Journal of Computer Applications
计算机应用,2014,34(3):754 -759
ISSN 1001-9081
CODEN JYIIDU
2014-03-10
http://www.joca.cn

文章编号:1001-9081(2014)03-0754-06
doi:10.11772/j.issn.1001-9081.2014.03.0754

基于高斯扰动的粒子群优化算法

朱德刚[1],孙　辉[2*],赵　嘉[2],余　庆[2]
(1.南昌航空大学 信息工程学院,南昌 330063;　2.南昌工程学院 信息工程学院,南昌 330099)
(*通信作者电子邮箱 sunhui@nit.edu.cn)

摘　要:针对标准粒子群优化(PSO)算法易陷入局部最优、进化后期收敛速度慢和收敛精度低的缺点,提出一种基于高斯扰动的粒子群优化算法。该算法采用对粒子个体最优位置加入高斯扰动策略,有效地防止算法陷入局部最优,加快收敛并提高收敛精度。在固定评估次数的情况下,对8个常用的经典基准测试函数在30维上进行了仿真。实验结果表明,所提算法在收敛速度和寻优精度上优于一些知名的粒子群优化算法。

关键词:粒子群优化算法;高斯扰动;快速收敛;全局搜索
中图分类号:TP301.6;TP18　　**文献标志码**:A

Particle swarm optimization algorithm based on Gaussian disturbance

ZHU Degang[1], SUN Hui[2*], ZHAO Jia[2], YU Qing[2]
(1. *School of Information Engineering, Nanchang Hangkong University, Nanchang Jiangxi 330063, China*;
2. *School of Information Engineering, Nanchang Institute of Technology, Nanchang Jiangxi 330099, China*)

Abstract: As standard Particle Swarm Optimization (PSO) algorithm has some shortcomings, such as getting trapped in the local minima, converging slowly and low precision in the late of evolution, a new improved PSO algorithm based on Gaussian disturbance (GDPSO) was proposed. Gaussian disturbance was put into in the personal best positions, which could prevent falling into local minima and improve the convergence speed and accuracy. While keeping the same number of function evaluations, the experiments were conducted on eight well-known benchmark functions with dimension of 30. The experimental results show that the GDPSO algorithm outperforms some recently proposed PSO algorithms in terms of convergence speed and solution accuracy.

Key words: Particle Swarm Optimization (PSO) algorithm; Gaussian disturbance; fast convergence; global search

0　引言

粒子群优化(Particle Swarm Optimization, PSO)算法[1-2]是由美国社会心理学家Kennedy等于1995年通过对鸟群、鱼群等聚集生物的觅食行为进行研究后提出的一种群体智能优化算法。由于该算法简单、易于实现、参数少、无需梯度信息,一经提出得到众多学者的关注和研究,并将其应用到电力系统优化[3]、函数优化[4]、模式识别[5]、目标跟踪[6]、烧结配料[7]、图像处理[8]等众多领域。

PSO算法应用领域比较广泛,但是算法具有后期收敛速度慢、易陷入局部最优等缺点,严重制约了它的发展。van Den Bergh[9]曾证明了标准PSO算法并不能收敛于全局最优解,为此,国内外许多学者都对标准PSO算法进行了大量改进,取得了众多的成果。通常算法的改进主要是对参数改进或自适应选择、更新公式改进、学习策略改进等。常见经典的改进算法有:Shi等[10]提出全局版本的PSO(Global Version PSO, GPSO)算法,第一次在粒子群算法中增加惯性权重参数,并分析了这个参数对粒子群算法的重要意义和影响;Clerc等[11]于2002年通过研究PSO收敛性,使用压缩因子来保证收敛,有效控制粒子运动轨迹,提出了局部版本的PSO(Local Version PSO, LPSO)算法;Mendes等[12]通过深入研究不同拓扑结构对粒子群性能的影响,并在粒子间信息交流时加入了"small world"概念,种群中每个粒子并非简单地只受最好粒子影响,而是充分利用了种群中所有粒子的搜索结果,提出了信息全部共享的粒子群(Fully Informed Particle Swarm, FIPS)优化算法;文献[13]提出一种线性时变加速因子的粒子群(Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, HPSO-TVAC)算法,加速因子c_1和c_2分别随着迭代次数逐渐减小和增大,用来平衡算法局部搜索和全局搜索能力;Liang等[14]基于局部版本的PSO和一个全新的邻域拓扑结构,首先将种群分成多个动态子群,然后在一定的迭代次数之后随机重新组合原有子群并形成新子群,提出动态多子群的粒子群优化(Dynamic Multi-Swarm Particle Swarm Optimizer, DMS-PSO)算法;Liang等[15]以一种动态邻域拓扑结构作为粒子学习样本,基于这种策略提出综合学习粒子群优化(Comprehensive learning particle swarm optimizer for global optimization of multi-modal function, CLPSO)算法;Zhan等[16]为了提高算法收敛速度和增强算法

收稿日期:2013-08-14;**修回日期**:2013-10-30。　**基金项目**:国家自然科学基金资助项目(61261039);江西省自然科学基金资助项目(20122BAB201043,20132BAB211031);江西省教育厅落地计划项目(KJLD13096);江西教育厅科技项目(GJJ13761,GJJ13745)。
作者简介:朱德刚(1988-),男,安徽芜湖人,硕士研究生,主要研究方向:群智能优化算法;　孙辉(1959-),男,江西九江人,教授,博士,主要研究方向:智能计算、Rough集与粒计算、变分不等原理与变分不等式;　赵嘉(1981-),男,江西九江人,副教授,主要研究方向:智能优化算法;　余庆(1990-),男,江西上饶人,硕士研究生,主要研究方向:智能信息处理。

跳出局部最优的能力,将粒子进化状态分为勘探、开采、收敛和跳出 4 个阶段,并自适应地控制惯性权重和加速系数,采用了精英高斯学习,提出自适应粒子群优化(Adaptive Particle Swarm Optimization, APSO)算法;高哲等[17]基于粒子的平均速度的切换、模拟退火算法和退火温度的更新公式,使粒子群在保持较快的寻优速度条件下,仍能跳出局部最小值,提出基于平均速度的混合自适应粒子群算法;高卫峰等[18]利用局部搜索算法的局部快速收敛性,对整个粒子群目前找到的最优位置进行局部搜索,再利用学习算子,保持粒子多样性,跳出局部最优,提出一种高效粒子群优化算法。上述改进算法虽然在不同的方面对原有的 PSO 进行了相关的改进,在一定程度上提高了算法的性能,但是仍然存在很多问题,比如改进算法过于复杂、算法收敛速度慢等。因此,如何能够更好地权衡算法收敛速度、收敛精度,逃离局部最优能力,且让算法更加简洁,值得进一步研究。

针对上述问题,本文提出一种基于高斯扰动的粒子群优化(Particle Swarm Optimization based on Gaussian Disturbance, GDPSO)算法。通过对粒子的速度更新时增加高斯扰动,防止算法"早熟",增强其逃离局部最优的能力,从而提高算法收敛速度和收敛精度。相比其他改进算法,GDPSO 算法思路简单,实现容易,并在典型测试函数上的仿真实验结果显示,GDPSO 算法有着非常好的寻优能力、较强的逃离局部最优能力和非常快的收敛速度。

1　标准 PSO 算法

PSO 算法具体可以描述为总数为 N 的粒子在一个 D 维的搜索空间飞行,第 i 个粒子在第 t 次迭代时位置为 $x_i^{(t)} = (x_{i1}^{(t)}, x_{i2}^{(t)}, \cdots, x_{iD}^{(t)})$,微粒 $i(i = 1,2,\cdots,N)$ 的速度定义为每次迭代中微粒移动的距离,粒子 i 的第 t 次迭代时速度表示为 $v_i^{(t)} = (v_{i1}^{(t)}, v_{i2}^{(t)}, \cdots, v_{iD}^{(t)})$,粒子 i 在第 $t+1$ 次迭代时第 $d(d = 1,2,\cdots,D)$ 维子空间的速度 $v_{id}^{(t+1)}$ 和位置 $x_{id}^{(t+1)}$ 根据下式调整:

$$v_{id}^{(t+1)} = wv_{id}^{(t)} + c_1r_1(p_{id}^{(t)} - x_{id}^{(t)}) + c_2r_2(p_{gd}^{(t)} - x_{id}^{(t)}) \quad (1)$$

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

其中:w 为惯性权重,主要作用是为了平衡粒子局部和全局的搜索能力,w 太大时粒子有较强的全局搜索能力,w 较小时粒子有较强的局部搜索能力;c_1、c_2 为学习因子(也称加速因子),主要作用是让粒子自我学习和向群体中其他优秀粒子学习,以便于粒子快速地向自己的历史最优位置和全局的最优位置运动,一般均取 2.0;r_1、r_2 是两个服从均匀分布的随机数,范围在区间(0,1);$p_{id}^{(t)}$ 为粒子 i 在第 t 次迭代时历史最优位置记录;$p_{gd}^{(t)}$ 为第 t 次迭代时整个粒子群的历史最优位置记录。

2　GDPSO 算法

本文提出的 GDPSO 算法,主要针对标准粒子群优化算法易陷入局部最优、进化后期收敛速度慢和收敛精度不高的问题,采用高斯扰动策略,增强算法逃离局部最优的能力,提高算法的收敛速度。

GDPSO 算法与标准 PSO 最大区别就是速度进化公式的改进,在"自我认知"部分添加了高斯扰动项,进化模式如下:

$$v_{id}^{(t+1)} = wv_{id}^{(t)} + c_1r_1(p_{id}^{(t)} + r_2gauss_{id}^{(t)} - x_{id}^{(t)}) + c_2r_3(p_{gd}^{(t)} - x_{id}^{(t)}) \quad (3)$$

$$x_{id}^{(t+1)} = x_{id}^{(t)} + v_{id}^{(t+1)} \quad (4)$$

$$gauss_{id}^{(t)} = r_4gaussian(\mu,\sigma^2) \quad (5)$$

其中:w 为惯性权重,c_1 和 c_2 为加速因子,r_1、r_2、r_3 和 r_4 表示(0,1)区间服从均匀分布的随机数,$v_{id}^{(t)}$ 为粒子 i 在第 t 次迭代时的速度,$p_{ij}^{(t)}$ 为粒子 i 在第 t 次迭代时的历史最优位置,$gauss_{ij}^{(t)}$ 表示粒子 i 在第 t 次迭代时产生的高斯扰动,μ 表示均值,σ^2 表示方差,$x_{ij}^{(t)}$ 是粒子 i 在第 t 次迭代时的位置,$p_{gd}^{(t)}$ 是第 t 次迭代时种群的最优位置。

GDPSO 算法的具体步骤如下:

1) 初始化所有粒子,设置相关参数。

2) 评价和计算每个粒子适应值。

3) 计算每个粒子的个体历史最优位置和种群最优粒子位置,同时计算每次迭代时粒子个体历史最优位置的高斯扰动值。

4) 将粒子按照式(3)和(4)更新速度和位置。

5) 检验是否满足终止条件,若满足,则停止迭代,输出全局最优粒子位置 p_{gd} 及其对应的适应值;否则转到 2)。

GDPSO 算法的详细流程如图 1 所示。

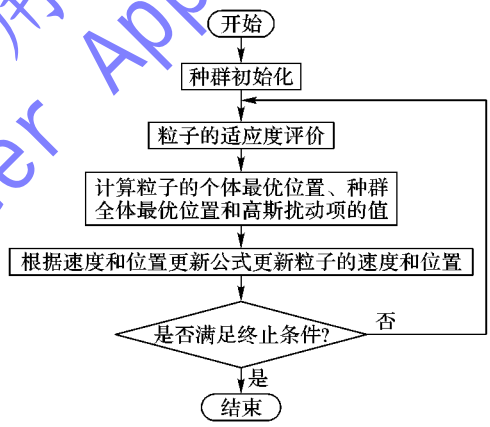

图 1　GDPSO 算法流程

3　GDPSO 算法分析

3.1　GDPSO 算法稳定性分析

由式(3)和(4)经变换,整理可得:

$$x_{id}^{(t+1)} = x_{id}^{(t)} + w(x_{id}^{(t)} - x_{id}^{(t-1)}) + \varphi(\rho - x_{id}^{(t)}) \quad (6)$$

其中,$\varphi = c_1r_1 + c_2r_3$,$\rho = \dfrac{c_1r_1(p_{id}^{(t)} + r_2gauss_{id}^{(t)}) + c_2r_3p_{gd}^{(t)}}{c_1r_1 + c_2r_3}$。由式(3) ~ (4)可知,速度和位置更新时在维度之间各自独立进行,因此关于稳定性分析过程可简化到一维空间,且只考虑种群中任意一个粒子,这种分析方法是由 Ozcan 等[19]提出,是目前 PSO 算法最有效的分析方法。当算法达到稳定状态时,个体极值和全局极值会在多次进化过程中保持不变,假设 p_{id} 和 p_{gd} 保持不变,分别记作 p 和 g。根据式(5)可知,由于 $gaussian_{id}$ 的数值是由均值 μ 和方差 σ 决定,而本文选取 $\mu = 0$,$\sigma^2 = |p_{id}|$,所以当 p_{id} 在多次进化过程中保持不变,$gaussian_{id}$ 也会不变,记作 s。在上述假设前提下,对式(6)重新整理可得其一维表达式:

$$x_{id}^{(t+1)} = (1 + w - \varphi)x_{id}^{(t)} - wx_{id}^{(t-1)} + \varphi\rho \quad (7)$$

取 $c_1 = c_2 = \lambda > 0$,$r_1,r_2,r_3,r_4 \sim U(0,1)$,有 $E(r_1) =$

$E(r_2) = E(r_3) = E(r_4) = 1/2$，则 $E(\varphi) = \lambda, E(\varphi\rho) = \frac{1}{2}\lambda\left(p + \frac{1}{2}s + g\right)$ 记作 τ，则式(7)的数学期望为：

$$E[x^{(t+1)}] = \theta E[x^{(t)}] - wE[x^{(t-1)}] + \tau \tag{8}$$

其中 $\theta = 1 + w - \lambda$，由式(8)可知：

$$E[x]^* = \frac{\tau}{1 - \theta + w} = \frac{2p + s + 2g}{4} \tag{9}$$

则式(8)相应的特征方程为：

$$P(z) = z^2 - \theta z + w = 0; \quad n = 2 \tag{10}$$

由 Jury 判据[20]的稳定条件，可得式(8)对应的二阶离散线性定常系统在 $E[x]^*$ 处渐近稳定的充要条件为 θ 和 w 满足如下条件：

$$\begin{cases} |w| < 1 \\ 1 - \theta + w > 0 \\ 1 + \theta + w > 0 \end{cases} \tag{11}$$

把 $\theta = 1 + w - \lambda$ 代入到式(11)，整理可得：

$$\begin{cases} |w| < 1 \\ \lambda > 0 \\ \lambda < 2 + 2w \end{cases} \Rightarrow \begin{cases} |w| < 1 \\ 0 < \lambda < 2 + 2w \end{cases} \tag{12}$$

当 w 和 λ 取值满足式(12)，且个体极值、高斯扰动值和全局极值保持不变时，算法在 $E[x]^*$ 处渐近稳定。

3.2 GDPSO 算法进化模式分析

令 $c_1 = c_2$，为了简化分析，假设 GDPSO 算法的速度进化式(3)中 $r_1 = r_2 = r_3 = r$，则种群中所有粒子大致可以分成以下几种情况。

1）粒子 i 当前位置为其历史最优位置，但不是全局最优位置，即 $x_{id} = p_{id}, x_{id} \neq p_{gd}$。则算法进化模式为：

$$\begin{aligned} v_{id}^{(t+1)} &= wv_{id}^{(t)} + c_1 r_1 (p_{id}^{(t)} + r_2 gauss_{id}^{(t)} - x_{id}^{(t)}) + c_2 r_3 (p_{gd}^{(t)} - x_{id}^{(t)}) = wv_{id}^{(t)} + c_1 r_1 \cdot r_2 gauss_{id}^{(t)} + c_2 r_3 (p_{gd}^{(t)} - x_{id}^{(t)}) = \\ & wv_{id}^{(t)} + c_1 r^2 gauss_{id}^{(t)} + c_2 r (p_{gd}^{(t)} - x_{id}^{(t)}) \end{aligned} \tag{13}$$

此时 GDPSO 相当于在“社会模型”上添加了一个高斯扰动项。粒子速度增量比标准 PSO 更大，全局搜索能力更强，收敛速度更快，且添加的高斯扰动还能有效帮助算法逃离局部最优。

2）粒子 i 当前位置为其历史最优位置，同时又是全局最优位置，即 $x_{id} = p_{id} = p_{gd}$。则算法进化模式为：

$$\begin{aligned} v_{id}^{(t+1)} &= wv_{id}^{(t)} + c_1 r_1 (p_{id}^{(t)} + r_2 gauss_{id}^{(t)} - x_{id}^{(t)}) + c_2 r_3 (p_{gd}^{(t)} - x_{id}^{(t)}) = wv_{id}^{(t)} + c_1 r_1 r_2 gauss_{id}^{(t)} = \\ & wv_{id}^{(t)} + c_1 r^2 gauss_{id}^{(t)} \end{aligned} \tag{14}$$

此时标准 PSO 中个体历史最优位置和全局最优位置对当前粒子已经没有任何影响，粒子 i 只依赖于原有速度惯性运动，粒子很容易陷入局部最优。从式(14)可以看出，GDPSO 除了原有速度项还有高斯扰动项，所以此时粒子 i 的速度比标准 PSO 要快，有助于粒子搜索到更好的解，跳出当前局部最优位置。

3）粒子 i 当前位置不是其历史最优位置，同时又不是全局最优位置，但是粒子 i 的历史最优位置恰好是全局最优位置，即 $x_{id} \neq p_{id}, x_{id} \neq p_{gd}$，且 $p_{id} = p_{gd}$。算法的进化模式为：

$$\begin{aligned} v_{id}^{(t+1)} &= wv_{id}^{(t)} + c_1 r_1 (p_{id}^{(t)} + r_2 gauss_{id}^{(t)} - x_{id}^{(t)}) + c_2 r_3 (p_{gd}^{(t)} - x_{id}^{(t)}) = wv_{id}^{(t)} + c_1 r_1 (p_{gd}^{(t)} + r_2 gauss_{id}^{(t)} - x_{id}^{(t)}) + \\ & c_2 r_3 (p_{gd}^{(t)} - x_{id}^{(t)}) = wv_{id}^{(t)} + 2c_1 r [p_{gd}^{(t)} - x_{id}^{(t)} + \frac{1}{2} rgauss_{id}^{(t)}] \end{aligned} \tag{15}$$

此时，如果是标准 PSO 算法，种群中粒子完全朝着当前全局最优位置运动，收敛速度很快。但若当前全局最优位置只是众多局部极值点中的一个而非最优点，则算法陷入局部最优很难逃离。通过式(15)可知，此时 GDPSO 算法的进化模式相当于两个“社会”模型：一方面能够保证算法收敛速度比标准 PSO 更快；另一方面由于增加了高斯扰动项，很好地避免了算法陷入局部最优。

4）粒子 i 当前位置不是其历史最优位置，同时又不是全局最优位置，且粒子 i 的历史最优位置不是当前种群全局最优位置，即 $x_{id} \neq p_{id}, x_{id} \neq p_{gd}$，且 $p_{id} \neq p_{gd}$。算法的进化模式可以写为：

$$\begin{aligned} v_{id}^{(t+1)} &= wv_{id}^{(t)} + c_1 r_1 (p_{id}^{(t)} + r_2 gauss_{id}^{(t)} - x_{id}^{(t)}) + c_2 r_2 (p_{gd}^{(t)} - x_{id}^{(t)}) = wv_{id}^{(t)} + c_1 r_1 (p_{id}^{(t)} - x_{id}^{(t)}) + c_1 r_1 r_2 gauss_{id}^{(t)} + \\ & c_2 r_2 (p_{gd}^{(t)} - x_{id}^{(t)}) \end{aligned} \tag{16}$$

将式(16)与标准 PSO 的速度式(1)比较可知，主要增加了高斯扰动项。作用在于增加速度增量，扩大粒子搜索范围，增强全局寻优能力，以便粒子需找到更好的解，提高算法的收敛速度。

4 仿真实验

4.1 基准测试函数

为了验证 GDPSO 算法的性能，本文选取 4 个单峰函数和 4 个多峰函数来测试 GDPSO 算法。其中：$f_1 \sim f_4$ 是单模态函数，在给定搜索范围内只有一个极值点，主要检验算法的收敛速度和寻优精度；$f_5 \sim f_8$ 是多模态函数，在给定搜索范围内有多个局部极值点，主要考察算法的全局搜索能力和逃离局部最优能力。

1）Sphere 函数(取值范围为 $[-100, 100]^D$，理论最优值为 0)：

$$f_1(x) = \sum_{i=1}^{D} x_i^2$$

2）Schwefel's P2.22 函数(取值范围为 $[-10, 10]^D$，理论最优值为 0)：

$$f_2(x) = \sum_{i=1}^{D} |x_i| + \prod_{i=1}^{D} |x_i|$$

3）Quadric 函数(取值范围为 $[-100, 100]^D$，理论最优值为 0)：

$$f_3(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$$

4）Quadric Noise 函数(取值范围为 $[-1.28, 1.28]^D$，理论最优值为 0)：

$$f_4(x) = \sum_{i=1}^{D} i \cdot x_i^4 + \text{random}[0, 1)$$

5）Rastrigin 函数(取值范围为 $[-5.12, 5.12]^D$，理论最优值为 0)：

$$f_5(x) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$$

6）Noncontinuous Rastrigin 函数(取值范围为 $[-5.12, 5.12]^D$，理论最优值为 0)：

$$f_6(x) = \sum_{i=1}^{D} [y_i^2 - 10\cos(2\pi y_i) + 10];$$

$$y_i = \begin{cases} x_i, & |x_i| < 0.5 \\ \text{round}(2x_i)/2, & |x_i| \geqslant 0.5 \end{cases}$$

7)Ackley 函数(取值范围为$[-32,32]^D$,理论最优值为 0):

$$f_7(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right) + 20 + e$$

8)Griewank 函数(取值范围为$[-600,600]^D$,理论最优值为 0):

$$f_8(x) = \frac{1}{4000}\sum_{i=1}^{D}x_i^2 - \prod_{i=1}^{D}\cos(x_i/\sqrt{i}) + 1$$

4.2　算法的参数设置

种群规模$N=10$,评估次数$T_Fes=200\,000$,学习因子$c_1=c_2=2.0$,惯性权重$w=0.6-(0.6-0.1)\times t/iterNum$,$t$ 表示当前的迭代次数,$iterNum$ 表示设定的最大迭代次数,均值$\mu=0$,方差$\sigma^2=|p_{ij}^{(t)}|$。

为了验证 GDPSO 算法的性能,本文选取了 FIPS[12]、HPSO-TVAC[13]、DMS-PSO[14]、CLPSO[15]和 APSO[16]共 5 个经典改进粒子群算法进行对比实验。各算法的参数设置见文献[16]。

4.3　仿真实验结果

表 1 给出了 6 种算法在 30 维时的寻优结果。表 1 中 Mean、Std. Dev 表示在限定的评估次数下算法的平均最优适应值及标准差,标准差反映了算法的稳定性。为了消除算法的随机性的影响,算法独立运行了 30 次,以最终的平均值作为算法的最后寻优结果。

表 1　6 种优化算法在 30 维时的实验对比

算法	f_1		f_2		f_3		f_4	
	Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev
FIPS	3.21E-30	3.60E-30	1.13E-17	7.86E-18	0.77	0.86	**2.55E-03**	**6.25E-04**
HPSO-TVAC	3.38E-41	8.50E-41	6.90E-23	6.89E-23	2.89E-07	2.97E-07	5.54E-02	2.08E-02
DMS-PSO	3.85E-54	1.75E-53	2.26E-29	6.60E-29	47.50	56.40	1.10E-02	3.94E-03
CLPSO	1.89E-19	1.49E-19	1.01E-13	6.51E-14	395.00	142.00	3.92E-03	1.14E-03
APSO	1.45E-150	5.73E-150	5.15E-84	1.44E-83	**1.00E-10**	**2.13E-10**	4.66E-03	1.70E-03
GDPSO	**1.12E-224**	**0**	**7.90E-226**	**0**	1.12E-01	2.90	5.54E-03	3.40E-02

算法	f_5		f_6		f_7		f_8	
	Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev	Mean	Std. Dev
FIPS	29.98	10.92	35.910	9.490	7.69E-15	9.33E-16	9.04E-04	2.78E-03
HPSO-TVAC	2.39	3.71	1.830	2.650	2.06E-10	9.45E-10	1.07E-02	1.14E-02
DMS-PSO	28.10	6.42	32.800	6.490	8.52E-15	1.79E-15	1.31E-02	1.73E-02
CLPSO	2.57E-11	6.64E-11	0.167	0.379	2.01E-11	9.22E-13	6.45E-13	2.07E-12
APSO	5.80E-15	1.01E-14	**4.14E-16**	**1.45E-15**	1.11E-14	3.55E-15	1.67E-02	2.14E-02
GDPSO	**0**	**0**	4.87E-01	14.38	**3.43E-15**	**6.61E-15**	**0**	**0**

从表 1 对比结果可知,GDPSO 算法在总体上与其他 5 种经典改进算法相比,不论是寻优精度,还是稳定性上都有很大优势。在单模态函数f_1和f_2上,GDPSO 算法的收敛精度远远高于其他算法。在多模态函数上,GDPSO 算法在f_5、f_7和f_8上也有非常好表现;特别是f_8函数,是典型的非线性多模态函数,具有广泛的搜索空间,通常被认为是优化算法很难处理的复杂多模态问题,一般算法很难搜索到最优位置,而 GDPSO 算法却能够很好地搜索到其全局最优位置。

4.4　t 检验结果和 Friendman 检验结果

4.4.1　t 检验结果

除了对算法的收敛精度进行比较外,本文为了判断 GDPSO 算法与其他 5 种算法的性能是否存在显著性差异,进行了 t 检验。t 检验的分位数为单侧 0.05,自由度为 30,t 检验的临界值通过查表为 1.697。当$t>1.697$时说明 GDPSO 优于其他算法,标记为"+";当$t<-1.697$时说明 GDPSO 差于其他算法,标记为"-";否则,说明 GDPSO 与其他算法无明显差异,标记为"="。"$w/t/l$"表示 GDPSO 算法与所选算法相比在w个函数上优于该算法,t个函数上无明显差异,l个函数上差于该算法。表 2 给出了 GDPSO 算法与其他 5 种算法的 t 检验结果。

表 2　GDPSO 算法与其他 5 种算法的 t 检验结果

算法	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	$w/t/l$
FIPS	+	+	+	+	+	+	+	+	8/0/0
HPSO-TVAC	+	+	+	+	+	+	=	+	7/1/0
DMS-PSO	=	+	+	+	+	+	+	+	7/1/0
CLPSO	+	+	+	+	+	+	+	+	8/0/0
APSO	=	+	+	+	+	=	+	+	6/2/0

4.4.2　Friendman 检验结果

为了进一步在统计意义上比较多个算法的性能,采用 Friendman 检验[21]对结果进行分析。表 3 给出 FIPS、HPSO-TVAC、DMS-PSO、CLPSO、APSO 和 GDPSO 共 6 种算法在 8 个测试函数上总体性能的平均排名。算法秩均值越小,性能越好,排名越高,排名最高的算法秩均值用粗体显示。从表 3 可知,6 种算法的性能由好到差依次如下:GDPSO、APSO、CLPSO、FIPS、DMSPSO 和 HPSO-TVAC,GDPSO 算法在 6 种算法中性能最好。

表 3　6 种优化算法的 Friendman 检验结果

算法	秩均值	算法	秩均值
GDPSO	**1.88**	FIPS	4.00
APSO	2.62	DMSPSO	4.25
CLPSO	4.00	HPSO-TVAC	4.25

4.5 算法收敛性能

为了更加直观地说明本文算法在进化过程中的收敛情况，比较了 GDPSO、APSO、CLPSO、DMS-PSO、HPSO-TVAC 和 FIPS 共 6 种算法在 8 个测试函数上关于评估次数和适应值之间的关系。图 2 ~9 是 8 个测试函数在 30 维上的进化过程曲线。

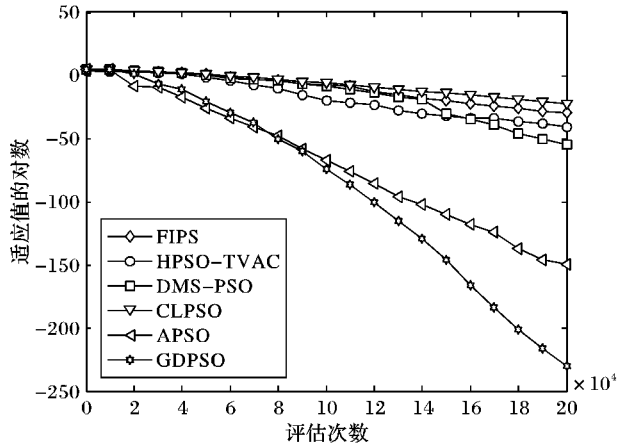

图 2 6 种算法在 f_1 函数上的进化过程曲线对比

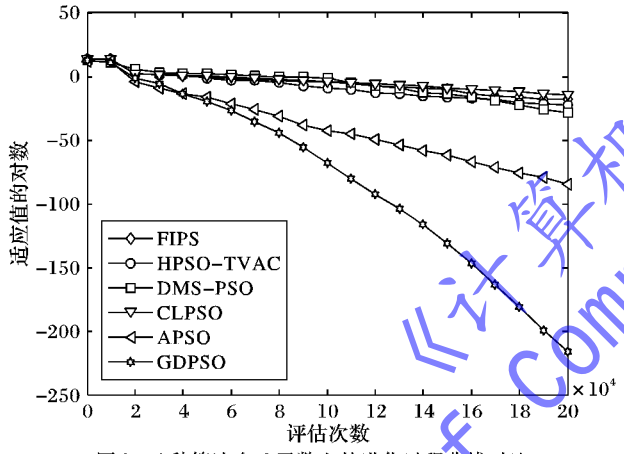

图 3 6 种算法在 f_2 函数上的进化过程曲线对比

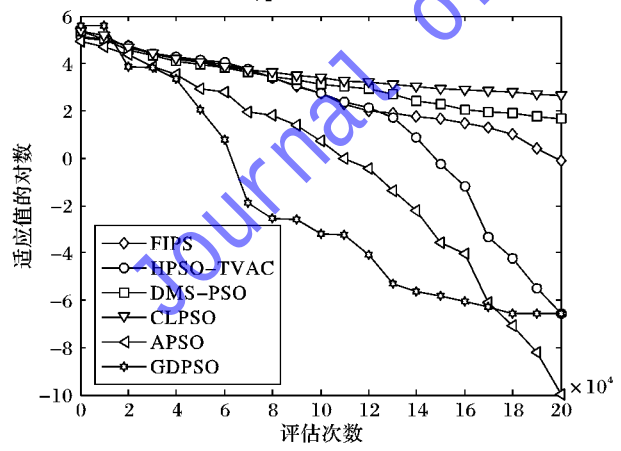

图 4 6 种算法在 f_3 函数上的进化过程曲线对比

由图 2 ~9 可知，本文提出的 GDPSO 算法，通过对个体历史最优位置进行高斯扰动，对于增强算法逃离局部最优能力，加速算法后期收敛速度有着很大的帮助。由图 2 ~3 可知：GDPSO 算法在处理单模态函数时，初期的优势并不明显；但是随着评估次数的增加，算法的收敛速度优势越来越明显，几乎呈直线下降。同时，观察图 6 ~9 可知，GDPSO 算法在处理复杂的多模态函数时，收敛速度依然具有很强的优势，特别是 f_6、f_7 和 f_9 在评估次数达到 40 000 次左右就能够寻找到最优位置(0,0,…,0)，然而其他几种算法很容易陷入局部最优，导致收敛速度变慢，甚至停滞不前。

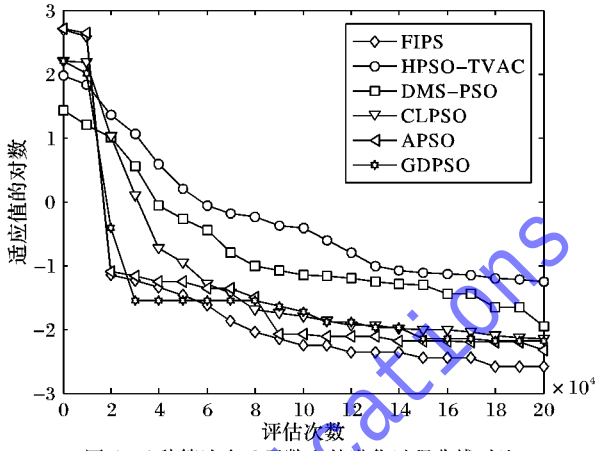

图 5 6 种算法在 f_4 函数上的进化过程曲线对比

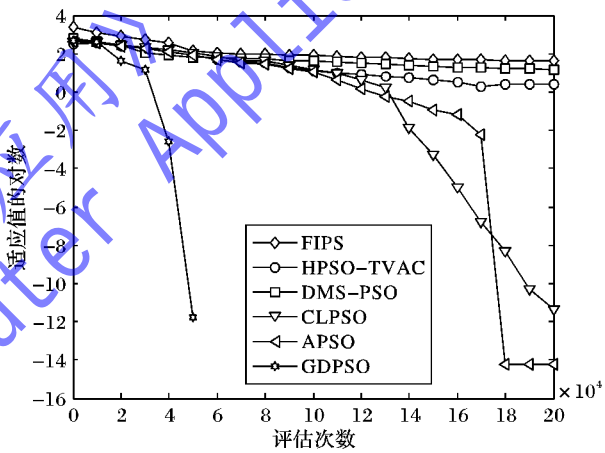

图 6 6 种算法在 f_5 函数上的进化过程曲线对比

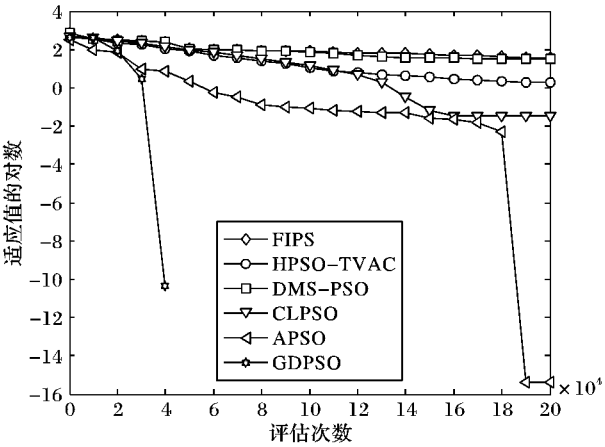

图 7 6 种算法在 f_6 函数上的进化过程曲线对比

5 结语

本文在标准 PSO 算法的基础上，提出一种基于高斯扰动的粒子群优化(GDPSO)算法，很好地改善了标准 PSO 算法收敛速度慢、收敛精度不高和易陷入局部最优的缺点。通过对标准 PSO 向粒子个体历史最优位置学习时，增加一个高斯扰动项，防止算法陷入局部最优，或帮助其逃离局部最优，且能有效提高算法的收敛速度。仿真实验表明，GDPSO 算法在相

同的评估次数下比 FIPS、HPSO-TVAC、DMS-PSO、CLPSO 和 APSO 算法收敛速度更快、精度更高;t 检验和 Friendman 检验结果进一步说明了 GDPSO 算法的优越性。如何更好地设置高斯扰动值,提高算法的性能,是该算法今后需要进一步研究的内容。

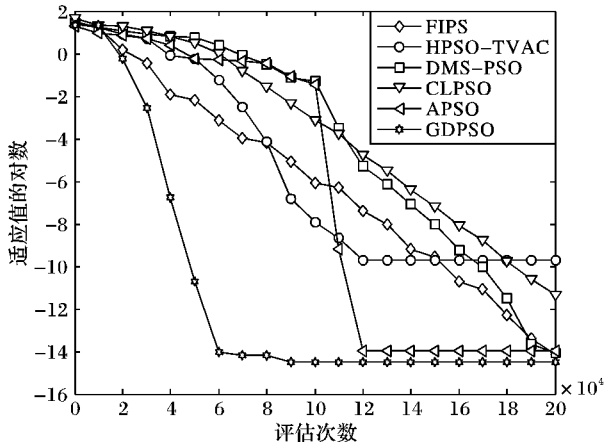

图 8　6 种算法在 f_7 函数上的进化过程曲线对比

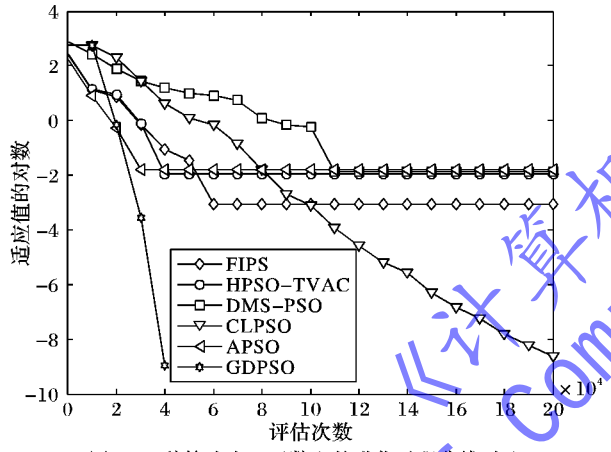

图 9　6 种算法在 f_8 函数上的进化过程曲线对比

参考文献:

[1] KENNEDY J, EBERHART R. Particle swarm optimization [C]// Proceedings of the 1995 IEEE International Conference on Neural Networks, Piscataway, NJ: IEEE Press, 1995: 1942 – 1948.

[2] EBERHART R, KENNEDY J. A new optimizer using particle swarm theory [C]// Proceedings of the 6th International Symposium on Micro Machine and Human Science. Piscataway, NJ: IEEE Press, 1995: 39 – 43.

[3] MA L, SHAN C, QU N. An improved swarm optimization for reactive power optimization [J]. Control Engineering of China, 2012, 19(6): 1077 – 1080. (马立新, 单冠华, 屈娜娜. 基于改进粒子群算法的电力系统无功优化[J]. 控制工程, 2012, 19(6): 1077 – 1080.)

[4] GAO L, LIU X. Resilient particle swarm global optimization algorithm based on chaos [J]. Control and Decision, 2009, 24(10): 1545 – 1548. (高雷阜, 刘旭旺. 基于混沌的弹性粒子群全局优化算法[J]. 控制与决策, 2009, 24(10): 1545 – 1548.)

[5] XIA T, WANG X, LIANG S, *et al.* ANN trained by PSO with adaptive genetic operator and its application [J]. Journal of PLA University of Science and Technology: Natural Science Edition, 2011, 12(1): 70 – 74. (夏天, 王新晴, 梁升, 等. 带自适应遗传算子的粒子群神经网络及其应用[J]. 解放军理工大学学报: 自然科学版, 2011, 12(1): 70 – 74.)

[6] ZOU D. Reconstructing 3D motion trajectory of particle swarms [D]. Shanghai: Fudan University, 2010. (邹丹平. 运动粒子群三维轨迹获取方法研究[D]. 上海: 复旦大学, 2010.)

[7] LI Y, WU M, GAO W, *et al.* A multi-objective optimization algorithm for sintering proportion based on linear programming and genetic algorithm particle swarm optimization [J]. Control Theory and Applications, 2011, 28(12): 1740 – 1746. (李勇, 吴敏, 曹卫华, 等. 基于线性规划和遗传-粒子群算法的烧结配料多目标综合优化方法[J]. 控制理论与应用, 2011, 28(12): 1740 – 1746.)

[8] ZHANG W, SUI Q. Coal dust image segmentation based on improved particle swarm optimization and fuzzy entropy [J]. Control and Decision, 2011, 26(2): 276 – 279. (张伟, 隋青美. 基于改进粒子群优化的模糊熵煤尘图像分割[J]. 控制与决策, 2011, 26(2): 276 – 279.)

[9] van DEN BERGH F. An analysis of particle swarm optimizers [D]. Pretoria, South Africa: University of Pretoria, 2001.

[10] SHI Y, EBERHART R C. A modified particle swarm optimizer [C]// ICEC'98: Proceedings of the 1998 IEEE International Conference on Evolutionary Computation. Piscataway, NJ: IEEE Press, 1998: 69 – 73.

[11] 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.

[12] MENDES R, KENNEDY J, NEVES J. The fully informed particle swarm: simpler, maybe better [J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 204 – 210.

[13] RATNAWEERA A, HALGAMUGE S, WATSON H. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients [J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 240 – 255.

[14] LIANG J J, SUGANTHAN P N. Dynamic multi-swarm particle swarm optimizer [C]// Proceedings of the 2005 Swarm Intelligence Symposium. Piscataway, NJ: IEEE Press, 2005: 124 – 129.

[15] LIANG J J, QIN A K, SUGANTHAN P N, *et al.* Comprehensive learning particle swarm optimizer for global optimization of multimodal function [J]. IEEE Transactions on Evolutionary Computation, 2006, 10(3): 281 – 295.

[16] ZHAN Z H, ZHANG J, LI Y, *et al.* Adaptive particle swarm optimization [J]. IEEE Transactions on Systems, Man, and Cybernetics, 2009, 39(6): 1362 – 1381.

[17] GAO Z, LIAO X. Hybrid adaptive particle swarm optimization based on average velocity [J]. Control and Decision, 2012, 27(1): 152 – 155. (高哲, 廖晓钟. 基于平均速度的混合自适应粒子群算法[J]. 控制与决策, 2012, 27(1): 152 – 155.)

[18] GAO W, LIU S. An efficient particle swarm optimization [J]. Control and Decision, 2011, 26(8): 1158 – 1162. (高卫峰, 刘三阳. 一种高效粒子群优化算法[J]. 控制与决策, 2011, 26(8): 1158 – 1162.)

[19] OZCAN E, MOHAN C K. Particle swarm optimization: surfing the waves [C]// Proceedings of the 1999 IEEE International Conference on Evolutionary Computation. Piscataway, NJ: IEEE Press, 1999: 1934 – 1944.

[20] JURY E I. Inners and stability of dynamic systems [M]. Hoboken, NJ: wiley, 1974.

[21] FRIEDMAN M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance [J]. Journal of the American Statistical Association, 1937, 32(200): 675 – 701.