计算机科学
COMPUTER SCIENCE

基于IMQ惯性权重策略的自适应灰狼优化算法

于明洋, 李婷, 许静

引用本文

于明洋, 李婷, 许静. 基于IMQ惯性权重策略的自适应灰狼优化算法[J]. 计算机科学, 2024, 51(7): 354-361.

YU Mingyang, LI Ting, XU Jing. Adaptive Grey Wolf Optimizer Based on IMQ Inertia Weight Strategy[J]. Computer Science, 2024, 51(7): 354-361.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230600181

基于 IMQ 惯性权重策略的自适应灰狼优化算法

于明洋[1]　李　婷[2,3]　许　静[1]
1 南开大学人工智能学院　天津 300350
2 南开大学计算机学院　天津 300350
3 天津津航技术物理研究所　天津 300350
(2120220567@mail.nankai.edu.cn)

摘　要　针对灰狼优化算法(Grey Wolf Optimizer,GWO)寻优精度低、收敛速度慢的问题,提出了一种基于 IMQ 惯性权重策略的自适应灰狼优化算法(ISGWO)。该算法利用 IMQ 函数的特性,实现对惯性权重的非线性调整,从而更好地平衡算法的全局勘探能力和局部开发能力;同时,基于 Sigmoid 指数函数自适应更新个体位置,更好地搜索和优化问题的解空间。采用 6 个基本函数和 29 个 CEC2017 函数对 ISGWO 进行测试,并与 6 种常用的算法进行比较,实验结果表明 ISGWO 具有更优的收敛精度和速度。

关键词:IMQ 函数;惯性权重;自适应;灰狼优化算法;收敛速度;寻优精度
中图分类号　TP301

Adaptive Grey Wolf Optimizer Based on IMQ Inertia Weight Strategy

YU Mingyang[1],LI Ting[2,3] and XU Jing[1]
1 College of Artificial Intelligence,Nankai University,Tianjin 300350,China
2 College of Computer Science,Nankai University,Tianjin,300350,China
3 Tianjin Jinhang Institute of Technical Physics,Tianjin,300350,China

Abstract　Aiming at the problems of low optimization accuracy and slow convergence speed of grey wolf optimizer(GWO),this paper proposes an adaptive grey wolf optimization algorithm(ISGWO) based on IMQ inertia weighting strategy. This algorithm utilizes the properties of the IMQ function to achieve a nonlinear adjustment of the inertia weights,which better balances the global exploration ability and local exploitation ability of the algorithm. At the same time,it adaptively updates the position of individuals based on the Sigmoid exponential function to better search and optimize the solution space of the problem. Six basic functions and 29 CEC2017 functions are used to test ISGWO and compare it with six commonly used algorithms,and the experimental results show that ISGWO has superior convergence accuracy and speed.

Keywords　IMQ function,Inertia weight,Adaptive,Grey wolf optimizer,Convergence speed,Optimization accuracy

1　引言

近年来,随着启发式智能优化算法在数值优化求解中的广泛应用,各种群体智能算法相继被提出,如粒子群优化算法(Particle Swarm Optimization,PSO)[1]、萤火虫算法(Firefly Algorithm,FA)[2]、正余弦优化算法(Sine Cosine Algorithm,SCA)[3]、风驱动优化算法(Wind Driven Optimization,WDO)[4]、果蝇优化算法(Fruit Fly Optimization Algorithm,FOA)[5]等。

灰狼优化算法(Grey Wolf Optimizer,GWO)是受灰狼群体捕猎行为启发而提出的[6]。GWO 自 2014 年问世以来,因简单、高效而受到国内外学者的广泛关注,成为解决复杂优化问题的重要工具[7]。然而,与其他优化算法类似,GWO 算法也存在容易出现早熟现象和易陷入局部最优等问题,尤其是在求解多模态函数问题时。分析认为,随着迭代过程的推进,受社会等级机制的影响,狼群多样性会下降,种群盲目聚集将导致搜索过早收敛、当前最优解无法跳出局部最优[8]。此外,当全局勘探向局部挖掘过渡时,算法可能会失去探索更广泛解空间的能力,而过于集中在对特定区域进行详细搜索。

过去几年里,许多学者尝试对 GWO 进行改进以增强算法的搜索性能,包括调整算法参数、改进速度和位置公式,以及将其与其他算法相结合等方面。Zhang 等[9]融合灰狼优化算法和算术优化算法以平衡算法的全局探索和局部开发能力。Xi 等[10]将 GWO 与多隐层极限学习机相结合,用于虚假数据注入攻击检测。Zhang 等[11]将自适应控制参数调整策略、自适应惯性权重策略、最优学习策略与跳出局部最优策略

到稿日期:2023-06-24　返修日期:2023-11-06
基金项目:天津市自然科学基金(21JCYBJC00110)
This work was supported by the Natural Science Foundation of Tianjin,China(21JCYBJC00110).
通信作者:李婷(t24725@126.com)

改进 GWO 算法融合，在增强种群多样性的同时，提高算法的搜索能力。Kang 等[12]混合改进的花授粉算法与灰狼算法，用于特征选择。Liu 等[13]设计了一种采用 GWO 改进的自适应噪声的完备集成经验模态分解的混合储能系统功率分配策略。Liu 等[14]提出了一种差分进化与 GWO 混合算法，提高了优化精度和收敛速度。Wang 等[15]提出了一种滤波增强版的 GWO，并融合了一种高效的自适应重网格方法以加快每次 EM 模拟调用的数值分析过程。Lu 等[16]根据探路者中跟随者的更新机制改变 GWO 中灰狼个体的位置，进而平衡算法的全局搜索和局部搜索能力。Liu 等[17]提出一种基于 Tent 混沌映射初始化种群的改进 GWO，并将其用于解决移动机器人全局路径规划问题。Xie 等[18]从初始种群、收敛因子等方面着手，改善 GWO 的局部搜索能力及收敛速度。Zhang 等[19]通过自适应调整改进 GWO，提高了算法的局部搜索与全局搜索能力。

为了进一步提升 GWO 性能，本文提出了一种基于 IMQ 惯性权重策略的自适应灰狼优化算法。采用 IMQ 函数来调整惯性权重，并基于 Sigmoid 函数的自适应机制更新个体位置，以增强种群逃离局部最优解的能力。采用基本测试函数和 CEC2017 测试函数对所提算法进行测试，以验证其寻优性能。通过参数对比实验、消融实验、可扩展性实验以及算法对比分析了上述算法的收敛速度和精度。

2 灰狼优化算法

GWO 模拟了灰狼的领导和狩猎机制，根据灰狼的社会等级将它们分为 α，β，δ 和 ω 狼。其中，α 狼领导整个狼群进行捕食；β 狼协助 α 狼作出决策，并在 α 狼死后成为 α 狼的最候选者；δ 狼听从 α 狼和 β 狼的命令，在狼群中起侦察作用；ω 狼是除上述 3 种等级外的狼群，数量多，主要负责搜索猎物[20]。GWO 的实现流程如下：

(1)初始化灰狼种群，设定种群规模 N、最大迭代次数 M、空间搜索维度 D 以及迭代系数 a 等相关的开发参数。

(2)计算初始种群的适应度值并进行排序，将适应度值最优的前 3 只个体依次命名为 α，β，δ 狼，剩余狼群均为 ω 狼。

(3)根据式(1)、式(2)更新当前灰狼位置。

$$\begin{cases} \boldsymbol{X}_1=\boldsymbol{X}_\alpha-\boldsymbol{A}_1\boldsymbol{D}_\alpha, & \boldsymbol{D}_\alpha=|\boldsymbol{C}_1\boldsymbol{X}_\alpha-\boldsymbol{X}| \\ \boldsymbol{X}_2=\boldsymbol{X}_\beta-\boldsymbol{A}_2\boldsymbol{D}_\beta, & \boldsymbol{D}_\beta=|\boldsymbol{C}_2\boldsymbol{X}_\beta-\boldsymbol{X}| \\ \boldsymbol{X}_3=\boldsymbol{X}_\delta-\boldsymbol{A}_3\boldsymbol{D}_\delta, & \boldsymbol{D}_\delta=|\boldsymbol{C}_3\boldsymbol{X}_\delta-\boldsymbol{X}| \end{cases} \tag{1}$$

$$\boldsymbol{X}(t+1)=[\boldsymbol{X}_1(t)+\boldsymbol{X}_2(t)+\boldsymbol{X}_3(t)]/3 \tag{2}$$

其中，$\boldsymbol{X}_1$，$\boldsymbol{X}_2$ 和 $\boldsymbol{X}_3$ 为 ω 狼群朝 α，β，δ 狼方向移动的步长；$\boldsymbol{X}(t)$ 和 $\boldsymbol{X}(t+1)$ 为第 t 次更新前和更新后的狼群位置；$\boldsymbol{A}$ 为包围步长，$\boldsymbol{C}$ 为移动方向，$\boldsymbol{C}=2\boldsymbol{r}_1$，$\boldsymbol{A}=2a\boldsymbol{r}_2-a$，$\boldsymbol{r}_1$ 和 $\boldsymbol{r}_2$ 均为(0，1)区间的随机向量。迭代系数的计算式为：

$$a=2-2t/M \tag{3}$$

(4)检查更新后个体位置是否越界，若越界，则对该个体位置重新随机初始化。计算狼群的适应度值，并更新 α 狼、β 狼、δ 狼和全局最优解。

(5)判断是否达到指定的停止条件(即是否满足 $t\geqslant M$)，若未达到，则重复步骤 2－步骤 5；否则输出最优结果。最后得到的 α 狼的位置即为最优解，对应的适应度值为最优解的优劣程度。

3 改进的灰狼优化算法

3.1 IMQ 惯性权重策略

IMQ 函数是一种基于逆多次平方原理的递减函数，在神经网络中常被用作正则化方法，比如在支持向量机中充当核函数[21]。根据 IMQ 函数的特性，本文将其应用于 GWO 式(1)中进行种群位置更新。IMQ 惯性权重 w 及修改后狼群的更新公式如式(4)、式(5)所示：

$$w=a\mathrm{e}^{-b\mathrm{e}^{-d}}+d \tag{4}$$

$$\begin{cases} \boldsymbol{X}_1=w\boldsymbol{X}_\alpha-\boldsymbol{A}_1\boldsymbol{D}_\alpha \\ \boldsymbol{X}_2=w\boldsymbol{X}_\beta-\boldsymbol{A}_2\boldsymbol{D}_\beta \\ \boldsymbol{X}_3=w\boldsymbol{X}_\delta-\boldsymbol{A}_3\boldsymbol{D}_\delta \end{cases} \tag{5}$$

其中，参数组$[a,b,c,d]$分别取$[0.6,0.02,0.05,0.3]$。惯性权重 w 随迭代次数的变化曲线如图 1 所示。由图 1 可知，当算法处于迭代前中期时，此时惯性权重 w 取值较大，待更新位置受 α 狼、β 狼、δ 狼支配影响较大，有利于种群快速向最优解靠近，有效避免盲目搜索导致的搜索资源浪费，从而提高了种群质量；当开发进行至中后期时种群高度密集，若高等级狼陷入局部最优，受其牵引的低阶层狼群同样无法跳出局部最优。此时应将 w 值降低到较低水平，从而扩大种群的自主搜索能力，避免陷入早熟。

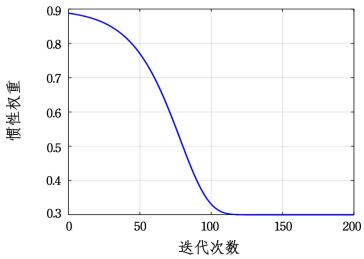


图 1 IMQ 惯性权重迭代曲线

Fig. 1 Iterative curves of IMQ inertia weights

3.2 自适应更新机制

基于 IMQ 惯性权重的种群更新机制一定程度上降低了种群聚集密度，然而，受 GWO 自身机制的影响，迭代过程中的新生狼群仍不可避免地集中向 α，β，δ 狼指引的位置迁移。对此，本文将某个体的适应度值与平均种群适应度值定义为聚集系数，用于描述当前解与最优解的疏远程度。在极小化问题中，适应度值越小越优。当聚集系数较小时，表明当前解较优，此时允许其在自身附近进行小幅度更新。相反，若个体聚集系数较大，表明其所处位置较差，可对其施加一个大扰动使其跃迁至其他位置[22]。基于上述分析，本文引入 Sigmoid 函数构造不同聚集系数下种群的自适应更新幅度，如式(6)、式(7)所示：

$$\phi=[1+(\mathrm{e}^{-f_i/f_{\mathrm{ave}}})^{\theta}]^{-1} \tag{6}$$

$$\boldsymbol{X}_i(t+1)=\boldsymbol{X}_i(t)\cdot\phi \tag{7}$$

其中，f_i 和 f_{ave} 分别表示第 i 个个体的适应度值和种群平均适应度；θ 为指数系数。

ISGWO 的算法实施流程如图 2 所示。与标准的 GWO 相比，ISGWO 引入 IMQ 惯性权重平衡了算法的全局勘探和局部搜索，表现为增强了种群的多样性、加快了算法的收敛速度。此外，结合聚集系数以及 Sigmoid 函数自适应更新机制

的引入进一步避免了种群盲目聚集。不同个体根据自身与最优解疏远程度采取不同幅度的更新，既提升了种群质量，也增强了算法在迭代后期抵抗局部最优的能力。

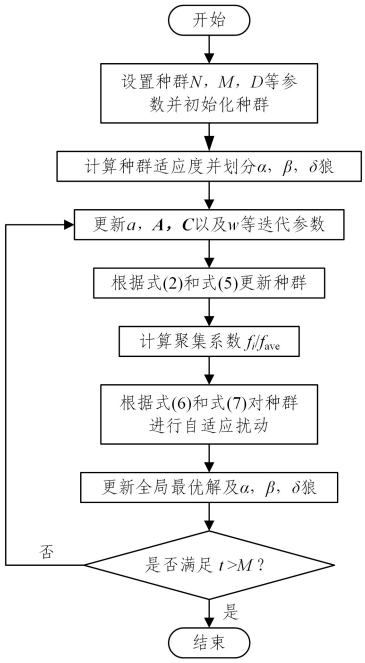


图 2 ISGWO 实施流程

Fig. 2 Implementation process of ISGWO

3.3 时间复杂度分析

时间复杂度是评判算法计算效率的重要指标，对于启发式智能算法，其时间复杂度主要由种群规模、搜索维度和最大迭代次数决定。GWO 算法的时间复杂度可记为 $O(N \cdot D \cdot M)$[23]。本文提出的 ISGWO 在 GWO 基础上引入了式(5)—式(7)进行联合改进，具体包括添加 IMQ 惯性权重以抑制高等级狼对低等级狼的过度引导，补充基于 Sigmoid 函数的种群自适应更新，增强算法的全局开发性能。

与式(1)相比，式(5)仅加入了权重系数，而计算权重系数的时间复杂度可忽略不计。式(6)更新不受种群规模、搜索维度限制，仅与最大迭代次数有关，其时间复杂度为 $O(M)$；式(7)的时间复杂度可记为 $O(N \cdot M)$。综上，ISGWO 的时间复杂度为：

$$O(ISGWO)=O(N \cdot D \cdot M)+O(M)+O(N \cdot M) \approx O(N \cdot D \cdot M) \tag{8}$$

与标准 GWO 相比，ISGWO 的时间复杂度没有明显增加，初步验证了其搜索时效性。

4 仿真实验和结果分析

本文仿真环境为 Windows10，64 位操作系统，CPU 为 AMD Ryzen 7 4800H，主频 2.30GHz，内存 16GB，算法基于 Matlab2018b 平台实现。

4.1 测试函数以及参数设置

为验证本文提出的 ISGWO 的有效性，选取了 6 个基本测试函数和 29 个 CEC2017 测试函数对其进行测试。上述函数的基本信息如表 1、表 2 所列。其中，6 个基本测试函数包括 3 个单峰函数(f_1-f_3)和 3 个多峰函数(f_4-f_6)，其最优值均为 0。29 个 CEC2017 测试函数是由多种基本测试函数混合构成，其中 f_1 和 f_3 为单峰函数，f_4-f_{10} 为多峰函数，$f_{11}-f_{20}$ 为混合函数，$f_{21}-f_{30}$ 为复合函数，f_2 函数由于高维的不稳定性已被剔除。CEC2017 测试函数集的搜索范围均为 $[-100,100]^D$。

实验中，设置算法的种群数量 N 为 50，最大迭代次数 M 为 200，分别独立运行 30 次并记录每组实验的最优适应度值。统计 30 次重复实验的最优值(Best)、最差值(Worst)、平均值(Mean)和标准差 Std。

表 1 6 个基本测试函数

Table 1 Six basic test functions

函数	函数名称	范围	函数	函数名称	范围
f_1	Sphere Function	[−100,100]	f_4	Schwefel's Problem 2.21	[−100, 100]
f_2	Schwefel's Problem 2.22	[−10,10]	f_5	Generalized Rastrigin's Function	[−5.12,5.12]
f_3	Schwefel's Problem 1.2	[−100,100]	f_6	Generalized Griewank's Function	[−600,600]

表 2 29 个 CEC2017 测试函数

Table 2 Twenty-nine CEC2017 test functions

函数	函数名称	最值	函数	函数名称	最值
f_1	Shifted and Rotated Bent Cigar Function	100	f_{17}	Hybrid Function 6(N=4)	1700
f_3	Shifted and Rotated Zakharov Function	300	f_{18}	Hybrid Function 6(N=5)	1800
f_4	Shifted and Rotated Rosenbrock's Function	400	f_{19}	Hybrid Function 6(N=5)	1900
f_5	Shifted and Rotated Rastrigin's Function	500	f_{20}	Hybrid Function 6(N=6)	2000
f_6	Shifted and Rotated Expanded Scaffer's F6 Function	600	f_{21}	Composition Function 1(N=3)	2100
f_7	Shifted and Rotated Lunacek Bi−Rastrigin Function	700	f_{22}	Composition Function 2(N=3)	2200
f_8	Shifted and Rotated Non−Continuous Rastrigin's Function	800	f_{23}	Composition Function 3(N=4)	2300
f_9	Shifted and Rotated Lévy Function	900	f_{24}	Composition Function 4(N=4)	2400
f_{10}	Shifted and Rotated Schwefel's Function	1000	f_{25}	Composition Function 5(N=5)	2500
f_{11}	Hybrid Function 1(N=3)	1100	f_{26}	Composition Function 6(N=5)	2600
f_{12}	Hybrid Function 2(N=3)	1200	f_{27}	Composition Function 7(N=6)	2700
f_{13}	Hybrid Function 3(N=3)	1300	f_{28}	Composition Function 8(N=6)	2800
f_{14}	Hybrid Function 4(N=4)	1400	f_{29}	Composition Function 9(N=3)	2900
f_{15}	Hybrid Function 5(N=4)	1500	f_{30}	Composition Function 10(N=3)	3000
f_{16}	Hybrid Function 5(N=4)	1600	—	—	—

4.2 参数 θ 的选取

在本文提出的 ISGWO 中，自适应参数 θ 决定了 Sigmoid 函数的幅值，在不同的 θ 取值下，ISGWO 将表现出不同的搜索性。为了确定参数 θ 的最佳取值，本文采用表 1 中

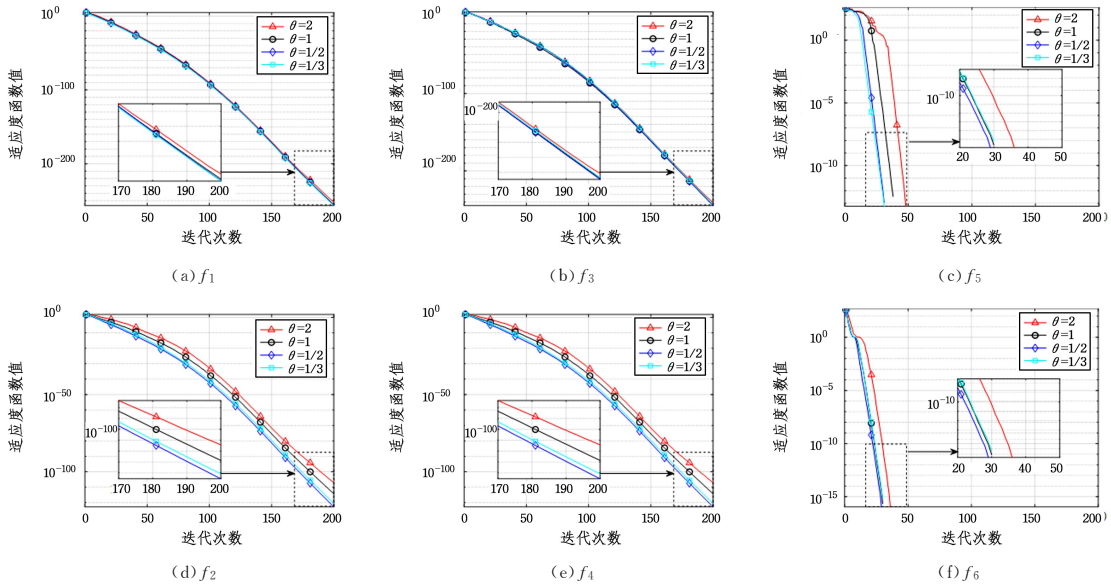
6 个基本测试函数对不同 $\theta \in [2,1,1/2,1/3]$ 取值下 ISGWO 的性能进行测试，测试结果如表 3 所列，重复实验的最优值用粗体表示，测试维度 $D=30$。从寻优精度来看，$\theta=1/2$ 时的统计最优值略优于其他取值的情况，但总体相差不大。

进一步地，图 3 给出了 θ 不同取值时，ISGWO 对基本测试函数 f_1-f_6 的测试情况。对于多峰函数 f_5 和 f_6，上述 4 种取值条件下 ISGWO 均能搜索到全局最优值 0，而 $\theta=1/2$ 时 ISGWO 的收敛速度更快，表明其应对复杂函数时具备更高的搜索效率。

表 3　参数 θ 取不同值时 ISGWO 的寻优结果

Table 3　Results of ISGWO with different values of θ

函数	$\theta=2$				$\theta=1$			
	Best	Worst	Mean	Std	Best	Worst	Mean	Std
f_1	3.3×10^{-258}	3.1×10^{-255}	2.8×10^{-256}	0	3.8×10^{-261}	1.2×10^{-257}	6.9×10^{-259}	0
f_2	1.1×10^{-116}	3.4×10^{-111}	3.1×10^{-112}	7.5×10^{-112}	1.4×10^{-127}	7.6×10^{-120}	2.9×10^{-121}	1.38×10^{-120}
f_3	1.6×10^{-246}	3.0×10^{-241}	1.8×10^{-242}	0	1.6×10^{-246}	8.6×10^{-244}	1.4×10^{-244}	0
f_4	6.8×10^{-118}	5.5×10^{-110}	3.0×10^{-111}	10.0×10^{-111}	1.8×10^{-122}	1.9×10^{-118}	1.7×10^{-119}	3.58×10^{-119}
f_5	0	0	0	0	0	0	0	0
f_6	0	0	0	0	0	0	0	0
函数	$\theta=1/2$				$\theta=1/3$			
	Best	Worst	Mean	Std	Best	Worst	Mean	Std
f_1	$\mathbf{4.4\times10^{-262}}$	5.2×10^{-259}	3.3×10^{-260}	0	7.3×10^{-262}	3.5×10^{-259}	4.0×10^{-260}	0
f_2	$\mathbf{2.1\times10^{-129}}$	9.5×10^{-127}	1.5×10^{-127}	2.2×10^{-127}	4.3×10^{-128}	7.8×10^{-124}	1.8×10^{-125}	1.6×10^{-125}
f_3	$\mathbf{1.8\times10^{-247}}$	6.2×10^{-244}	2.7×10^{-245}	0	4.9×10^{-247}	9.3×10^{-245}	3.4×10^{-245}	0
f_4	2.7×10^{-126}	2.4×10^{-122}	2.9×10^{-123}	5.3×10^{-123}	$\mathbf{1.4\times10^{-126}}$	1.5×10^{-122}	3.7×10^{-123}	4.5×10^{-123}
f_5	0	0	0	0	0	0	0	0
f_6	0	0	0	0	0	0	0	0



(a) f_1　(b) f_3　(c) f_5

(d) f_2　(e) f_4　(f) f_6

图 3　参数 θ 不同取值下的 ISGWO 迭代曲线

Fig. 3　Iteration curve of ISGWO with different values of parameter θ

4.3　消融实验

为进一步阐明 IMQ 惯性权重策略和自适应位置更新策略对 GWO 的影响，本文对此开展了消融对比实验。将分别采用惯性权重改进策略、自适应位置更新机制，以及同时采用上述两种策略的算法依次命名为 IMQGWO，SGWO，ISGWO，利用 6 个经典函数对其进行测试。运行次数、种群规模、测试维度以及迭代最大次数均与 4.2 节保持一致，比较不完全改进算法的 Best，Worst，Mean 和 Std，结果如表 4 所列。

表 4　消融实验结果

Table 4　Results of ablation experiments

函数	IMQGWO				SGWO			
	Best	Worst	Mean	Std	Best	Worst	Mean	Std
f_1	1.5×10^{-143}	7.9×10^{-142}	2.1×10^{-142}	2.2×10^{-142}	3.0×10^{-116}	5.0×10^{-112}	3.9×10^{-113}	9.9×10^{-113}
f_2	2.3×10^{-73}	2.7×10^{-72}	7.7×10^{-73}	5.1×10^{-73}	3.2×10^{-54}	5.2×10^{-53}	1.4×10^{-53}	1.1×10^{-53}
f_3	3.1×10^{-129}	1.8×10^{-126}	3.6×10^{-127}	5.2×10^{-127}	1.6×10^{-111}	4.1×10^{-109}	6.8×10^{-110}	8.6×10^{-110}
f_4	5.9×10^{-70}	1.1×10^{-68}	4.2×10^{-69}	3.1×10^{-69}	2.1×10^{-51}	3.5×10^{-50}	1.3×10^{-50}	9.0×10^{-5}
f_5	0	0	0	0	0	32.468	3.599	7.204
f_6	0	0	0	0	0	0.195	0.001	0.004

（续表）

函数	ISGWO			
	Best	Worst	Mean	Std
f_1	**4.4×10^{-262}**	**5.2×10^{-259}**	**3.3×10^{-260}**	0
f_2	**2.1×10^{-129}**	**9.5×10^{-127}**	**1.5×10^{-127}**	**2.2×10^{-127}**
f_3	**1.8×10^{-247}**	**6.2×10^{-244}**	**2.7×10^{-245}**	0
f_4	**2.7×10^{-126}**	**2.4×10^{-122}**	**2.9×10^{-123}**	**5.3×10^{-123}**
f_5	0	0	0	0
f_6	0	0	0	0

由表 4 可知，在 IMQGWO，SGWO 和 ISGWO 的测试结果中，ISGWO 表现最优，IMQGWO 次之。SGWO 在多峰上的表现略逊色于 IMQGWO 和 ISGWO，这表明 IMQ 惯性权重策略更有利于提升 GWO 处理多峰函数的能力。对于上述测试函数，仅由 IMQ 惯性权重或自适应更新策略改进后的 GWO 搜索精度有限，而联合策略改动的 IMQGWO 的搜索结果能较充分地接近理论最优解。

4.4 可扩展性分析

为测试 ISGWO 在高维函数上的处理能力，将测试函数的搜索维度扩展至 100 维和 300 维，并与 PSO，FA，SCA，WDO，FOA，GWO 这 6 个典型优化算法的实验结果进行对比。

这些对比算法的参数设置与原文献保持一致，测试结果如表 5 所列。

表 5 可拓展性分析结果

Table 5 Results of scalability analysis

函数	算法	$D=30$				$D=100$				$D=100$			
		Best	Worst	Mean	Std	Best	Worst	Mean	Std	Best	Worst	Mean	Std
f_1	PSO	7.4×10^{2}	2.0×10^{3}	1.7×10^{3}	2.5×10^{2}	1.4×10^{4}	2.1×10^{4}	1.7×10^{4}	1.9×10^{3}	1.0×10^{5}	1.5×10^{5}	1.2×10^{5}	1.2×10^{4}
	FA	3.2	8.9	5.4	1.3	2.4×10^{2}	4.5×10^{2}	3.4×10^{2}	5.6×10	3.2×10^{4}	7.3×10^{4}	5.0×10^{4}	9.7×10^{3}
	FOA	1.6×10^{4}	2.5×10^{4}	2.1×10^{4}	2.4×10^{3}	1.1×10^{5}	1.5×10^{5}	1.3×10^{5}	9.8×10^{3}	4.9×10^{5}	5.8×10^{5}	5.4×10^{5}	2.2×10^{4}
	GWO	6.5×10^{-12}	1.6×10^{-10}	4.3×10^{-11}	3.5×10^{-11}	1.0×10^{-3}	4.0×10^{-3}	2.0×10^{-3}	1.0×10^{-3}	5.5	1.7×10	1.0×10	2.5
	WDO	1.8×10^{-7}	6.5×10^{-4}	3.9×10^{-5}	1.2×10^{-4}	3.0×10^{-6}	1.0×10^{-3}	3.4×10^{-4}	5.1×10^{-4}	9.1×10^{-8}	2.1×10^{-2}	1.0×10^{-3}	4.0×10^{-3}
	SCA	1.1×10	1.9×10^{3}	5.7×10^{2}	5.5×10^{2}	5.7×10^{3}	8.4×10^{4}	2.6×10^{4}	1.7×10^{4}	6.8×10^{4}	2.5×10^{5}	1.6×10^{5}	5.0×10^{4}
	ISGWO	**4.4×10^{-262}**	**5.2×10^{-259}**	**3.3×10^{-260}**	**0**	**1.5×10^{-249}**	**1.9×10^{-245}**	**1.1×10^{-246}**	**0**	**3.3×10^{-243}**	**3.7×10^{-239}**	**4.7×10^{-240}**	**0**
f_2	PSO	3.7×10^{7}	1.1×10^{12}	8.1×10^{10}	2.1×10^{11}	5.8×10^{41}	1.0×10^{51}	3.5×10^{49}	1.8×10^{50}	8.7×10^{143}	5.3×10^{156}	2.7×10^{155}	Inf
	FA	6.6	1.1×10	8.5	9.2×10	8.1×10	2.1×10^{2}	1.2×10^{2}	2.3×10	5.9×10^{36}	5.0×10^{77}	1.7×10^{76}	9.1×10^{76}
	FOA	5.6×10	4.0×10^{5}	2.8×10^{4}	7.6×10^{4}	4.9×10^{20}	6.0×10^{31}	3.2×10^{30}	1.1×10^{31}	4.9×10^{93}	2.8×10^{114}	9.5×10^{112}	5.1×10^{113}
	GWO	1.1×10^{-7}	7.0×10^{-7}	3.6×10^{-7}	1.5×10^{-7}	6.0×10^{-3}	1.3×10^{-2}	1.0×10^{-2}	2.0×10^{-3}	1.5	2.2	1.8	2.2×10^{-1}
	WDO	3.4×10^{-4}	6.0×10^{-3}	2.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}	3.9×10^{-2}	1.5×10^{-2}	1.0×10^{-2}	5.0×10^{-3}	1.9×10^{-1}	4.6×10^{-2}	4.5×10^{-2}
	SCA	2.0×10^{-2}	3.1	9.7×10^{-1}	9.4×10^{-1}	4.1	7.2×10	2.0×10	1.3×10	2.7×10	1.7×10^{2}	8.8×10	4.0×10
	ISGWO	**2.1×10^{-129}**	**9.5×10^{-127}**	**1.5×10^{-127}**	**2.2×10^{-127}**	**1.4×10^{-119}**	**1.8×10^{-117}**	**4.5×10^{-118}**	**4.5×10^{-118}**	**3.0×10^{-115}**	**3.7×10^{-114}**	**1.2×10^{-114}**	**8.3×10^{-115}**
f_3	PSO	3.0×10^{3}	7.0×10^{3}	4.9×10^{3}	1.1×10^{3}	5.9×10^{4}	1.2×10^{5}	9.1×10^{4}	1.8×10^{4}	5.5×10^{5}	1.3×10^{6}	8.5×10^{5}	1.5×10^{5}
	FA	9.4	5.1×10	2.6×10	1.0×10	3.7×10^{4}	3.0×10^{5}	7.0×10^{4}	4.6×10^{4}	5.3×10^{5}	8.9×10^{5}	7.1×10^{5}	7.5×10^{4}
	FOA	3.1×10^{4}	5.9×10^{4}	4.5×10^{4}	6.6×10^{3}	3.1×10^{5}	6.8×10^{5}	5.1×10^{5}	9.8×10^{4}	4.4×10^{6}	1.4×10^{7}	7.0×10^{6}	2.3×10^{6}
	GWO	2.6×10^{2}	1.2	3.2×10^{-1}	3.0×10^{-1}	2.1×10^{3}	1.2×10^{4}	6.2×10^{3}	2.7×10^{3}	1.2×10^{5}	2.8×10^{5}	2.0×10^{5}	4.7×10^{4}
	WDO	3.0×10^{-6}	7.0×10^{-3}	1.0×10^{-3}	1.0×10^{-3}	2.5×10^{-4}	3.2×10^{-1}	6.5×10^{-2}	7.7×10^{-2}	1.5×10^{-2}	1.0×10	1.1	1.9
	SCA	3.8×10^{3}	3.6×10^{4}	1.7×10^{4}	6.9×10^{3}	1.9×10^{5}	4.1×10^{5}	2.9×10^{5}	5.2×10^{4}	2.4×10^{6}	4.2×10^{6}	3.1×10^{6}	4.9×10^{5}
	ISGWO	**1.8×10^{-247}**	**6.2×10^{-244}**	**2.7×10^{-245}**	**0**	**8.1×10^{-240}**	**2.2×10^{-236}**	**1.0×10^{-237}**	**0**	**1.4×10^{-235}**	**7.4×10^{-233}**	**5.3×10^{-234}**	**0**
f_4	PSO	7.4	9.1	8.6	3.8×10^{-1}	8.9	9.7	9.5	1.4×10^{-1}	9.7	9.9	9.8	3.6×10^{-2}
	FA	1.8	4.4	3.1	7.0×10^{-1}	8.6	9.5	9.3	2.1×10^{-1}	9.4	9.8	9.7	7.9×10^{-2}
	FOA	4.9	5.5	5.3	1.7×10^{-1}	7.3	7.9	7.6	1.5×10^{-1}	8.4	8.8	8.7	8.8×10^{-2}
	GWO	2.4×10^{-4}	2.0×10^{-3}	7.4×10^{-4}	3.8×10^{-4}	2.3×10^{-1}	2.1	8.3×10^{-1}	4.1×10^{-1}	3.8	7.6	5.5	8.4×10^{-1}
	WDO	6.8×10^{-6}	1.0×10^{-3}	3.4×10^{-4}	3.1×10^{-4}	3.0×10^{-5}	4.0×10^{-3}	7.9×10^{-4}	8.3×10^{-4}	1.5×10^{-4}	3.0×10^{-3}	8.4×10^{-4}	6.2×10^{-4}
	SCA	1.8	7.0	4.8	1.2	8.8	9.5	9.2	1.9×10^{-1}	9.7	9.9	9.2	4.3×10^{-2}
	ISGWO	**2.7×10^{-126}**	**2.4×10^{-122}**	**2.9×10^{-123}**	**5.3×10^{-123}**	**4.1×10^{-118}**	**4.1×10^{-112}**	**1.5×10^{-113}**	**7.6×10^{-113}**	**1.2×10^{-104}**	**1.8×10^{-97}**	**2.7×10^{-98}**	**4.5×10^{-98}**
f_5	PSO	3.6×10^{2}	4.5×10^{2}	4.2×10^{2}	2.4×10	1.5×10^{3}	1.7×10^{3}	1.61×10^{3}	3.9×10	4.9×10^{3}	5.2×10^{3}	5.1×10^{3}	6.9×10
	FA	1.5×10^{2}	2.2×10^{2}	1.9×10^{2}	1.8×10	9.6×10^{2}	1.1×10^{3}	1.0×10^{3}	5.8×10	3.8×10^{3}	4.1×10^{3}	4.0×10^{3}	7.5×10
	FOA	2.5×10^{2}	3.4×10^{2}	2.9×10^{2}	2.1×10	1.2×10^{3}	1.3×10^{3}	1.2×10^{3}	3.4×10	4.0×10^{3}	4.3×10^{3}	4.2×10^{3}	6.1×10
	GWO	2.6×10^{-6}	2.5×10	1.1×10	6.2	2.8×10	1.1×10^{2}	5.5×10	1.8×10	2.1×10^{2}	4.8×10^{2}	3.3×10^{2}	7.3×10
	WDO	5.5×10	1.4×10^{2}	9.7×10	2.8×10	4.9×10^{2}	6.5×10^{2}	5.7×10^{2}	4.1×10	1.6×10^{-5}	2.4×10^{3}	1.5×10^{3}	1.0×10^{3}
	SCA	4.7	1.7×10^{2}	7.1×10	3.9×10	1.1×10^{2}	7.3×10^{2}	3.6×10^{2}	1.4×10^{2}	3.6×10^{2}	1.5×10^{3}	9.7×10^{2}	3.3×10^{2}
	ISGWO	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_6	PSO	1.7	2.2	1.9	1.4×10^{-1}	9.6	1.3×10	1.1×10	8.6×10^{-1}	8.6×10	1.1×10^{2}	9.9×10	7.7
	FA	2.0×10^{-1}	1.0×10^{-1}	2.9×10^{-1}	5.1×10^{2}	1.0	1.1	1.1	2.4×10^{-2}	2.5×10^{2}	6.3×10^{2}	3.9×10^{2}	9.4×10
	FOA	1.3×10^{2}	2.3×10^{2}	1.9×10^{2}	2.2×10	1.0×10^{3}	1.4×10^{3}	1.2×10^{3}	8.7×10	4.4×10^{3}	5.3×10^{3}	4.8×10^{3}	1.9×10^{2}
	GWO	1.2×10^{-11}	3.0×10^{-2}	4.0×10^{-3}	8.0×10^{-3}	4.9×10^{-4}	1.0×10^{-1}	2.6×10^{-2}	3.9×10^{-2}	8.1×10^{-2}	1.1	9.9×10^{-1}	9.3×10^{-2}
	WDO	8.6×10^{-7}	3.8×10^{-1}	3.4×10^{-2}	9.2×10^{-2}	4.3×10^{-6}	1.0×10^{-3}	3.1×10^{-4}	3.2×10^{-4}	4.1×10^{-7}	5.0×10^{-3}	7.4×10^{-4}	1.0×10^{-3}
	SCA	1.0	9.4	3.6	2.1	6.7×10	4.1×10^{2}	2.1×10^{2}	9.5	5.6×10^{2}	2.6×10^{3}	1.5×10^{3}	5.6×10^{2}
	ISGWO	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**

表 5 显示，不论 $D=30$，100 或 300，在多峰函数 f_5，f_6 测试下 ISGWO 均能收敛到最优值，这表明 ISGWO 摆脱局部最优的能力较强。在单峰函数 f_1-f_4 中，尽管 ISGWO 未达到最优值，但收敛精度和速度均优于其他 6 种算法，表明 ISGWO 更适合处理单极值问题。此外，统计平均值 Mean 和标准差 Std 可知，ISGWO 获取目标精度的稳定性更强。

图 4 出了 7 种算法在测试函数的搜索维度 $D=300$ 时的收敛曲线。综上所述，ISGWO 算法在收敛速度、求解精度和稳定性方面表现更出色。

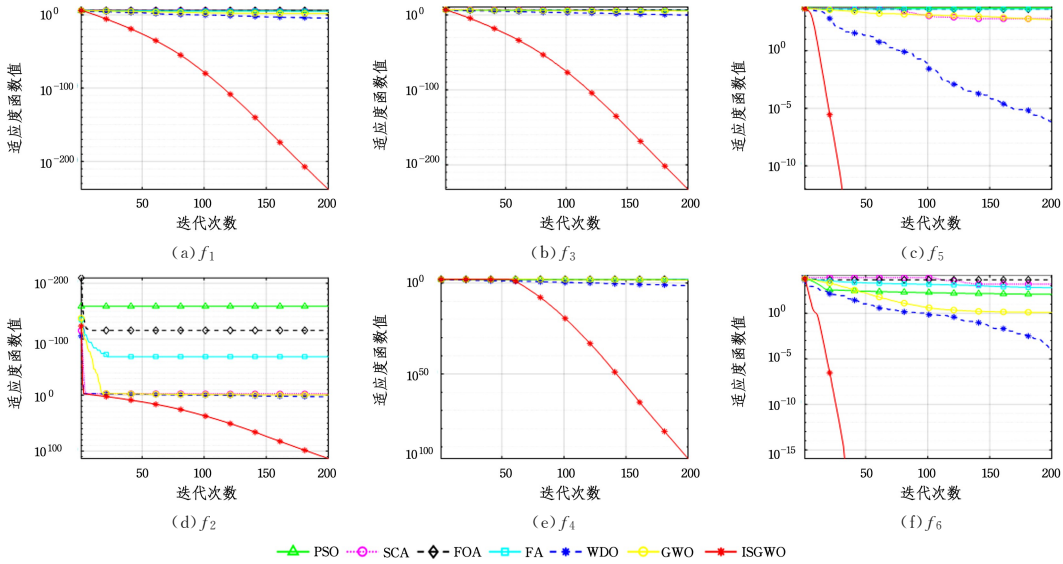


图 4 收敛曲线($D=300$)

Fig. 4 Convergence curve($D=300$)

4.5 CEC 系列函数测试

CEC 系列函数可对实际问题的复杂性进行模拟，这对于新算法的研究具有指导意义。对此，本文采用表 2 中的 29 个 CEC2017 测试函数对 4.4 节中 7 种优化算法进行测试，搜索维度 $D=100$。种群规模、迭代次数、统计次数等基本参数设置与前文保持一致。

经反复实验，计算重复实验的最优值(Best)和平均值(Mean)，具体如表 6 所列。

表 6 CEC2017 测试结果

Table 6 Test results of CEC2017

函数	PSO Mean(Std)	SCA Mean(Std)	FA Mean(Std)	FOA Mean(Std)
f_1	$6.3\times10+11(4.3\times10^{10})$	$2.3\times10^{11}(9.8\times10^{9})$	$5.1\times10^{9}(4.3\times10^{9})$	$1.6\times10^{12}(9.3\times10^{10})$
f_3	$4.0\times10^{11}(6.8\times10^{11})$	$4.2\times10^{5}(3.2\times10^{4})$	$5.7\times10^{5}(5.1\times10^{4})$	$3.8\times10^{17}(5.7\times10^{17})$
f_4	$2.9\times10^{5}(4.6\times10^{4})$	$6.4\times10^{4}(6.2\times10^{3})$	$1.2\times10^{3}(1.4\times10^{2})$	$1.8\times10^{6}(2.1\times10^{5})$
f_5	$3.1\times10^{3}(1.4\times10^{2})$	$2.1\times10^{3}(5.1\times10)$	$1.3\times10^{3}(9.9\times10)$	$6.3\times10^{3}(1.2\times10^{2})$
f_6	$7.7\times10^{2}(7.0)$	$7.1\times10^{2}(4.2)$	$6.5\times10^{2}(1.3\times10)$	$8.9\times10^{2}(1.4\times10)$
f_7	$1.2\times10^{4}(7.0\times10^{2})$	$3.9\times10^{3}(9.5\times10)$	$2.7\times10^{3}(2.6\times10^{2})$	$3.6\times10^{4}(4.2\times10^{2})$
f_8	$3.6\times10^{3}(1.4\times10^{2})$	$2.5\times10^{3}(4.7\times10)$	$1.6\times10^{3}(9.8\times10)$	$6.0\times10^{3}(1.0\times10^{2})$
f_9	$2.4\times10^{5}(1.8\times10^{4})$	$8.5\times10^{4}(4.0\times10^{3})$	$4.5\times10^{4}(1.9\times10^{4})$	$2.9\times10^{5}(5.2\times10^{4})$
f_{10}	$3.7\times10^{4}(1.0\times10^{3})$	$3.3\times10^{4}(5.0\times10^{2})$	$2.1\times10^{4}(2.3\times10^{3})$	$3.2\times10^{4}(1.5\times10^{3})$
f_{11}	$5.1\times10^{7}(1.3\times10^{8})$	$1.8\times10^{5}(1.6\times10^{4})$	$2.4\times10^{4}(1.2\times10^{4})$	$6.0\times10^{7}(1.8\times10^{8})$
f_{12}	$3.8\times10^{8}(3.9\times10^{10})$	$1.5\times10^{11}(2.0\times10^{11})$	$9.4\times10^{8}(9.0\times10^{8})$	$9.9\times10^{11}(7.3\times10^{11})$
f_{13}	$9.1\times10^{10}(1.0\times10^{10})$	$3.4\times10^{10}(4.5\times10^{9})$	$1.1\times10^{7}(3.3\times10^{6})$	$3.1\times10^{11}(2.0\times10^{10})$
f_{14}	$1.3\times10^{8}(2.1\times10^{8})$	$4.7\times10^{7}(6.6\times10^{6})$	$5.4\times10^{6}(4.3\times10^{6})$	$5.0\times10^{8}(3.4\times10^{8})$
f_{15}	$5.3\times10^{10}(1.0\times10^{10})$	$1.5\times10^{10}(2.8\times10^{9})$	$2.5\times10^{6}(8.3\times10^{5})$	$1.6\times10^{11}(8.3\times10^{9})$
f_{16}	$4.7\times10^{4}(6.4\times10^{3})$	$1.8\times10^{4}(1.2\times10^{3})$	$7.1\times10^{3}(7.9\times10^{2})$	$1.9\times10^{5}(3.7\times10^{4})$
f_{17}	$1.3\times10^{8}(8.5\times10^{7})$	$1.1\times10^{6}(4.8\times10^{5})$	$6.2\times10^{3}(8.2\times10^{2})$	$7.8\times10^{8}(9.3\times10^{8})$
f_{18}	$1.9\times10^{9}(9.3\times10^{8})$	$9.5\times10+07(2.8\times10^{7})$	$7.1\times10^{6}(3.9\times10^{6})$	$1.1\times10^{9}(5.8\times10^{8})$
f_{19}	$6.0\times10^{10}(8.1\times10^{9})$	$1.1\times10^{10}(1.5\times10^{9})$	$4.8\times10^{6}(4.3\times10^{6})$	$1.4\times10^{11}(2.4\times10^{10})$
f_{20}	$9.9\times10^{3}(4.0\times10^{2})$	$8.1\times10^{3}(2.4\times10^{2})$	$6.2\times10^{3}(5.0\times10^{2})$	$9.6\times10^{3}(4.6\times10^{2})$
f_{21}	$5.5\times10^{3}(3.0\times10^{2})$	$4.4\times10^{3}(1.1\times10^{2})$	$3.1\times10^{3}(7.4\times10)$	$8.6\times10^{3}(1.3\times10^{2})$
f_{22}	$4.0\times10^{4}(1.0\times10^{3})$	$3.6\times10^{4}(5.1\times10^{2})$	$2.4\times10^{4}(1.5\times10^{3})$	$3.8\times10^{4}(8.4\times10^{2})$
f_{23}	$8.4\times10^{3}(9.9)$	$5.9\times10^{3}(2.2\times10^{2})$	$3.7\times10^{3}(1.2\times10^{2})$	$8.0\times10^{3}(2.9\times10^{2})$
f_{24}	$1.4\times10^{4}(2.0\times10^{3})$	$9.0\times10^{3}(4.1\times10^{2})$	$4.2\times10^{3}(1.3\times10^{2})$	$1.2\times10^{4}(8.1\times10^{2})$
f_{25}	$1.4\times10^{5}(1.5\times10^{4})$	$2.2\times10^{4}(1.8\times10^{3})$	$3.9\times10^{3}(2.5\times10^{2})$	$1.1\times10^{6}(1.1\times10^{5})$
f_{26}	$9.7\times10^{4}(1.1\times10^{4})$	$4.8\times10^{4}(2.9\times10^{3})$	$1.6\times10^{4}(1.5\times10^{3})$	$7.3\times10^{4}(4.3\times10^{3})$
f_{27}	$1.7\times10^{4}(1.6\times10^{3})$	$1.1\times10^{4}(1.0\times10^{3})$	$3.7\times10^{3}(1.1\times10^{2})$	$2.6\times10^{4}(2.5\times10^{3})$
f_{28}	$7.5\times10^{4}(7.7\times10^{3})$	$3.0\times10^{4}(1.6\times10^{3})$	$5.2\times10^{3}(2.1\times10^{3})$	$2.1\times10^{5}(2.2\times10^{4})$
f_{29}	$2.9\times10^{7}(3.2\times10^{7})$	$1.4\times10^{5}(1.1\times10^{5})$	$8.1\times10^{3}(8.0\times10^{2})$	$4.0\times10^{8}(2.3\times10^{8})$
f_{30}	$7.7\times10^{10}(1.4\times10^{10})$	$2.7\times10^{10}(4.7\times10^{9})$	$1.6\times10^{7}(5.8\times10^{6})$	$2.9\times10^{11}(2.7\times10^{10})$
+/=/−	**29/0/0**	**29/0/0**	**26/3/0**	**27/2/0**

（续表）

函数	WDO Mean(Std)	GWO Mean(Std)	ISGWO M×10an(Std)
f_1	$9.5\times10^{10}(1.1\times10^{10})$	$6.0\times10^{10}(1.2\times10^{10})$	**$3.4\times10^{3}(3.5\times10^{3})$**
f_2	**$3.2\times10^{5}(1.5\times10^{4})$**	$6.8\times10^{5}(1.4\times10^{5})$	$6.3\times10^{5}(5.7\times10^{4})$
f_3	$1.5\times10^{4}(1.7\times10^{3})$	$8.1\times10^{3}(1.8\times10^{3})$	**$6.0\times10^{2}(3.0\times10)$**
f_4	$1.8\times10^{3}(7.6\times10)$	$1.3\times10^{3}(1.5\times10^{2})$	**$9.7\times10^{2}(2.1\times10)$**
f_5	$6.9\times10^{2}(4.3)$	$6.5\times10^{2}(5.8)$	**$6.0\times10^{2}(2.1\times10^{13})$**
f_6	$3.2\times10^{3}(1.2\times10^{2})$	$2.3\times10^{3}(1.9\times10^{2})$	**$1.2\times10^{3}(3.1\times10)$**
f_7	$2.1\times10^{3}(5.8\times10)$	$1.6\times10^{3}(5.3\times10)$	**$1.3\times10^{3}(3.1\times10)$**
f_8	$7.5\times10^{4}(6.4\times10^{3})$	$5.9\times10^{4}(1.6\times10^{4})$	**$2.1\times10^{4}(2.4\times10^{3})$**
f_9	$2.8\times10^{4}(1.3\times10^{3})$	$2.1\times10^{4}(9.6\times10^{2})$	**$1.2\times10^{4}(3.7\times10^{2})$**
f_{10}	**$1.0\times10^{4}(2.0\times10^{4})$**	$1.3\times10^{5}(3.4\times10^{4})$	$6.7\times10^{4}(1.8\times10^{4})$
f_{11}	$2.4\times10^{10}(4.8\times10^{9})$	$1.4\times10^{10}(6.8\times10^{9})$	**$2.0\times10^{7}(6.9\times10^{6})$**
f_{12}	$2.7\times10^{9}(1.1\times10^{9})$	$1.2\times10^{9}(5.3\times10^{8})$	**$4.2\times10^{3}(1.5\times10^{3})$**
f_{13}	$9.7\times10^{6}(3.3\times10^{6})$	$1.4\times10^{7}(8.7\times10^{6})$	**$6.5\times10^{6}(2.1\times10^{6})$**
f_{14}	$3.7\times10^{8}(1.8\times10^{8})$	$2.4\times10^{8}(3.0\times10^{8})$	**$2.3\times10^{3}(5.3\times10^{2})$**
f_{15}	$1.2\times10^{4}(1.8\times10^{3})$	$7.7\times10^{3}(1.6\times10^{3})$	**$4.9\times10^{3}(4.1\times10^{2})$**
f_{16}	$8.0\times10^{3}(5.6\times10^{2})$	$7.1\times10^{3}(2.3\times10^{3})$	**$4.3\times10^{3}(2.0\times10^{2})$**
f_{17}	$8.2\times10^{6}(3.0\times10^{6})$	$1.3\times10^{7}(5.8\times10^{6})$	**$4.1\times10^{6}(9.6\times10^{4})$**
f_{18}	$3.5\times10^{8}(2.2\times10^{8})$	$2.3\times10^{8}(2.3\times10^{8})$	**$2.9\times10^{3}(1.2\times10^{3})$**
f_{19}	$6.7\times10^{3}(6.4\times10^{2})$	$6.1\times10^{3}(1.2\times10^{3})$	**$4.6\times10^{3}(3.0\times10^{2})$**
f_{20}	$4.2\times10^{3}(1.6\times10^{2})$	$3.2\times10^{3}(1.1\times10^{2})$	**$2.8\times10^{3}(4.2\times10)$**
f_{21}	$3.2\times10^{4}(2.3\times10^{3})$	$2.7\times10^{4}(6.1\times10^{3})$	**$1.5\times10^{4}(3.9\times10^{2})$**
f_{22}	$5.7\times10^{3}(5.6\times10^{2})$	$3.8\times10^{3}(1.1\times10^{2})$	**$3.1\times10^{3}(1.4\times10)$**
f_{23}	$6.0\times10^{3}(2.7\times10^{2})$	$4.8\times10^{3}(2.5\times10^{2})$	**$3.7\times10^{3}(3.4\times10)$**
f_{24}	$1.1\times10^{4}(8.8\times10^{2})$	$7.0\times10^{3}(1.0\times10^{3})$	**$3.3\times10^{3}(3.8\times10)$**
f_{25}	$3.0\times10^{4}(3.2\times10^{3})$	$1.9\times10^{4}(1.4\times10^{3})$	**$1.1\times10^{4}(3.4\times10^{2})$**
f_{26}	$5.8\times10^{3}(7.5\times10^{2})$	$4.4\times10^{3}(1.8\times10^{2})$	**$3.4\times10^{3}(2.4\times10)$**
f_{27}	$1.4\times10^{4}(1.7\times10^{3})$	$1.0\times10^{4}(2.1\times10^{3})$	**$3.4\times10^{3}(4.0\times10)$**
f_{28}	$1.5\times10^{4}(1.4\times10^{3})$	$1.0\times10^{4}(6.5\times10^{2})$	**$7.0\times10^{3}(1.7\times10^{2})$**
f_{29}	$1.8\times10^{9}(4.8\times10^{8})$	$2.5\times10^{9}(2.2\times10^{9})$	**$2.4\times10^{4}(6.1\times10^{3})$**
+/=/−	**24/3/2**	**26/3/0**	Wilcoxon

采用秩和检验对测试结果进行评估，其中，“+”“−”和“=”分别表示 ISGWO 寻优效果强于、弱于和接近于其他算法。统计结果表明，ISGWO 在与 PSO 的对比中有绝对优势，仅在不超过两个基准函数上的寻优效果差于 WDO。与原始 GWO 相比，ISGWO 在 CEC 测试函数集上的表现更出色。ISGWO 仅在 Shifted and Rotated Zakharov Function f_3 中的表现弱于 WDO。这可能是由于 Shifted and Rotated Zakharov Function 坡面较长，而最优值位置在搜索空间边界。惯性权重和自适应更新机制对种群更新幅度的影响有限，以致于 ISGWO 在应对坡面较长的单峰函数时很难进一步地接近理论解。

结束语 本文提出了一种基于 IMQ 惯性权重策略的自适应灰狼优化算法(ISGWO)。该算法利用 IMQ 函数和 Sigmoid 函数的特性，通过非线性调整惯性权重，自适应地更新种群个体的位置。

与其他 6 种常用优化算法相比，本文提出的 ISGWO 在收敛速度和精度方面具有显著的优势。通过时间复杂度分析，验证了 ISGWO 的改进与 GWO 相比并未增加算法的计算负担，确保了实用性和高效性。参数选取实验、消融实验、可拓展性实验结果表明，ISGWO 在优化问题中表现出明显的优越性，在实际应用中能够有效地解决复杂优化问题，提供更准确、高效的解决方案。

然而，本文提出的算法还存在一些改进的空间。首先，可以进一步优化算法的参数设置，以提高算法在不同问题领域的适应性和鲁棒性。其次，可以结合其他优化策略或者混合算法，进一步增强算法的搜索能力和稳定性。此外，还可以探索并引入更多的启发式机制，以进一步提升算法的性能。

参 考 文 献

[1] KENNEDY J, EBERHART R. Particle Swarm Optimization [C]// International Conference on Neural Networks. New York: IEEE Press, 1995: 1-2.

[2] YANG X S. Firefly algorithms for multimodal optimization [C]// International symposium on stochastic algorithms. Berlin, Heidelberg: Springer, 2009: 169-178.

[3] MIRJALILI S. SCA: a sine cosine algorithm for solving optimization problems[J]. Knowledge-Based Systems, 2016, 96: 120-133.

[4] BAYRAKTAR Z, KOMURCU M, WERNER D H. Wind Driven Optimization(WDO): A novel nature-inspired optimization algorithm and its application to electromagnetics[C]// 2010 IEEE Antennas and Propagation Society International Symposium. New York: IEEE Press, 2010: 1-4.

[5] PAN W T. A new fruit fly optimization algorithm: taking the financial distress model as an example[J]. Knowledge-Based Systems, 2012, 26: 69-74.

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

[7] FAN X Z, YU M. Coverage Optimization of WSN Based on Improved Grey Wolf Optimizer [J]. Computer Science, 2022, 49(S1): 628-631.

[8] MAO M X, XU Z, CUI L C, et al. Research on Multi-Peak

MPPT of Photovoltaic Array Based on Modified Gray Wolf Optimization Algorithm[J]. Acta Energiae Solaris Sinica(Journal of Solar Energy),2023,44(3):450-456.

[9] ZHANG W N,ZHOU Q L,JIAO Z Y,et al. Hybrid Algorithm of Grey Wolf Optimizer and Arithmetic Optimization Algorithm for Class Integration Test Order Generation[J]. Computer Science,2023,50(5):72-81.

[10] XI L,HE M,ZHOU B Q,et al. Research on False Data Injection Attack Detection in Power System Based on Improved Multi Layer Extreme Learning Machine[J]. Acta Automatica Sinica (Acta Automatica),2023,49(4):881-890.

[11] ZHANG A,YANG M,BI W H,et al. Task allocation of heterogeneous multi-UAVs in uncertain environment based on multi-strategy integrated GWO[J]. Acta Aeronautica et Astronautica Sinica(Journal of Aeronautics),2023,44(8):148-164.

[12] KANG Y,WANG H N,TAO L,et al. Hybrid Improved Flower Pollination Algorithm and Gray Wolf Algorithm for Feature Selection[J]. Computer Science,2022,49(S1):125-132.

[13] LIU Y,LIU D P,MU Y,et al. Power Distribution Strategy of Hybrid Energy Storage System Based on GWO Optimization of ICEEMDAN Decomposition[J]. Journal of Electrical Engineering,2022,17(4):257-267.

[14] LIU Y,JIANG Y,ZHANG X,et al. An improved grey wolf optimizer algorithm for identification and location of gas emission [J]. Journal of Loss Prevention in the Process Industries,2023,82:105003.

[15] WANG H,ZOU Q,LIN H. A Quasi-Optimal Shape Design Method for Electromagnetic Scatterers Based on NURBS Surfaces and Filter-Enhanced GWO[J]. IEEE Transactions on Antennas and Propagation,2023,71(5):4236-4245.

[16] LU M,QU L D,HE D X. Pathfinder Grey Wolf Algorithm for Solving Multiple-roots Nonlinear Equations[J]. Chinese Journal of Engineering Mathematics,2022,39(6):957-968.

[17] LIU Z Q,HE L,YUAN L,et al. Path Planning of Mobile Robot Based on TGWO Algorithm[J]. Journal of Xi'an Jiaotong University,2022,56(10):49-60.

[18] XIE S P,WU B S,ZHAO X T,et al. Structural damage identification based on improved gray wolf optimization algorithm[J]. Chinese Journal of Computational Mechanics,2024,41(2):256-262.

[19] ZHANG L,ZHENG L D,LENG X B,et al. Research on Multi-Objective Optimization Strategy of Wind-Photovoltaic-Pumped Storage Combined System Based on Gray Wolf Algorithm[J/OL]. Journal of Shanghai Jiao Tong University. https://doi.org/10.16183/j.cnki.jsjtu.2023.049.

[20] YU X,XU W Y,WU X,et al. Reinforced exploitation and exploration grey wolf optimizer for numerical and real-world optimization problems[J]. Applied Intelligence,2022,52(8):8412-8427.

[21] RATHAN S,SHAH D,KUMAR T H,et al. Adaptive IQ and IMQ-RBFs for solving Initial Value Problems:Adam-Bashforth and Adam-Moulton methods[J]. arXiv:2302.06113,2023.

[22] E J T,LIU J,WAN Z. A novel adaptive algorithm of particle swarm optimization based on the human social learning intelligence[J]. Swarm and Evolutionary Computation,2023,80:101336.

[23] BANAIE-DEZFOULI M,NADIMI-SHAHRAKI M H,BEHESHTI Z. BEGWO:Binary extremum-based grey wolf optimizer for discrete optimization problems[J]. Applied Soft Computing,2023,146:110583.

YU Mingyang, born in 2000, postgraduate. His main research interests include deep learning and intelligent optimization algorithm.

LI Ting, born in 1988, master, senior engineer. Her main research interests include image big data governance, algorithm parity and intelligent recognition algorithm.

（责任编辑：杨雪敏）