-），男，硕士，主要研究方向为神经网络；林晓勇（1979-），男（通信作者），副教授，博士，主要研究方向为数据挖掘、人工智能（linxy@mail.buct.edu.cn）.

度下降法在全局最优解附近加速局部搜索。将所提出的 ICMICPSO-BP 算法用于训练前馈神经网络的权值和阈值,利用四个 Benchmark 标准函数问题来检验算法,并和文献结果进行对比,说明算法的有效性。

## 1　BPNN、PSO 及混沌映射基本思想

### 1.1　基于误差反传的神经网络(BPNN)

BP 神经网络是在导师的指导下,其学习过程分为正向传播和反向传播两个过程。已经证明一个三层 BP 神经网络能以任意精度逼近非线性函数,以三层神经网络为例。设输入有 $L$ 节点,隐含层有 $m$ 个节点,输出层有 $n$ 个节点。均方误差函数计算式如下:

$$E=\sum_{j=1}^{q}E_j/(q*k)\ \text{ where }E_j=\sum_k\varepsilon_k^2=\sum_k(d_k-c_k)^2 \tag{1}$$

其中:$q$ 是输入样本数;$\varepsilon_k$ 是输出层第 $k$ 个节点的输出误差;$d_k$ 是输出层第 $k$ 个节点的期望输出值;$c_k$ 是输出层第 $k$ 个节点的实际输出值。

### 1.2　粒子群优化(PSO)算法

粒子群优化(PSO)算法是由 Kennedy 和 Eberhart 于 1995 年提出的一种新型智能优化算法。算法随机初始化一群粒子,粒子通过追逐两个极值来更新自己的位置,即:自身所找到的当前最优解,称为个体极值 $P_b$;群体当前最优解,称为全局极值 $P_g$。假设在 $D$ 维搜索空间中,有 $m$ 个粒子组成一个种群,粒子 $i$ 在 $D$ 维空间中的位置可以表示为 $X_i=(x_{i1},x_{i2},\cdots,x_{iD})$,第 $i$ 个粒子经历过的最好位置记为 $P_i=(p_{i1},p_{i2},\cdots,p_{iD})$,每个粒子的飞行速度为 $V_i=(v_{i1},v_{i2},\cdots,v_{iD})(i=1,2,\cdots,m)$,在整个群体中所有粒子经历过的最好位置为 $P_g=(P_{g1},P_{g2},\cdots,P_{gD})$,每一代粒子通过跟踪个体极值和全局极值来更新自身的速度和位置如式(2)和(3)所示:

$$v_{id}^{k+1}=\omega\times v_{id}^k+c_1\times\text{rand}()\times(p_{id}-x_{id}^k)+c_2\times\text{rand}()\times(p_{gd}-x_{id}^k) \tag{2}$$

$$x_{id}^{k+1}=x_{id}^k+v_{id}^{k+1} \tag{3}$$

其中:$k$ 为迭代次数;$c_1$ 和 $c_2$ 为学习因子;rand( )为介于[0,1]之间的随机数;$\omega$ 为惯性权重,本文利用线性递减惯性权重,$\omega$ 取值[3]为

$$\omega=\omega_{\max}-t\frac{\omega_{\max}-\omega_{\min}}{T_{\max}} \tag{4}$$

其中:$\omega_{\max}$和 $\omega_{\min}$分别是惯性权重上下限;$T_{\max}$为算法的最大迭代次数;$t$ 为算法当前迭代次数。

PSO 算法早熟判断—适应度方差法:

随着种群不断进化,个体之间的差异度逐渐变小,并出现粒子聚集现象,导致算法在训练过程中极易出现早熟。为了判断种群是否出现早熟,可根据粒子适应度的整体变化情况进行判断。设种群中粒子数为 $N$,第 $i$ 个粒子的适应度为 $f_i$,当前种群的平均适应度为 $f_{\text{avg}}$,种群的适应度方差为 $\sigma^2$,定义为[9]

$$\sigma^2=\sum_{i=1}^{N}\left(\frac{f_i-f_{\text{avg}}}{f}\right) \tag{5}$$

其中:$f$ 为归一化定标因子,作用是限制 $\sigma^2$ 的大小,$f$ 的取值为

$$f=\begin{cases}\max\limits_{1\leqslant i\leqslant N}|f_i-f_{\text{avg}}| & \max|f_i-f_{\text{avg}}|>1\\ 1 & \text{其他}\end{cases} \tag{6}$$

式(6)表示 $\sigma^2$ 为群体的适应度方差,反映的是粒子群中所有粒子的聚集程度。$\sigma^2$ 越小,表示粒子聚集程度越大。随着迭代次数的增多,种群个体适应度会逐渐接近,$\sigma^2$ 会越来越小,当 $\sigma^2>C$($C$ 为某一给定的阈值)且适应度不满足结束条件时,进行早熟处理,即混沌变换操作。

### 1.3　混沌映射(chaotic maps)

混沌是存在于非线性系统中的一种较为普遍的现象,混沌运动具有良好的遍历性、随机性和规律性等特点,能够在一定范围内按照自身规律不重复遍历所有状态。利用这种遍历状态和粒子群算法结合,可以帮助粒子逃出局部最优解区域,并增强全局搜索能力[10~12]。一维可逆映射是一种最简单的混沌运动系统,He 等人[13,14]提出了无限折叠迭代混沌映射(iterative chaotic map with infinite collapses,ICMIC)形式,并从数学角度严格讨论了它的混沌性,该混沌形式有较大的 Lyapunov 指数,相比于 Logistic 等传统混沌映射,ICMIC 映射具有更好的混沌特性,本文采用 ICMIC 映射来产生混沌变量。ICMIC 映射产生一个[-1,1]之间分布的变量序列:

$$x_{k+1}=\sin(a/x_k)\quad -1\leqslant x_k\leqslant 1,x_k\neq 0 \tag{7}$$

其中:$x_k$ 为第 $k$ 个分量;$a$ 为控制参量,一般 $a$ 可以取 5.65。利用混沌对初值的敏感特点,取 $N$ 个微小差异的初值粒子,可以得到 $N$ 个轨迹不同的混沌变量。

在实际的应用中,各变量的定义区间不同,需要将混沌变量由区间(-1,1)载波到粒子群解向量的区间。设 $x_{i\max}$ 和 $x_{i\min}$分别为第 $i$ 维变量的搜索上下界。按照下式映射为(-1,1)之间的混沌变量 $cx_i$[15]:

$$cx_i=\frac{x_i-x_{i\min}}{x_{i\max}-x_{i\min}} \tag{8}$$

按照式(8)更新混沌变量 $cx_i$,按式(9)转换成决策变量 $x_i$:

$$x_i=x_{i\min}+cx_i\times(x_{i\max}-x_{i\min}) \tag{9}$$

## 2　ICMICPSO-BP 算法及 FNNs 训练

### 2.1　混合 ICMICPSO-BP 算法

ICMICPSO-BP 是将 ICMIC 混沌映射与基于误差反传的梯度下降法结合。其中 PSO 算法有较强的全局最优解搜索能力,但当搜索到最优解附近时,搜索速度变慢并且容易出现早熟现象。为了解决 PSO 算法的上述缺陷,本文引入 ICMIC 和基于误差反传的梯度下降算法来改善粒子群优化算法的性能。通过早熟判断机制,即当 PSO 算法陷入局部最优解时,引入混沌映射,带领群体逃离局部极值点,继续在全局范围内搜索,避免早熟现象的发生。同时利用基于误差反传的梯度下降法可以加速粒子在最优解附近的局部搜索。根据收敛情况,算法能够自动地在全局和局部搜索之间切换。

### 2.2　ICMICPSO-BP 算法训练前馈神经网络

将前馈神经网络的权值和阈值作为粒子的位置向量,利用本文提出的算法训练网络参数。

根据 PSO 算法早熟判断机制,当适应度方差 $\sigma^2$ 小于某个给定的阈值时,说明群体出现早熟,此时以当前全局最优位置为初始点,按照一定概率进行混沌映射,带领粒子逃离局部极值区域。为了加速局部搜索过程调用 BP 算法,并再次进入 PSO 寻优过程,如此进行迭代,直到算法终止。

本算法利用混沌映射可以获得种群的多样性,使用梯度下降信息可以加速 PSO 局部搜索能力,因此粒子能够在快速局部搜索的前提下对整个空间进行全局搜索。具体步骤如下:

a)算法各参数初始化:

(a)设置粒子种群规模 $M$,算法总迭代次数 $T$,算法当前迭代次数 $TN$,最小训练停止误差 $\lambda$,混沌变换概率 $P_m$,学习因子 $C_1$ 和 $C_2$,惯性权重 $\omega$,适应度方差阈值 $C$,BP 算法迭代次数 $T_{\mathrm{BP}}$,学习速率 $\eta$,动量因子 $\alpha$。

(b)随机初始化粒子速度 $V_i=(v_{i1},v_{i2},\cdots,v_{iD})^{\mathrm{T}}$ 和位置 $X_i=(x_{i1},x_{i2},\cdots,x_{iD})^{\mathrm{T}}$ 来表征神经网络的权值和阈值,其中 $D$ 为神经网络权值和阈值维数之和。

(c)计算粒子适应度值 $f(x_i)$,令初始种群中具有最佳适应度值的粒子作为初始算法搜索的全局极值位置 $P_g$ 和整个算法的全局极值位置 $P_{\mathrm{best}}$,并令 $TN=1$。

b)如果 $TN\geqslant T$,则保存最优结果 $\min\{f(p_g),f(p_i)\}$,结束;否则,执行以下步骤:

(a)根据式(2)和(3)更新粒子的速度 $V_i$ 和位置 $X_i$。

(b)根据群体中各粒子的信息,更新 $P_i$ 和 $P_g$,并记录下全局最优粒子下标 $g_{\mathrm{best}}$。

c)如果 $\sigma^2\leqslant C$,则继续执行以下步骤,否则,返回步骤 b)。

d)以全局最优粒子的位置 $X_{g_{\mathrm{best}}}$ 为初始点,调用 BP 算法,从而更新 $X_{g_{\mathrm{best}}}$,$P_i$ 和 $TN$。

e)如果 $TN\geqslant T$,保存优化结果 $\min\{f(p_g),f(p_i)\}$,结束;否则,继续。

f)对每个粒子,生成[0,1]之间的随机数 $r$,如果 $r\leqslant P_m$,且下标 $i\neq g_{\mathrm{best}}$,则在混沌搜索空间进行 ICMIC 混沌映射,并计算新位置对应的目标值 $f_i^{k+1}$,更新 $TN$、$P_g$。

g)返回步骤 b)。

### 2.3 ICMICPSO-BP 算法性能分析

ICMICPSO-BP 算法结合了粒子群、混沌映射及 BP 算法,其中粒子群和混沌映射算法较单纯 BP 算法步骤简单,训练速度快。在复杂度方面,ICMICPSO-BP 算法相当于 PSO-BP 和 GA-BP 等算法复杂度。在收敛性方面,引入混沌映射机制,使粒子有较强的全局搜索能力;引入 BP 能够加速粒子局部寻优能力,PSO 的性能得到了良好发挥,所以算法的收敛性能更加优越,收敛速度较对比文献算法更快。

## 3 仿真实验

### 3.1 函数列表及实验参数设置

本文采用四个典型的 Benchmark 函数来检验算法的有效性,并和文献[16]中的结果进行对比。四个 Benchmark 函数如表 1 所示。

表 1 测试 Benchmark 函数集

| 函数 | 范围 |
|---|---|
| $f_1=100\times(x_1^2-x_2)^2+(1-x_1)^2$ | $x_i\in[-10,10],i=1,2$ |
| $f_2=x_1^2+x_2^3-x_1x_2x_3+x_3-\sin x_2^2-\cos(x_1x_2^3)$ | $x_i\in[-2\pi,2\pi],i=1,2,3$ |
| $f_3=x_1^{x_2^3}+x_3^{x_4^{x_5}}$ | $x_i\in[0,2],i=1,2,\cdots,5$ |
| $f_4=\sum_{i=1}^{8}x_i^i$ | $x_i\in[-1,1],i=1,2,\cdots,8$ |

为了保持和对比文献中训练参数设置的一致性,参数设置如下:粒子群种群规模 $M=40$,程序最大的迭代次数设置为 $T=1\ 000$ 次,最小训练停止误差 $\lambda=10^{-6}$,粒子群学习因子 $C_1=C_2=1.4$,网络权值和阈值初始化为[-1,1]范围,初始化惯性权重 $w$ 的范围为[0.4,0.9],$r_1$ 和 $r_2$ 是[0,1]范围内的两个随机数。设粒子最大的速度是 10,最小的速度为 -10,如果粒子的速度超过了限制范围,将会被重置。利用 ICMIC 映射作为混沌映射,混沌变换概率 $P_m=0.2$,适应度方差阈值 $C=0.01$,$v_{\max}=(x_{\max}-x_{\min})/2$,$v_{\min}=-v_{\max}$,粒子位置和速度的维数(包括权值和阈值)$D=i\times h+h\times 1+h+1=h\times(i+2)+1$,其中 $i$ 表示输入层神经元个数,$h$ 表示隐含层神经元个数。设置 BP 的学习速率 $\eta=0.7$,动量因子 $\alpha=0.3$,BP 子程序的最大迭代次数为 30 次。在自变量取值范围内,随机初始化训练样本 150 组,泛化样本 50 组。将提出的算法与文献中的结果进行对比,其中 $E_1$ 为训练均方差,$E_2$ 为泛化均方差,$E_3$ 为训练平均绝对误差、$E_4$ 为泛化平均绝对误差、$E_5$ 为全部样本平均绝对误差。其中 $f_1$ 函数的网络隐含层节点数 $h=3,4,5,6,7,8,9,10$;$f_2$ 函数的网络隐含层节点数 $h=3,4,5,6,7,8,9,10$;$f_3$ 函数的网络隐含层节点数 $h=3,4,5,6,7,8,9,10,11,12$;$f_4$ 函数的网络隐含层节点数为 $h=4,5,6,7,8,9,10,11,12$。

表 2~5 列出了函数 $f_1$~$f_4$ 具有不同隐含层节点数时的上述五种误差,表 6 列出的是 ICMICPSO-BPNN 算法和对比算法训练四个函数的误差比较。

表 2 $f_1$ 函数不同网络结构时训练误差对比

| 隐含层节点数 $h$ | 算法 | $E_1$ | $E_2$ | $E_3$ | $E_4$ | $E_5$ |
|---|---|---|---|---|---|---|
| 3 | ICMICPSO-BPNN | 3.17E-05 | 0.000962 | 0.004823 | 0.027465 | 0.010484 |
| | PSO-BPNN | 5.2469E-05 | 0.001506 | 0.006068 | 0.034836 | 0.013260 |
| 4 | ICMICPSO-BPNN | 4.15E-05 | 0.000743 | 0.002387 | 0.023674 | 0.007709 |
| | PSO-BPNN | 7.48E-05 | 0.003765 | 0.004926 | 0.040358 | 0.013784 |
| 5 | ICMICPSO-BPNN | 8.43E-05 | 0.000648 | 0.003464 | 0.022652 | 0.008261 |
| | PSO-BPNN | 0.000140 | 0.057845 | 0.008578 | 0.192467 | 0.054550 |
| 6 | ICMICPSO-BPNN | 3.63E-06 | 0.000547 | 0.001782 | 0.023257 | 0.007149 |
| | PSO-BPNN | 6.06E-06 | 0.009995 | 0.001906 | 0.074016 | 0.019933 |
| 7 | ICMICPSO-BPNN | 1.04E-06 | 4.2843E-05 | 0.000867 | 0.010234 | 0.003201 |
| | PSO-BPNN | 4.87E-06 | 2.98E-04 | 0.001886 | 0.014980 | 0.005160 |
| 8 | ICMICPSO-BPNN | 8.37E-06 | 0.000797 | 0.001032 | 0.018561 | 0.005414 |
| | PSO-BPNN | 4.75141E-05 | 0.001074 | 0.005482 | 0.030187 | 0.011658 |
| 9 | ICMICPSO-BPNN | 1.26E-05 | 0.000864 | 0.001932 | 0.024634 | 0.007607 |
| | PSO-BPNN | 3.33E-05 | 0.003742 | 0.004869 | 0.039807 | 0.013603 |
| 10 | ICMICPSO-BPNN | 1.05E-05 | 0.000946 | 0.00225 | 0.025193 | 0.007985 |
| | PSO-BPNN | 1.70E-05 | 0.002909 | 0.003310 | 0.039772 | 0.012425 |

表 3 $f_2$ 函数不同网络结构时训练误差对比

| 隐含层节点数 $h$ | 算法 | $E_1$ | $E_2$ | $E_3$ | $E_4$ | $E_5$ |
|---|---|---|---|---|---|---|
| 3 | ICMICPSO-BPNN | 2.63E-05 | 0.002728 | 0.003152 | 0.025436 | 0.008723 |
| | PSO-BPNN | 4.49E-05 | 0.004118 | 0.005569 | 0.052501 | 0.017302 |
| 4 | ICMICPSO-BPNN | 1.35E-05 | 0.003265 | 0.003614 | 0.034752 | 0.011398 |
| | PSO-BPNN | 3.92E-05 | 0.006128 | 0.005224 | 0.063825 | 0.019874 |
| 5 | ICMICPSO-BPNN | 6.38E-06 | 0.004179 | 0.003825 | 0.039164 | 0.012659 |
| | PSO-BPNN | 3.58E-05 | 0.007362 | 0.004882 | 0.065222 | 0.019967 |
| 6 | ICMICPSO-BPNN | 4.67E-06 | 8.29E-05 | 0.002835 | 0.017457 | 0.006488 |
| | PSO-BPNN | 3.56E-05 | 9.26E-04 | 0.004992 | 0.025081 | 0.010014 |
| 7 | ICMICPSO-BPNN | 4.39E-05 | 8.23E-05 | 0.003326 | 0.011254 | 0.007808 |
| | PSO-BPNN | 2.86E-04 | 5.15E-04 | 0.012571 | 0.018063 | 0.013944 |
| 8 | ICMICPSO-BPNN | 3.54E-05 | 0.000863 | 0.003746 | 0.022758 | 0.008493 |
| | PSO-BPNN | 3.76E-05 | 0.001524 | 0.004963 | 0.030166 | 0.011264 |
| 9 | ICMICPSO-BPNN | 3.08E-05 | 0.007651 | 0.002973 | 0.021415 | 0.007581 |
| | PSO-BPNN | 4.63E-05 | 0.019533 | 0.005353 | 0.112832 | 0.032223 |
| 10 | ICMICPSO-BPNN | 3.95E-05 | 0.004132 | 0.002865 | 0.027159 | 0.008936 |
| | PSO-BPNN | 4.13E-05 | 0.006777 | 0.005272 | 0.064394 | 0.020052 |

表 4 $f_3$ 函数不同网络结构时训练误差对比

| 隐含层节点数 $h$ | 算法 | $E_1$ | $E_2$ | $E_3$ | $E_4$ | $E_5$ |
|---|---|---|---|---|---|---|
| 3 | ICMICPSO-BPNN | 0.003156 | 0.005417 | 0.034265 | 0.053243 | 0.039010 |
| | PSO-BPNN | 0.005884 | 0.007459 | 0.055645 | 0.070014 | 0.059238 |
| 4 | ICMICPSO-BPNN | 0.006471 | 0.008294 | 0.023848 | 0.041521 | 0.028263 |
| | PSO-BPNN | 0.010471 | 0.010909 | 0.067656 | 0.076700 | 0.069917 |
| 5 | ICMICPSO-BPNN | 0.007562 | 0.009426 | 0.028641 | 0.049763 | 0.033922 |
| | PSO-BPNN | 0.010771 | 0.010075 | 0.070969 | 0.080524 | 0.073358 |
| 6 | ICMICPSO-BPNN | 0.007454 | 0.008315 | 0.026346 | 0.045231 | 0.031063 |
| | PSO-BPNN | 0.011222 | 0.007552 | 0.075375 | 0.063986 | 0.072528 |
| 7 | ICMICPSO-BPNN | 0.007165 | 0.007268 | 0.031723 | 0.046526 | 0.035424 |
| | PSO-BPNN | 0.007343 | 0.007399 | 0.057853 | 0.062319 | 0.058970 |
| 8 | ICMICPSO-BPNN | 0.006472 | 0.008562 | 0.021652 | 0.047184 | 0.028033 |
| | PSO-BPNN | 0.011044 | 0.009478 | 0.074014 | 0.070769 | 0.073203 |
| 9 | ICMICPSO-BPNN | 0.004125 | 0.006318 | 0.024963 | 0.042541 | 0.029355 |
| | PSO-BPNN | 0.009145 | 0.013129 | 0.065892 | 0.079063 | 0.069184 |
| 10 | ICMICPSO-BPNN | 0.003156 | 0.003264 | 0.020015 | 0.032815 | 0.023215 |
| | PSO-BPNN | 0.005850 | 0.003419 | 0.044258 | 0.043698 | 0.044118 |
| 11 | ICMICPSO-BPNN | 0.003682 | 0.008165 | 0.021874 | 0.039265 | 0.026218 |
| | PSO-BPNN | 0.005876 | 0.015719 | 0.047325 | 0.066078 | 0.052013 |
| 12 | ICMICPSO-BPNN | 0.003861 | 0.009825 | 0.023146 | 0.042512 | 0.066781 |
| | PSO-BPNN | 0.004666 | 0.014449 | 0.046185 | 0.061496 | 0.050012 |

表 5　$f_4$ 函数不同网络结构时训练误差对比

| 隐含层节点数 $h$ | 算法 | $E_1$ | $E_2$ | $E_3$ | $E_4$ | $E_5$ |
|---|---|---|---|---|---|---|
| 4 | ICMICPSO-BPNN | 0.010635 | 0.018326 | 0.074564 | 0.092817 | 0.079127 |
| | PSO-BPNN | 0.015260 | 0.022104 | 0.096349 | 0.117707 | 0.101688 |
| 5 | ICMICPSO-BPNN | 0.005462 | 0.007493 | 0.052674 | 0.063462 | 0.055371 |
| | PSO-BPNN | 0.009349 | 0.009811 | 0.075112 | 0.072731 | 0.074517 |
| 6 | ICMICPSO-BPNN | 0.005735 | 0.007929 | 0.052173 | 0.068154 | 0.0561682 |
| | PSO-BPNN | 0.014198 | 0.017239 | 0.092526 | 0.101285 | 0.094716 |
| 7 | ICMICPSO-BPNN | 0.005228 | 0.006742 | 0.041791 | 0.062839 | 0.047052 |
| | PSO-BPNN | 0.007367 | 0.008337 | 0.068605 | 0.068605 | 0.069402 |
| 8 | ICMICPSO-BPNN | 0.007191 | 0.009281 | 0.042655 | 0.068547 | 0.049012 |
| | PSO-BPNN | 0.009399 | 0.013577 | 0.079761 | 0.100246 | 0.084882 |
| 9 | ICMICPSO-BPNN | 0.008192 | 0.009347 | 0.052163 | 0.074152 | 0.057660 |
| | PSO-BPNN | 0.012823 | 0.015750 | 0.092703 | 0.103527 | 0.095409 |
| 10 | ICMICPSO-BPNN | 0.008292 | 0.009728 | 0.063276 | 0.075251 | 0.067265 |
| | PSO-BPNN | 0.009363 | 0.010234 | 0.077145 | 0.076798 | 0.077059 |
| 11 | ICMICPSO-BPNN | 0.009014 | 0.011417 | 0.071615 | 0.084215 | 0.074792 |
| | PSO-BPNN | 0.015711 | 0.021223 | 0.098027 | 0.120967 | 0.103762 |
| 12 | ICMICPSO-BPNN | 0.009176 | 0.012431 | 0.072964 | 0.086213 | 0.076253 |
| | PSO-BPNN | 0.008876 | 0.014913 | 0.074183 | 0.092894 | 0.078861 |

表 6　ICMICPSO-BPNN 和对比算法用于四个函数误差比较

| 算法 | 误差 | $f_1$(2-7-1) | $f_2$(3-6-1) | $f_3$(5-10-1) | $f_4$(8-7-1) |
|---|---|---|---|---|---|
| ICMICPSO-BPNN | $E_1$ | 1.04E-06 | 4.67E-06 | 0.003156 | 0.005228 |
| | $E_2$ | 4.28E-05 | 8.29E-05 | 0.007764 | 0.006742 |
| | $E_3$ | 0.000867 | 0.002835 | 0.020015 | 0.041791 |
| | $E_4$ | 0.010234 | 0.016457 | 0.032815 | 0.062839 |
| | $E_5$ | 0.003201 | 0.006488 | 0.023215 | 0.047052 |
| 改进 PSO-BPNN | $E_1$ | 4.87E-06 | 3.56E-05 | 0.005850 | 0.007367 |
| | $E_2$ | 2.98E-04 | 9.26E-04 | 0.003419 | 0.008337 |
| | $E_3$ | 0.001886 | 0.004992 | 0.044258 | 0.068605 |
| | $E_4$ | 0.014980 | 0.025081 | 0.043698 | 0.071793 |
| | $E_5$ | 0.005160 | 0.010014 | 0.044118 | 0.069402 |
| 传统 PSO-BPNN | $E_1$ | 4.16E-05 | 2.54E-04 | 0.008941 | 0.015331 |
| | $E_2$ | 0.110844 | 0.066119 | 0.002599 | 0.019901 |
| | $E_3$ | 0.005701 | 0.013349 | 0.061929 | 0.098510 |
| | $E_4$ | 0.250629 | 0.224780 | 0.041604 | 0.111654 |
| | $E_5$ | 0.066933 | 0.066207 | 0.056848 | 0.101796 |
| GA-BPNN | $E_1$ | 5.37E-04 | 3.30E-04 | 0.009272 | 0.011973 |
| | $E_2$ | 0.041519 | 0.064065 | 0.003002 | 0.016747 |
| | $E_3$ | 0.017857 | 0.015369 | 0.058552 | 0.086193 |
| | $E_4$ | 0.197877 | 0.221804 | 0.045065 | 0.105106 |
| | $E_5$ | 0.062862 | 0.066978 | 0.055180 | 0.090921 |
| 基本 BPNN | $E_1$ | 9.9995E-05 | 3.6850E-04 | 0.0114 | 0.0120 |
| | $E_2$ | 0.4178 | 0.1350 | 0.1949 | 0.0132 |
| | $E_3$ | 0.0082 | 0.0151 | 0.0730 | 0.0857 |
| | $E_4$ | 0.5980 | 0.3629 | 0.3813 | 0.0944 |
| | $E_5$ | 0.2075 | 0.1361 | 0.2001 | 0.1172 |

### 3.2　仿真结果及分析

a)按照 3.1 节中的算法设置,将本文所提出的ICMICPSO-BPNN 算法与对比文献[16]中相同参数设置环境下改进的 PSO-BPNN 算法结果进行对比,表 2 ~ 6 列出了所测试的四种 Benchmark 函数的结果。

表 2 是 $f_1$ 函数的测试结果,可以看出,ICMICPSO-BPNN 训练误差 $E_1$ 和泛化误差 $E_2$ 明显小于对比算法,当 $h=7$ 时,ICMICPSO-BPNN 的泛化误差为 4.2843E-05 远小于对比算法的 2.98E-04,算法具有良好的学习能力及对新样本良好的适应能力,同样在 $E_3$、$E_4$ 和 $E_5$ 上也有体现。通过不同隐含层节点数的结果对比,当隐含层节点数 $h=7$ 时算法表现出了较好的性能。

表 3 是 $f_2$ 函数的对比结果,可以看出,ICMICPSO-BPNN 的 $E_1$、$E_2$、$E_3$、$E_4$ 和 $E_5$ 误差明显小于对比算法,特别当 $h=6$ 时,$E_1$ 和 $E_2$ 要比对比算法小将近一个数量级;对于训练和泛化绝对误差 $E_3$、$E_4$ 和 $E_5$,本文算法也要明显优于对比算法,充分说明了混沌映射在粒子群优化算法中的重要作用,即带领粒子逃离局部最优解。

表 4 显示的是 $f_3$ 函数误差结果对比,除 $h=6$ 时 $E_2$ 误差略大于对比算法外,其余训练误差和泛化误差结果都优于对比算法的结果,充分说明了 ICMICPSO-BPNN 算法具有较高的训练精度和良好的学习能力。当 $h=10$ 时,五种测试误差均最小,算法表现出更好的性能。

表 5 显示的是 $f_4$ 函数各误差的对比结果,可以看出 $E_3$、$E_4$ 和 $E_5$ 都小于对比算法的结果,说明 ICMICPSO-BPNN 的网络输出值和真实值更加接近。除 $h=12$ 时,$E_1$ 略大于对比算法外,其余 $E_1$ 和 $E_2$ 均小于对比算法,说明算法泛化误差更小,对新样本有更强的学习和适应能力,当 $h=7$ 时,五种误差最小,性能最优。

b)表 6 显示的是 ICMICPSO-BPNN 与文献[16]中给出的改进 PSO-BPNN、传统 PSO-BPNN、GA-BPNN、BPNN 四种算法的对比,其中四个函数的网络结构分别是在 2-7-1、3-6-1、5-10-1、8-7-1 结构下进行测试。从结果可以看出,ICMICPSO-BPNN 在训练均方差、泛化均方差、训练绝对误差、泛化绝对误差、总样本绝对误差五个测试指标都要优于其他对比算法一个数量级,说明所提算法适合建模,具有较高的训练精度和泛化精度,并且对新样本有更好的适应性。同样也能说明,引入混沌和适应度方差的有效性,利用适应度方差判断粒子的早熟,利用混沌特性来带领粒子逃离局部最优解,利用基于误差反传的梯度下降方法加速粒子的局部搜索,这样的策略是算法高效和优于对比算法的主要原因。

## 4　结束语

本文提出了一种基于反传的 ICMIC 粒子群(ICMICPSO-BP)方法来训练前馈神经网络。此算法充分利用了基于误差反传的梯度下降信息,并且引入混沌映射避免粒子陷入局部最优解,能够在全局范围内调整网络参数。该方法能够自动地在误差反传的梯度下降法、混沌映射和 PSO 算法间切换。

本文利用四个标准 Benchmark 函数集来证明 ICMICPSO-BPNN 的有效性,并和文献中的改进 PSO-BPNN、传统 PSO-BPNN、GA-BPNN 及 BPNN 算法的结果进行对比。结果表明 ICMICPSO-BPNN 算法具有较高的训练精度和泛化精度,适宜于软测量建模,同时也证明了在粒子群算法的基础上引入混沌概念和误差反传梯度下降概念的合理性和正确性。

**参考文献:**


[1] HORNIK K, STINCHCOMBE M, WHITE H. Multilayer feedforward networks are universal approximators[J]. Neural Networks, 1989, 2(5): 359-366.

[2] KENNEDY J, EBERHART R C. Particle swarm optimization[C]// Proc of IEEE International Conference on Neural Networks. Piscataway: IEEE Press, 1995: 1942-1948.

[3] SHI Yu-hui, EBERHART R C. Empirical study of particle swarm optimization[C]//Proc of IEEE Congress on Evolutionary Computation. Washington DC: IEEE Press, 1999: 1945-1950.

[4] Van Den BERGH F, ENGELBRECHT A P. A new locally convergent particle swarm optimizer[C]//Proc of IEEE International Conference on Systems, Man and Cybernetics. [S. l.]: IEEE Press, 2002.

[5] Van Den BERGH F, ENGELBRECHT A P. A cooperative approach to particle swarm optimization[J]. IEEE Trans on Evolutionary Computation, 2004, 8(3): 225-239.


(下转第 133 页)

$$\begin{cases} y(t) = y_0 \quad 0 \leqslant t \leqslant \tau \\ \dfrac{\mathrm{d}y(t)}{\mathrm{d}(t)} = ry(t)(1 - y(t)/k) \end{cases} \tag{11}$$

对上述微分方程求解可得

$$y(t) = k/(1 + \mathrm{e}^{a-rt}), y_0 = k/(1 + \mathrm{e}^{a}) \tag{12}$$

式(12)为谷氨酸菌体的生长模型,其中 $y(t)$ 为菌体浓度,$t$ 为菌体生长时间,$r$、$a$、$k$ 为待定模型参数。

本文用差分进化算法对生长模型的参数 $r$、$a$、$k$ 进行优化辨识,目的就是要确定菌体生长模型中的模型参数,使得模型的实际观测值与估计值有较高的拟合度。改进差分进化算法的参数取值为:维数 $D=3$,种群规模 $NP=20$,变异率 $F_0=0.3$,交叉概率 $CR_{\min}=0.5$,$CR_{\max}=0.85$,待辨识参数为 $r$、$a$、$k$,取值范围为[0,5]。本文利用文献[15]中提供的实际观测数据(上海天厨味精厂某发酵罐的批报数据)作为拟合点,对模型参数进行估计,并与文献[14]的人工神经网络(ANN)、文献[15]中的粒子群算法(PSO)进行了比较,得到的结果如表 3 所示。

表3　参数寻优结果

| 参数 | ANN(文献[14]) | PSO(文献[15]) | MCDE(本文) |
|---|---|---|---|
| $r$ | 0.370009 | 0.378983 | 0.437141 |
| $a$ | 1.698644 | 1.747781 | 1.853419 |
| $k$ | 0.908464 | 0.907380 | 0.905365 |

根据表 3 中的数据,由式(12)可计算出谷氨酸菌体浓度拟合值,结果如表 4 所示。图 7 是采用表 3 中 MCDE 算法结果构建的模型和实际曲线的拟合。由图可知,估计参数构建的模型与实际测量值接近。从 8 min 以后两条曲线拟合程度较高,由此可以看出,对于微生物发酵这类复杂、非线性很强的生化过程而言,改进差分进化算法是一种较好的参数估计算法。本文在标准算法的基础上,通过引入变异率算子,并动态增加交叉概率的方法,有效地提高了算法的搜索能力和收敛速度。

表4　菌体浓度实际值与模型拟合值的比较

| 实际值/g/L | ANN(文献[14]) | PSO(文献[15]) | MCDE(本文) |
|---|---|---|---|
| 0.32 | 0.2518 | 0.2459 | 0.2472 |
| 0.35 | 0.3243 | 0.3193 | 0.3329 |
| 0.36 | 0.4048 | 0.4013 | 0.4289 |
| 0.4 | 0.4886 | 0.4870 | 0.5272 |
| 0.58 | 0.5700 | 0.5704 | 0.5272 |
| 0.64 | 0.6443 | 0.6461 | 0.6187 |
| 0.74 | 0.7079 | 0.7106 | 0.6968 |
| 0.78 | 0.7598 | 0.7623 | 0.7687 |
| 0.82 | 0.8003 | 0.8032 | 0.8048 |
| 0.85 | 0.8309 | 0.8334 | 0.8378 |
| 0.86 | 0.8534 | 0.8554 | 0.8605 |
| 0.87 | 0.8697 | 0.8711 | 0.8759 |
| 0.87 | 0.8814 | 0.8822 | 0.8861 |
| 0.89 | 0.8897 | 0.8900 | 0.8928 |
| 0.9 | 0.8953 | 0.8954 | 0.8972 |
| 0.9 | 0.8993 | 0.8992 | 0.9001 |
| 0.9 | 0.9021 | 0.9017 | 0.9019 |
| 0.9 | 0.9041 | 0.9035 | 0.9039 |
| 0.9 | 0.9054 | 0.9047 | 0.9044 |
| 0.9 | 0.9064 | 0.9056 | 0.9048 |

为了进一步验证本文方法的有效性,根据表 4 中的数据,采用式(13)所示的均方差作为评价标准:

$$S = \sum_{i=1}^{n} \sqrt{\frac{(y_i - \hat{y})^2}{n-2}} \tag{13}$$

比较结果为:采用 MCDE 算法得到的模型均方差为 0.05954;采用 ANN 算法得到的模型均方差为 0.08828;采用 PSO 算法得到的模型均方差为 0.08672。从模型拟合值和模型均方差的结果可以看出,ANN 和 PSO 算法也可以得到较好的拟合结果,但是基于 MCDE 算法的模型参数辨识精度比 ANN 和 PSO 算法更高一些。可见,将 MCDE 算法用于非线性模型参数辨识具有有效性和可行性。

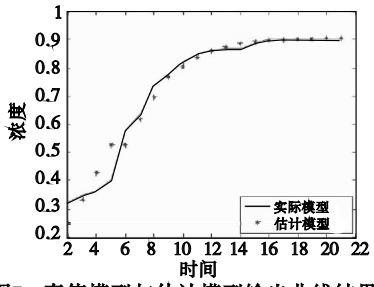


**图7　真值模型与估计模型输出曲线结果比较**

## 5　结束语

本文融合自适应变异率和动态非线性增加交叉概率两种策略的优点,提出了一种精度高、收敛速度快的改进差分进化算法。通过对几类非线性模型的参数辨识结果,说明该算法搜索能力强、收敛速度快,可以用来解决非线性模型参数辨识问题。以谷氨酸菌体生长模型参数估计为实例进行应用验证,证明了基于改进差分进化算法的非线性模型参数辨识是有效、可行的。将改进的差分进化算法应用于更多的实际工程中,以及研究在线估计,将是进一步的研究目标。

**参考文献:**

[1] TA M,De BRUNNER V. Minimum entropy estimation as a near maximum-likelihood method and its application in system identification with non-Gaussian noise[C]//Proc of IEEE International Conference on Acustics,Speed,and Signal Processing. 2004:545-548.

[2] LIU Yan-ming. Study on the technology of quadric surface extracting base on least square method[C]//Proc of the 2nd International Conference on Mechanic Automation and Control Engineering. 2011:5328-5331.

[3] WANG Jian-lin,XUE Yao-yu,YU Tao,*et al*. Run-to-run optimization for fed-bath fermentation process with swarm energy conservation particale swarm optimizatin algorithm[J]. Chinese Journal of Chemical Engineering,2010,18(5):787-794.

[4] KIM Y,MALLICK,R,BHOWMILCK S, et al. Nonlinear system identification of large-scale smart pavement systems[J]. Expert System with Applications,2013,40(9):3551-3560.

[5] 姜波,汪秉文. 基于遗传算法的非线性系统模型参数估计[J]. 控制理论与应用,2000,17(1):151-153.

[6] 王凌,李令莱,郑大钟,等. 非线性系统参数估计的一类有效搜索策略[J]. 自动化学报,2003,29(6):953-958.

[7] 吴亮红,王耀南,曾照福,等. 基于复合微粒群算法的非线性系统模型参数估计[J]. 系统仿真学报,2006,18(7):1942-1945.

[8] 于龙文,刘国志. 基于改进微粒群算法的非线性系统模型参数估计[J]. 科学技术与工程,2010,10(5):1259-1261.

[9] STORN R,PRICE K. Differential evoduation:a simple and efficient adaptive scheme for global optimization over continuous spaces[J]. Global Optimization,1997,11:341-359.

[10] 颜学锋,余娟,钱锋,等. 基于改进差分进化算法的超临界水氧化动力学参数估计[J]. 华东理工大学学报:自然科学版,2006,32(1):94-97

[11] MOHAMED A W,SABRY H Z. Constrained optimization based on modified differential evolution algorithm[J]. Information Sciences,2012,194:171-208.

[12] 丁峰. 系统辨识理论与方法[M]. 北京:中国电力出版社,2011.

[13] 林星卫,张惠娣,刘士荣,等. 应用粒子群优化算法辨识 Hammerstein 模型[J]. 仪器仪表学报, 2006, 27(1):76-79.

[14] 蔡煜东,陈常庆,周斌,等. 用人工神经网络辨识发酵动力学模型参数[J]. 生物数学学报,1994,9(4):103-107.

[15] 苏成利,徐志成,王树青. PSO 算法在非线性系统模型参数估计中的应用[J]. 信息与控制,2005,34(1):123-125.