自适应惯性权重的改进粒子群算法

敖永才，师奕兵，张 伟，李焱骏

(电子科技大学自动化工程学院 成都 611731)

【摘要】针对标准PSO算法求解高维非线性问题时存在的大量无效迭代(经过一轮迭代后全局最优位置保持不变)，提出了一种自适应惯性权重的改进粒子群算法。基于单次迭代中单粒子运动状态的分析，提出并证明了论点：上一轮迭代适应度值变差的粒子，当前迭代中其惯性分量将引导粒子往适应度值变差的方向运动，导致粒子群体无效迭代次数增加。设计了标准PSO算法改进方案，将上一轮迭代中适应度值变差的全体粒子的惯性权重置为零，消除当前迭代中不利惯性分量对算法收敛的不良影响。采用6个标准测试函数，将该算法与标准PSO算法、固定惯性权重PSO算法和具有领袖的PSO算法进行性能对比分析。试验表明，该改进算法无效迭代次数更少，在收敛率、收敛速度和收敛稳定性上均具有明显的优势。

关 键 词 自适应惯性权重； 收敛性能； 惯性分量； 无效迭代； 粒子群优化算法

中图分类号 TP273 **文献标志码** A doi:10.3969/j.issn.1001-0548.2014.06.014

Improved Particle Swarm Optimization with Adaptive Inertia Weight

AO Yong-cai, SHI Yi-bing, ZHANG Wei, and LI Yan-jun

(School of Automation Engineering, University of Electronic Science and Technology of China Chengdu 611731)

Abstract To reduce the invalid iterations (the iteration in which the global optimum position is unchanged) of the particle swarm while solving the high-dimensional nonlinear problems by the standard particle swarm optimization (PSO) algorithm, an improved PSO algorithm with adaptive inertia weight is proposed in this paper. Based on the analysis of the instantaneous movement of single particle at each iteration, a significant argument is given and proved. In the improved algorithm, the inertia weights of the particles whose fitness become worse at the last iterations are set to zero. Six benchmark functions were used to test the proposed improved PSO algorithm, the standard PSO algorithm, the fixed inertia weight PSO algorithm, and the PSO algorithm with the leader. Experiments show that the invalid iterations of the proposed algorithm are less and it has obvious superiority on the convergence ratio, the convergence speed, and the convergence stability.

Key words adaptive inertia weight; convergence performance; inertial component; invalid iteration; particle swarm optimization (PSO) algorithm

文献[1]通过对鸟群捕食行为的仿真研究，首次提出粒子群优化算法。为了更好控制粒子群体的搜索行为，文献[2]对PSO算法给出了重大改进，为速度迭代公式增加了惯性权重(inertia weights)系数ω，并在文献[3]中给出了ω的选择方法，指出当ω随着迭代次数线性减小时，算法具有更好的收敛性能，形成了目前广泛应用的标准PSO算法。标准PSO算法简单，易实现，效率高，能有效解决复杂优化任务，已经引起了众多学者的重视。为了提升标准PSO算法的收敛性能，近年来，研究人员从全局收敛能力增强[4-8]、惯性权重ω控制[8-11]和粒子学习网络拓扑关系分析几个方面展开研究。

标准PSO算法直接采用粒子群体中适应度值最优的粒子为全局最优粒子。文献[4]将粒子群体的几何中心视为一个附加粒子引入到全局最优粒子的计算过程中。文献[5]将重力搜索算法引入PSO算法中，赋予全局最优粒子重力中心的地位，提高了PSO算法的全局收敛能力。当粒子群体可能早熟收敛时，对全局最优粒子执行变异操作，可从一定程度避免PSO算法收敛到局部最优解[6,8]，但算法的收敛速度有所降低。惯性权重ω是PSO算法的核心参数，对算法收敛性能有重要影响。为了改善PSO算法的收敛能力，文献[9]提出了当惯性权重固定时，PSO算法具有较快的收敛速度，但实际应用中该算

收稿日期：2013 – 11 – 21；修回日期：2014 – 02 – 28
基金项目：国家自然科学基金(61201131)；中央高校基本科研业务费(ZYGX2012J092, ZYGX2012K094)
作者简介：敖永才(1976 –)，男，博士，主要从事模式识别、智能优化算法方面的研究.

法对高维测试函数的求解能力较弱。文献[10]采用随机数加权平衡粒子的原始速度向量、自我学习向量和社会学习向量，增强了PSO算法的全局搜索能力。文献[8]将PSO搜索过程分为探索搜索和开发搜索两部分，并指出在探索搜索阶段赋予全局最优粒子远大于其他粒子的惯性权重，可改善PSO算法的全局收敛能力。

本文提出了一种自适应惯性权重的改进粒子群算法，在PSO算法的每一轮迭代中均实时监测粒子群体的运动状态，并根据粒子群体运动状态动态调整粒子群体中各粒子的惯性权重，消除了算法迭代过程中粒子惯性分量的不良影响，大大减少了粒子群体的无效迭代次数，使PSO算法的收敛性能得到显著提升。

1　标准PSO算法基础

文献[2]对PSO算法的数学描述为：在 D 维解搜索空间 Ω 中，粒子群粒子个数为 n，每个粒子有位置和速度两个特征。第 i 个粒子的位置(表示问题的一个可能解)为 $\boldsymbol{x}_i=(x_{i1},x_{i2},\cdots,x_{iD})$，速度为 $\boldsymbol{v}_i=(v_{i1},v_{i2},\cdots,v_{iD})$。第 i 个粒子迭代过程中的历史最优位置为 $\mathbf{pbest}_i^t$，全体粒子迭代过程中的最优位置为 $\mathbf{gbest}^t$。PSO算法经随机初始化后，每个粒子均按照式(1)和式(2)进行迭代，直至满足收敛准则。

$$\boldsymbol{v}_i^{t+1}=\omega_i^t\boldsymbol{v}_i^t+c_1r_1(\mathbf{pbest}_i^t-\boldsymbol{x}_i^t)+c_2r_2(\mathbf{gbest}^t-\boldsymbol{x}_i^t) \tag{1}$$

$$\boldsymbol{x}_i^{t+1}=\boldsymbol{x}_i^t+\boldsymbol{v}_i^{t+1} \tag{2}$$

式中，位置和速度均为向量；$i=1,2,\cdots,n$；ω_i^t 为惯性权重；c_1、c_2 为加速因子，一般取 $c_1=c_2=2$；r_1、r_2 为[0,1]区间均匀分布的随机数；$\boldsymbol{v}_i^t$ 为粒子 i 上一轮迭代结束时的速度；$(\mathbf{gbest}^t-\boldsymbol{x}_i^t)$ 为社会学习向量；$(\mathbf{pbest}_i^t-\boldsymbol{x}_i^t)$ 为自我学习向量。随着迭代次数的增加，ω 线性减小[3]是一种广泛应用的PSO算法惯性权重控制方案：

$$\omega_i^t=\omega_{\max}-\frac{\omega_{\max}-\omega_{\min}}{\text{iterNum}}t \tag{3}$$

式中，ω_i^t 为粒子 i 在第 t 次迭代中的取值；iterNum为预先设定的最大迭代次数；t 为当前迭代数；$\omega_{\max}$ 为惯性权重最大值，典型取值为0.9；$\omega_{\min}$ 为惯性权重最小值，典型取值为0.4。

2　单粒子运动状态分析

由仿真试验数据可知，当测试函数维度较高时($D\geqslant 30$)，标准PSO算法的收敛率和收敛速度均急速下降。采用标准PSO算法求解复杂的非线性问题时，算法迭代过程中存在大量的无效迭代，导致PSO算法的收敛性能降低。图1给出了采用标准PSO算法对 $f_1(D=30)$ 的一次典型求解过程的全局最优适应度值下降曲线(算法参数：ω_i^t 线性减小，$\omega_{\max}=0.9$，$\omega_{\min}=0.4$，$c_1=c_2=2$，粒子数30，算法迭代的最大次数5 000)。

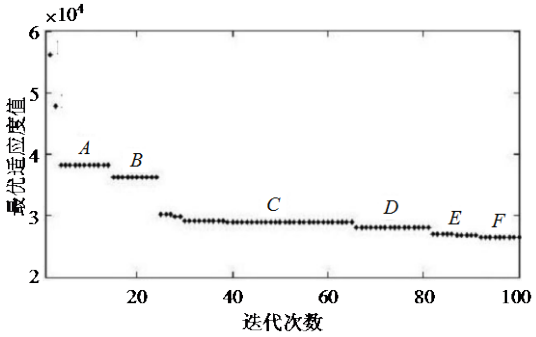


图1　求解 f_1 测试函数的全局最优适应度值下降曲线

在图1的迭代过程中，初始迭代阶段(前几轮迭代)的每一轮迭代中粒子群体均能搜索到适应度值更优的位置，全局最优适应度值连续下降，但随后的迭代过程中全局最优适应度值下降曲线出现水平直线段(图1中的A～F)，粒子群体执行了大量的无效迭代，严重影响了PSO算法的收敛性能。

粒子群体的无效迭代过程与群体中单个粒子 i 的瞬时运动状态密切相关。由式(1)可知，粒子 i 在当前迭代中的更新速度 $\boldsymbol{v}_i^{t+1}$ 由3个相互独立的速度分量唯一确定：惯性分量 $\omega_i^t\boldsymbol{v}_i^t$、自我学习分量 $c_1r_1(\mathbf{pbest}_i^t-\boldsymbol{x}_i^t)$ 和社会学习分量 $c_2r_2(\mathbf{gbest}^t-\boldsymbol{x}_i^t)$。任意粒子 i 的自身历史最优位置 $\mathbf{pbest}_i^t$ 和粒子群体的全局历史最优位置 $\mathbf{gbest}^t$ 的适应度值均优于粒子 i 的当前位置 x_i^t 的适应度值，粒子在两个学习分量的作用下，倾向于往适应度值更优的位置运动。但是，在一定的条件下(定理1)，粒子的惯性分量 $\omega_i^t\boldsymbol{v}_i^t$ 可能引导粒子往适应度值变差的方向运动，从而削弱甚至抵消粒子的学习效果，导致粒子群体的无效迭代，从而使得PSO算法的收敛性能降低。

假设 1　本文主要讨论采用PSO算法求解最小值优化问题，粒子位置的目标函数值为粒子的适应度值，适应度值越小，则其适应度值越优。

定理 1　设 $f(\boldsymbol{x})$ 是 D 维空间中的连续函数，$\nabla f(\boldsymbol{x})$ 存在，当采用标准PSO算法搜索 $f(\boldsymbol{x})$ 的全局最优解时，对于 D 维空间中的任意粒子 k (在上一轮迭代中以速度 $\boldsymbol{v}_k^t$ 从 $\boldsymbol{x}_k^{t-1}$ 运动到 $\boldsymbol{x}_k^t$，在当前迭代中将以速度 $\boldsymbol{v}_k^{t+1}$ 从 $\boldsymbol{x}_k^t$ 运动到 $\boldsymbol{x}_k^{t+1}$)，假设 $\nabla f(\boldsymbol{x}_k^t)\neq 0$，即 $\boldsymbol{x}_k^t$ 不是 $f(\boldsymbol{x})$ 的极值点，若粒子 k 在上一轮迭代中运动到适应度值更差的位置，即：

$$f(\boldsymbol{x}_k^t) > f(\boldsymbol{x}_k^{t-1}) \tag{4}$$

则当前迭代中，惯性分量 $\omega_k^t \boldsymbol{v}_k^t$ 将引导粒子 k 往适应度值更差的方向运动，对PSO算法的收敛不利；反之，惯性分量 $\omega_k^t \boldsymbol{v}_k^t$ 将引导粒子 k 往适应度值更优的方向运动，对PSO算法的收敛有利。

证明：标准PSO算法的迭代公式(1)和式(2)均可视为 D 维解搜索空间 Ω 中向量组的线性组合，即粒子在 Ω 中的运动方式是线性的。可在粒子的当前位置 $\boldsymbol{x}_k^t$ 用直线方程来近似 $f(\boldsymbol{x})$。将 $f(\boldsymbol{x})$ 在粒子 k 的当前位置 $\boldsymbol{x}_k^t$ 的某个邻域 $[\boldsymbol{x}_k^t - \delta, \boldsymbol{x}_k^t + \delta]$ 内泰勒级数展开，并保留前两项(直线方程)，有：

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}_k^t) + [\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} (\boldsymbol{x} - \boldsymbol{x}_k^t) \tag{5}$$

式中，$\nabla f(\boldsymbol{x}_k^t)$ 为 $f(\boldsymbol{x})$ 在 $\boldsymbol{x}_k^t$ 的梯度。将 $\boldsymbol{x}_k^{t-1}$ 代入式(1)，有：

$$f(\boldsymbol{x}_k^{t-1}) \approx f(\boldsymbol{x}_k^t) + [\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} (\boldsymbol{x}_k^{t-1} - \boldsymbol{x}_k^t) \tag{6}$$

代入式(2)后化简得到：

$$f(\boldsymbol{x}_k^{t-1}) - f(\boldsymbol{x}_k^t) \approx -[\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} \boldsymbol{v}_k^t \tag{7}$$

由式(4)和式(7)可得：

$$[\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} \boldsymbol{v}_k^t > 0 \tag{8}$$

式(8)的几何意义为：上一轮迭代中，粒子 k 的运动速度 $\boldsymbol{v}_k^t$ 在 $\boldsymbol{x}_k^t$ 的梯度方向 $\nabla f(\boldsymbol{x}_k^t)$ 的投影为正值，粒子沿 $\boldsymbol{v}_k^t$ 方向的运动将导致其适应度值增大。再将 $\boldsymbol{x}_k^{t+1}$ 代入式(5)，有：

$$f(\boldsymbol{x}_k^{t+1}) \approx f(\boldsymbol{x}_k^t) + [\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} (\boldsymbol{x}_k^{t+1} - \boldsymbol{x}_k^t) = f(\boldsymbol{x}_k^t) + [\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} \boldsymbol{v}_k^{t+1} \tag{9}$$

将式(1)代入式(9)得到：

$$f(\boldsymbol{x}_k^{t+1}) \approx f(\boldsymbol{x}_k^t) + [\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} [c_1 r_1 (\mathbf{pbest}_k^t - \boldsymbol{x}_k^t)] + [\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} [c_2 r_2 (\mathbf{gbest}^t - \boldsymbol{x}_k^t)] + \omega_i^t [\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} \boldsymbol{v}_k^t \tag{10}$$

式中，右边第1项为粒子 k 当前位置的适应度值；第2项为粒子的自我学习带来的适应度值改变量；第3项为粒子的社会学习带来的适应度值改变量；第4项为粒子 k 的惯性分量 $\omega_k^t \boldsymbol{v}_k^t$ 带来的适应度值改变量。前3项之和表示粒子 k 在两个学习分量的作用下运动到了某个新的位置，并且这个新位置在概率意义上比 $\boldsymbol{x}_k^t$ 有更优的适应度值。当上一轮迭代中粒子 k 运动到适应度值更差的位置时，由 $\omega_k^t > 0$ 和式(8)可知，式(10)右边第4项为：

$$\omega_k^t [\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} \boldsymbol{v}_k^t > 0 \tag{11}$$

惯性分量 $\omega_k^t \boldsymbol{v}_k^t$ 将导致粒子更新位置 $\boldsymbol{x}_k^{t+1}$ 的适应度值 $f(\boldsymbol{x}_k^{t+1})$ 增大，对PSO算法的收敛不利。

同理，当上一轮迭代中粒子 k 运动到适应度值更优的位置时，式(10)右边第4项为：

$$\omega_k^t [\nabla f(\boldsymbol{x}_k^t)]^{\mathrm{T}} \boldsymbol{v}_k^t < 0 \tag{12}$$

惯性分量 $\omega_k^t \boldsymbol{v}_k^t$ 将导致粒子更新位置 $\boldsymbol{x}_k^{t+1}$ 的适应度值 $f(\boldsymbol{x}_k^{t+1})$ 减小，对PSO算法收敛有利，证毕。

3 PSO算法的改进

采用标准PSO算法优化求解实际问题时，为了充分发挥粒子群体智能，常常有多个粒子参与迭代。而在每一轮的算法迭代中，必定有一部分粒子运动到适应度值更优的位置，其他粒子则运动到适应度值更差的位置。同样以求解 f_1 测试函数($D=30$)为例，算法参数与第2节相同。采用标准PSO算法执行一次对 f_1 的典型求解，其第2～11轮迭代后粒子群体中适应度值变差的粒子数分别为17,14,16,13,16,16,17,8,18和11个，其平均数为14.6个，达到粒子总数的49%。由第2节可知，若上一轮迭代中粒子运动到适应度值更差的位置，当前迭代中粒子的惯性分量将导致粒子的适应度值变差，对粒子收敛到全局最优解不利。若在标准PSO算法的每一轮迭代(除开第1轮)中均根据上一轮迭代适应度值的变化情况动态调整各粒子在惯性方向的运动，对粒子群体的快速收敛将十分有利。

基于上述分析，本文提出标准PSO算法的改进方案：在每一轮迭代中，根据各粒子在上一轮迭代中的适应度值变化，将粒子群体分为两类(适应度值变差的粒子类和适应度值变优的粒子类)，对于适应度值变差的粒子类，粒子的惯性分量对PSO算法的收敛不利，令其惯性权重 $\omega_i^t = 0$，以消除惯性分量带来的不利影响；对于适应度值变优的粒子类，粒子的惯性分量对PSO算法的收敛有利，令其惯性权重 ω_i^t 保持不变。

自适应惯性权重的改进粒子群算法流程如下：

1) 随机初始化粒子群体中粒子的位置与速度，令迭代次数 $t=0$。

2) 计算各粒子的适应度值，设 $\mathbf{pbest}_i^t$ 为粒子当前位置，$\mathbf{gbest}^t$ 为初始群体中适应度值最优的粒子位置。

3) 算法成功收敛？是，转到步骤10)，否则执行步骤4)。

4) 令 $t = t+1$，若 $t<2$，通过式(3)计算出当前条件下全体粒子统一的惯性权重 ω_i^t，并转到步骤8)，否则，执行步骤 5)。

5) 通过式(3)计算出当前条件下全体粒子统一的惯性权重 ω_i^t。

6) 通过式(13)计算全体粒子上一轮迭代结束后适应度值的变化：

$$\delta f(\boldsymbol{x}_i^t) = f(\boldsymbol{x}_i^t) - f(\boldsymbol{x}_i^{t-1}) \tag{13}$$

式中，$i=1,2,\cdots,n$，$t\geqslant 2$；$f(\boldsymbol{x}_i^t)$ 表示粒子 i 在第 t 次迭代后的适应度值；$\delta f(\boldsymbol{x}_i^t)$ 表示粒子 i 在第 t 次迭代后的适应度值变化。

7) 确定各粒子在当前迭代中惯性权重的最终取值为：

$$\omega_i^t = \begin{cases} \omega_i^t & \delta f(\boldsymbol{x}_i^t) < 0 \\ 0 & \delta f(\boldsymbol{x}_i^t) \geqslant 0 \end{cases} \tag{14}$$

8) 通过式(1)和式(2)更新粒子群体的位置与速度。

9) 计算各粒子的适应度值，存储 $\mathbf{pbest}_i^t$ 和 $\mathbf{gbest}^t$，并转到步骤3)。

10) 输出 $\mathbf{gbest}^t$，算法运行结束。

4 仿真试验

为了验证本文提出的改进PSO算法(表示为PSO-AIW)的有效性，将算法PSO-AIW与标准PSO算法[3](表示为PSO-S)，惯性权重固定PSO算法[9](表示为PSO-F)和拥有领导机制的PSO算法[8](表示为PSO-L)进行性能对比分析。

分别采用4种算法对表1所示的6个标准测试函数进行独立求解。f_1、f_2 为典型的单峰函数，在解搜索空间上仅有一个极值点；f_3 为病态函数，虽然在解搜索空间上有一个最小值点，但一般算法很难得到其最优解；f_5 是多峰函数，在解搜索空间上有6个极值点；f_4 和 f_6 为多峰函数，在解搜索空间有大量局部极值点。各测试函数的名称、表达式、解空间范围、理论最优解 $\mathbf{x}^*$、$\mathbf{x}^*$ 对应的适应度值 $f(\mathbf{x}^*)$ 如表1所示。算法实现和数据分析通过Matlab软件完成，硬件平台为一台PC机，AMD Athlon X2@2.8 GHz，2.0 GB RAM。

表1　标准测试函数及其参数

函数名	函数表达式	解空间	$\boldsymbol{x}^*$	$f(\boldsymbol{x}^*)$	f_{err}
Sphere	$f_1=\sum_{i=1}^{D} x_i^2$	$[-100,100]^D$	$(0)^D$	0	0.01
Schwefel 1.2	$f_2=\sum_{i=1}^{D}\left(\sum_{j=1}^{i} x_j\right)^2$	$[-100,100]^D$	$(0)^D$	0	0.01
Rosenbrock	$f_3=\sum_{i=1}^{D-1}100(x_{i+1}-x_i^2)^2+(x_i-1)^2$	$[-30,30]^D$	$(1)^D$	0	0.01
Rastrigin	$f_4=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$	$[-5.12,5.12]^D$	$(0)^D$	0	0.1
Six-hump Camel-back	$f_5=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4+2$	$[-5,5]^2$	$(0.09,-0.71)$ $(-0.09,0.71)$	0.968	0.001
Ackey f_1	$f_6=-20\mathrm{e}^{-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}}-\mathrm{e}^{\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)}+20+\mathrm{e}$	$[-32,32]^D$	$(0)^D$	0	0.1

4.1 算法参数设置

算法PSO-AIW和PSO-S的惯性系数 ω 在 $[\omega_{\min},\omega_{\max}]$ 区间线性减小，其中 $\omega_{\max}=0.9$，$\omega_{\min}=0.4$，$c_1=c_2=2$；算法PSO-F采用固定参数 $\omega=0.729\,84$，$c_1=c_2=2.05$；对于算法PSO-L，在探索搜索阶段全局最优粒子的 $\omega=0.9$，其他粒子 $\omega=0.4$。粒子的位置和速度的初始化范围均为解空间范围。为获取有效统计数据，针对每种算法，每个测试函数的求解过程被独立的执行50次。每次求解过程算法迭代的最大次数为5 000次，若算法迭代5 000次仍未能成功收敛，则当次试验失败。

4.2 算法收敛准则

如式(15)所示，当粒子群体的全局最优位置 $\mathbf{gbest}^t$ 的适应度值与理论最优解或期望最优解的适应度值间的距离小于 f_{err} 时，认为算法成功收敛。其中 f_{err} 为预先设定的收敛阈值(为了达到一定的收敛精度，不同的优化问题，一般设定不同的收敛阈值，如表1所示)。

$$\left|f(\mathbf{gbest}^t)-f(\boldsymbol{x}^*)\right|<f_{\text{err}} \tag{15}$$

4.3 算法性能指标

收敛率(converge ratio，CR)：算法成功收敛次数与算法执行总次数的比值，反映算法的可收敛性；平均迭代次数(average iteration times，AIT)：算法成功收敛时算法的平均迭代次数，反映算法的收敛速度；平均迭代运行时间(average iteration running time，AIRT)：算法迭代运行多次并成功收敛时所需的平均CPU时间，反映算法的收敛速度；迭代次数标准偏差(iteration times standard deviation，ITSD)：算法成功收敛时迭代次数的标准偏差，反映算法的收敛稳定性。

4.4 试验结果及分析

表2列出了求解6个标准测试函数时算法PSO-AIW、PSO-S、PSO-F和PSO-L的性能指标计算结果。表中，“-”表示对应的算法PSO-S的收敛率为0，从表中可以看出，算法PSO-F的收敛性能与其他3种算法有较大差异，表现在：1) 当测试函数维度较高($D \geqslant 10$)时，对于表1中的全体测试函数，算法PSO-F的收敛率均为0，无法收敛到全局最优解；2) 当测试函数维度较低($D=2$)时，对于单峰函数 f_1 和 f_2，多峰函数 f_4 和 f_6，算法PSO-F的收敛速度最慢(AIT最大)，而对于单峰函数 f_3 和多峰函数 f_5，算法PSO-F的收敛速度最快(AIT最小)。算法PSO-F无法求解高维非线性问题，后文将不对其展开对比讨论。

表2 算法运行的性能指标

函数	维度	PSO-S				PSO-F				PSO-L				PSO-AIW			
		CR/%	AIT	AIRT/ms	ITSD	CR/%	AIT	AIRT/ms	ITSD	CR/%	AIT	AIRT/ms	ITSD	CR/%	AIT	AIRT/ms	ITSD
f_1	2	100	33	28	35	98	43	32	31	100	19	16	34	100	6	6	3
	10	78	201	164	197	0	-	-	-	100	106	87	113	100	33	30	20
	30	24	264	227	168	0	-	-	-	66	187	161	141	100	53	49	27
	50	10	448	415	312	0	-	-	-	40	307	284	169	100	78	77	75
f_2	2	100	36	34	50	100	39.7	36	29.1	100	15	16	17	100	6	8	4
	10	54	229	283	139	0	-	-	-	94	182	227	146	100	41	53	22
	30	0	-	-	-	0	-	-	-	12	525	1 075	473	98	127	266	124
	50	0	-	-	-	0	-	-	-	0	-	-	-	92	349	1 018	295
f_3	2	86	457	37	294	100	79	64	33	82	387	317	464	90	318	255	207
f_4	2	98	50	43	54	98	33.7	29	32	100	21	19	24	100	7	8	8
	10	68	179	162	167	0	-	-	-	100	126	110	133	100	38	36	23
	30	34	371	357	252	0	-	-	-	66	243	2 37	166	100	56	57	30
	50	10	380	395	330	100	-	-	-	38	335	354	148	98	66	73	38
f_5	2	100	274	219	168	100	37.2	28	18.7	100	192	160	134	100	141	115	81
f_6	2	100	38	38	49	2	46.4	46	31.5	100	26	27	32	100	6	8	3
	10	84	167	168	145	0	-	-	-	100	110	110	93	100	33	36	24
	30	28	301	327	191	0	-	-	-	64	218	240	140	98	58	70	31
	50	0	-	-	-		-	-	-	20	328	385	111	92	84	100	112

表3 算法PSO-AIW和PSO-L的相对AIRT

函数	维度	PSO-L	PSO-AIW	函数	PSO-L	PSO-AIW
f_1	2	1.8	4.7	f_4	2.3	5.4
	10	1.9	5.5		1.5	4.5
	30	1.4	4.6		1.5	6.3
	50	1.5	5.4		1.1	5.4
	平均	1.6	5.1		1.5	4.7
f_2	2	2.1	4.3	f_6	1.4	4.8
	10	1.2	5.3		1.5	4.7
	30	-	-		1.4	4.7
	50	-	-		-	-
	平均	1.6	4.8		1.4	4.7
f_3	2	1.2	1.5	f_5	1.4	1.9

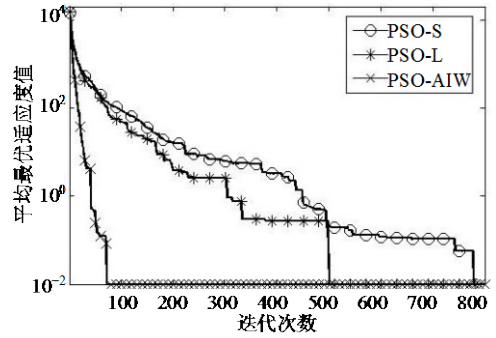
收敛率指标CR反映了算法的可收敛性，CR值越大，算法的可收敛性越好。对于表1中的全体测试函数，算法PSO-AIW的收敛率指标全面优于算法PSO-S和算法PSO-L，具体表现在：

1) 当 $D \leqslant 30$ 时，对于 f_1、f_2、f_4 和 f_6，算法PSO-AIW的收敛率几乎全部达到100%。相比之下，测试函数维度的增加带来算法PSO-S和PSO-L收敛率的快速下降，如表2所示，当 $D=30$ 时，对于测试函数 f_2，算法PSO-S和PSO-L的收敛率分别下降为0和12%；

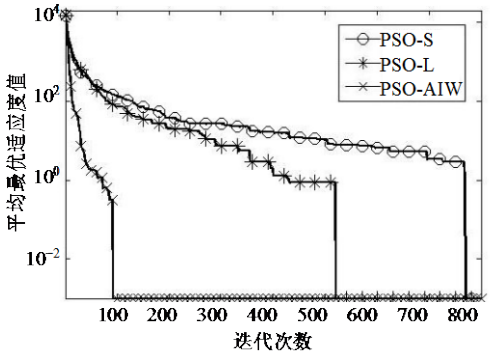
2) 算法PSO-AIW在求解高维测试函数时具有更高的收敛率，当 $D=50$ 时，对于 f_1、f_2、f_4 和 f_6，算法PSO-AIW的收敛率均超过90%；对于 f_1、f_4 和 f_6，算法PSO-S和PSO-L的收敛率均低于40%；对于 f_2，算法PSO-S和PSO-L的收敛率均为0；

3) 对于 f_3 和 f_5，算法PSO-AIW的收敛率略高于算法PSO-S和PSO-L；对于病态函数 f_3，当 $D \geqslant 10$ 时，4种算法均无法收敛到全局最优解(收敛率均为0，表2中未列出)。平均迭代次数指标AIT反映了算法的收敛速度，AIT值越小，则算法收敛到全局最优解的速度越快。如表2所示，对于表1中的全体测试函数，无论测试函数的维度是多少，算法PSO-AIW的AIT值均更小，即具有更快的收敛速度。表2中的平均迭代运行时间AIRT指标更直观显示出几种算法的收敛速度。因算法PSO-AIW和PSO-L均是对算法PSO-S的改进，为方便比较各算法的收敛速度，以算法PSO-S的AIRT指标为基准，计算出算法PSO-AIW和PSO-L的相对于算法PSO-S的平均迭代运行时间(Relative AIRT,以算法PSO-S的AIRT除以算法PSO-AIW或PSO-L的AIRT指标)。显

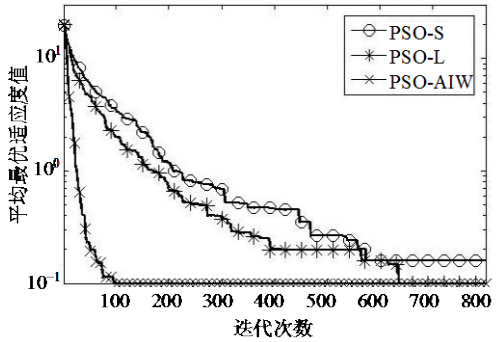
然，算法的相对AIRT值越大，表示算法PSO-AIW或PSO-L对算法PSO-S收敛速度的提升越大。表3列出了采用算法PSO-AIW和PSO-L求解表1中的全体测试函数的相对AIRT。由表3可知，相对于算法PSO-S：1) 求解 f_1、f_2、f_4 和 f_6 时，算法PSO-AIW的收敛速度提升显著，收敛速度的平均提升接近5倍；2) 求解 f_1、f_2、f_4 和 f_6 时，算法PSO-L的收敛速度提升不显著，收敛速度的平均提升仅为1.5倍；3) 求解 f_3 和 f_5 时，算法PSO-AIW和PSO-L的收敛速度提升均不显著，算法PSO-AIW的收敛速度提升约为1.7倍，算法PSO-L的收敛速度提升仅为1.3倍。



a. f_1测试函数



b. f_4测试函数



c. f_6测试函数

图2　平均最佳适应度下降曲线

本文通过粒子群体的平均最优适应度值下降曲线对几种算法的收敛过程做出进一步的对比分析。图2给出了采用算法PSO-AIW，PSO-S和PSO-L分别求解 f_1、f_4 和 f_6（$D=10$）并成功收敛时的平均最优适应度值下降曲线。

由图2可以看出，对于 f_1、f_4 和 f_6：1) 算法PSO-AIW满足式(15)收敛准则需要的迭代次数最少，在整体上表现出最快的收敛速度；2) 在初始探索搜索阶段，3种算法的平均最优适应度值曲线均连续下降，表明3种算法在其探索搜索阶段均可实现有效搜索(无效迭代次数少)，其中算法PSO-AIW的平均最优适应度值曲线下降最快，而算法PSO-S的平均最优适应度值曲线下降最慢；3) 在算法中后期开发搜索阶段，算法PSO-AIW的平均最优适应度值曲线保持连续下降趋势(无效迭代次数少)，并快速的收敛到全局最优解，而算法PSO-S和PSO-L的平均最优适应度值下降曲线均出现水平线段，表明存在大量的无效迭代，导致其收敛速度下降。

迭代次数标准偏差ISTD作为一种典型的随机搜索算法，PSO算法受到随机因素的影响，即使求解同一问题，其迭代次数仍为一个随机变量。迭代次数标准偏差指标是算法成功收敛时迭代次数的变化范围，反映了算法的收敛稳定性，显然，ISTD值越小，算法的收敛稳定性越好。对于表1中的全体测试函数，算法PSO-AIW的ISTD值均显著小于算法PSO-L和PSO-S对应的ISTD值，算法PSO-AIW表现出更优的收敛稳定性(限于篇幅，未进行详细计算)。

5　结　论

为了改善PSO算法在高维非线性优化应用中的收敛性能，本文提出了一种自适应惯性权重的改进粒子群算法。不同于传统PSO改进方法仅针对全局最优粒子展开研究或无视粒子群体运动状态仅对惯性权重进行粗糙的调整，本文提出的改进方法通过对粒子群体的运动状态的动态监测，并根据监测结果在算法的每一轮迭代中均实时调整全体粒子的惯性权重，消除了算法迭代过程中粒子惯性分量的不良影响，大大减少了粒子群体的无效迭代次数。通过求解6个标准测试函数，本文提出的改进算法PSO-AIW在收敛率，收敛速度和收敛稳定性几个指标上全面优于算法PSO-S和PSO-L。

参　考　文　献

[1] KENNEDY J, EBERHART R C. Particle swarm optimization[C]//Proceedings of IEEE International Conference on Neural Networks. [S.l.]: IEEE, 1995:

1942-1948.

[2] SHI Y H, EBERHART R C. A modified particle swarm optimizer[C]//Proceedings of IEEE Congress on Evolutionary Computation(CEC 1998). Piscataway: IEEE, 1998: 69-73.

[3] SHI Y H, EBERHART R C. Parameter selection in particle swarm optimization[C]//7th International Conference, Evolutionary Programming VII. Berlin Heidelberg: Springer, 1998, 1447: 591-600.

[4] LIU Y, ZHENG Q, ZHEWEN S, et al. Center particle swarm optimization[J]. Neurocomputing, 2007, 70(4): 672-679.

[5] TSAI Hsing-chih, TYAN Yaw-yauan, WU Yun-wu, et al. Gravitational particle swarm[J]. Applied Mathematics and Computation, 2013, 219(17): 9106-9117.

[6] WORASUCHEEP C. A particle swarm optimization with stagnation detection and dispersion[C]//IEEE Congress on Evolutionary Computation. [S.l.]: IEEE, 2008: 424-429.

[7] 吕振肃, 侯志荣. 自适应变异的粒子群优化算法[J]. 电子学报, 2004, 32(3): 416-420.
LÜ Zhen-su, HOU Zhi-rong. Particle swarm optimization with adaptive mutation[J]. Acta Electronica Sinica, 2004, 32(3): 416-420.

[8] ZHOU L, SHI Y, LI Y, et al. Parameter selection, analysis and evaluation of an improved particle swarm optimizer with leadership[J]. Artificial Intelligence Review, 2010, 34(4): 343-367.

[9] CLERC M, KENNEDY J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(1): 58-73.

[10] CHALERMCHAIARBHA S, ONGSAKUL W. Stochastic weight trade-off particle swarm optimization for nonconvex economic dispatch[J]. Energy Conversion and Management, 2013, 70: 66-75.

[11] TATSUMI K, IBUKI T, TANINO T. A chaotic particle swarm optimization exploiting a virtual quartic objective function based on the personal and global best solutions[J]. Applied Mathematics and Computation, 2013, 219(17): 8991-9011.

编 辑 漆 蓉