一种改进的灰狼优化算法

龙 文[1,2],蔡绍洪[1],焦建军[2],伍铁斌[3]
(1. 贵州财经大学贵州省经济系统仿真重点实验室,贵州贵阳 550025;
2. 贵州财经大学数学与统计学院,贵州贵阳 550025;3. 湖南人文科技学院能源与机电工程学院,湖南娄底 417000)

摘 要: 灰狼优化算法是最近提出的一种较有竞争力的优化技术.然而,它的位置更新方程存在开发能力强而探索能力弱的缺点.受差分进化和粒子群优化算法的启发,构建一个修改的个体位置更新方程以增强算法的探索能力;受粒子群优化算法的启发,提出一种控制参数 a 随机动态调整策略.此外,为了提高算法的全局收敛速度,用混沌初始化方法产生初始种群.采用 18 个高维测试函数进行仿真实验,结果表明:对于绝大多数情形,在相同最大适应度函数评价次数下,本文算法的性能明显优于标准灰狼优化算法.

关键词: 灰狼优化算法;差分进化;粒子群优化;控制参数;混沌初始化
中图分类号: TP301.6 **文献标识码:** A **文章编号:** 0372-2112 (2019)01-0169-07
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2019.01.022

An Improved Grey Wolf Optimization Algorithm

LONG Wen[1,2],CAI Shao-hong[1],JIAO Jian-jun[2],WU Tie-bin[3]
(1. *Key Laboratory of Economics System Simulation, Guizhou University of Finance & Economics, Guiyang, Guizhou* 550025, *China*;
2. *School of Mathematics and Statistics, Guizhou University of Finance & Economics, Guiyang, Guizhou* 550025, *China*;
3. *School of Energy and Electrical Engineering, Hunan University of Humanities Science & Technology, Loudi, Hunan* 417000, *China*)

Abstract: Grey wolf optimization (GWO) algorithm is a relatively novel optimization technique which has been shown to be competitive to other population-based algorithms. However, there is still an insufficiency in canonical GWO regarding its position update equation, which is good at exploitation but poor at exploration. Inspired by differential evolution and particle swarm optimization, the personal best information and the random selected individual from population are used to construct a modified position update equation for enhancing the exploration. Inspired by particle swarm optimization, a random adjustment strategy of control parameterais proposed. In addition, to enhance the global convergence, when producing the initial population, the chaos method is employed. Simulation experiments were conducted on the 18 high-dimensional conventional test functions. The simulation results show that the proposed algorithm provides better performance than basic GWO algorithms in the same or less number of maximum fitness function evaluation in most cases.

Key words: grey wolf optimization algorithm; differential evolution; particle swarm optimization; control parameter; chaotic initialization

1 引言

灰狼优化(Grey Wolf Optimization, GWO)算法是 Mirjalili 等于 2014 年提出的一种新型群体智能优化算法[1].它主要模拟自然界中灰狼群体等级机制和捕食行为,通过灰狼群体搜索、包围和追捕攻击猎物等过程实现优化搜索的目的.函数优化测试表明[2],GWO 算法与遗传算法(Genetic Algorithm, GA)、粒子群优化(Particle Swarm Optimization, PSO)、差分进化(Differential Evolution, DE)和万有引力搜索算法(Gravitational Search Algorithm, GSA)相比是有竞争力的.目前,GWO 算法已成功应用于电力系统[3]、无人机路径规划[4]、经济调度指派[5]、PI 控制器优化[6]、车间调度[7]等领域中.

然而,与其他群体智能算法类似,标准 GWO 也存在易陷入局部最优、后期收敛速度慢、求解精度不高等缺点.针对这些问题,学者们提出了许多改进策略来提

收稿日期:2016-12-23;修回日期:2018-07-15;责任编辑:蓝红杰
基金项目:国家自然科学基金(No.61463009);贵州省高校科技拔尖人才支持计划(黔教合 KY 字.[2017]070)

高 GWO 算法的性能. 文献[8]结合 DE 和 GWO,提出一种混合算法用于求解全局优化问题. 文献[9]引入佳点集方法初始化灰狼种群、设计基于正切三角函数描述的非线性控制参数策略和修改的位置更新方程,提出一种协调探索和开发能力的 GWO 算法. 文献[10]将算术交叉和多样性变异引入到 GWO 算法中,提出一种嵌入遗传算子的改进 GWO 算法. 文献[11]将动态进化种群技术嵌入到标准 GWO 算法中以加强其全局探索能力. 然而,到目前为止,改进 GWO 都很难在加快算法收敛速度和避免陷入局部最优两方面取得平衡.

在标准 GWO 算法中,它的位置更新方程仅以全局最优个体(α 狼)引导,即位置更新方程具有局部开发能力强而全局探索能力弱的缺点. 另外,距离控制参数 a 对协调算法的探索和开发能力起着关键作用,但其设置为随进化迭代次数从 2 线性递减到 0. 然而,在实际优化问题中,由于算法搜索过程极为复杂,控制参数 a 线性递减策略很难适应搜索实际情况. 针对上述不足,受 DE 和 PSO 算法的启发,本文设计出一个改进的个体位置更新方程;受 PSO 算法的启发,将随机分布函数引入到距离控制参数 a 中以协调算法的探索和开发能力;采用混沌进行初始化以加快算法的全局收敛速度. 仿真实验表明该算法能显著提高优化性能.

2 灰狼优化算法

GWO 算法是模拟自然界中灰狼群体的社会等级机制和捕猎行为而衍生出的一种新型群体智能优化算法. 在 GWO 算法中,每只灰狼代表种群中 1 个候选解,其中,群体中最优解称为 α,次最优解称为 β,第三最优解称为 δ,其他解均称为 ω. 假设灰狼种群规模为 N,搜索空间为 D 维,第 i 只灰狼在第 D 维空间中的位置可表示为 $\vec{X}_i=(x_i^1,x_i^2,\cdots,x_i^D)$,$i=1,2,\cdots,N$. 灰狼群体通过式(1)逐渐接近并包围猎物:

$$X_i^d(t+1)=X_p^d(t)-A\cdot|C\cdot X_p^d(t)-X_i^d(t)| \tag{1}$$

其中,t 为当前迭代次数,$X_p=(x_p^1,x_p^2,\cdots,x_p^D)$ 为猎物位置,$A\cdot|C\cdot X_p^d(t)-X_i^d(t)|$ 为包围步长,A 和 C 是系数,可定义为:

$$A=2a\cdot r_1-a \tag{2}$$

$$C=2\cdot r_2 \tag{3}$$

其中,r_1 和 r_2 为[0,1]之间的随机数,a 称为距离控制参数,随迭代次数增加从 2 线性减小到 0,即

$$a=2-\frac{2t}{t_{\max}} \tag{4}$$

式中 $t_{\max}$ 为最大迭代次数.

狼群中其他灰狼个体 X_i 根据 α、β 和 δ 的位置 X_α、X_β 和 X_δ 来更新各自的位置:

$$\begin{cases}X_{i,\alpha}^d(t+1)=X_\alpha^d(t)-A\cdot|C\cdot X_\alpha^d(t)-X_i^d(t)|\\ X_{i,\beta}^d(t+1)=X_\beta^d(t)-A\cdot|C\cdot X_\beta^d(t)-X_i^d(t)|\\ X_{i,\delta}^d(t+1)=X_\delta^d(t)-A\cdot|C\cdot X_\delta^d(t)-X_i^d(t)|\end{cases} \tag{5}$$

$$X_i^d(t+1)=\frac{X_{i,\alpha}^d(t)+X_{i,\beta}^d(t)+X_{i,\delta}^d(t)}{3} \tag{6}$$

3 改进的灰狼优化算法

3.1 基于混沌的种群初始化方法

混沌具有随机性、遍历性和规律性等特点,目前已广泛应用于 PSO、DE 和 ABC 等群体智能优化算法中以提高算法的搜索效率[12]. 为了使初始种群个体尽可能地利用解空间的信息,本文采用 Skew Tent 映射产生混沌序列用来进行种群初始化,其数学模型为[13]:

$$\begin{cases}x_{k+1}=x_k/\varphi, & 0<x_k<\varphi\\ x_{k+1}=(1-x_k)/(1-\varphi), & \varphi<x_k<1\end{cases} \tag{7}$$

当 $\varphi\in(0,1)$ 且 $x\in[0,1]$ 时,系统(7)处于混沌状态.

利用 Skew Tent 混沌产生初始群体的具体步骤如算法 1 所示.

算法 1 基于 Skew Tent 混沌的种群初始化方法

设置种群规模 N 和最大混沌迭代步数 K.
for $i=1$ to N do
  for $j=1$ to D do
    随机产生一个数 $\varphi_{0,j}\in(0,1)$
    for $k=1$ to K do
      if $0<x_{k,j}<\varphi_{k-1,j}$ do
        $x_{k,j}=x_{k-1,j}/\varphi_{k-1,j}$
      else
        $x_{k,j}=(1-x_{k-1,j})/(1-\varphi_{k-1,j})$
      end if
    end for
    $x_{i,j}=x_{\min,j}+x_{k,j}\cdot(x_{\max,j}-x_{\min,j})$
  end for
end for

3.2 修改位置更新方程

由位置更新方程(5)和(6)可以看出,标准 GWO 算法在个体位置迭代更新过程中只考虑了个体当前位置信息和群体历史最优位置信息,由于有群体最优位置 X_α(α 狼的位置)进行引导搜索,因此,该位置更新方程具有较好的开发能力而忽略了算法的探索能力,在求解多峰值问题时易陷入局部最优.

为进一步增强 GWO 算法的探索能力和加快收敛速度,受 DE 和 PSO 算法的启发,从群体中随机选取个体与当前个体进行差分搜索,同时将 PSO 算法中对粒子自身运动历史最优解进行记忆保存的思想引入到 GWO 算法中,对个体的记忆功能加以改进,使其能够记

忆自身进化过程中的最优解. 因此,本文设计出一种修改的位置更新方程替代原位置更新方程(6):

$$X_i^d(t+1)=\frac{X_{i,\alpha}^d(t)+X_{i,\beta}^d(t)+X_{i,\delta}^d(t)}{3}+b_1\cdot r_3\cdot(P_{i,\text{best}}^d(t)-X_i^d(t))+b_2\cdot r_4\cdot(X_j^d(t)-X_i^d(t)) \tag{8}$$

其中,$b_1\in[0,1]$称为个体记忆系数,$b_2\in[0,1]$称为交流系数,r_3、r_4为[0,1]间的随机数,$P_{i,\text{best}}^d$表示第 i 只灰狼个体所经历的最佳位置,X_j为群体中随机选择的个体且 $j\neq i$,通过调节系数 b_1 和 b_2 的值,可以协调群体和个体记忆对 GWO 算法搜索的影响.

从式(8)所示的位置更新方程可以看出,与原位置更新方程(6)相比,式(8)的右边增加了两个部分,右边第一部分引入了个体自身最优信息,以进一步加强算法的开发能力和加快收敛速度;右边第二部分以随机选择个体作为引导搜索,从而增强群体的多样性和全局探索能力. 通过调节系数 b_1 和 b_2,从而平衡算法的开发和探索能力.

3.3　控制参数随机调整策略

由式(2)可以看出,在迭代过程中,系数 A 的值随距离控制参数 a 的变化而不断变化. 换句话说,距离控制参数 a 的设置影响了 GWO 的探索和开发能力之间的平衡. 但是,由式(4)可知,距离控制参数 a 随迭代次数增加从 2 线性减小到 0. 在迭代初期较大的距离控制参数 a 值使得搜索步长较大,探索能力较强,避免算法出现早熟收敛;在迭代后期较小的 a 值使得群体集中在某个区域搜索,开发能力较强,加快收敛速度. 然而,在实际优化问题中,由于算法搜索过程极为复杂,控制参数 a 线性递减策略很难适应搜索实际情况. 文献[1]仿真结果表明,GWO 在解决多峰值问题时易陷入局部最优.

若距离控制参数 a 设定为服从某种分布的随机数,利用随机变量的特性调整控制参数 a 的值,有利于算法跳出局部最优;若在最优个体附近,随机分布控制参数能产生相对较小的值,这样有利于加快算法的收敛速度. 原因在于随机性使得控制参数既能在迭代初期有机会取得较大或较小的值,又能在算法迭代后期取得较小或较大的值. 基于上述分析,本文提出一种随机分布调整控制参数策略,即

$$a(t)=a_{initial}-(a_{initial}-a_{final})\times rand()+\sigma\times randn() \tag{9}$$

其中,$a_{initial}$和 a_{final}分别为距离控制参数 a 的初始值和终止值,t 为当前迭代次数,$rand()$为[0,1]服从均匀分布的随机数,$randn()$为服从正态分布的随机数,σ(方差)用来度量随机变量控制参数 a 与其数学期望(即均值)之间的偏离程度,是为了控制取值中的参数 a 误差,使控制参数 a 有利于向期望控制参数 a 方向进化.

3.4　EGWO 算法步骤

综上所述,本文提出的高效灰狼优化算法(记为 EGWO)的算法步骤如算法 2 所示.

算法 2　EGWO 算法

```
begin
    设置算法参数:种群规模 N,最大迭代次数 t_max,距离控制参数 a
    的初始值 a_initial和终止值 a_final,群体交流系数 b1 和个体记忆系
    数 b2.
    利用算法 1 初始化产生灰狼种群{X_i,i=1,2,…,N}.
    计算群体中每个个体的适应度值{f(X_i),i=1,2,…,N},并记录
    当前最优个体 α、次最优个体 β 和第三最优个体 δ,其对应位置分
    别为 X_α、X_β 和 X_δ.
    while (t < t_max) do
      for i=1 to N do
        for j=1 to D do
          根据式(9)计算距离控制参数 a 的值.
          由式(2)和(3)计算参数 A 和 C 的值.
          根据式(6)和(8)更新个体的位置.
        end for
      end for
      计算群体中个体的适应度值{f(X_i),i=1,2,…,N}.
      更新最优个体 α、次最优个体 β 和第三最优个体 δ 及其对应位置
      X_α、X_β 和 X_δ.
      t=t+1;
    end while
end
```

4　仿真实验

4.1　测试函数及参数设置

为了测试 EGWO 算法处理高维函数优化问题的能力,本节将它用于求解 18 个标准测试函数,即 Sphere(f_1)、Schwefel2.22(f_2)、Schwefel1.2(f_3)、Schwefel2.21(f_4)、Rosenbrock(f_5)、Step(f_6)、Quartic(f_7)、Sumsquare(f_8)、Rastrigin(f_9)、Ackley(f_{10})、Griewank(f_{11})、Alpine(f_{12})、Levy(f_{13})、Cosine Mixture(f_{14})、Levy Montalo(f_{15})、Sumpower(f_{16})、Elliptic(f_{17})和 Zakharov(f_{18}),其中,f_1-f_8为单峰函数,f_9-f_{18}为多峰函数. 18 个函数的维数分别设置为 $D=30$、$D=100$、$D=500$ 和 $D=1000$ 维,EGWO 算法参数设置如下:种群规模均为 $N=30$,最大迭代次数均为 $t_{\max}=500$,距离控制参数初始值 $a_{initial}=2$,终止值 $a_{final}=0$,个体记忆系数 $b_1=0.1$,群体交流系数 $b_2=0.9$.

4.2　参数 b_1 和 b_2 对算法性能的影响分析

由位置更新方程(8)可知,个体记忆系数 b_1 和群体交流系数 b_2 对协调算法的探索和开发能力具有一定的影响. 本小节通过对个体记忆系数 b_1 和群体交流系数

b_2对选取不同的值,即 b_1 分别取 0.1、0.3、0.5、0.7 和 0.9,b_2分别取 0.1、0.3、0.5、0.7 和 0.9,组合 b_1 和 b_2 的值进行数值实验,分析其对 EGWO 算法性能的影响. 表 1 给出了不同 b_1 和 b_2 组合值对 18 个测试函数($D=30$ 维)的寻优结果比较,需要说明的是,除了参数 b_1 和 b_2,其余参数的值与 4.1 小节相同. 表中黑体字为当 EGWO 算法选取不同的 b_1 和 b_2 值中的最好结果.

从表 1 中结果可知,总体来说,当 $b_1=0.1$ 且 $b_2=0.9$ 时算法的性能最优. 与 $b_1=0.3$ 且 $b_2=0.7$ 和 $b_1=0.5$ 且 $b_2=0.5$ 时相比,当 $b_1=0.1$ 且 $b_2=0.9$ 时算法在 12 个函数上获得较好的寻优结果;在其余函数上获得了相似的结果. 与 $b_1=0.7$ 且 $b_2=0.3$ 相比,算法当 $b_1=0.1$ 且 $b_2=0.9$ 时在 13 个函数上获得了较好的寻优结果;在四个函数上得到了全局最优解 0. 与 $b_1=0.9$ 且 $b_2=0.1$ 相比,当 $b_1=0.1$ 且 $b_2=0.9$ 时算法在 15 个函数上获得较好的寻优结果;在函数 f_6 和 f_{14} 上获得了相似的结果. 对于 f_5,当 $b_1=0.1$ 且 $b_2=0.9$ 时获得的寻优结果稍差. 从上述比较结果可知,$b_1=0.1$ 且 $b_2=0.9$ 是一个合适的参数设置.

表 1 不同的 b_1 和 b_2 值对 EGWO 算法性能的影响结果比较($D=30$ 维)

函数	$b_1=0.1, b_2=0.9$		$b_1=0.3, b_2=0.7$		$b_1=0.5, b_2=0.5$		$b_1=0.7, b_2=0.3$		$b_1=0.9, b_2=0.1$	
	Mean	St. dev	Mean	St. dev	Mean	St. dev	Mean	St. dev	Mean	St. dev
f_1	**1.43E-226**	**0**	1.21E-188	0	1.33E-147	4.10E-147	3.00E-100	9.20E-100	8.65E-056	2.57E-057
f_2	**3.13E-120**	**3.10E-119**	9.57E-101	2.00E-100	7.56E-080	1.35E-079	2.92E-055	7.27E-055	5.71E-032	8.28E-032
f_3	**2.90E-173**	**0**	2.95E-134	9.30E-134	6.01E-093	1.90E-092	1.61E-052	3.66E-052	1.87E-013	4.17E-013
f_4	**1.31E-100**	**4.90E-100**	3.42E-082	9.12E-082	7.22E-062	1.95E-061	1.78E-035	4.87E-035	1.27E-007	2.38E-007
f_5	28.7769	0.25365	28.6917	0.32026	28.5622	0.38824	28.2015	0.61641	28.5611	0.37908
f_6	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_7	**3.61E-005**	**3.36E-005**	3.95E-005	4.15E-005	1.04E-004	8.00E-005	1.79E-004	1.18E-004	7.42E-004	4.82E-004
f_8	**4.44E-224**	**0**	8.85E-191	0	7.28E-149	2.00E-148	5.87E-102	1.80E-101	6.80E-051	1.56E-050
f_9	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	2.67E+001	3.63E+001
f_{10}	**4.44E-015**	**0**	4.44E-015	0	5.15E-015	1.50E-015	6.93E-015	1.71E-015	1.15E-014	3.35E-015
f_{11}	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	5.37E-003	8.65E-003
f_{12}	**1.67E-120**	**8.90E-120**	1.90E-102	1.44E-102	2.83E-075	8.95E-075	2.36E-034	7.36E-034	5.63E-004	6.98E-004
f_{13}	**9.45E-226**	**0**	5.56E-193	0	6.36E-152	9.60E-152	2.08E-101	5.69E-101	6.74E-055	2.11E-054
f_{14}	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_{15}	**1.71E-228**	**0**	2.57E-193	0	1.40E-151	2.60E-151	9.71E-104	2.70E-103	2.51E-058	3.75E-058
f_{16}	**0**	**0**	**0**	**0**	**0**	**0**	6.02E-285	0	6.01E-174	0
f_{17}	**3.80E-218**	**0**	9.90E-188	0	1.61E-144	3.60E-144	5.46E-097	1.31E-096	1.19E-050	2.00E-050
f_{18}	**4.85E-162**	**0**	1.40E-135	4.40E-135	1.08E-098	2.32E-098	6.41E-062	9.24E-062	1.11E-020	3.13E-020

4.3 与标准 GWO 算法的比较

采用 EGWO 算法对 18 个标准测试函数进行求解,并与标准 GWO 算法的结果进行比较,通过统计两种算法不同性能测度的结果,对比分析它们的优化性能,因而验证 EGWO 算法的有效性和可行性. 所有仿真实验均在 Intel Core Quad,CPU:Q8300、2G 内存、2.50GHz 主频的计算机上实现,程序采用 MATLAB 7.0 语言实现.

在对比实验中,为了比较的公平性,标准 GWO 算法和 EGWO 算法采用相同的实验参数. 对每个测试函数,标准 GWO 算法和 IGWO 算法在上述参数设置下均独立运行 20 次实验,分别记录两种算法的最优精度值(Best)、平均精度值(Mean)、最差精度值(Worst)和精度标准差值(St. dev). 其中,Best、Worst 反映了解的质量;Mean 显示了在给定的迭代次数(函数评价次数)下算法所能达到的精度,反映了算法的收敛速度;St. dev 反映了算法的稳定性和鲁棒性. 结果如表 2 所示.

从表 2 中实验结果可知,当函数维数设置为 $D=30$ 时,EGWO 算法在 5 个测试函数上 30 次实验均能一致地收敛到理论最优值 0. 对于函数 f_1,f_2,f_3,f_4,f_8,f_{12},f_{13},f_{15},f_{17} 和 f_{18},EGWO 算法虽然没有收敛到理论最优值,但其获得的值非常接近理论最优值. 对于函数 f_5,EGWO 算法没有收敛. 与标准 GWO 算法相比,EGWO 在 13 个函数上取得了较好的寻优结果. 对于测试函数 f_6,f_{11} 和 f_{14},标准 GWO 算法和 EGWO 算法获得了相似的求解结果. 然而,对于函数 f_5,标准 GWO 算法获得了稍好的结果. 对于函数 f_9,两种算法得到了相似的最优值(Best),而 EGWO 取得了较好的平均值(Mean)、最差值(Worst)和标准差(St. dev).

表 2　EGWO 算法和 GWO 算法对 18 个函数的寻优结果比较

函数	维数	GWO 算法				EGWO 算法			
		Best	Mean	Worst	St. dev	Best	Mean	Worst	St. dev
f_1	30	3.24E－030	1.69E－029	5.18E－029	1.09E－029	8.98E－230	1.43E－226	3.00E－225	0
f_2	30	1.44E－018	4.35E－018	1.42E－017	2.87E－018	2.96E－121	3.13E－120	6.16E－119	3.10E－119
f_3	30	1.71E－007	1.22E－006	5.78E－005	1.79E－005	2.57E－179	2.90E－173	2.64E－172	0
f_4	30	3.20E－008	7.30E－008	9.34E－007	5.18E－007	1.47E－102	1.31E－100	1.57E－099	4.90E－100
f_5	30	26.3078	27.3005	28.5613	0.84522	28.1005	28.7769	28.9191	0.25365
f_6	30	0	0	0	0	0	0	0	0
f_7	30	5.87E－004	1.64E－003	3.45E－003	1.02E－003	8.69E－007	3.61E－005	1.22E－004	3.36E－005
f_8	30	1.30E－031	1.69E－030	1.18E－029	3.58E－030	9.05E－231	4.44E－224	1.65E－223	0
f_9	30	0	2.09E＋000	1.07E＋001	4.42E＋000	0	0	0	0
f_{10}	30	5.77E－014	6.80E－014	7.90E－014	7.94E－015	4.44E－015	4.44E－015	4.44E－015	0
f_{11}	30	0	0	0	0	0	0	0	0
f_{12}	30	8.53E－018	2.27E－004	9.17E－004	3.38E－004	1.99E－122	1.67E－120	2.89E－119	8.90E－120
f_{13}	30	2.22E－032	6.19E－031	1.71E－030	5.96E－031	1.88E－233	9.45E－226	7.64E－225	0
f_{14}	30	0	0	0	0	0	0	0	0
f_{15}	30	7.50E－035	8.40E－033	7.20E－032	2.23E－032	6.86E－234	1.71E－228	2.08E－227	0
f_{16}	30	2.12E－110	1.30E－102	1.15E－101	2.70E－101	0	0	0	0
f_{17}	30	1.13E－027	3.96E－026	4.73E－025	4.19E－026	4.30E－223	3.80E－218	3.70E－217	0
f_{18}	30	2.01E－010	1.17E－008	3.39E－008	1.11E－008	1.27E－169	4.85E－162	1.40E－161	0

为了更直观地反映算法的寻优效果,图 1 给出了 EGWO 算法与标准 GWO 算法对 6 个测试函数的收敛曲线. 从图 1 可以清晰地看出,无论是收敛精度还是收敛速度,EGWO 算法比 GWO 算法有了显著的提高.

为了进一步验证 EGWO 算法求解高维优化问题的能力,将 18 个测试函数的维数分别设置为 $D=100$、$D=$

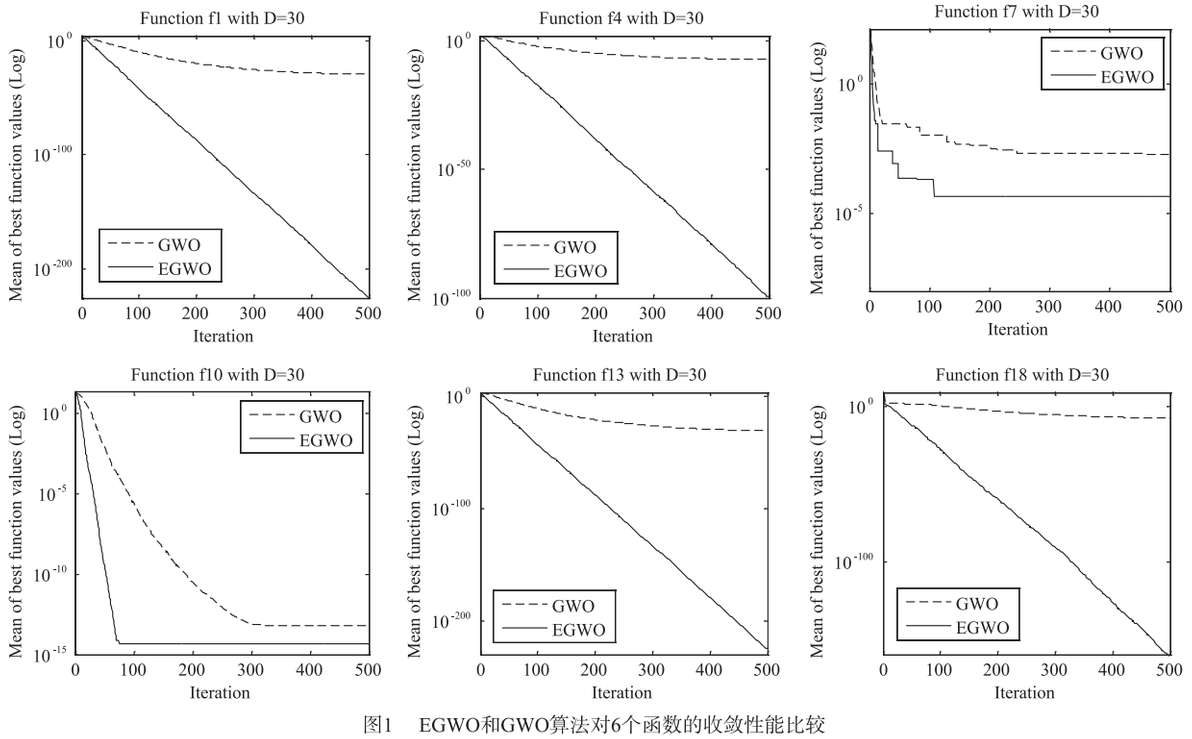

图1　EGWO和GWO算法对6个函数的收敛性能比较

500 和 $D=1000$ 维. EGWO 算法的参数设置与求解 $D=30$ 一样,即种群规模为 30,最大迭代次数为 500. 表 3 给出了 EGWO 算法在最大适应度函数评价次数为 15000 次下,30 次实验得到的平均值(Mean)和标准差(St. dev).

表 3 EGWO 算法对 **18** 个函数 **30** 次实验的寻优结果比较

函数	D = 100 维		D = 500 维		D = 1000 维	
	Mean	St. dev	Mean	St. dev	Mean	St. dev
f_1	2.91E - 188	0	8.48E - 170	0	1.44E - 162	0
f_2	5.39E - 100	9.40E - 100	3.65E - 092	7.30E - 092	1.86E - 088	3.19E - 088
f_3	5.94E - 145	1.90E - 144	3.06E - 133	8.10E - 133	1.42E - 123	4.50E - 123
f_4	8.70E - 088	2.51E - 087	4.91E - 080	8.45E - 080	8.28E - 078	1.09E - 077
f_5	9.88E + 001	8.79E - 002	4.99E + 002	3.41E - 002	9.99E + 002	4.55E - 002
f_6	0	0	0	0	0	0
f_7	1.09E - 004	1.13E - 004	1.17E - 004	2.82E - 004	1.34E - 004	3.41E - 004
f_8	2.24E - 191	0	1.28E - 169	0	1.63E - 163	0
f_9	0	0	0	0	0	0
f_{10}	4.44E - 015	0	4.44E - 015	0	5.86E - 015	1.83E - 015
f_{11}	0	0	0	0	0	0
f_{12}	1.72E - 100	2.00E - 100	1.07E - 089	1.36E - 089	2.41E - 086	4.39E - 086
f_{13}	7.92E - 192	0	4.29E - 172	0	9.38E - 166	0
f_{14}	0	0	0	0	0	0
f_{15}	8.12E - 193	0	1.56E - 173	0	1.30E - 167	0
f_{16}	0	0	0	0	0	0
f_{17}	2.71E - 188	0	5.85E - 168	0	8.08E - 161	0
f_{18}	2.20E - 103	7.00E - 103	3.43E - 037	1.06E - 036	5.55E - 021	1.76E - 020

从表 3 中结果可以看出,当测试函数的维数分别为 $D=100$、$D=500$ 和 $D=1000$ 时,除了函数 f_5 和 f_{18} 以外,EGWO 算法在其他 16 个测试函数上获得的性能变化不明显,这充分说明 EGWO 算法具有较强的求解高维优化问题的能力,较小的标准差值说明 EGWO 算法具有较强的鲁棒性和稳定性.

5 结束语

受差分进化算法和粒子群优化算法的启发,本文提出了一种高效的灰狼优化算法(简记为 EGWO). 通过引入修改位置更新方程和非线性调整距离控制参数策略,平衡了算法的探索和开发能力. 对 18 个标准测试函数的实验结果表明,本文提出的 EGWO 算法在优化效率、优化性能和鲁棒性方面比标准灰狼优化算法及改进的灰狼优化算法有了较大的改善. 然而,EGWO 算法也有自身的局限性. 对很难求解的 f_5 - Rosenbrock 函数,EGWO 算法和 GWO 算法的表现都不佳. 下一步研究内容将 EGWO 算法应用于大规模函数优化问题、多目标优化问题、约束优化问题和实际工程问题中.

参考文献

[1] MIRJALILI S, MIRJALILI S M, LEWIS A. Grey wolf optimizer[J]. Advances in Engineering Software, 2014, 69(3): 46 - 61.

[2] LONG W, LIANG X, JIAO J, et al., An exploration-enhanced grey wolf optimizer to solve high-dimensional numerical optimization[J]. Engineering Application of Artificial Intelligence, 2018, 68: 63 - 80.

[3] GUHA D, ROY P K, BANERJEE S. Load frequency control of interconnected power system using grey wolf optimization[J]. Swarm and Evolutionary Computation, 2016, 27: 97 - 115.

[4] 姚鹏,王宏伦. 基于改进流体扰动算法与灰狼优化的无人机三维航路规划[J]. 控制与决策, 2016, 31(4): 701 - 708.
YAO Peng, WANG Hong-lun. Three-dimensional path planning for UAV based on improved interfered fluid dynamical system and grey wolf optimizer[J]. Control and Decision, 2016, 31(4): 701 - 708. (in Chinese)

[5] SONG H, SULAIMAN M, MOHAMED M. An application

of grey wolf optimizer for solving combined economic emission dispatch problems [J]. International Review on Modeling and Simulation, 2014, 7(5): 838 – 844.

[6] GUPTA E, SAXENA A. Robust generation control strategy based on grey wolf optimizer[J]. Journal of Electrical Systems, 2015, 11(2): 174 – 188.

[7] KOMAKI G, KAYVANFAR V. Grey wolf optimizer algorithm for the two-stage assembly flow shop scheduling problem with release time [J]. Journal of Computational Science, 2015, 8(3): 109 – 120.

[8] ZHU A, XU C, LI Z, et al. Hybridizing grey wolf optimization with differential evolution for global optimization and test scheduling for 3D stacked SoC[J]. Journal of Systems Engineering and Electronics, 2015, 26(2): 317 – 328.

[9] 龙文,伍铁斌. 协调探索和开发能力的改进灰狼优化算法[J]. 控制与决策,2017,32(10):1749 – 1757.
LONG Wen, WU Tiebin. Improved grey wolf optimization algorithm coordinating the ability of exploration and exploitation[J]. Control and Decision, 2017, 32(10): 1749 – 1757. (in Chinese)

[10] 徐松金,龙文. 嵌入遗传算子的改进灰狼优化算法[J]. 兰州理工大学学报,2016,42(4):102 – 108.
XU Song-jin, LONG Wen. Improved grey wolf optimization embedded with genetic operators [J]. Journal of Lanzhou University of Technology, 2016, 42(4): 102 – 108. (in Chinese)

[11] MIRJALILI S. How effective is the grey wolf optimizer in training multilayer perceptrons [J]. Applied Intelligence, 2015, 42(2): 608 – 619.

[12] 周凌云,丁立新,彭虎,等. 一种邻域重心反向学习的粒子群优化算法[J]. 电子学报,2017,45(11):2815 – 2824.
ZHOU Ling-yun, DING Li-xin, PENG Hu, et al., Neighborhood centroid opposition-based particle swarm optimization[J]. Acta Electronica Sinica, 2017, 45(11): 2815 – 2824. (in Chinese)

[13] 江善和,王其申,江巨浪. 一种新型 Skew Tent 映射的混沌混合优化算法[J]. 控制理论与应用,2007,24(2):269 – 273.
JIANG Shan-he, WANG Qi-shen, JIANG Ju-lang. Chaotic hybrid optimization algorithm of a new Skew Tent map [J]. Control Theory & Applications, 2007, 24(2): 269 – 273. (in Chinese)

作者简介

龙　文　男,1977 年生于湖南隆回,教授,博士,博士生导师. 主要研究方向为智能优化和电力系统建模与控制.
E-mail: longwen227@ mail. gufe. edu. cn

蔡绍洪　男,1958 生于贵州仁怀,教授,博士,博士生导师. 主要研究方向为复杂非线性系统建模和非平衡系统控制.
E-mail: caish@ mail. gufe. edu. cn

焦建军　男,1973 年生于湖南邵阳,教授,博士,博士生导师. 主要研究方向为复杂系统建模与控制、计算智能.
E-mail: jiaojianjun@ mail. gufe. edu. cn

伍铁斌　男,1981 年生于湖南新化,副教授,博士. 主要研究方向为智能优化、工业生产过程建模与控制.
E-mail: wutiebin81@ csu. edu. cn