微粒群算法参数效能的统计分析

彭 宇, 彭喜元, 刘兆庆
(哈尔滨工业大学 339#, 黑龙江哈尔滨 150001)

摘 要: 微粒群算法已在函数优化和组合优化问题中取得了比较理想的应用效果, 其显著特点是算法表达简单, 设置参数少, 易于操作. 但是其中关键参数的选择方法对算法特性有显著影响. 本文利用经典统计分析中的方差分析方法, 针对基本微粒群算法中的惯性权、加速常数 C_1 和 C_2 的设置对算法基本性能的影响进行了分析. 实验结果证明: 按照方差分析选择适应的参数设置水平, 能够获得稳健和高效的优化效果.

关键词: 微粒群算法; 参数设置; 方差分析
中图分类号: TP18 文献标识码: A 文章编号: 0372-2112 (2004) 02-0209-05

Statistic Analysis on Parameter Efficiency of Particle Swarm Optimization

PENG Yu, PENG Xi-yuan, LIU Zhao-qing
(No. 339#, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China)

Abstract: Satisfactory results have been acquired in functional and combinational optimization by adopting particle swarm optimization. Its significant feature is simpler expression, less parameters and easier operation. However, selection of key parameters has great influence on algorithm effects. So the variance analysis of statistic theory is applied to analyze the effect of inertia weight and accelerating constants. It was proved that robust and effective optimization can be achieved by setting parameters according to the variance analysis results discussed.

Key words: swarm optimization; parameter efficiency; statistic analysis

1 引言

PSO 是 Kennedy 和 Eberhart 在 1995 年提出的一种新型计算智能方法[1], 该算法最初是受到飞鸟集群活动的规律性启发, 进而利用群体智能建立的一个简化模型. 微粒群算法在对动物集群活动行为观察的基础上, 利用群体中的个体对信息的共享使整个群体的运动在问题求解空间中产生从无序到有序的演化过程, 从而获取最优解[2]. 研究初期, 通过对近邻的速度匹配、消除不必要的变量、并考虑了多维搜索和根据距离的加速, 形成了 PSO 的雏形. 随后引入了惯性权重来实现对解空间搜索更好的控制, 逐步形成了目前普遍应用的基本微粒群算法[3, 4].

微粒群算法简单的计算形式和参数设置, 良好的算法收敛性已经吸引了越来越多的关注. 在优化问题研究领域显示出了一定的潜力, 包括带约束的优化问题在内的多元函数优化是微粒群算法最直接的应用[5]. 控制系统中的控制参数选择和寻优, 神经网络的训练都已成为微粒群算法的主要应用领域. 在一些研究实验中, 随着优化问题复杂程度的加深, 微粒群算法比遗传算法显示出更强的适用性[6, 7]. 同时, 微粒群算法也被用于了应用优化问题, 如: 对特殊的电气设备的功率反馈和电压进行控制, 实现虚拟仪器参数自适应优化设置[8].

虽然微粒群算法发展迅速并取得了可观的研究成果, 但其理论基础仍相对薄弱. 尤其是算法基本模型中的参数设置和优化问题还缺乏成熟的理论论证和研究. 目前比较通用的参数设置方法是经验法和实验法. A. Eberhart 在其研究论文中利用统计方法分析了微粒群规模、速度极限和迭代次数对算法性能的影响[9]. 但是这三个参数与具体问题的特性密切相关, 并不能完全说明参数对算法基本特性的影响. 因此本文利用统计中的方差分析, 通过抽样实验方法, 论证了微粒群算法中关键参数因子: 惯性权、加速常数对算法整体性能的影响效果, 并提出了参数设置的指导原则.

2 PSO 算法

2.1 算法原理

微粒群算法和其他演化算法相似, 也是根据对环境的适应度将群体中的个体移动到好的区域, 因此有人认为它属于

收稿日期: 2003-03-26; 修回日期: 2003-07-16
基金项目: 高等学校博士学科点专项科研基金(No. 20020213012)

演化算法的一种. 不同之处在于它不象其它演化算法一样对个体使用演化算子, 而将每个个体看作是 d 维搜索空间中的一个没有体积没有质量的微粒, 在搜索空间中以一定的速度飞行, 并根据对个体和集体的飞行经验的综合分析来动态调整这个速度. 设群体中第 i 个微粒为$X_i=(x_{i1}, x_{i2}, \cdots, x_{id})$, 它经历过的位置为 $P_i=(p_{i1}, p_{i2}, \cdots, p_{id})$, 其中最佳位置记为 pbest. 当前组成群体的所有微粒经历过的最佳位置记为 pgbest, 微粒 i 的速度用 $V_i=(V_{i1}, V_{i2}, \cdots, V_{id})$ 表示. 对每一次迭代, 微粒 i 在 d 维(1[d[D)空间的运动遵循如下方程进行[10]:

$$v_{id}^{k+1}= X* v_{id}^{k}+ c_1* Rand()* (p_{id}- x_{id}^{k}) + c_2* Rand()* (pgbest^{k}- x_{id}^{k}) \quad (1)$$

$$x_{id}^{k+1}= x_{id}^{k}+ v_{id}^{k} \quad (2)$$

其中 X 为惯性权(inertia weight), 它使微粒保持运动的惯性, 使其有能力探索新的区域. c_1 和 c_2 为加速常数, 它们使每个微粒向 pbest 和 pgbest 位置加速运动. Rand() 为[0, 1]范围里变化的随机数. 此外, 微粒的速度 V_i 被一个最大速度 V_{max} 所限制. 如果当前对微粒的加速将导致它在某维的速度 v_{id} 超过该维的最大速度 V_{maxd}, 则该维的速度被限制为该维最大速度 V_{maxd}. 它决定了微粒在解空间的搜索精度, 如果 V_{max} 太高, 微粒可能会飞过最优解, 如果 V_{max} 太小, 微粒陷入局部搜索空间而无法进行全局搜索.

212 算法流程

基本微粒群算法流程如下[10]:

(a) 初始化一群规模为 m 的微粒, 其中所有微粒的位置和速度都是随机的;

(b) 对每个微粒, 将它的当前位置和它经历过的最好位置 pbest 作比较, 如果当前位置更好, 则将其作为当前的最好位置 pbest;

(c) 对每个微粒, 将它的当前位置和群体中所有微粒所经历的最好位置 pgbest 作比较, 如果这个微粒的位置更好, 则将其设置为当前的最好位置 pgbest;

(d) 根据方程(1)和(2)更新微粒的速度和位置;

(e) 如未达到结束条件(通常为预设的运算精度或迭代次数), 返回步骤 b, 开始下一轮迭代计算. 否则, 取当前 pgbest 作为最优解.

213 算法解析

公式(1)中, 第一部分为微粒当前速度乘以权值进行加速, 表示微粒对当前自身运动状态的信任, 依据自身的速度进行惯性运动. 第二部分为0认知0部分, 表示微粒本身的思考, 即一个得到加强的随机行为在将来的出现几率增大. 这里的行为即0认知0, 并假设获得正确的知识是得到加强的, 从而实现一个增强学习过程. 第三部分为0社会0部分, 表示微粒间的信息共享与相互合作. 也就是当观察到一个模型在加强某一行为时, 将增加它实行该行为的概率. 即微粒本身的认知将被其它微粒所模仿.

微粒群算法的这些心理学假设是无争议的, 即在群体寻求一致的认知过程中, 个体往往记住自身的信念, 同时考虑其他个体的信念. 当个体察觉其他个体的信念较好的时候, 它将进行适应性调整.

公式(2)表示了微粒在求解空间中, 由于相互影响导致的运动位置调整. 整个求解过程中, 惯性权重 X、加速常数 c_1 和 c_2 和最大速度 V_{max} 共同维护微粒对全局和局部搜索能力的平衡. 这四个算法参数的设置与具体问题密切相关, 目前比较常用的方法是针对具体求解问题, 利用充分的实验来确定.

3 参数特性的统计分析

由于算法中四个主要参数与具体问题有着密不可分的联系, 目前针对这些算法参数选择问题的普遍性研究还很欠缺. J Kennedy 和 R Eberhart 曾在论文中建议设置 $c_1= c_2= 2.0$ 以保证公式(1)中的随机乘积均值等于 1. 另外, 他们还建议采用惯性权动态调整算法, 在 0.8< X< 1.4 范围内从大到小逐步调整其取值, 从而实现搜索空间由全局向局部的稳步过渡[2]. Gerhard Venter 的研究论文中指出适当减少每个微粒自身的信任度(较小的 c_1), 并适当增加对群体的信任度(较大的 c_2)在结构设计优化中能够获得更好的效果[11]. 对这些研究成果的分析说明算法参数选择的理论性研究和实验分析论证是非常必要的. 只有充分了解参数设置的基本问题, 才能够为利用微粒群算法解决具体优化问题提供更加明确的算法特性说明. 以下将利用统计分析中的试验因子方差分析方法, 对微粒群算法的参数设置与算法系统性能之间的内在联系程度进行剖析.

311 方差分析

方差分析(analysis of variance)是分析试验数据的一种方法. 利用方差分析可以分析抽样测得的实验数据, 在不同观测条件(即同一因素的不同水平或不同因素的各个水平)所得试验结果的异同, 从而, 探求在系统性差异下观测条件的不同而引起的试验结果变化. 利用此方法亦可分析算法中同一参数的不同水平或者不同参数的各个水平对算法性能影响的差异性, 从而探究不同参数设置范围与算法系统性能之间的潜在关系[12].

单因子方差分析是通过观察一个因子的量值变化, 分析这个因子变化对整个试验的影响程度. 利用这种方法可考查微粒群算法中 c_1 和 c_2 这两个关键的参数因子各自对算法性能的影响. 在试验中, 为了评价试验的性质需要进行多次测试, 其结果就是被测试方法的性能指标. 在试验中影响指标的因素称为因子, 因子所处的状态, 所取的等级称为因子水平. 在本文的算法性能研究中, 采取相等的试验次数进行方差分析. 首先, 假定因子 A 有 m 个水平, 在每种水平下, 做 k 次试验, 在每次试验后可得一试验值, 记做 x_{ij}, 它表示在第 i 个水平下的第 j 个试验值 i= 1, 2, ⋯, m; j= 1, 2, ⋯, k, 如表 1 所示.

在考察因子 A 对试验结果的影响程度时, 把因子 A 的 m 个水平 $A_1, A_2, \cdots, A_m$ 看成是 m 个正态总体, 因此可设 $X_{ij}\sim N(a_i, R^2)$, i= 1, 2, ⋯, m, j= 1, 2, ⋯, k. 其中, $a_i= L+ E_i$, E_i 是因子 A 的第 i 水平 A_i 所引起的差异. 因此检验因子 A 的各水平之间是否有显著的差异, 相当于判断公式(3):

H_{01}: $a_1 = a_2 = , = a_m = L$ 或 H_{01}: $E_1 = E_2 = , = E_m = 0$ (3)

表 1　单因子方差分析

试验次数 / 因子水平	1, 2, , ,j, , , k	$\sum_{j=1}^{k} x_{ij}$	$x_i = \frac{1}{k}\sum_{j=1}^{k} x_{ij}$
A_1	$x_{11}, x_{12}, , , x_{1j}, , , x_{1k}$	$\sum x_{1j}$	x_1
A_2	$x_{21}, x_{22}, , , x_{2j}, , , x_{2k}$	$\sum x_{2j}$	x_2
,	,	,	,
A_i	$x_{i1}, x_{i2}, , , x_{ij}, , , x_{ik}$	$\sum x_{ij}$	x_i
,	,	,	,
A_m	$x_{m1}, x_{m2}, , , x_{mj}, , , x_{mk}$	$\sum x_{mj}$	x_m

利用公式(4)表述的平方和 S_T 分解公式可将总的离差平方和进行分解,从而将因子水平不同而造成的结果差异与随机因素影响而造成的结果差异从量值上区分开来.

$$S_T = S_e + S_A,\ S_T = \sum_{i=1}^{m}\sum_{j=1}^{k}(x_{ij} - \bar{x})^2 \qquad (4)$$

其中: $S_A = \sum_{i=1}^{m}\sum_{j=1}^{k}(\bar{x}_i - \bar{x})^2 = k\sum_{i=1}^{m}(\bar{x}_i - \bar{x})^2$, $S_e = \sum_{i=1}^{m}\sum_{j=1}^{k}(x_{ij} - \bar{x}_i)^2$, $\bar{x}_i = \frac{1}{k}\sum_{j=1}^{k} x_{ij}$, $\bar{x} = \frac{1}{m}\sum_{i=1}^{m}\bar{x}_i = \frac{1}{mk}\sum_{i=1}^{m}\sum_{j=1}^{k}\bar{x}_{ij}$

总离差平方和 S_T 是所有观察值 x_{ij} 与其总平均值 $\bar{x}$ 之差的平方和,是描述全部数据离散程度的数量指标. 由于 x_{ij} 是服从正态分布的随机量,当公式(2)成立时 x_{ij} 是独立同分布,同正态分布的随机变量,则有公式(5)是服从 $f_T = mk - 1$ 的 V^2 分布:

$$\frac{S_T}{R^2} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{k}(x_{ij} - \bar{x})^2}{R^2} \qquad (5)$$

S_e 是观察值 x_{ij} 与组内平均值 $\bar{x}_i$ 之差的平方和,也就是组内平均和,它反映了组内(同一水平下)样本的随机波动,其自由度为 $f_e = mk - m$. S_A 是组内平均值 $\bar{x}_i$ 与总平均值之差的平方和,即组间平方和. 它在一定程度上反映了因子各个水平不同而引起的差异. 其自由度为 $f_A = m - 1$.

平方和分解公式说明观察值关于其总平均值之间的差异是由组内平方和组间平方和组成的. 因此,公式(5)表示的 S_A 和 S_e 之间的比值 F 就反映了两种差异所占的比重.

$$F = \frac{S_A/(m-1)}{S_e/m(k-1)} \qquad (6)$$

F 越大说明因子各水平不同引起的差异越显著,所以统计量 F 可用来检验各因子的影响效应.

3.2　实验方法

利用 J. D. Schaffer 提出的函数:

$$f(X) = \frac{\sin^2\sqrt{x_1^2 + x_2^2} - 0.5}{[1 + 0.001(x_1^2 + x_2^2)]^2} + 0.5, \quad |x_i| \le 100$$

作为测试函数,其最优状态和最优值为: $\min(f(X^*)) = f(0, 0) = 0$. 此函数在距全局最优点大约 3.14 范围内存在无穷多个局部极小将其包围,并且函数强烈振荡. 在算法实验设计中,微粒群规模设为 160, $V_{max} = 100$. 分别选择三个主要参数惯性权重 X、加速常数 c_1 和 c_2 的不同设置水平,每个设置水平进行 10 次测试. 通过单因子方差分析,说明不同参数水平对算法速率性能)迭代次数和算法优化性能)近似最优解的影响能力. 各参数的不同设置水平如下表所示.

表 2　单因子方差分析参数设置水平

	1	2	3	4	5	6	相关条件
X	0.3	0.4	0.5	0.6	0.7	0.8	$c_1 = c_2 = 2.0$
c_1	1.5	1.6	1.7	1.8	1.9	2.0	$c_2 = 2.0$, 动态调整
c_2	1.5	1.6	1.7	1.8	1.9	2.0	$c_1 = 2.0$, 动态调整

实验分析结果以盒子图方式表示. 其中的横坐标对应各次实验中不同的参数设置水平,纵坐标分别代表算法迭代次数或算法最小误差平方和. 高位横线和低位横线代表一组实验数据中的最大值和最小值,中位线代表一组数据的均值. 各盒子中位线水平的不同反映了各组实验结果的组间差异.

通过表 3、表 5 和表 7 的方差分析数据可知这三个参数对算法整体性能的影响都是很显著的. 而惯性权的作用更为显著,因为其 Prob> F 的量值远远小于两个加速常数. 由图 1(图中横坐标对应表 1 中不同的参数设置水平)可知在 0.3 F XF 0.5 的范围内,能够获得比较一致的迭代次数均值,且在此范围内进行更细致的单因子方差分析进一步证明较小的惯性权值能够提高算法速率. 表 4 列出了 X 设置水平对优化效果的单因子方差分析结果. 其中(Prob> F) = 0.4741> 0.05,故在此取值范围内 X 设置水平差异不会导致优化效果的显著变化. 也就是说,在需要较高计算速率的应用中,可适当减小惯性权值.

表 3　X设置水平对算法迭代次数的单因子方差分析

	SS	df	MS	F	Prob> F
Columns	295418.8	5	59083.3	11.07	2.2786e-7
Error	288180.9	54	5336.7		
Total	583599.6	59			

表 4　X设置水平对优化效果的单因子方差分析

	SS	df	MS	F	Prob> F
Columns	5.2236e-9	6	8.706e-10	0.94	0.4741
Error	5.84471e-8	63	9.27732e-10		
Total	6.36707e-8	69			

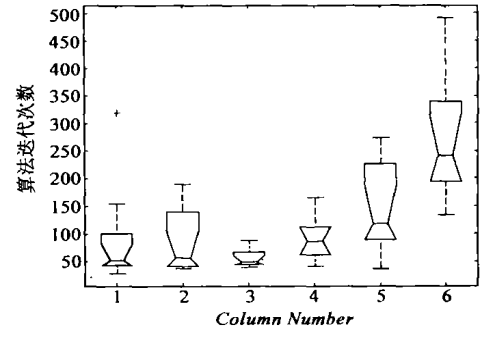

图 1　X 设置水平对迭代次数的影响

由图 2 可见, c_1 在[1.6, 1.8]范围内具有比较一致的算法迭代次数. 表 6 说明在 1.5 F c_1 F 2.0 范围内, c_1 对算法优化效果的影响是很小的. 这也证明在整个群体搜索解空间的过程中, 个体加速特性的改变对算法整体特性的影响是比较弱的. 另外, 从图 4 可以看出, 同样是在[1.6, 1.8]范围内, 算法具有整个实验范围内相对较好的优化效果. 因此在改善算法性能时, c_1 不必作为优先考虑的因素.

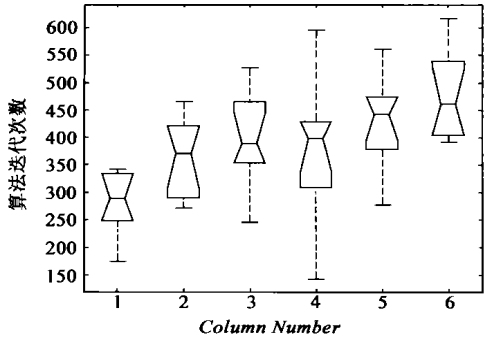

图 2 c_1 设置水平对迭代次数的影响

由图 3 可见, 在[1.6, 1.9]范围内具有相对比较一致的算法迭代次数. 但是通过 c_2 对算法精度影响的单因子方差分析可知 c_2 的变化对算法优化效能有着显著的影响. 表 8 和图 5 充分说明了这一问题. 这个结果再次证明了微粒间信息共享与相互合作的重要性. 因此, 在优化效果要求较高的应用中, 应优先考虑调整 c_2 的设置水平, 以改变微粒间相互影响的程度, 从而提高每个个体察觉其他个体信念的能力, 实现更好的适应性调整.

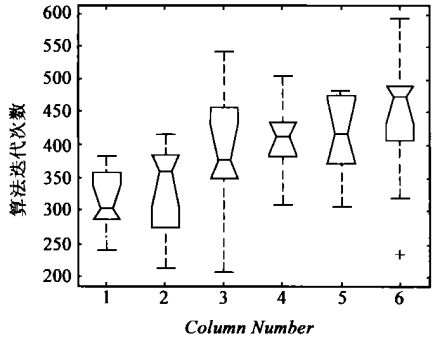

图 3 c_2 设置水平对迭代次数的影响

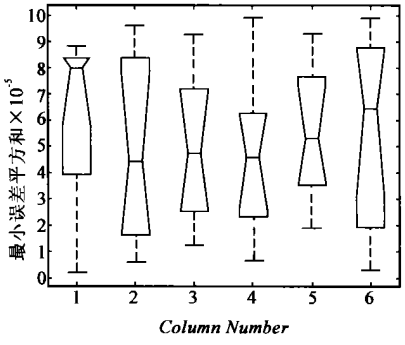

图 4 c_1 对优化效果的单因子方差分析

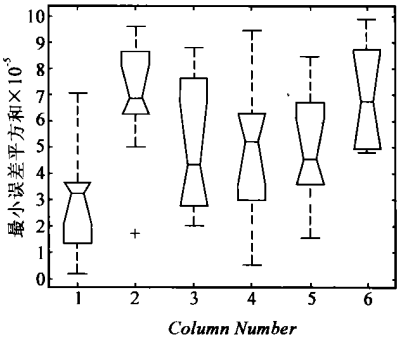

图 5 c_2 对优化效果的单因子方差分析

表 5 c_1 设置水平对算法迭代次数的单因子方差分析

	SS	df	MS	F	Prob> F
Columns	226011.7	5	45202.3	5.81	0.0002
Error	420333.2	54	7783.9		
Total	646344.9	59			

表 6 c_1 设置水平对优化效果的单因子方差分析

	SS	df	MS	F	Prob> F
Columns	2.05593e-9	5	4.11187e-10	0.43	0.8248
Error	5.1476e-8	54	9.53259e-10		
Total	5.35319e-8	59			

表 7 c_2 设置水平对算法迭代次数的单因子方差分析

	SS	df	MS	F	Prob> F
Columns	1177681.8	5	235531.8	4.12	0.0027
Error	3027371.2	54	56061.2		
Total	420506	59			

表 8 c_2 设置水平对优化效果的单因子方差分析

	SS	df	MS	F	Prob> F
Columns	9.91275e-9	5	1.98255e-9	3.77	0.0053
Error	2.83829e-8	54	5.25609e-10		
Total	3.82957e-8	59			

4 结论

在微粒群算法应用过程中, 关键参数的选择决定了算法整体性能的差异. 本文通过对微粒群算法中惯性权和加速常数的单因子方差分析, 探讨了不同参数设置水平与算法性能之间的基本联系. 通过实验数据分析获得的参数设置指导原则已在约束条件函数优化和实际组合优化问题(虚拟仪器参数的微粒群优化方法)中进行了验证[8], 并取得了与本文分析结论基本一致的效果. 另外, 算法参数间的相互影响和制约效果还有待更进一步的实验分析证明.

参考文献:

[1] J Kennedy, R Eberhart. Particle swarm optimization [A]. Proc. IEEE Int. Conf. on Neural Networks [C]. Perth, WA, Australia, 1995. 1942 - 1948.

[2] R Eberhart, J Kennedy. A new optimizer using particle swarm theory

[A]. Proc. 6th Int. Symposium on Micro Machine and Human Science [C]. Nagoya, Japan: Nagoya Municipal Industrial Research Institute, 1995. 39- 43.

[3] M M Millonas. Swarms, Phase Transition, and Collective Intelligence [M]. In C. G. Langton, Ed, Artificial Life Ó. MA: Addison Wesley, 1994.

[4] E Ozcan, C Mohan. Particle swarm optimization: surfing the waves [A]. Proceedings of the Congress on Evolutionary Computation [C]. Washington D.C, USA: IEEE Press, 1999. 1939- 1944.

[5] T Ray, K M Liew. A swarm with an effective information sharing mechanism for unconstrained and constrained single objective optimization problems [A]. Proc. IEEE Int. Conf. on Evolutionary Computation [C]. South Korea: IEEE Press, Seoul, 2001. 75- 80.

[6] P J Angeline. Evolutionary optimization versus particle swarm optimization: philosophy and performance difference [A]. Proc. of the 7th Annual Conf. on Evolutionary Programming [C]. 1998. 601- 610.

[7] J Kennedy, W M Spears. Matching algorithms to problems: an experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator [A]. Proc. IEEE Int. Conf. on Evolutionary Computation [C]. Anchorage, AK, USA, 1998. 78- 83.

[8] PENG Yu, PENG Xi2yuan, MENG Sheng2wei. Virtual instrument parameter calibration with particle swarm optimization [A]. Proceedings of 2003 IEEE Swarm Intelligence Symposium [C]. IEEE Press, 2003. 42- 45.

[9] El2Gallad, A, El2Hawary, M, Sallam, A, Kalas, A. Enhancing the particle swarm optimizer via proper parameters selection [A]. IEEE CCECE02 Proceedings [C]. Canadian: IEEE Press, 2002. 2. 792- 797.

[10] J Kennedy, R Eberhart, Yuhui Shi. Swarm Intelligence [M]. San Francisco: Morgan Kaufmann, 2001. 287- 299.

[11] Gerhard Venter. Particle swarm optimization [A]. 43rd AIAA/ ASME/ ASCE/ AHS/ ASC Structures, Structural Dynamics and Materials Conference [C]. Denver, Colorado, 2002. 22- 25.

[12] 5现代数学应用手册6编委会. 概率统计与随机过程卷(第一版) [M]. 北京: 清华大学出版社, 2000. 276- 302.

作者简介:

彭　宇　男, 1973 年生于西安, 哈尔滨工业大学自动化测试与控制系博士生, 主要研究领域为计算智能和智能故障诊断理论.

彭喜元　男, 1961 年生于内蒙古四子王旗, 哈尔滨工业大学自动化测试与控制系教授, 博士生导师, 主要研究方向为自动测试技术和智能故障诊断.