

基于模式搜索法的鸡群优化算法

张慕雪, 张达敏, 何锐亮

(贵州大学 大数据与信息工程学院, 贵州 贵阳 550025)

摘 要: 针对标准的鸡群容易陷入局部最优的缺陷, 提出一种基于模式搜索的鸡群算法, 在算法的每次迭代中加入对最优解的检测判断, 当发现该解多次出现不变时, 认为算法陷入局部最优, 引入模式搜索帮助算法跳出局部最优解; 利用算法全局最优解和全局次优解之间的差分结果牵引每次迭代中的最优解进行邻域搜索, 在保证算法收敛精度的同时, 提高收敛速度. 通过对 6 个典型的基准测试函数的仿真表明, 改进算法具有较强的全局搜索能力, 同时寻优精度和收敛速度比原算法也有较大的提高, 尤其是在处理高维函数问题上, 改进算法表现出了较强的优势.

关键词: 鸡群算法; 模式搜索; 差分结果; 局部搜索

中图分类号: TP301

文献标识码: A

文章编号: 1000—7180(2018)04—0046—07

A Chicken Swarm Algorithm Based on Hooke-Jeeves

ZHANG Mu-xue, ZHANG Da-min, HE Rui-liang

(College of Big Data and Information Engineering, Guizhou University, Guiyang 550025, China)

Abstract: A Chicken Swarm Optimization based on Hooke-Jeeves(HJCSO) was proposed to solve the defect that traditional Chicken Swarm Optimization(CSO) is easy to fall into local optimum. In the iteration of the algorithm, the examination and judgment of the optimal solution was added. When it was found that the solution remained unchanged many times, the algorithm was considered to be locally optimal. Hooke-Jeeves algorithm was introduced to help the algorithm to jump out of the local optimum. The differential results between the global optimal solution and the global suboptimal solution was used to drag the optimal solution in each iteration for neighborhood search. When it comes to the convergence precision of the algorithm is guaranteed, the convergence speed is improved. The results on five typical standard test functions show that the improved CSO not only improves the global search ability, but also the search efficiency, search accuracy and convergence rate are better than the traditional CSO. Especially in dealing with high-dimensional function problems, the improved HJCSO shows a strong advantage.

Key words: chicken swarm optimization; hooke-jeeves; differential results; local search

1 引言

鸡群算法(Chicken Swarm Optimization, CSO)是 2014 年由 Xianbing Meng 等提出的模拟鸡群中的等级制度和觅食行为的一种优化算法. 与粒子群算法(Particle Swarm Optimization, PSO)、蝙蝠算法(Bat Algorithm, BA)、蚁群算法(Ant Colony Optimization, ACO)等其他优化算法相比, 其具有较好的寻优精度和鲁棒性^[1-2]. 但标准的鸡群算法

由于小鸡粒子容易陷入局部最优解而使算法得不到全局最优解, 降低了算法寻优能力^[2].

模式搜索法(又叫 Hooke-Jeeves 法, HJ)是一种解决最优化问题的直接局部搜索算法, 其目的是找到函数值下降的最佳方向进行搜索而得到最优解^[3], 具有结构简单、求解精度高、不依赖函数性态、局部搜索能力强等特点^[4].

本文提出一种基于模式搜索的鸡群算法(A Chicken swarm algorithm based on Hooke-Jeeves,

HJCSO). 在算法的每一次迭代中,利用全局最优解和全局次优解的差分结果来指导粒子进行邻域局部搜索,增强算法的局部搜索能力,提高算法精度;同时对算法每次迭代的最优解进行判断,如果该解多次出现不变时,认为算法出现早熟停滞,引入模式搜索算法,使算法跳出局部最优。

2 鸡群算法

鸡群算法^[2,5-6]模拟自然界中鸡群的等级制度和觅食行为,将搜索的粒子比作鸡群。算法做如下理想化规定:

(1)将整个鸡群分成若干子群,每个子群都由一只公鸡,若干母鸡和一群小鸡组成。

(2)根据每只鸡自身的适应度值,将若干的鸡分成公鸡、母鸡和小鸡。拥有最好适应度值的鸡作为每个子群的公鸡,每个子群中的公鸡是该子群的头目;具有最差适应度值的鸡被指定为子群中的小鸡;剩余个体为母鸡。子群中小鸡和母鸡的母子关系是随机建立的。

(3)鸡群中的等级制度、支配关系和母子关系一旦建立就保持不变,直到进化代数之后才更新。

(4)鸡群中母鸡和小鸡跟随他们子群中的公鸡寻找食物,同时可以阻止其他子群中的鸡抢夺食物。每只小鸡跟随他们的鸡妈妈寻找食物,且小鸡可以随机偷食其他个体发现的食物。在每个子群中公鸡具有良好的竞争优势,它们优先于其他个体找到食物^[7]。

在解决优化问题时,鸡群中每个个体对应优化问题中的一个解。个体位置 $x_{i,j}(t)$ 表示第 i 个体的第 j 维在第 t 次迭代的值。鸡群中个体寻食的位置更新公式与子群分配的角色有关^[7]。

公鸡作为子群中具有最好适应度值的粒子,拥有更广泛的区域来接近食物。其对应的位置更新公式如下:

$$x_{i,j}^{t+1} = x_{i,j}^t * (1 + \text{Randn}(0, \sigma^2)) \quad (1)$$

$$\sigma^2 = \begin{cases} 1, & \text{if } f_i \leq f_k \\ \exp\left(\frac{f_k - f_i}{|f_i| + \epsilon}\right), & \text{otherwise} \end{cases}$$

$$k \in [1, N], K \neq i \quad (2)$$

式中: $\text{Randn}(0, \sigma^2)$ 为均值为 0, 标准差为 σ^2 的一个高斯分布; ϵ 为一个很小的常熟; k 为所有公鸡中除去 i 后的任一个体。

对于母鸡粒子,它们跟随子群中的公鸡寻找食物。此外,它们还可以随机的偷食其他鸡找到的更好

的食物。其位置更新公式如下:

$$x_{i,j}^{t+1} = x_{i,j}^t + S_1 * \text{Rand} * (x_{r_1,j}^t - x_{i,j}^t) + S_2 * \text{Rand} * (x_{r_2,j}^t - x_{i,j}^t) \quad (3)$$

$$S_1 = \exp\left(\frac{f_i - f_{r_1}}{\text{abs}(f_i) + \epsilon}\right) \quad (4)$$

$$S_2 = \exp((f_{r_2} - f_i)) \quad (5)$$

式中: Rand 为 $[0, 1]$ 之间均匀分布的随机数; r_1 是第 i 只母鸡所在子群中的公鸡; r_2 是整个鸡群中公鸡和母鸡中随机选取的任意个体,且 $r_1 \neq r_2$ 。

小鸡围绕着它们的鸡妈妈寻找食物。其位置更新公式如下:

$$x_{i,j}^{t+1} = x_{i,j}^t + FL * (x_{m,j}^t - x_{i,j}^t) \quad (6)$$

式中: $x_{m,j}^t$ 代表第 i 只小鸡对应鸡妈妈的位置; FL 为 $[0, 2]$ 之间服从均匀分布的随机数,表示小鸡跟随鸡妈妈寻找食物的跟随系数,考虑到个体差异,每只小鸡的 FL 随机在 $[0, 2]$ 之间选择。

鸡群算法伪代码如下:

初始化:鸡群大小 N , 公鸡粒子个数 N_R , 母鸡粒子个数 N_H , 小鸡粒子个数 N_C , 鸡妈妈粒子个数 N_M , 进化代数 G , 最大迭代次数 M

Begin

1. 计算 N 只鸡的适应度值, $t = 0$;
2. WHILE($t < \text{Max_Generation}$)
3. IF($t \% G == 0$)
4. 对鸡的适应度值进行排序,并建立等级制度;
5. 根据排序的适应度值将每只鸡分到不同的群体,并确定小鸡和母鸡之间的母子关系;
6. END IF
7. FOR($i = 1:N$)
8. IF($i == \text{rooster}$)
9. 根据公式(1)更新公鸡粒子的适应度值;
10. END IF
11. IF($i == \text{hen}$)
12. 根据公式(3)更新母鸡粒子的适应度值;
13. END IF
14. IF($i == \text{chick}$)
15. 根据公式(6)更新小鸡粒子的适应度值;
16. END IF
17. 更新鸡群的粒子当前最好位置和鸡群全局最好位置;
18. END FOR
19. END WHILE
- END

3 模式搜索法

模式搜索算法主要由两种移动过程组成:探测移动和模式移动。探测移动是沿坐标轴方向的移动,模式移动则是沿相邻两个探测点连线方向上的移动。两种移动模式交替进行,目的是顺着函数值下降的最佳方向进行搜索^[4,8]。算法流步骤如下:

(1) 给定初始点 $x^{(0)}$, 初始步长 $\delta^0 = (\delta_1^0, \delta_2^0, \dots, \delta_n^0)^T > 0$, 加速系数 $\gamma > 0$, 收缩系数 $\theta \in (0, 1)$ 及精度 $\varepsilon > 0$, 置 $k = 0$;

(2) 令 $y = x^k$;

(3) 从 y 出发, 依次作平行于单位矢量 $e^j (j = 1, \dots, n)$ 的轴向探测移动;

正向探测: 若 $f(y + \delta_j^k e^j) < f(y)$, 则令 $y = y + \delta_j^k e^j$, 否则做负向探测;

负向探测: 若 $f(y - \delta_j^k e^j) \geq f(y)$, 则令 $y = y - \delta_j^k e^j$, 否则令 $y = y$;

(4) 令 $x^{k+1} = y$, 若 $f(x^{k+1}) < f(x^k)$, 则对 x^{k+1} 沿加速方向 $p^k = x^{k+1} - x^k$ 作模式移动, 令 $y = x^{k+1} + \gamma p^k$, $\delta^{k+1} = \delta^k$, $k = k + 1$, 转(3), 否则转(5);

(5) 若 $|\delta^k| < \varepsilon$, 则停止迭代, 输出 x^k , 否则当 $x^{k+1} \neq x^k$ 时, 令 $y = x^{k+1}$, $\delta^{k+1} = \delta^k$, $k = k + 1$, 转(3), 当 $x^{k+1} = x^k$ 时, 令 $y = x^{k+1}$, $\delta^{k+1} = \theta \delta^k$, $k = k + 1$, 转(3)。

4 改进的鸡群算法

4.1 具有局部搜索能力的鸡群算法

小鸡粒子在更新位置时, 只向其鸡妈妈学习, 当鸡妈妈陷入局部最优时, 小鸡因为只获取鸡妈妈的位置信息也会陷入局部最优。为使算法在保证种群多样性的同时, 增加算法局部搜索能力, 在所有粒子完成一次迭代更新适应度值之后, 记录全局最优解 $fMin_1$ 和全局次优解 $fMin_2$, 利用全局最优解和全局次优解的差分结果来指导 $fMin_1$ 进行邻域局部搜索, 增强算法的局部搜索能力。具体操作如下:

$$fMin_1 = fMin_1 + \alpha \cdot p_t (fMin_1 - fMin_2) \quad (7)$$

$$p_{t+1} = p_t (1 - \frac{t}{T}) \quad (8)$$

式中: α 是服从 $[-1, +1]$ 均匀分布的随机数, 主要的控制局部搜索的方向; p_t 是第 t 代进化时的缩放因子, 其随着迭代次数 t 的增加而减小, 目的在迭代初期, $fMin_1$ 离最优解较远, 这时需要较大的局部搜索范围, 而在迭代的后期, $fMin_1$ 离最优解越来越近, 这时需要较小的局部搜索范围来进行精细搜索。

4.2 引入模式搜索法

结合 CSO 和 HJ 各自的优点, 通过 CSO 进行全局搜索, 在迭代的过程中对每一次迭代的优化解进行检测判断, 当出现早熟停滞现象时, 引入模式搜索算法, 使其跳出局部最优, 平衡算法的搜索能力和开发能力。

在迭代的过程中, 当全局最优解 $fMin_1$ 连续多次出现不变的情况时, 认为此时陷入局部最优, 出现早熟停滞现象, 引入模式搜索算法, 对当前解进行模式搜索, 以跳出局部最优。

改进后的算法步骤如下:

(1) 初始化鸡群大小 N , 公鸡粒子个数 N_R , 母鸡粒子个数 N_H , 小鸡粒子个数 N_C , 鸡妈妈粒子个数 N_M , 进化代数 G , 最大迭代次数 M 等相关参数;

(2) 计算每个鸡群粒子的适应度值, $t = 0$;

(3) 判断是否满足结束条件, 满足转步骤(10), 不满足转步骤(4);

(4) 判断是否满足更新鸡群等级制度条件, 满足转步骤(5), 不满足转步骤(6);

(5) 对鸡群粒子的适应度值进行排序, 并建立等级制度, 根据排序的适应度值将每个粒子分到不同的群体, 并随机确定小鸡和母鸡之间的母子关系。

(6) 根据公式(1) ~ (6) 分别对公鸡、母鸡和小鸡粒子进行适应度值更新, 并更新鸡群粒子得当前最优解、全局最优解和全局次优解;

(7) 根据公式(7) ~ (8) 对全局最优解 $fMin_1$ 进行局部搜索;

(8) 判断粒子是否早熟停滞, 满足转步骤(9), 不满足转步骤(4);

(9) 对当前解进行模式搜索, 转步骤(4);

(10) 找到最优解, 输出优化结果。

5 实验仿真

为了测试改进算法 HJCSO 的有效性和客观性, 本文进行了 CSO、HJCSO、PSO、ICSO^[5] 和 RLPSO^[9] 在 6 个典型的基准测试函数中的优化性能比较。本文所用的 6 个典型基准测试函数均来自全局优化测试函数库^[10]。函数形式、搜索区间、理论极小值如表 1 所示。其中, Sphere 和 Rosenbrock 为连续单峰函数, 通常用于衡量算法的收敛速度。函数 Ackley、Rastrigin、Griewank 和 Schaffer 是非线性的多峰函数, 存在大量的局部极值, 通常用于衡量算法的全局搜索性能。

表 1 算法测试函数

函数名称	函数形式	搜索区间	最小值
$f_1(x)$ Sphere	$f_1(x) = \sum_{i=1}^n x_i^2$	$[-100,100]$	0
$f_2(x)$ Rosenbrock	$f_2(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30,30]$	0
$f_3(x)$ Ackley	$f_3(x) = 20 + e - 20\exp(-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^n x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^n \cos(2\pi x_i))$	$[-32,32]$	0
$f_4(x)$ Rastrigin	$f_4(x) = \sum_{i=1}^n [x_i^2 - 10\cos(2\pi x_i) + 10]$	$[-100,100]$	0
$f_5(x)$ Griewank	$f_5(x) = \frac{1}{4000}\sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600,600]$	0
$f_6(x)$ Schaffor	$f_6(x) = 0.5 + \frac{\sin^2(\sqrt{\sum_{i=1}^n x_i^2} - 0.5)}{[1.0 + 0.001(\sum_{i=1}^n x_i^2)]^2}$	$[-100,100]$	0

在对算法进行测试时本文拟从以下三个方面进行. 第一,在种群规模、维数等相关参数一定的情况下,比较寻优精度、稳定性和迭代次数. 第二,在其他参数一定的情况下,比较不同维度下算法的寻优精度. 第三,将 HJCSO 与参考文献中的几种算法进行寻优性能对比.

5.1 在相关参数固定时性能测试

为测试算法性能,实验一固定种群规模为 100,粒子维数为 30,最大迭代次数为 200,对比标准 CSO,PSO 和改进算法 HJCSO 的寻优精度、稳定性和迭代次数. 算法具体相关参数如表 2 所示. 为使结果更客观有效,将三种算法分别对选用的六个函数进行独立实验 50 次,测试结果如表 3 所示. 表 3 分别从最差解、最优解、平均解和方差四个方面来进行对比,对三种算法的性能可以有更直观的认识. 从实验结果可以看出,无论是对连续单峰函数 Sphere 和 Rosenbrock,还是对复杂的非线性多峰函数 Ackley、Rastrigin、Griewank 和 Schaffor,HJCSO 的寻优精度均比标准的 CSO 高出 1~4 个数量级,而 HJCSO 的寻优精度更是远远高出标准的 PSO.

在对函数 $f_2(x)$ 进行的 50 次独立实验中,出现过一次较差的解,使得 HJCSO 对 $f_2(x)$ 的优化结果很接近 CSO 对 $f_2(x)$ 的优化结果. 但从方差来看,CSO 的方差比 HJCSO 的方差平均低了 5 个数量级. 单看对 $f_2(x)$ 的优化,HJCSO 的方差也比 CSO 高出 3 个数量级. 虽然标准的 CSO 与其他优化算法相比,具有更优的寻优能力,但改进的算法在此基础上不仅得到更优的寻优精度,在稳定性方面更是得到了很大地提高.

表 2 算法相关参数设置

算法	参数
CSO	$N=100,G=10,FL\in \text{rand}(0,4,1),D=30$
	$N_R=0.2\cdot N,N_H=0.6\cdot N,N_C=N-N_R-N_H$ $N_M=0.1\cdot N_H$
PSO	$c_1=c_2=1.5,\omega=0.8,N=100$
	$V_{\max}=5,V_{\min}=-5,D=30$
HJCSO	$N=100,G=10,FL\in \text{rand}(0,4,1),D=30$
	$N_R=0.2\cdot N,N_H=0.6\cdot N,N_C=N-N_R-N_H$ $N_M=0.1\cdot N_H$

表 3 CSO,PSO 和 HJCSO 的性能测试对比

测试函数	算法名称	最差解	最优解	平均解	方差
$f_1(x)$	PSO	1.377 6	0.352 9	0.971 4	0.131 2
	CSO	4.052 1e-006	1.461 3e-008	1.106 3e-007	2.043 1e-012
	HJCSO	4.385 4e-009	1.135 1e-010	1.775 3e-009	2.379 1e-018
$f_2(x)$	PSO	272.987 7	74.803 3	197.680 0	3.4876e+03
	CSO	6.591 0e-006	1.416 7e-008	2.942 7e-007	4.791 3e-012
	HJCSO	9.539 2e-007	1.620 3e-009	4.912 2e-008	1.420 5e-015
$f_3(x)$	PSO	2.563 3	0.377 6	1.337 8	0.525 8
	CSO	4.383 9e-004	2.673 0e-005	1.513 8e-004	2.529 0e-008
	HJCSO	7.574 8e-006	5.447 8e-007	2.278 9e-006	4.147 8e-012

测试函数	算法名称	最差解	最优解	平均解	方差
$f_4(x)$	PSO	195.889 1	110.083 2	155.503 2	861.336 1
	CSO	3.166 4e-004	4.387 9e-007	8.627 0e-005	1.190 0e-008
	HJCSO	5.749 8e-008	1.411 1e-009	2.221 2e-008	4.555 5e-016
$f_5(x)$	PSO	0.102 0	0.026 1	0.060 1	7.168 6e-004
	CSO	1.323 0e-005	6.972 9e-009	2.073 1e-006	1.723 9e-011
	HJCSO	8.105 9e-009	2.318 1e-010	3.158 0e-009	6.119 6e-018
$f_6(x)$	PSO	0.440 5	0.279 8	0.355 0	0.003 4
	CSO	1.568 8e-006	5.839 5e-010	2.788 5e-007	3.179 4e-013
	HJCSO	9.275 1e-011	2.692 8e-012	3.637 0e-011	9.238 9e-017

设置目标精度为 10^{-5} ，在固定测试精度的情况下，分别对 HJCSO 和 CSO 进行 20 次独立的实验，比较两种算法初次达到目标精度时的迭代次数，如图 1 所示。由于最大迭代次数为 200 时， $f_3(x)$ 达不到目标精度，所以对 $f_3(x)$ 的最大迭代次数设置为 300。由图 1 可以看出，在达到目标精度时，HJCSO 的迭代次数明显比 CSO 的迭代次数少（HJCSO 达

到目标精度时的迭代次数为 115、83、126、134、109、75；而 CSO 达到目标精度时的迭代次数为 170、149、218、181、167、139）；HJCSO 比 CSO 收敛快（HJCSO 对 6 个函数的收敛次数大约为 42、22、50、63、61、60；而 CSO 对 6 个函数的收敛次数大约 80、64、125、100、122、118）。

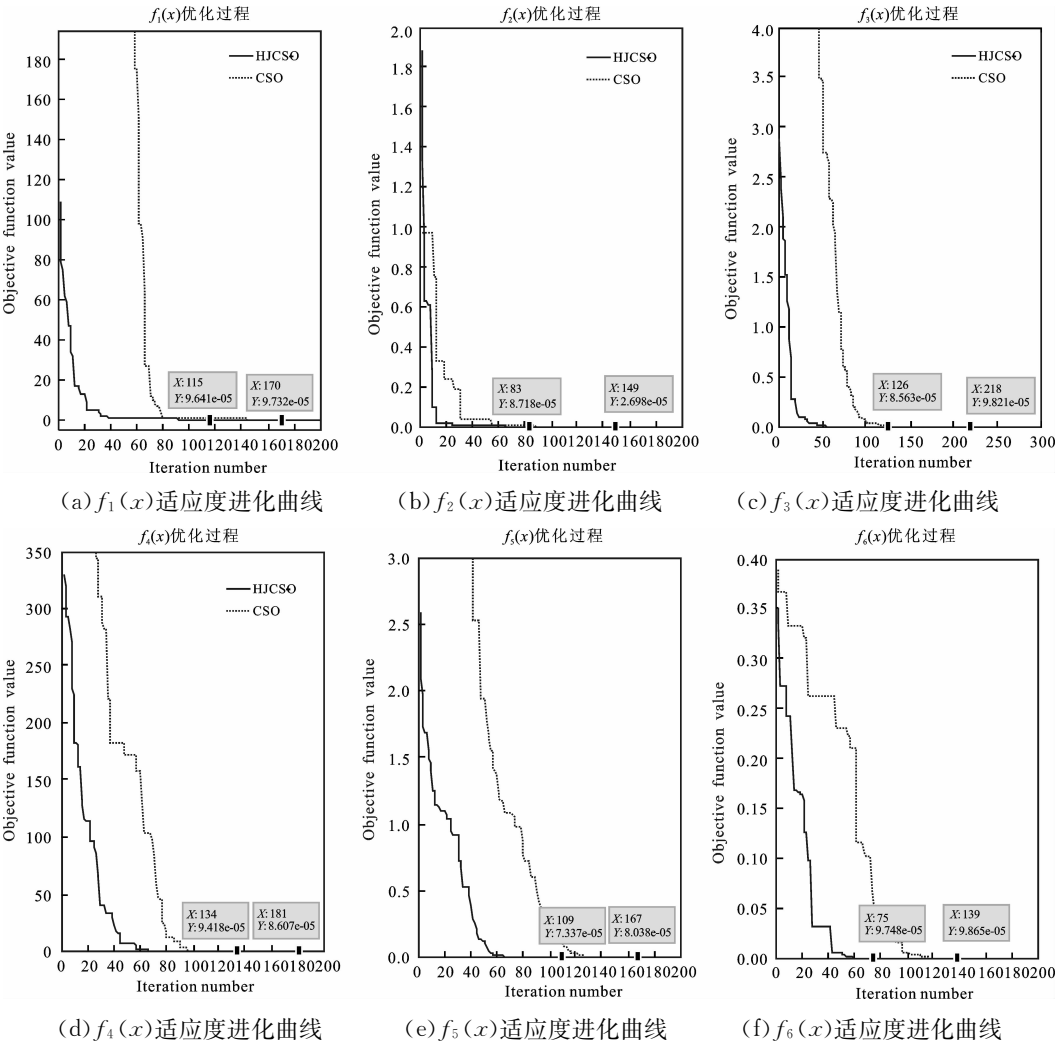


图 1 算法适应度进化曲线

5.2 不同维度下寻优精度对比

设置最大迭代次数为 200,种群规模为 100,分别比较 HJCSO 和 CSO 在维度分别为 10 和 100 下寻优精度,结果如表 4 所示. 对于 $D=100$ 而言, HJCSO 对 6 个函数的寻优结果都远远优于 CSO 的寻优结果,尤其是对函数 $f_1(x)$ 、 $f_4(x)$ 和 $f_6(x)$. 当 $D=10$ 时,由于维数较少,简化了函数的表达式, CSO 对函数 $f_4(x)$ 和 $f_5(x)$ 能得到最优解,但对其余四个函数而言, HJCSO 的求解结果还是比 CSO 的求解结果好.

表 4 不同维度下算法性能对比

函数名称	维数	HJCSO	CSO
$f_1(x)$	10	1.665 2e-26	6.389 9e-22
	100	2.032 9	833.753 5
$f_2(x)$	10	3.523 1e-017	6.183 6e-014
	100	0.941 7	3.465 2
$f_3(x)$	10	6.217 2e-15	4.858 3e-13
	100	0.088 2	6.558 6
$f_4(x)$	10	0	0
	100	0.592 9	423.192 4
$f_5(x)$	10	0	0
	100	0.078 6	5.333 7
$f_6(x)$	10	0	3.330 7e-16
	100	8.169 0e-05	0.4212

5.3 与参考文献的算法性能对比

为比较本文算法与其他改进算法的优劣,将本文算法与参考文献中的几种算法进行比较,引用文献[5]和文献[9]的几组数据,如表 5 所示(“—”表示参考文献未给出).

首先, HJCSO 与 CSO、PSO 的性能对比, HJCSO 比 CSO、PSO 具有更优的性能,从表 5 可知, ICSO 的优化解比 HJCSO 的要好,这是因为 ICSO 的实验结果是在维数为 10,最大迭代次数为 1 000 的情况下得出的. 当维数为 10 时, HJCSO 对函数 $f_4(x)$ 和 $f_5(x)$ 在最大迭代次数为 200 时就能达到最优解为 0 的结果,而对函数 $f_2(x)$ 的寻优结果也远远优于 ICSO. 设置维数为 10,最大迭代次数为 1 000 时, HJCSO 对函数 $f_1(x)$ 和 $f_3(x)$ 的最优解分别为 6.778 7e-135、0, 综上可知, HJCSO 比 ICSO 具有更优的寻有能力. 对比本文算法和文献[10]算法, HJCSO 在对函数 $f_2(x)$ 、 $f_4(x)$ 和 $f_5(x)$ 进行优化时, HJCSO 表现出了比 RLPSO 更明显的优势,但在对函数 $f_1(x)$ 和 $f_3(x)$ 的优化上, RLPSO 具有更优的优化能力. 当增加维数到 100 时, HJCSO 在 6 个函数上的寻有能力均表现出了优于 RLPSO, 这表明,在处理高维函数优化问题上, HJCSO 具有更优的优势.

表 5 与参考文献中算法性能的对比

函数名称	平均最优值				
	HJCSO	CSO	PSO	ICSO ^[5]	RLPSO ^[9]
$f_1(x)$	1.775 3e-09	1.106 3e-07	0.971 4	6.148 0e-132	1.230 0e-13
$f_2(x)$	4.912 2e-08	2.942 7e-07	197.680 0	6.829 1	2.030 0e+02
$f_3(x)$	2.278 9e-06	1.513 8e-04	1.337 8	0	4.990 0e-08
$f_4(x)$	2.221 2e-08	8.627 0e-05	155.503 2	0	1.560 0e+00
$f_5(x)$	3.158 0e-09	2.073 1e-06	0.060 1	0	1.630 0e-02
$f_6(x)$	3.637 0e-11	2.788 5e-07	0.355 0	—	—

6 结束语

CSO 在寻优能力上优于其他的优化算法, 但因其小鸡粒子容易陷入局部最优, 使算法易出现早熟停滞现象. 本文根据较优解之间的差分结果来牵引全局最优解进行局部搜索, 并在判断算法陷入局部最优之后利用模式搜索帮助算法跳出局部最优解, 增加算法寻优能力, 在保证求解精度的情况下, 减少算法的迭代次数, 提高算法收敛速度和稳定性. 通过与原算法和参考文献中的算法对比表明, 本文算法

有效可行.

参考文献:

[1] 张建春,康凤举,梁洪涛,等. 基于鸡群优化的粒子滤波算法研究[J]. 系统仿真学报,2017,29(2):295-300.
[2] 李振璧,王康,姜媛媛. 基于模拟退火的改进鸡群优化算法[J]. 微电子学与计算机,2017,34(2):30-33.
[3] 谢朝政,樊晓光,嵇真福. 引入模式搜索算子的粒子群优化算法[J]. 微电子学与计算机,2015,32(2):97-99.
[4] 龚纯,王正林. 精通 MATLAB 最优化计算[M]. 北京: 电子工业出版社,2012.

[5] 孔飞,吴定会. 一种改进的鸡群算法[J]. 江南大学学报(自然科学版),2015,14(6):681-688.

[6] 黄霞,叶春明,郑军. 混合改进搜索策略的鸡群优化算法[J/OL]. <http://kns.cnki.net/kcms/detail/11.2127.TP.20170304.1727.004.html>. 2017-03-04.

[7] Chen Y L. Combining penalty function with modified chicken swarm optimization for constrained optimization[C]// Proceedings of International Conference on Information Sciences,Machinery,Materials and Energy (ICISMME 2015),2015:9.

[8] 吴建辉,王博华,张小刚,陈华. 基于模式搜索法的云模型粒子群算法[J/OL]. <http://kns.cnki.net/kcms/detail/21.1124.TP.20170619.1006.006.html>, 2017-06-19/2017-07-05.

[9] 夏学文,刘经南. 具备反向学习和局部学习能力的粒子

群算法[J]. 计算机学报,2015,38(7):1397-1407.

[10] HEDAR J. Test functions for unconstrained global optimization[EB/OL]. http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar_files/Test_GO_files/Page364.htm.

作者简介:

张慕雪 女,(1992-),硕士研究生. 研究方向为智能优化算法、数据挖掘.

张达敏(通讯作者) 男,(1967-),博士,教授. 研究方向为优化计算、网络拥塞控制. E-mail:1203813362@qq.com

何锐亮 男,(1990-),硕士研究生. 研究方向为数据挖掘与分析.

(上接第 45 页)

[2] 刘美容,张立玮. 基于小波分解和模糊聚类的模拟电路软故障诊断[J]. 微电子学与计算机, 2014,31(12): 140-143.

[3] 张征凯,谷立臣,曹向辉,等. 基于最优小波包的电机定子电流故障特征提取方法[J]. 煤矿机械, 2016, 37(8):149-151.

[4] 曹岩枫,徐诚. 基于果蝇算法优化广义回归神经网络的机枪枪管初速衰减建模与预测[J]. 兵工学报, 2017, 38(1):1-8.

[5] 史峰,郁磊,王辉. MATLAB 智能算法 30 个案例分析[M]. 北京:北京航空航天大学出版社, 2011: 242-246.

[6] 赵乃刚,李勇. 基于二次飞行和随机扰动的改进蝙蝠算法[J]. 微电子学与计算机, 2017, 34(5):21-25.

[7] 张义良,罗晓娟. 一种基于拟牛顿法蝙蝠算法的网络故障检测方法[J]. 微电子学与计算机, 2017, 34(1): 128-131.

[8] 秦波,刘永亮,王建国,等. 基于小波包最优节点能量特征的 BA-ELM 齿轮故障诊断方法[J]. 机械传动, 2016(4):33-37.

作者简介:

胡鸿志 男,(1977-),博士. 研究方向为测控技术与仪器、电路故障诊断和海洋环境监测.

岑德炼(通讯作者) 男,(1993-),硕士. 研究方向为电路故障诊断和智能优化算法. E-mail:2250645962@qq.com.

吴汝琴 女,(1992-),硕士. 研究方向为心电信号处理.

滕全进 男,(1991-),硕士. 研究方向为嵌入式测控系统.