文章编号: 1001-0920(2008)12-1321-06

改进协同微粒群优化的模糊神经网络控制系统设计

都延丽, 吴庆宪, 姜长生, 周 丽
(南京航空航天大学 自动化学院, 南京 210016)

摘 要: 针对协同微粒群算法不能保证收敛到局部或全局最优值的问题,提出一种改进协同微粒群算法(ICPSO),并证明了该算法能以概率 1 收敛于全局最优解.应用 ICPSO 建立一类非线性对象的神经网络辨识模型,并对系统的模糊神经网络自适应控制器的参数进行了离线和在线优化.仿真结果表明,ICPSO 能提高系统的建模精度,增强模型的泛化能力,而且由 ICPSO 训练的控制器可以达到良好的控制效果.

关键词: 改进协同微粒群算法; 全局收敛; 神经网络辨识; 模糊神经网络控制器
中图分类号: TP273.2 文献标识码: A

Improved cooperative particle swarm optimizer for design of fuzzy neural network control system

DU Yan-li, WU Qing-xian, JIANG Chang-sheng, ZHOU Li
(College of Automation Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China. Correspondent: DU Yan-li, E-mail: duylnuaa@sohu.com)

Abstract: An improved cooperative PSO (ICPSO) is proposed for the cooperative PSO incapable of converging at the local or global optimum. It is proved that the algorithm can converge at the global optimization solution with probability one. ICPSO is applied to the neural network modeling of a nonlinear plant, and also employed to the offline and online training of the fuzzy neural network adaptive controller in the system. Simulation results show that ICPSO has advantages of increasing the precision and enhancing the generalization capability of the model, and the fuzzy neural network (FNN) controller trained by ICPSO is effective for the system control.

Key words: Improved cooperative particle swarm optimization; Global convergence; Neural network identification; Fuzzy neural network controller

1 引 言

航空航天、生产制造过程等领域中存在着一大类非线性系统.通常,系统中仅含有控制对象的输入输出数据或操作过程中积累起来的经验信息,难以建立对象的精确的数学模型.对于该类系统,模糊神经网络自适应控制(FNNAC)方法提供了一条行之有效的解决途径[1-3].FNNAC 利用神经网络的自学习、强适应和强信息融合能力来实现模糊控制规则的自适应调整,适用于处理无精确模型的非线性控制问题.因此,该控制结构中的神经网络辨识器(NNI)和模糊神经网络控制器(FNNC)的参数优化便成为一个关键性问题.通常,二者大多采用梯度下降的方法和改进的遗传算法进行学习[4].梯度法要求目标函数连续可导,而遗传算法操作较为繁琐,不便于工程实现.

近几年倍受关注的微粒群算法(PSO),由于概念简单、工程实现容易,目前已成功地应用于控制领域的一些优化问题.在控制器设计方面,PSO 多用于对 PID 控制器以及模糊控制器进行参数优化[5-7],而对于待学习参数较多的 FNN 控制器优化的应用还比较少,这在相当程度上是因为多数改进的 PSO 算法在解决高维优化问题上效果不佳.Van den Bergh 等提出的协同微粒群算法[8](CPSO)通过降维处理、增加评价次数,有效地解决了这一问题,很适合解决诸如 NN 学习等高维优化问题.但是,CPSO-S_k(k 为分裂因子)可能出现伪最小值现

收稿日期: 2007-09-20; 修回日期: 2008-01-10.
基金项目: 国家自然科学基金项目(90405011, 90716028).
作者简介: 都延丽(1977-),女,陕西延安人,工程师,博士生,从事智能控制、飞行控制等研究; 吴庆宪(1955-),男,江苏扬州人,教授,博士生导师,从事鲁棒控制、智能控制方法等研究.

象, 并且算法不能保证收敛到局部或全局最小值[8]. 其后的改进算法 CPSO2Hk 能消除伪最小值现象, 但同样不能保证算法收敛到全局最优值. 此外, 文献[9]提出的随机 PSO(SPSO), 通过对优化向量施加随机变化因素, 增强了微粒的全局搜索能力.

本文将 SPSO 的思想引入 CPSO2Sk 的设计中, 提出一种能保证收敛到全局最优值的改进协同微粒群算法(ICPSO), 并将其应用于实际工业过程控制对象的 FNN 自适应控制系统的设计. 实验结果表明, ICPSO 优化的 NN 模型在精度和泛化能力上相对于 CPSO 等算法有明显的提高, 用于 FNNC 的离线和在线训练也收到了良好的控制效果, 值得进一步实验并扩展其应用.

2 改进协同微粒群算法

对于标准 PSO 以及多数改进 PSO, 它们具备一个共同特征, 即群体中的每一个微粒代表的都是可行解空间中一个完整的解向量, 因此每一步对微粒的更新都是在一个 n 维向量上进行的. 于是很有可能被优化向量的某些分量靠近了较优值, 而另外一些分量却远离了较优的值. 尤其对于高维优化问题, 这种现象更为常见. CPSO 将 n 维解空间分割成 K 个子空间, 在每一操作代中, 问题的解是基于 K 个子群体依次更新的, 从而有效地扩大了搜索范围, 克服了标准 PSO 算法在优化高维问题上性能急剧下降、容易陷入局部极小值的缺点. 但是, 正是由于每次更新只对部分分向量进行变动, 每步操作只能搜索一个子空间, 从而使得 CPSO 存在伪最小值的现象. 因此, 本文考虑在 CPSO 执行后再加入能使算法逃离伪最小值, 并保证收敛到全局最优值的操作, 即将随机 PSO 的思想引入 CPSO 的设计中, 构造一种能保证全局收敛的 CPSO 算法.

2.1 随机 PSO

标准 PSO 算法的进化方程为

$$v_{i,j}(t+1) = \chi[v_{i,j}(t) + c_1 r_{1,i}(t)[y_{i,j}(t) - x_{i,j}(t)] + c_2 r_{2,i}(t)[\hat{y}_j(t) - x_{i,j}(t)], \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1). \quad (2)$$

式中: $v_{i,j}$ 为第 i 个微粒在第 j 维搜索空间上的飞行速度; 权重因子 $\chi \in [0,1]$; x 为相应微粒的位置; $y_i(t)$ 为每个微粒经历过的最好位置, 整个微粒群迄今为止经历过的最好位置定义为 $\hat{y}(t)$.

随机 PSO 令 $\chi = 0$, 因此当 $x_k(t) = y_k(t) = \hat{y}(t) = P_g$ 时, 从式(1)可以得出 $v_k = 0$, 第 k 个微粒将停止进化, 微粒群将收缩到当前的最好位置, 从而使局部搜索能力增强, 全局搜索能力减弱. 为改善全局搜索能力, 随机 PSO 保留微粒群历史最好位置, 而在搜索空间中重新随机产生微粒 k 的位置 $x_k(t+1)$, 其他微粒 i 则以式(3)进化, 产生 $x_i(t+1)$, $i \neq k$. 算法具体表达如下:

$$\begin{aligned} P_k &= x_k(t+1), \text{ 随机产生;} \\ P_i &= \begin{cases} P_i, & f(P_i) < f(x_i(t+1)); \\ x_i(t+1), & f(P_i) \geqslant f(x_i(t+1)); \end{cases} \\ P_{\hat{g}} &= \arg\min\{f(P_i) \mid i = 1, \cdots, s\}; \\ P_g &= \arg\min\{f(P_{\hat{g}}), f(P_g)\}. \end{aligned} \quad (3)$$

此操作能保证算法以概率 1 收敛到全局最优值[9], 当然也就不会产生伪最小值现象.

2.2 ICPSO 涉及的基本概念

定义 1 子群. CPSO 将 s 个微粒中的每一个 n 维向量都划分为 K 个部分:

$$(x_{1,1}, x_{1,2}, \cdots, x_{1,n_1})_1, (x_{1,1}, x_{1,2}, \cdots, x_{1,n_1})_2, \cdots, (x_{1,1}, x_{1,2}, \cdots, x_{1,n_1})_{K_1}, (x_{1,1}, x_{1,2}, \cdots, x_{1,n_2})_{K_1+1}, \cdots, (x_{1,1}, x_{1,2}, \cdots, x_{1,n_2})_K;$$

$$\vdots$$

$$(x_{i,1}, x_{i,2}, \cdots, x_{i,n_1})_1, (x_{i,1}, x_{i,2}, \cdots, x_{i,n_1})_2, \cdots, (x_{i,1}, x_{i,2}, \cdots, x_{i,n_1})_{K_1}, (x_{i,1}, x_{i,2}, \cdots, x_{i,n_2})_{K_1+1}, \cdots, (x_{i,1}, x_{i,2}, \cdots, x_{i,n_2})_K;$$

$$\vdots$$

$$(x_{s,1}, x_{s,2}, \cdots, x_{s,n_1})_1, (x_{s,1}, x_{s,2}, \cdots, x_{s,n_1})_2, \cdots, (x_{s,1}, x_{s,2}, \cdots, x_{s,n_1})_{K_1}, (x_{s,1}, x_{s,2}, \cdots, x_{s,n_2})_{K_1+1}, \cdots, (x_{s,1}, x_{s,2}, \cdots, x_{s,n_2})_K.$$

其中: $K_1 = n \bmod K$, $K_2 = K - K_1$, $n = \lfloor n/K \rfloor + 1$, $n_2 = \lfloor n/K \rfloor$. 于是, 任意 n 维微粒群被划分为 K 个子群($K = K_1 + K_2$), 每个子群有 s 个个体, 前 K_1 个子群的个体维数是 n_1 维, 后 K_2 个子群的个体维数是 n_2 维. 子群划分也可以根据具体问题灵活处理.

定义 2 关联向量. CPSO 运行的每一个操作代中, 问题的解是基于 K 个子群体依次进行更新的, 不可能单独对更新的分向量进行评价, 因此需要设计如下关联向量:

$$b(j, z) = (P_1\hat{y}, \cdots, P_{j-1}\hat{y}, z, P_{j+1}\hat{y}, \cdots, P_K\hat{y}).$$

其中: $P_1 \sim P_K$ 代表 K 个子群, $P_j x_i$ 表示第 j 个子群中第 i 个微粒分量($0 \leqslant i \leqslant s$), $P_j y_i$ 为第 j 个子群中第 i 个微粒经历过的最好位置, $P_j\hat{y}_i$ 表示第 j 个子群中评价函数最优的个体. 这样, $b(j, z)$ 就代表了一个 n 维向量, 此向量除第 j 个子群涉及的分量, 其余分向量都由其他子群中评价函数最优的个体构成. 于是, $b(1, P_1\hat{y}) = (P_1\hat{y}, P_2\hat{y}, \cdots, P_K\hat{y})$ 就代表了当前最好的关联向量.

定义 3 最优粒子池. ICPSO 操作要在 CPSO 与 SPSO 操作中间传递当前最优的个体 $b(1, P_1\hat{y})$,

并不断地对此个体进行更新. 因此, 需在内存中开辟一块存储区, 专门用于存放 b(1, P1ŷ). 此存储区称为最优粒子池.

2.3　设计过程

在优化的开始阶段, 利用 CPSO2Sk 的快速收敛性, 首先执行 CPSO, 然后进行 SPSO 的操作, 利用 SPSO 的全局收敛性, 使整体算法通过多代的更新最终收敛到全局最优值. ICPSO 的伪代码表示如下:

定义 b(j, z) S (P1ŷ, , , Pj- 1ŷ, z, Pj+1ŷ, , , PKŷ)

K1 = nmodK; K2 = K - K1

n1 = 0n/K4, n2 = 8n/K <

初始化 K1 个 n1 维分量的子群 Pj, j = 1, , , K1

初始化 K2 个 n2 维分量的子群 Pj, j = K1 + 1, , , K

初始化 1 个 n 维 Q 群, 用于执行随机 PSO 操作

repeat:

for 每个子群 j I [1, , , K]

for 每个微粒 i I 1, , , s]

if f(b(j, PjXi)) < f(b(j, PjYi))

then PjYi = PjXi

if f(b(j, PjYi)) < f(b(j, Pjŷ))

then Pjŷ = PjYi

endfor

对 Pj 应用式(1) 和(2) 进行微粒更新

endfor

将 b(1, P1ŷ) 放入最优粒子池

设 X= 0, Q 群中随机选取个体满足 QyK X Qŷ

令 QxK = b(1, P1ŷ)

for 每个微粒 j I [1, , , s]

if f(Qxj) < f(Qyj)

then Qyj = Qxj

if f(Qyj) < f(Qŷ)

then Qŷ = Qyj

endfor

按式(3) 执行随机 PSO 操作

将 Qŷ 放入最优粒子池, 令其为关联向量

until 满足算法停止条件

2.4　收敛性分析

文献[10] 给出了随机优化算法以概率 1 收敛于全局最优解的条件, 其判断准则如下:

引理 1　$\{x_k\}_{k=0}^{]}$ 为随机算法产生的解序列, $R_{E,M}$ 为全局最优点集合, 当满足如下假设:

假设 1　若 f(D(x, N)) [f(x), N I S, 则 f(D(x, N)) [f(N);

假设 2　对于 S 的任意 Borel 子集 A, 若 Lebesgue 测度 v(A) > 0, 则

$$\prod_{k=0}^{]} [1 - L_k(A)] = 0.$$

从而有 $\lim_{k\to]} P[x_k \text{ I } R_{E,M}] = 1$ 成立. 其中: f 为一可测目标函数, D 为产生问题解的函数, S 为 R^n 的一可测子集, N 为从概率测度空间(R^n, B, Lk) 产生的随机向量, B 为 R^n 子集的 R2 代数, Lk 为 B 上的概率测度, P[xk I RE,M] 为算法第 k 步产生的解属于 RE,M 的概率.

定理 1　ICPSO 算法生成的解序列 $\{x_k\}_{k=0}^{]}$ 以概率 1 收敛于解空间 S 的全局最优值.

证明　ICPSO 算法的迭代函数 D 可定义为

$$D(p_{g,t}, x_i(t)) = \begin{cases} p_{g,t}, & f(p_{g,t}) \text{ [} f(x_i(t)); \quad (4) \\ x_i(t), & f(p_{g,t}) > f(x_i(t)). \quad (5) \end{cases}$$

若 f(D(pg,t, xi(t))) [f(pg,t), xi(t) I S, 则当式(4) f(pg,t) [f(xi(t)) 时, f(D(pg,t, xi(t))) = f(pg,t) [f(pg,t) 成立, 即满足 f(D(pg,t, xi(t))) = f(pg,t) [f(xi(t)); 若 f(D(pg,t, xi(t))) [f(pg,t), xi(t) I S, 则当式(5) f(pg,t) > f(xi(t)) 时, f(D(pg,t, xi(t))) = f(xi(t)) [f(pg,t) 成立, 也满足不等式 f(D(pg,t, xi(t))) [f(xi(t)). 即 ICPSO 算法满足假设 1.

要满足假设 2, 规模为 m 的微粒群样本空间的并必须包含 S, 即 $S \text{ A } \prod_{i=1}^{m} M_{i,k}$, 其中 Mi,k 为第 k 代微粒 i 的样本空间的支撑集. 由于 ICPSO 算法 P 群个体替换 Q 群某个体, 但并不替换掉其最优个体, 而是与最优个体一同进行评价, 即 P 群的个体并不影响 Q 群后半部分执行的 SPSO 算法. 故对于满足 xj(t) = pk = pg 的微粒 j, 根据式(3), pg 在 S 域中取随机值, 有 Mj,k = S. 而对于其他微粒 i 则有下式成立:

$$M_{i,k} = x_{i,j}(k) + c_1 r_{1,i}(t)[y_{i,j}(k) - x_{i,j}(k)] + c_2 r_{2,i}(t)[P_g - x_{i,j}(k)]. \quad (6)$$

由于 pg 的随机性, 同样 Mi,k = S. 因此, 取样空间的支集 Mk = S. 定义 S 的 Borel 子集 A = Mk, 则有 Lk[A] = Lk[Mk] = Lk[S] = 1, 从而满足假设 2. 因此由引理 1 可以得出, ICPSO 以概率 1 收敛于全局最优解. t

本文算法能保证收敛于最优, 也就能够避免出现伪最小值的现象, 并且 SPSO 的最优个体不断注入 CPSO2Sk 执行过程, 提高了 CPSO2Sk 算法的探测

能力, 使 CPSO2Sk 执行部分出现伪最小值的可能性大为降低.

3 ICPSO 优化的 FNN 自适应控制系统设计

以某药厂霜、液剂温度控制系统为例, 检验 ICPSO 进行控制系统优化的效果. 此系统是一典型带有热焓过程的慢变系统, 它通过调整调节阀的开度、改变水介质的流量来控制介质的温度, 经热传导最终控制加热罐内药剂的温度. 由于介质管道输送距离远, 其间阀门较多, 被控对象存在较大的纯滞后; 加热罐内药剂的种类经常根据生产的需要而发生改变, 致使对象的大时间常数时常发生变动, 而且以介质流量为调节手段来控制某个工艺参数的热焓过程, 从过程动态学的角度看, 是一种参数非线性过程. 因此, 该系统被控对象具有纯滞后、大惯性、非线性及参数不确定等特征, 易产生振荡或较大的超调, 应用传统的控制策略很难满足工艺要求的性能指标. 模糊神经网络不仅空间结构清晰, 而且具有良好的自学习和非线性逼近能力, 其特点有望较好地解决系统中存在的非线性和时变不确定问题, 故本文针对这类对象实施模糊神经网络自适应控制策略[11]. 系统结构如图 1 所示.

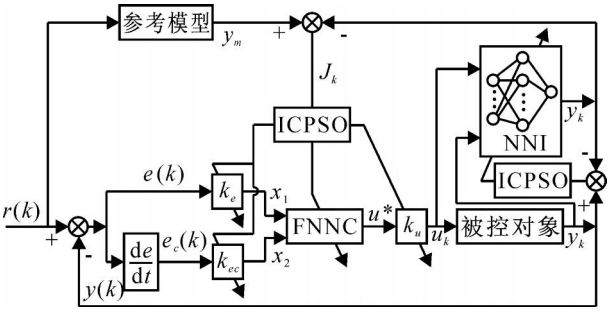


图 1 ICPSO 优化的模糊神经网络自适应控制系统

3.1 NNI 的建立

3.1.1 NNI 结构的确定

建模前采集到被控对象的输入/ 输出历史数据{u(k), y(k)} 共 160 组, 分别作为训练和测试样本集. 全部数据进行归一化处理, 使 u(k) I [- 1, 1], y(k) 由 0~ 100e 映射到区间[0, 1]. 以所采集数据为基础, 建立如图 1 中的神经网络辨识模型(NNI) 如下:

$$\hat{y}(k) = N(W(k), y(k-1), y(k-2), u(k-1), u(k-2)).$$

神经网络隐层的激励函数为双曲正切 S 型函数, 输出层激励函数为线性传输函数. 网络以系统的实际输出与辨识器输出(经过归一化处理的数值) 之差值的平方作为误差函数对神经网络进行离线训练. 定义误差函数为 E1, 其中 p 为学习样本的数目.

$$E_1 = \frac{1}{2}\sum_{i=1}^{p}(y_i - \hat{y}_i)^2. \quad (7)$$

应用改进浮点遗传算法(FGA) 选取 NNI 隐层神经元个数, 从 42421 结构开始到 42821 结束, 初始种群染色体个数设为 50. 经实验, 当模型为 42621 形式, 即权系数为 37 个时, 训练和测试数据集适应度数值(定义为 1/E1) 最大. 执行评价函数 15 @10^4 次后, 适应度为 147. 14, E1 = 1/ 147. 14 = 0. 0068, 误差变化如图 2 所示. 由于优化精度较低, 考虑应用 ICPSO 对 NNI 权系数进行学习.

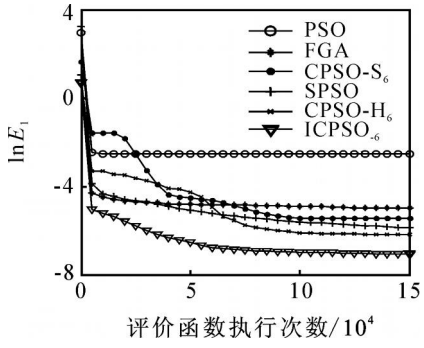


图 2 不同微粒群算法训练比较

3.1.2 ICPSO 学习 NNI 权值的过程

初始化生成两个种群 P 和 Q, 每个种群的微粒个数 s1 = s2 = 15, 个体向量由 37 个权系数排列而成, 因此维数 n= 37. P 群的分裂因子 K = 6, K1 = 37 mod 6= 1, K2 = K- K1 = 5, n1 = 0 37/ 6 4 = 7, n2 = 8 37/ 6 < = 6. 在前半部分 CPSO2Sk 算法执行过程中, 权重因子 X取值为服从正态分布的随机数值, 即 X~ N(0. 5, 0. 5^2), 限制范围为(0, 1], 改进后较之 X线性递减效果更好.

图 2 是 5 种不同的微粒群算法和 FGA 在测试数据下的训练比较, 编程环境为 Borland C++ 3. 1. 所有 PSO 算法的群体规模均为 15, v 和 x 变化范围是[- 2. 0, 2. 0], c1 = c2 = 1. 8, 惯性权重从 1 线性递减至 0. 1(除 ICPSO 外). 优化时每训练一次, 同时用测试数据测试一遍, 最终得到图 2 所示的测试数据一次实验均方误差曲线.

表 1 20 次测试数据结果比较

算法	测试数据集		
	平均误差	最大误差	最小误差
PSO	0. 144062	0. 937795	0. 000924
FGA	0. 007054	0. 007311	0. 006796
CPSO2S6	0. 003894	0. 009185	0. 0007043
SPSO	0. 002361	0. 007316	0. 000699
CPSO2H6	0. 002585	0. 006731	0. 000469
ICPSO26	0. 001331	0. 003929	0. 000232

从图 2 可以看出, 训练开始阶段 ICPSO 算法误差降低非常明显, 后期误差值达到最小. 表 1 同时列出了几种算法各执行 20 次实验的误差结果.

数据表明, 与其他算法相比, ICPSO 优化的平均误差最小, 说明算法经过改进不仅提高了优化精度, 而且进一步提高了 NNI 的泛化能力. 另外, ICPSO 最大误差与最小误差之间差值不大, 即学习过程比较稳定. 由此可见, ICPSO 是一种较为稳健的全局优化算法, 收敛精度高, 学习的 NNI 泛化能力较强.

选取 ICPSO 优化的一次结果作为 NNI 的权值参数, 将其作为图 1 的一个组成部分.

3.2　FNNC 的优化

为方便工程应用, 图 1 中的 FNNC 模型描述为

R_i: If e is A_1^i and e_c is A_2^i,

Then u is B^i, $i = 1, 2, \cdots, 5$.

其中: $e(k) = r(k) - y(k)$, $e_c(k) = e(k) - e(k-1)$; $A_1^i = [NB, NS, Z, PS, PB]$, $A_2^i = [NB, NS, Z, PS, PB]$. 设各变量论域为 $e = [-E, +E] = [-15, +15]$, $e_c = [-\hat{E}, +\hat{E}] = [-1.5, +1.5]$, $U = u_k = [-1, +1]$.

量化因子的初值设为 $K_{e_c} = n/e_c = 5/1.5 = 10/3$, $K_e = n/e = 5/15 = 1/3$, $K_u = U/n = 1/5 = 0.2$.

FNNC 采用的是基于标准模型的模糊神经网络结构[11], 具体结构如下:

第 1 层为输入层, 有 2 个节点 x_1 和 x_2. 如图 1 中 $x_1 = e \# K_e$, $x_2 = e_c \# K_{e_c}$.

第 2 层的节点从上到下依次为属于 x_1 和 x_2 的语言变量, 二者的隶属函数为

$$L_{ij} = e^{-(x_i - c_{ij})^2 / R_{ij}^2}, \quad i = 1, 2,\ j = 1, 2, \cdots, 5.$$

x_1 的初始隶属度函数设为

NB: $L_{A_1^1}(e) = \exp[-(x_1 + 4)^2 / 1.0^2]$,

NS: $L_{A_1^1}(e) = \exp[-(x_1 + 2)^2 / 1.0^2]$,

Z: $L_{A_1^1}(e) = \exp[-(x_1 + 0)^2 / 1.0^2]$,

PS: $L_{A_1^1}(e) = \exp[-(x_1 - 2)^2 / 1.0^2]$,

PB: $L_{A_1^1}(e) = \exp[-(x_1 - 4)^2 / 1.0^2]$.

x_2 的初始参数设置同 x_1.

第 3 层的每个节点代表一条模糊规则, 用于计算出每条规则的适用度, 即 $A_j = \min\{L_{i_1}^1, L_{i_2}^2\}$. 其中: $i_1 \in \{1, \cdots, 5\}$, $i_2 \in \{1, \cdots, 5\}$, $j = 1, \cdots, 25$.

第 4 层实现归一化计算, 即

$$A_j = A_j / \left(\sum_{j=1}^{25} A_j\right), \quad j = 1, 2, \cdots, 25.$$

第 5 层是输出层, 它实现清晰化计算, 即

$$u = \sum_{j=1}^{25} w_j A_j = (w_1 \ w_2 \ \cdots \ w_{25})(A_1 \ A_2 \ \cdots \ A_m)^T.$$

其中 c_{ij}, R_{ij} $(i = 1, 2,\ j = 1, 2, \cdots, 5)$ 以及 $w_1, \cdots, w_{25}$ 共 45 个参数为 FNNC 离线优化的参数.

3.2.1　ICPSO 离线优化 FNNC 控制律

根据系统的先验知识及实验结果, 经模糊推导, 总结出粗略的控制律如表 2 所示.

将表 2 作为训练样本集, 应用 ICPSO 对 FNNC 进行离线训练, 误差函数定义如下:

$$E_2 = \frac{1}{2}\sum_{i=1}^{121} (u_i^* - \hat{u}_i^*)^2. \tag{8}$$

ICPSO 的 P 和 Q 种群的微粒个数 $s_1 = s_2 = 15$, 被优化个体维数 $n = 45$. 设 P 群的分裂因子 $K = 9$, 则 $K_1 = 45 \bmod 9 = 0$, $K_2 = K - K_1 = 9$, $n_1 = n_2 =$

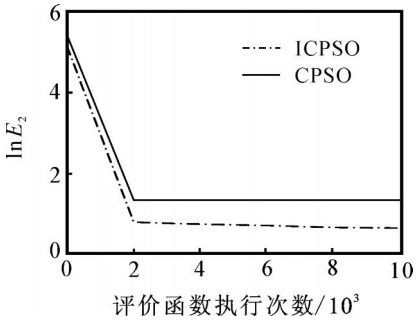


图 3　ICPSO 离线训练 FNNC 的误差变化

表 2　初始控制律

E \ E_c	- 5	- 4	- 3	- 2	- 1	0	1	2	3	4	5
- 5	- 5.0	- 5.0	- 5.0	- 5.0	- 4.5	- 4.5	- 4.0	- 4.0	- 3.5	- 3.5	- 3.0
- 4	- 4.5	- 4.5	- 4.0	- 4.0	- 3.5	- 2.5	- 2.0	- 2.0	- 2.0	- 1.5	- 1.5
- 3	- 4.0	- 3.5	- 3.0	- 2.5	- 2.0	- 2.0	- 1.5	- 1.5	- 1.5	- 1.0	- 1.0
- 2	- 2.5	- 2.5	- 2.0	- 1.5	- 1.5	- 1.0	- 1.0	- 0.5	- 0.5	0.0	0.0
- 1	- 2.0	- 2.0	- 1.5	- 1.0	- 1.0	- 0.5	- 0.5	0.0	0.0	0.0	0.5
0	- 1.0	- 0.5	0.0	0.0	0.5	1.5	2.5	3.0	3.0	3.5	3.5
1	1.0	1.5	1.75	2.5	3.0	4.0	4.0	4.5	4.5	5.0	5.0
2	2.5	3.25	3.5	3.6	3.75	4.0	4.0	4.5	4.5	5.0	5.0
3	3.5	3.75	4.0	4.1	4.25	4.5	4.5	5.0	5.0	5.0	5.0
4	3.75	4.0	4.1	4.25	4.5	4.5	4.5	5.0	5.0	5.0	5.0
5	4.4	4.5	4.6	4.75	4.9	5.0	5.0	5.0	5.0	5.0	5.0

0.45/9.4 = 5. 应用 ICPSO 和 CPSO 离线训练 FNNC 的误差变化如图 3 所示. 执行 1@10^4 次评价函数后, E_2 分别等于 1.9341 和 3.85479. 可见, 用 ICPSO 训练 FNNC 的精度明显高于 CPSO 训练的精度.

3.2.2 ICPSO 在线优化 FNNC

以一次离线优化得到的参数值作为 FNNC 在线学习的初始值, 同时加入 3 个量化因子的在线调整, 则微粒个体的维数 n = 48. 设种群的微粒个数 $s_1 = s_2 = 10$. P 群的 $K = 10, K_1 = 9, K_2 = 1, n_1 = 5, n_2 = 3$. 参数调整范围设为 $w \in [-2.0, 2.0]$, c_{ij}, $R_i \in [-1.0, 1.0]$, 在线学习的误差函数定义为

$$E_3 = \frac{1}{2}[y_k(k) - y_m(k)]^2. \qquad (9)$$

参考模型定义为 $y_m(k+1) = r(k)$.

NNI 经训练后 $\hat{y}_k \approx y_k$, 因此, 仿真过程中以 $\hat{y}_k$ 代替实际对象的输出对 FNNC 进行在线学习. 应用 C 语言实现的具体过程如下:

Step1: 读取 ICPSO 的初始设置参数;

Step2: 读取系统误差 e(k) 和误差导数 $e_c(k)$;

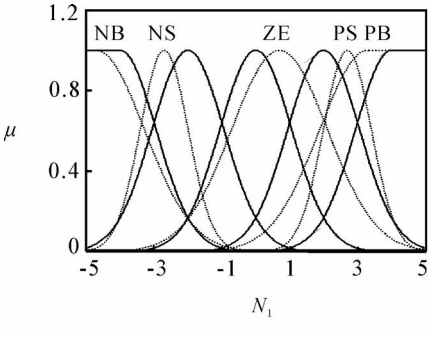
Step3: 实施 ICPSO 操作, 产生一组优化参数, 将参数代入 FNNC, 计算输出 $u^*(k)$;

Step4: 计算 NNI 的输出, 得到 $\hat{y}(k+1)$;

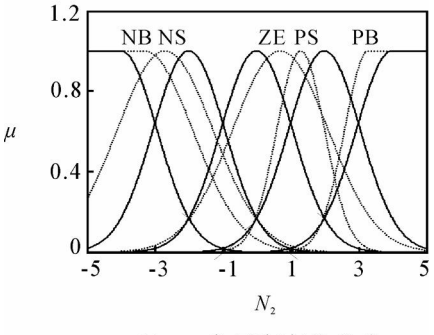
Step5: 达到采样次数, 结束; 否则, 转 Step2.

4 仿真结果及分析

执行仿真程序时, 每次采样后, ICPSO 操作的终止条件都是更新20代结束. 程序执行完毕后, 单次采样时间内 ICPSO 优化参数的时间 < 0.055s(Pentium IV 1.7G CPU), 训练时间较快, 因而方便工程实现.



(a) e 隶属度变化曲线



(b) e_c 隶属度变化曲线

图 4 隶属函数变化曲线

e 和 e_c 的隶属函数调整见图 4, 图中实线为 e 和 e_c 优化前的初始隶属函数, 虚线为训练后的隶属函数曲线. 图 5 是执行仿真程序后得到的系统输出曲线. 应用 FGA 建立 NNI 的精度较低, 因此 FGA 在线优化 FNNC 达到的稳态误差较大. 比较而言, ICPSO 优化 FNNC 的控制效果较好.

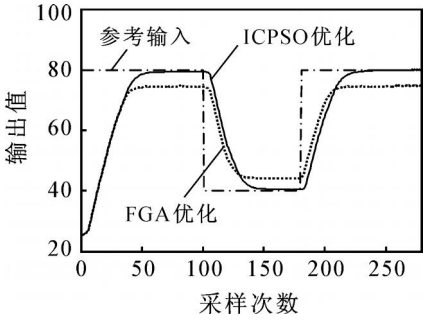


图 5 输出值变化

5 结 论

针对协同 PSO 不能保证全局收敛以及训练过程中易出现伪最小值现象的缺点, 本文提出了 ICPSO 算法. 分析表明, 此算法能保证以概率 1 收敛于全局最优值, 并具有求解高维优化问题的优越性. 在一类非线性对象的神经网络建模和 FNN 控制器优化的应用中, ICPSO 与实验的其他算法相比, 所训练的神经网络模型精度较高、泛化能力较强. 同时, 优化 FNN 控制器较之 FGA 达到了更好的控制效果. 算法编程实现容易, 优化速度较快, 有利于在实际系统建模和控制优化中进一步应用.

参考文献(References)

[1] Lin F J, Shen P H. Adaptive fuzzy-neural-network control for a DSP-based permanent magnet linear synchronous motor servo drive[J]. IEEE Trans on Fuzzy Systems, 2006, 14(4): 481-495.

[2] Chen C H, Lin C T, Lin C J. A functional-link-based fuzzy neural network for temperature control[C]. Proc of the 2007 IEEE Symposium on Foundations of Computational Intelligence. Hawaii, 2007: 53-58.

[3] 张承慧, 石庆升, 程金. 一种多电机同步传动模糊神经网络控制器的设计[J]. 控制与决策, 2007, 22(1): 30-34.
(Zhang C H, Shi Q S, Cheng J. Design of fuzzy neural network controller for synchronization drive in multi-motor systems[J]. Control and Decision, 2007, 22(1): 30-34.)

(下转第 1337 页)

状态估计的精度.

6 结　　论

本文提出了一种基于粒子滤波的模型自适应机动目标跟踪算法. 在粒子滤波算法框架下, 将模型信息引入到粒子的采样中, 实现了对模型的准确辨识, 进而提高了状态估计的精度. 同时, 该算法不受模型非线性条件的限制, 可有效解决一些 IMM 算法失效的情况: 高度非线性运动模型, 高度非线性测量模型及各种机动目标的跟踪. 另外, 在实时性和精度方面本文算法也优于 IMMPF 算法.

参考文献(References)

[1] Johnston L A, Krishnamurthy V. An improvement to the interacting multiple model (IMM) algorithm[J]. IEEE Trans on Signal Processing, 2001, 49(12): 2909-2923.

[2] Derbez E, Remillard B, Jouan A. A comparison of fixed gain IMM against two other filters[C]. Proc of 2000 Int Conf on Information Fusion. Paris, 2000: 10-13.

[3] 梁彦, 程咏梅, 贾宇岗, 等. 交互式多模型算法性能分析[J]. 控制理论与应用, 2001, 18(4): 487-492.
(Liang Y, Chen Y M, Jia Y G, et al. Analysis on the performance and properties of interacting multiple models algorithm[J]. Control Theory and Applications, 2001, 18(4): 487-492.)

[4] Kirubarajan T, Bar-Shalom Y. Kalman filter versus IMM estimator: When do we need the latter[J]. IEEE Trans on Aerospace and Electronic Systems, 2003, 39(4): 1452-1457.

[5] Lenon C, Rodney W. Sensor fault detection for UAVs using a nonlinear dynamic model and the IMM-UKF algorithm[C]. 2007 Information, Decision and Control. Adelaide, 2007: 1-6.

[6] Doucet A, Godsill S J, Andrieu C. On sequential Monte Carlo sampling methods for Bayesian filtering[J]. Statistics and Computing, 2000, 10(3): 197-208.

[7] Andrieu C, De Freitas N, Doucet A, et al. An introduction to MCMC for machine learning[J]. Machine Learning, 2003, 50(1/2): 5-43.

[8] Boers Y, Driessen J N. Interacting multiple model particle filter[J]. IEE Proc of Radar Sonar Navigation, 2003, 150(5): 334-2349.

[9] Doucet A, Gordon N J, Vikram K. Particle filters for state estimation of jump Markov linear systems[J]. IEEE Trans on Signal Processing, 2001, 49(3): 613-624.

[10] Gordon N J, Salmond D J, Smith A F M. Novel approach to nonlinear/non-Gussian Bayesian state estimation[J]. IEE Proc of Radar Signal Process, 1993, 140(2): 107-113.

[11] Gordon N J. A hybrid bootstrap filter for target tracking in clutter[J]. IEEE Trans on Aerospace and Electronic Systems, 1997, 33(3): 353-2358.

[12] Liu J S, Chen R. Sequential Monte Carlo methods for dynamic systems[J]. J of the American Statistical Association, 1998, 93(443): 1032-1044.

[13] Doucet A, Godsill S. On sequential Monte Carlo sampling methods for Bayesian filtering[R]. Cambridge: Department of Engineering University of Cambridge, 1998.

(上接第 1332 页)

[4] Zou Q Y, Jiang C S, Wu D. Evolutionary fuzzy guidance law with self-adaptive region[J]. Transactions of Nanjing University of Aeronautics & Astronautics, 2004, 21(3): 234-2240.

[5] Mukherjee V, Ghoshal S P. Intelligent particle swarm optimized fuzzy PID controller for AVR system[J]. Electric Power System Research, 2007, 77(12): 1689-1698.

[6] Kao C C, Chuang C W, Fung R F. The self-tuning PID control in a slider-crank mechanism system by applying particle swarm optimization approach[J]. Mechatronics, 2006, 16(8): 513-2522.

[7] 郝万君, 强文义, 柴庆宣, 等. 基于粒子群优化的一类模糊控制器设计[J]. 控制与决策, 2007, 22(5): 585-588.
(Hao W J, Qiang W Y, Chai Q X, et al. Design of fuzzy controller based on particle swarm optimization[J]. Control and Decision, 2007, 22(5): 585-2588.)

[8] Van den Bergh F, Engelbrech A P. A cooperative approach to particle swarm optimization[J]. IEEE Trans on Evolutionary Computation, 2004, 8(3): 225-2239.

[9] 曾建潮, 崔志华. 一种保证全局收敛的 PSO 算法[J]. 计算机研究与发展, 2004, 41(8): 1333-1338.
(Zeng J C, Cui Z H. A guaranteed global convergence particle swarm optimizer[J]. J of Computer Research and Development, 2004, 41(8): 1333-1338.)

[10] Solis F, Wets R. Minimization by random research techniques[J]. Mathematics of Operations Research, 1981, 6(1): 19-230.

[11] 胡德文, 王正志, 王耀南, 等. 神经网络自适应控制[M]. 长沙: 国防科技大学出版社, 2006.
(Hu D W, Wang Z Z, Wang Y N, et al. Neural networks for adaptive control[M]. Changsha: National University of Defence Technology Press, 2006.)