DOI:10.19652/j.cnki.femt.2406246

多策略融合的斑马优化算法

黄子介 代永强
(甘肃农业大学 兰州 730070)

摘 要:针对斑马优化算法易陷入局部最优和鲁棒性差的问题,提出了一种多策略融合的斑马优化算法。首先采用基于Logistics混沌的收敛因子作为步长控制因子,平衡了算法全局探勘与局部开发能力,提高了算法的寻优精度;其次采用位置扰动策略,避免了迭代过程种群多样性的减少,增强了算法跳出局部最优的能力;最后采用记忆更新策略,降低了位置更新策略的盲目性。利用14个标准测试函数,在收敛精度、收敛速度、统计检验3个方面对改进后算法的优良性进行实验检验。实验结果表明,改进策略有效地提升了斑马优化算法寻优精度与跳出局部最优的能力。在工程优化问题上进一步验证了算法处理实际优化问题的有效性与实用性。

关键词:斑马优化算法;Logistics混沌收敛因子;位置扰动策略;记忆更新策略;统计检验
中图分类号:TP301.6 **文献标识码**:A **国家标准学科分类代码**:520.604

Multi-strategy combined zebra optimization algorithm

Huang Zijie Dai Yongqiang
(Gasu Agricultural University, Lanzhou 730070, China)

Abstract: To address the issue of zebra optimization algorithm being prone to local optimum and poor robustness, a Multi-strategy combined zebra optimization algorithm is proposed. Utilizing a convergence factor based on Logistics chaos as the step control parameter balances the algorithm′s global exploration and local exploitation capabilities, thereby enhancing its optimization precision. Utilizing a position perturbation strategy avoid the decrease of population diversity in the iterative process and enhance the ability of the algorithm to jump out of local optimum. Utilizing a memory update strategy reduces the blindness of the location update strategy. Selecting Fourteen standard test functions examine the excellence of the improved algorithm in convergence accuracy, convergence speed, and statistical tests three aspects. Experimental results show that the improved strategy effectively improves the optimization accuracy and the ability to jump out of the local optimal of the zebra optimization algorithm. The effectiveness and practicability of the algorithm in dealing with practical optimization problems are further verified by engineering optimization problems.

Keywords: zebra optimization algorithm; Logistics chaotic convergence factor; position perturbation strategy; memory update strategy; statistical testlion algorithm

0 引 言

智能优化算法是一种启发式算法,其设计灵感源自于特定事物活动产生启发,并通过数学模型对该灵感进行建模以解决优化问题。这类算法具有鲁棒性、可扩充性和便于实现等特点,在实际应用领域得到了广泛应用[1]。近几年,该领域学者根据受各类动植物种群生活习性启发,开创了教与学优化算法[2]、学前教育优化算法[3]、算术优化算法[4]、蜣螂优化算法[5]、灰狼优化算法[6](GWO)等一系列具有求解高效、灵活度高等优点的新型优化算法并广泛应用于实际应用求解和其他科学领域,如路径规划[7-8]、工程调度[9]、特征选择[10]以及超参数优化[11]等问题。

斑马优化算法(ZOA)是由Trojovská等[12]提出一种新型的生物启发式算法。该算法通过模拟斑马群的觅食和防御活动,寻求全局最优解。与以往智能优化算法相比,斑马优化算法具有寻优能力强、收敛速度快等特点。

收稿日期:2024-08-07

目前已经在微电网频率控制领域实现应用[13]。然而该算法仍存在群智能算法的通病，如容易陷入局部最优和鲁棒性较差。针对这些问题，该领域的学者们提出了许多解决措施。文献[14]提出了一种自适应振荡权重和黄金正弦算子的斑马优化算法(IZOA)，自适应振荡权重可以调整算法的探索和利用策略，以平衡空间探索和局部区域利用，黄金正弦算子可以在算法的前期增加搜索空间避免局部优化，并在后期减少搜索空间以提高收敛精度。文献[15]提出了一种融合正切搜索与竞争交配的斑马优化算法，融合正切搜索提高了算法前期搜索空间，补足了斑马算法由最优个体导向更新产生的难以跳出局部最优的窘境；竞争交配策略由于根据最优个体进行更新，有效的提高了算法的寻优速度；文献[16]使用混沌正弦图增强斑马优化算法(ZOA)，该研究所使用方法在一定程度上加快了算法的收敛速度；文献[17]提出了一种动态策略，使每次迭代完成之后更新的种群，在各个体更新时就发生了更新，提高了种群多样性，避免陷入局部最优的能力也有所提升。

本文针对斑马算法易陷入局部最优及种群多样性差等问题，提出一种基于多策略融合的斑马优化算法。通过将 Logistics 混沌收敛因子替代觅食策略的步长控制因子，提高算法在迭代期间的全局搜索能力与局部开发能力；引入一种位置扰动策略，根据小概率原理对个体位置进行扰动，在不影响收敛速度的前提下提高算法迭代过程的种群多样性；提出一种记忆更新策略对防御阶段的位置更新公式进行修改，提高位置更新策略的有效性，避免算法陷入局部最优。

1 斑马优化算法

斑马优化算法是在数学上通过模拟斑马群觅食行为与防御行为建立的优化模型。该算法寻优过程主要有如下 3 个步骤组成：种群初始化、觅食行为及防御行为。

1.1 种群初始化

斑马优化算法中采用随机生成种群的方法，即：

$$x_{i,j}=lb_j+r\cdot(ub_j-lb_j) \tag{1}$$

式中：$x_{i,j}$ 是种群个体；ub_j 为第 i 维度下的寻优上边界；lb_j 为第 j 维度下的寻优下边界；r 是范围 (0,1) 内的随机数。

1.2 觅食行为

该过程是模拟斑马觅食行为，斑马种群向先锋斑马靠近的过程。具体更新公式表示为：

$$x_{t+1}=x_t+r\cdot(PZ-I\cdot x_t) \tag{2}$$

式中：r 是范围 (0,1) 的随机数；PZ 为先峰斑马；I 是集合 {1,2} 随机值。

1.3 针对捕食者的防御策略

斑马面对不同捕食者会采取不同的防御策略。斑马面对大型捕食者攻击，会选择 z 字形跑动来躲避捕食者攻击的防御策略；斑马面对小型捕食者攻击，会选择聚集反击的防御策略。具体更新公式如下：

$$x_{t+1}=\begin{cases}S_1:x_t+r_1\cdot(2r_2-1)\cdot\left(1-\dfrac{t}{T}\right)\cdot x_t, & r_3>0.5\\ S_2:x_t+r_4\cdot(AZ-I\cdot x_t), & \text{其他}\end{cases} \tag{3}$$

式中：S_1 为面对大型捕食者所采取的防御策略；S_2 为面对小型捕食者所采取的防御策略；r_1、r_2、r_3、r_4 为范围 (0，1) 的随机数；AZ 为被攻击的斑马。

2 多策略融合的斑马优化算法

斑马算法具有收敛速度快、寻优能力强等优点，但斑马优化算法的更新策略主要依靠于最优个体的位置且参数具有很大的随机性，导致算法存在易于陷入局部最优、鲁棒性差等缺点。针对该问题，本文引入了 Logistics 混沌收敛因子、位置扰动策略以及根据记忆更新算法的改进防御策略。

2.1 Logistics 混沌收敛因子

斑马优化算法的全局搜索与局搜索能力由随机数 r 决定，当 $r>0.5$ 时，算法全局搜索能力强，易于发现近似最优解的个体，当 $r<0.5$ 时，算法的局部搜索能力强，有利于寻找最优解个体。但在实际应用中，往往期望在迭代前期，步长控制因子尽可能大，以便算法能够在较大的范围内进行搜索，扩大搜索范围。在迭代后期，往往又期望步长控制因子维持在一个较小的范围内，以提高算法的寻优精度。原始斑马算法中的步长控制因子由随机数 r 决定，在迭代前期，因子可能因数值过小导致算法的收敛速度较慢，在迭代后期，因子可能因数值过大导致算法的局部搜索能力较差。因此，本文提出一种基于 Logistics 混沌的非线性收敛因子，以平衡算法迭代过程的全局搜索能力与局部搜索能力，该随机因子变动范围随着迭代的增加逐渐减小，在迭代前期，因子维持为一个较大数值，以保证迭代前期算法能够发现更多的近似最优个体；在迭代后期，因子维持为一个较小数值，以提高算法的搜索精度。Logistics 混沌收敛因子的具体计算公式如下：

$$a=l\cdot\left[\left(1-\left(\frac{t}{T}\right)^{\frac{1}{4}}\right)+\left(1-\left(\frac{t}{T}\right)^{\frac{1}{2}}-\left(1-\left(\frac{t}{T}\right)^{\frac{1}{4}}\right)\right)\right] \tag{4}$$

式中：t 为当前迭代次数；T 为最大迭代次数；l 为 Logistics 混沌扰动因子。Logistics 混沌收敛因子的收敛曲线如图 1 所示。

Logistics 混沌收敛因子引入觅食阶段的位置更新公式为：

$$x_{t+1}=x_t+a\cdot(PZ-I\cdot x_t) \tag{5}$$

2.2 局部扰动策略

斑马优化算法觅食阶段的个体更新策略以全局最优个体的位置为参考物，个体更新将呈现向最优个体聚拢的趋势。在此过程中，算法的种群多样性将会减小，从而导

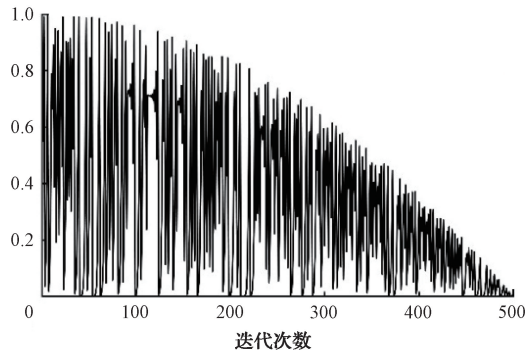

图 1 Logistics 混沌收敛因子曲线

Fig. 1 Logistics chaos convergence factor curve

致算法的全局搜索能力变弱、收敛速度变慢，并且当算法陷入局部最优时，算法后期跳出局部最优的能力较弱。因此本文提出一种位置扰动策略，根据小概率原理引入自适应因子对算法中随机个体的位置进行扰动，以减缓算法迭代过程多样性的损失、提高算法搜索范围并增强其跳出局部最优的能力。引入因子时结合小概率原理，能够避免对个体位置的过度扰动导致算法过早的陷入局部最优，同时能够提高算法运行效率，保证在不很大程度影响算法收敛速度的情况下，增强算法跳出局部最优的能力。该因子具体计算公式如下：

$$\omega = step \cdot \exp\left(-\frac{30t}{T}\right)^{10} \tag{6}$$

式中：

$$step = 0.1 \cdot (ub_j - lb_j) \tag{7}$$

式中：$step$ 表示移动步长；ub_j 第 j 维度下的寻优上边界；lb_j 第 j 维度下的寻优下边界；t 为当前迭代次数；T 为最大迭代次数。

与觅食策略进行结合，更新公式如下：

$$x_{t+1} = \begin{cases} x_t + a \cdot (PZ - I \cdot x_t), & r > 0.01 \\ \omega \cdot (x_t + a \cdot (PZ - I \cdot x_t)), & \text{其他} \end{cases} \tag{8}$$

其中，r 是范围 (0,1) 内的随机数。

2.3 记忆更新策略

斑马优化算法在迭代过程陷入局部最优时，往往根据防御策略实现跳出局部最优的功能，但是防御策略是根据随机个体位置和参数对个体进行更新，该更新策略具有较大的随机性。因此本文提出一种记忆学习策略，对防御阶段面对小型捕食者策略进行修改，引入记忆因子与学习因子，使该位置更新策略不仅仅依靠其他随机个体进行位置更新，还受更新个体的历史最优值影响，使个体朝向较优值的位置进行更新，有利于提高算法的收敛速度，更快寻得最优解。采用记忆学习策略的位置更新策略具体计算公式如下：

$$S_{2new}: x_{t+1} = x_t + c_1 \cdot r_1 \cdot (P_{best} - I \cdot x_t) + c_2 \cdot r_2 \cdot (AZ - I \cdot x_t) \tag{9}$$

式中：S_{2new} 为改进后的面对小型捕食者所采取的防御策略；c_1、c_2 为学习因子，经多次试验均取 1；r_1 为范围 (−1, 1) 内的随机数；r_2 为范围 (0,1) 的随机数；AZ 为被攻击的斑马；P_{best} 为更新个体斑马的历史最优值；I 是集合 {1, 2} 随机值。

将该策略引入防御策略进行修改，防御策略更新公式如下：

$$x_{t+1} = \begin{cases} S_1: x_t + r_1 \cdot (2r_2 - 1) \cdot \left(1 - \frac{t}{T}\right) \cdot x, & r_3 > 0.5 \\ S_{2new}, & \text{其他} \end{cases} \tag{10}$$

式中：r_1、r_2、r_3 均为 (0,1) 的随机数。

2.4 MSCZOA 算法实现步骤

步骤 1)初始化算法相关系数，设置最大迭代次数 T 、种群规模 N 、维度 D 、搜索上下界 ub 与 lb 等参数。

步骤 2)根据原始 ZOA 算法随机初始化种群。

步骤 3)计算个体适应度，选定群体中的最优个体为先锋斑马。

步骤 4)根据式(8)对全局个体位置进行更新，计算更新个体适应度值，与原本个体适应度值进行比较，保留更新后的较优个体。

步骤 5)根据式(10)对种群位置进行更新，计算更新个体适应度值，与原本个体适应度值进行比较，保留更新后的较优个体。若个体适应度值低于个体以往历代适应度值，记录该更新个体为该个体的历史最优个体。

步骤 6)判断当前迭代次数是否达到预先设置的最大迭代次数，若达到输出最优个体的适应度值与位置，若没达到返回步骤 3)。

3 实验与结果分析

为了验证改进策略对斑马算法的有效性，本文选用 14 个常用基准测试函数对改进后算法优良性进行测验。实验环境为 AMD Ryzen 7 4800H with Radeon Graphics 处理器，主频 2.90 GHz，内存 16 G，Windows10 64 位操作系统，采用 MATLAB2020b 进行仿真实验。在运行时为保证算法产生的实验数据无差异，本文所有算法实验的种群为 50，最大迭代次数为 500，并进行 30 次独立计算，取平均值与标准差作为评估算法优良性的测试指标，得到收敛曲线。

3.1 基准测试函数

本文采用的 14 种经典基准测试函数名称、具体公式、搜索范围、选取维度及理论最小值如表 1 所示，其中 F1～F7 为单峰函数，F8～F12 为多峰函数，F13、F14 为固定维度的多峰函数。

3.2 与其他算法的对比分析

为了分析与测试多策略融合的斑马算法(MSCZOA)的性能与实际效果，本文选用近几年提出并已经应用于多个领域的两个经典优化算法鲸鱼优化算法(WOA)[18]、

表 1 基准测试函数的基本信息

Table 1 Basic information about the benchmark function

名称	基准函数	Dim	搜索范围	min
Sphere	$F_1(X)=\sum_{i=1}^{n}x_i^2$	30	[−100,100]	0
Schwefel's Problem 2.22	$F_2(X)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	30	[−10,10]	0
Schwefel's Problem 1.2	$F_3(X)=\sum_{i=1}^{n}(\sum_{j=1}^{n}x_j)^2$	30	[100,100]	0
Schwefel's Problem 2.21	$F_4(X)=\max\{\lvert x_i\rvert,1\leqslant i\leqslant n\}$	30	[−100,100]	0
Generalized Rosenbrock	$F_5(X)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	30	[−30,30]	0
Step Function	$F_6(X)=\sum_{i=1}^{n}([x_i+0.5])^2$	30	[−100,100]	0
Quartic Function i.e. Noise	$F_7(X)=\sum_{i=1}^{n}i\cdot x_i^4+random[0,1)$	30	[−1.28,1.28]	0
Generalized Rastrigin	$F_8(X)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	30	[−5.12,5.12]	0
Ackley	$F_9(X)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right)+20+e$	30	[−32,32]	0
Griewank	$F_{10}(X)=\frac{1}{4\ 000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{i}\right)+1$	30	[−600,600]	0
Generalized Penalized Function	$F_{11}(X)=\frac{\pi}{n}(10\sin(\pi y_1)+\sum_{i=1}^{n}(y_i-1)^2(1+10\sin^2(\pi y_{i+1}))+(y_n-1)^2)+\sum_{i=1}^{n}u(x_i,10,100,4)$	30	[−50,50]	0
Generalized Penalized Function	$F_{12}(X)=0.1\{\sin^2(3\pi x_1)+\sum_{i=1}^{n-1}(x_i-1)^2[1+\sin(3\pi x_{i+1})]+(x_n-1)[1+\sin^2(2\pi x_n)]\}+\sum_{i=1}^{n}u(x_i,5,100,4)$	30	[−50,50]	0

GWO 算法[15]以及 IZOA 算法[14]、CZOA 算法[16]基于上述 14 个基准测试函数进行对比实验。各算法参数设置如表 2 所示。

对比实验结果如表 3 所示，从单峰函数 F1～F7 上看，MSCZOA 算法除函数 F7 外，均寻得函数的理论最优值，且相较于原算法与其他算法，除函数 F6 外，MSCZOA 算法的平均值与标准差均明显优于原算法与其他算法，说明改进策略有效地提高了 ZOA 算法的寻优能力。从多峰函数 F8～F12 与固定维度地多峰函数 F13、F14 的测试上看，除函数 F9、F11、F12 外，MSCZOA 算法均取得函数的理论最优值，说明 MSCZOA 算法具有较强全局勘探与局部开发能力。其中虽然 MSCZOA 算法在单峰函数 F7、多

表 2 算法参数设置

Table 2 Algorithm parameters setting

算法	参数设置
MSCZOA	I 为集合{1,2}的随机值，r 是[0,1]的随机数
ZOA	I 为集合{1,2}的随机值，r 是[0,1]的随机数
WOA	$a\in[0,2]$，并从 2 线性下降，p 是[0,1]的随机数，b 为 1
GWO	$a\in[0,2]$，并从 2 线性下降，$r1$，$r2$ 是[0,1]的随机数
IZOA	I 为集合{1,2}的随机值，r 是[0,1]的随机数
CZOA	I 为集合{1,2}的随机值，r 是[0,1]的随机数

峰函数 F11、F12 上未能寻得理论最优值，但是与其他算法相比，算法的准确性与鲁棒性明显优于其他算法，也说明了 MSCZOA 算法具有优越的性能。

与改进算法相比，从单峰函数 F1～F7 上看，MSCZOA 除在单峰函数 F7 上所得实验结果略劣于 CZOA，其稳定性与寻优能力均优于其他两种改进算法；从多峰函数 F8～F12 与固定维度地多峰函数 F13、F14 的测试上看，MSCZOA 除组合函数 F13 所得结果略劣于 IZOA，其探索与开发的能力和跳出局部最优能力均优于其他两种改进算法。

综上，从单峰、多峰、组合函数的测试结果来看，相比于两个经典优化算法及两个改进算法，MSCZOA 搜索到的最优值更加接近理论最优值，总体寻优性能更好，在收敛精度和鲁棒性更好，且能在一定程度上跳出局部最优。

表 3 MSCZOA 算法与其他算法的对比实验结果

Table 3 Experimental results of MSCZOA compared with other algorithms

函数		MSCZOA	ZOA	WOA	GWO	IZOA	CZOA
F1	平均值	0	5.96×10^{-252}(+)	4.52×10^{-73}(+)	8.01×10^{-28}(+)	5.52×10^{-182}(+)	0(=)
	标准差	0	0(=)	1.75×10^{-72}(+)	9.19×10^{-28}(+)	0(=)	0(=)
F2	平均值	0	1.58×10^{-131}(+)	2.26×10^{-50}(+)	1.01×10^{-16}(+)	1.23×10^{-95}(+)	2.46×10^{-262}(+)
	标准差	0	7.79×10^{-131}(+)	1.17×10^{-49}(+)	1.02×10^{-16}(+)	3.38×10^{-95}(+)	0.00×10(=)
F3	平均值	0	6.11×10^{-156}(+)	2.26×10^{-50}(+)	1.01×10^{-16}(+)	1.23×10^{-95}(+)	2.46×10^{-262}(+)
	标准差	0	3.33×10^{-155}(+)	1.17×10^{-49}(+)	1.02×10^{-16}(+)	3.38×10^{-95}(+)	0(=)
F4	平均值	0	7.20×10^{-115}(+)	5.79×10(+)	6.59×10^{-7}(+)	5.97×10^{-94}(+)	4.50×10^{-262}(+)
	标准差	0	1.51×10^{-114}(+)	2.57×10(+)	6.30×10^{-7}(+)	2.39×10^{-93}(+)	0(=)
F5	平均值	0.026 473 128	2.82×10(+)	2.80×10(+)	2.70×10(+)	2.70×10(+)	2.90×10(+)
	标准差	0.144 124 224	5.82×10^{-1}(+)	4.64×10^{-1}(+)	8.33×10^{-1}(+)	8.19×10(+)	2.17×10^{-2}(−)
F6	平均值	0	0(=)	0(=)	0(=)	0(=)	0(=)
	标准差	0	0(=)	0(=)	0(=)	0(=)	0(=)
F7	平均值	5.60×10^{-5}	7.68×10^{-5}(+)	2.51×10^{-3}(+)	2.25×10^{-3}(+)	7.48×10^{-5}(+)	5.42×10^{-5}(+)
	标准差	4.52×10^{-5}	4.09×10^{-5}(+)	2.43×10^{-3}(+)	1.17×10^{-3}(+)	7.07×10^{-5}(+)	4.56×10^{-5}(+)
F8	平均值	0	9.88×10^{-1}(+)	0(=)	2.91(+)	0(=)	0(=)
	标准差	0	5.41×10(+)	0(=)	3.69×10(+)	0(=)	0(=)
F9	平均值	8.88×10^{-16}	8.88×10^{-16}(=)	4.20×10^{-15}(+)	1.01×10^{-13}(+)	8.88×10^{-16}(=)	8.88×10^{-16}(=)
	标准差	0	0(=)	1.85×10^{-15}(+)	1.95×10^{-14}(+)	0(=)	0(=)
F10	平均值	0	0(=)	1.80×10^{-2}(+)	3.71×10^{-3}(+)	0(=)	0(=)
	标准差	0	0(=)	6.07×10^{-2}(+)	7.28×10^{-3}(+)	0(=)	0(=)
F11	平均值	4.73×10^{-21}	1.28×10^{-1}(+)	2.69×10^{-2}(+)	4.85×10^{-2}(+)	1.05×10^{-5}(+)	9.61×10^{-1}(+)
	标准差	2.59×10^{-20}	5.53×10^{-2}(+)	2.83×10^{-2}(+)	2.50×10^{-2}(+)	7.06×10^{-6}(+)	2.15×10^{-1}(+)
F12	平均值	2.05×10^{-9}	2.07×10(+)	4.42×10^{-1}(+)	5.81×10^{-1}(+)	1.48×10^{-4}(+)	3.00×10(+)
	标准差	1.12×10^{-8}	3.44×10^{-1}(+)	2.06×10^{-1}(+)	2.04×10^{-1}(+)	1.85×10^{-4}(+)	2.53×10^{-3}(+)
F13	平均值	3.10×10^{-4}	3.15×10^{-4}(=)	7.19×10^{-4}(+)	5.12×10^{-3}(+)	3.08×10^{-4}(−)	1.18×10^{-2}(+)
	标准差	2.26×10^{-6}	1.83×10^{-5}(+)	4.42×10^{-4}(+)	8.56×10^{-3}(+)	4.36×10^{-7}(−)	1.44×10^{-2}(+)
F14	平均值	-1.05×10	-9.95×10(+)	-6.29×10(+)	-9.99×10(+)	-1.05×10(=)	-5.42×10(−)
	标准差	3.08×10^{-5}	1.79×10(+)	3.37×10(+)	2.06×10(+)	3.08×10^{-5}(=)	2.08×10(+)
Wilcoxon 检验		+/−/=	10/4/0	11/3/0	13/1/0	8/5/1	8/5/1
Fri×10ndman 检验		1.79	3.61	4.46	4.82	2.75	3.57
Friendman 检验排名		1	4	5	6	2	3

在单峰函数F6、多峰函数F8～F10上MSCZOA算法与其他部分算法产生的实验数据相同，此时无法通过平均值与标准差来对MSCZOA算法的性能进行分析。MSCZOA算法与其他算法在上述14种基准测试函数的收敛曲线如图2所示，在收敛速度方面进一步对算法进行评价。

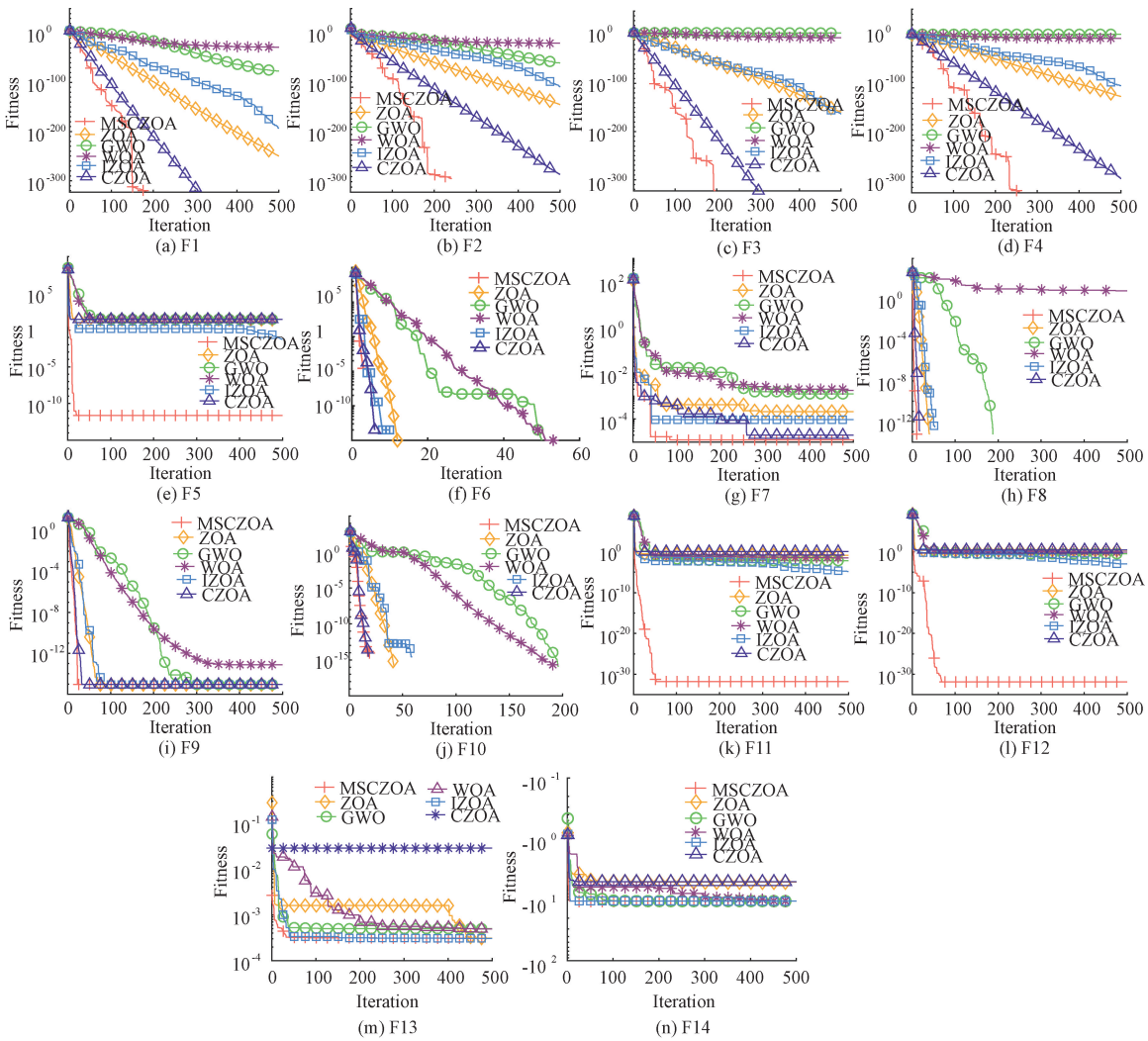

图2 MSCZOA与其他算法的收敛图

Fig. 2 Convergence plot of MSCZOA and other algorithms

由图2可以看出，MSCZOA算法在各函数的收敛速度上均优于原代码与其他算法。另外，虽然在单峰函数F6、多峰函数F8、F9、F10上，MSCZOA算法的平均值与标准差与其他部分算法数据相同，但是MSCZOA与其他算法相比，更早寻得理论最优值，说明多策略融合有效地提升了算法的收敛速度。

3.3 消融实验

为验证本文提出的改进策略对斑马优化算法改进的有效性与影响程度。本文采用多策略融合的斑马算法与分别根据3种改进策略单独改进的斑马算法进行消融实验，以验证改进策略对斑马算法性能提升的有效性。实验采用上述14个基准测试函数进行实验，结果如表4所示。其中MSCZOA算法为多策略融合的斑马优化算法，LZOA为采用基于Logistics混沌收敛因子改进的斑马优化算法，SZOA为采用位置扰动策略改进的斑马优化算法，PZOA为采用记忆更新策略的斑马优化算法。

由表4可知，从平均值与标准差综合方面上评价，3种单策略改进斑马优化算法在上述14个函数中有11个函数得到的实验数据明显优于原代码，说明3种改进策略有效地提升了算法地性能。在单峰函数F1～F7上，SZOA在7个单峰函数中有5个函数寻得了理论最优解，与其他单策略改进的斑马算法相比，得到的数据更优良，并且该策略数据与MSCZOA算法得到得数据相同，说明了位置扰动策略有效地提升了算法在单峰函数上的性能，在提升算法寻优能力方面有着较强效果。在多峰函数F8～F12上，3种单策略改进的斑马优化算法在函数F8～

表 4 MSCZOA 与单策略改进算法的实验结果

Table 4 Experimental results of MSCZOA and single-strategy improved algorithm

函数		MSCZOA	ZOA	LZOA	SZAO	PZOA
F1	平均值	0	5.96×10^{-252}(+)	0(=)	0(=)	0(=)
	标准差	0	0(=)	0(=)	0(=)	0(=)
F2	平均值	0	1.58×10^{-131}(+)	1.60×10^{-187}(+)	0(=)	6.21×10^{-234}(+)
	标准差	0	7.79×10^{-131}(+)	0(=)	0(=)	0(=)
F3	平均值	0	6.11×10^{-156}(+)	1.75×10^{-304}(+)	0(=)	0(=)
	标准差	0	3.33×10^{-155}(+)	0(=)	0(=)	0(=)
F4	平均值	0	7.20×10^{-115}(+)	4.88×10^{-189}(+)	0(=)	1.02×10^{-221}(+)
	标准差	0	1.51×10^{-114}(+)	0(=)	0(=)	0(=)
F5	平均值	0	2.82×10(+)	1.23×10^{-7}(+)	2.02×10^{-6}(+)	3.36×10^{-3}(+)
	标准差	0	5.82×10^{-1}(+)	4.91×10^{-7}(+)	1.01×10^{-5}(+)	1.84×10^{-2}(+)
F6	平均值	0	0(=)	0(=)	0(=)	0(=)
	标准差	0	0(=)	0(=)	0(=)	0(=)
F7	平均值	3.16×10^{-5}	7.68×10^{-5}(+)	4.50×10^{-5}(+)	3.19×10^{-5}(+)	2.93×10^{-5}(−)
	标准差	2.66×10^{-5}	4.09×10^{-5}(+)	3.69×10^{-5}(+)	3.17×10^{-5}(+)	2.20×10^{-5}(−)
F8	平均值	0	9.88×10^{-1}(+)	0(=)	0	0
	标准差	0	5.41×10(+)	0(=)	0	0
F9	平均值	8.88×10^{-16}	8.88×10^{-16}(=)	8.88×10^{-16}(=)	8.88×10^{-16}(=)	8.88×10^{-16}(=)
	标准差	0	0(=)	0(=)	0(=)	0(=)
F10	平均值	0	0(=)	0(=)	0(=)	0(=)
	标准差	0	0(=)	0(=)	0(=)	0(=)
F11	平均值	1.57×10^{-32}	1.28×10^{-1}(+)	1.09×10^{-25}(+)	6.90×10^{-19}(+)	1.57×10^{-32}(=)
	标准差	5.57×10^{-48}	5.53×10^{-2}(+)	5.95×10^{-25}(+)	3.78×10^{-18}(+)	5.57×10^{-48}(=)
F12	平均值	3.33×10^{-32}	2.07×10(+)	2.25×10^{-9}(+)	9.85×10^{-22}(+)	1.62×10^{-25}(+)
	标准差	1.08×10^{-31}	3.44×10^{-1}(+)	1.23×10^{-8}(+)	5.39×10^{-21}(+)	8.85×10^{-25}(+)
F13	平均值	3.08×10^{-4}	3.15×10^{-4}(+)	3.08×10^{-4}(=)	3.08×10^{-4}(=)	3.08×10^{-4}(=)
	标准差	3.46×10^{-7}	1.83×10^{-5}(+)	4.66×10^{-7}(=)	8.14×10^{-7}(+)	3.89×10^{-7}(+)
F14	平均值	-1.05×10	-9.95×10(+)	-1.05×10(=)	-1.05×10(=)	-1.05×10(=)
	标准差	2.78×10^{-5}	1.79×10(+)	2.42×10^{-5}(−)	2.63×10^{-5}(−)	2.42×10^{-5}(−)

F10 上得到的数据相同，但是在其他函数上 PZOA 取得的平均值与标准差均优于其他单一策略改进的斑马优化算法，并且在固定维度的多峰函数 F13、F14 上，虽然 PZOA 得到的平均值与其他改进算法相同，但是其标准差低于其他算法的标准差，说明 PZOA 的稳健性优于其他算法，因此综上验证了记忆更新策略对提高算法的全局探勘与局部开发能力有明显效果。LZOA 仅在函数 F5、F14 上的实验数据优于其他改进算法，但是 LZOA 与 SZOA、PZOA 相比，无论在提升算法的寻优能力，还是提升算法全局探勘与局部开发能力都没有明显的短板，因此引入 Logistics 混沌收敛因子改进一策略有利于全方面提高算法性能。从多策略融合的斑马算法的性能角度进行评价，在上述实验数据中，除在函数 F7、F14 外，MSCZOA 算法的平均值与标准差均小于等于其他改进算法。说明多策略的融合使得斑马优化算法结合了各个改进策略的优点，提高了算法的性能，并依靠不同策略的效果，中和了各个改进策略的缺点，避免了改进策略缺点对算法的影响。

3.4 统计检验

仅根据平均值、标准差和收敛曲线无法分析 MSCZOA 算法与其他算法每次运行算法的显著区别。本文采用 Wilcoxon 秩和检验与 Friedman 检验分别从逐一检验与综合检验两个方面基于算法 30 次运行的数据结果对算法差异性与优良性进行验证。其中 Wilcoxon 秩和检验，实验设定的显著差异(p 值)设定为 5%，当 p 值小于 5%

时，表示 MSCZOA 算法与该算法存在显著差异，反之则说明两算法相近。实验结果如表 3 所示，其中“+”、“−”、“=”分别表示 MSCZOA 算法优于、劣于和等于对比算法。根据 wilcoxon 秩和检验实验结果 MSCZOA 算法与对比算法在大部分函数所得到的检验 p 值均小于 5%，并在各函数上的性能表现中，MSCZOA 算法均以超过 50% 函数以上的性能表现超越对比算法，说明了算法相比于其他算法具有更优的性

Wilcoxon 秩和检验验证了 MSCZOA 算法与其他算法具有显著区别。为了进一步检验 MSCZOA 算法与其他算法的差异性与优良性，在综合方面对 MSCZOA 算法与其他算法再进行 friedman 检验，并根据检验结果对其性能优异进行排序。由表 3 可知，friedman 检验结果所示，MSCZOA 算法与其他算法相比，其性能排名第一并且所求得的秩平均值与其他算法求得的秩平均值存在显著差异，进一步说明了 MSCZOA 算法具有优秀的寻优性能。

3.5 时间复杂度分析

首先定义 ZOA 算法中种群规模为 N，维度为 J，最大迭代次数为 T，求个体适应度的时间为 $T(J)$，初始化种群所需时间为 a_1，阶段 1 更新种群个体每一维所需时间为 a_2，阶段 2 中第 1 种更新策略更新种群个体数为 $Pnum$，每一维所需时间为 a_{31}，阶段 2 中第 2 种更新策略更新种群个体数量为 $N-Pnum$，每一维所需时间为 a_{32}。因此种群初始化阶段及计算个体适应度的时间复杂度为 $T_1=O(N\cdot(T\cdot T(J)+J\cdot a_1))$，阶段 1 更新的时间复杂度为 $T_2=O(N\cdot T\cdot J\cdot a_2)$，阶段 2 更新的时间复杂度为 $T_3=O(Pnum\cdot T\cdot J\cdot a_{31}+(N-Pnum)\cdot T\cdot J\cdot a_{32})$，综上，ZOA 算法的时间复杂度为 $T=T_2+T_3+(T_1)$。

MSCZOA 算法的时间复杂度，首先定义种群规模为 N，维度为 J，最大迭代次数为 T，求个体适应度的时间为 $T(J)$，初始化种群所需时间为 a_1'，阶段 1 因位置扰动策略将阶段 1 分为两种更新形式，因此定义根据引入 Logistics 混沌因子的更新策略更新种群个体数量为 $Pnum1$，每一维所需时间为 a_{21}'，根据位置扰动策略更新种群个体数量为 $N-Pnum1$ 每一维所需时间为，阶段 2 第 1 种更新策略并没有发生改变，更新种群个体数量为 $Pnum2$ 每一维所需时间仍为 a_{31}，根据记忆更新策略更新种群个体数量为 $N-Pnum2$，每一维所需时间为 a_{32new}。MSCZOA 初始化种群及求个体适应度没有发生变化，因此时间复杂度为 $T_1'=O(N\cdot(T\cdot T(J)+J\cdot a_1'))$；阶段 1 更新个体位置公式有变化但是种群整体在此更新阶段发生更新次数并不发生改变，因此该阶段种群复杂度为 $T_2'=O(Pnum1\cdot(T\cdot a_{21}'\cdot J)+((N-Pnum1)\cdot T\cdot a_{22}'\cdot J))$，阶段 2 的更新策略没有改变种群更新次数，因此此阶段算法的时间复杂度为 $T_3'=(Pnum2\cdot(T\cdot a_{31}'\cdot J)+((N-Pnum2)\cdot(T\cdot a_{32}'\cdot J)$，综上 MSCZOA 的时间复杂度为 $T=T_2'+T_3'+(T_1)$。因此说明 MSCZOA 的优越性没有以增加算法的时间复杂度为代价。

4 工程优化问题应用

4.1 拉压弹簧设计问题

拉压弹簧设计问题作为优化工程应用问题中的经典案例，其主要目的是在一定约束条件下控制 3 个变量使重量最小化，其结构模型如图 3 所示。

图 3 压力弹簧模型

Fig. 3 Pressure spring model

该问题主要包括弹簧的横截面的直径 $d(x_1)$、弹簧圈的平均直径 $D(x_2)$以及弹簧的有效圈数 $N(x_3)$3 个设计变量以及最小挠度、剪切应力、振荡频率以及外径限制 4 个不等式约束。

拉压弹簧的 3 个属性用 x_1、x_2、x_3 表示。

$$X=[x_1,x_2,x_3]=[d,D,N] \tag{11}$$

目标函数为：

$$\min\ f(x)=(x_3+2)\cdot x_2\cdot x_1^2 \tag{12}$$

决策变量及取值范围：

$$\begin{cases}0.05\leqslant x_1\leqslant 2\\0.25\leqslant x_2\leqslant 1.3\\2\leqslant x_3\leqslant 15\end{cases} \tag{13}$$

约束条件公式为：

$$\begin{cases}g_1(X)=1-\dfrac{x_2^3\cdot x_3}{717\ 854x_1^4}\leqslant 0\\g_2(X)=\dfrac{4x_2^2-x_1x_2}{12\ 566(x_2\cdot x_1^3-x_1^4)}+\dfrac{1}{5\ 108x_1^2}\leqslant 0\\g_3(X)=1-\dfrac{140.45x_1}{x_2^2\cdot x_3}\leqslant 0\\g_4(X)=\dfrac{x_1+x_2}{1.5}-1\leqslant 0\end{cases} \tag{14}$$

为了检验改进后的斑马优化算法对于工程优化问题的有效性，将 MSCZOA 算法与 ZOA 算法、GWO 算法、WOA 算法、IZOA 算法和 CZOA 算法进行对比分析。6 种算法对拉压弹簧设计问题的实验结果如表 5 所示。

根据表 5 数据可知，MSCZOA 算法与 GWO 算法所得寻优结果相近且均为在拉压弹簧设计问题中实验算法所得的最优解，同时与原算法 ZOA 相比所得寻优结果有所提升，因此说明 MSCZOA 在求解复杂问题时比其他算法具有更高的稳定性和寻优性能，所使用的 Logistics 混沌因子与记忆更新策略很好的帮助了斑马粒子逃离局部最优解，而局部扰动测量则提高了斑马粒子的种群多样

表 5 弹簧设计问题结果

Table 5 Results of spring design problems

算法	x_1	x_2	x_3	$f(x)$
MSCZOA	0.052 40	0.374 17	10.342 09	0.012 682
ZOA	0.054 76	0.435 31	7.828 74	0.012 832
GWO	0.050 00	0.317 20	14.060 00	0.012 682
WOA	0.053 28	0.396 23	9.299 80	0.012 883
IZOA	0.052 88	0.386 18	9.750 04	0.012 691
CZOA	0.054 126	0.418 193	8.470 718	0.012 828

性,在局部搜索的精度与速度上发挥了重要作用。

4.2 齿轮设计问题

齿轮设计问题是机械工程中的一个实际优化问题,其没有约束条件且结构相对简单。其目的是控制 4 个齿轮的齿轮数以实现减少特定传动成本。其结构模型如图 4 所示。

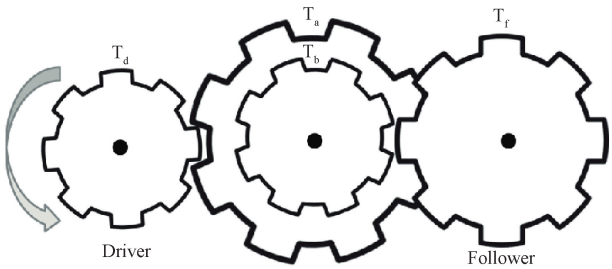

图 4 齿轮设计模型

Fig. 4 Gear design model

该问题主要包括齿轮的齿轮数 $T_a(x_1)$、$T_b(x_2)$、$T_d(x_3)$、$T_f(x_4)$4 个设计变量。

齿轮系设计的 4 个属性用 x_1、x_2、x_3、x_4 表示。

$$X=[x_1,x_2,x_3,x_4]=[T_a,T_b,T_d,T_f] \tag{15}$$

决策变量及取值范围:

$$\begin{cases}12\leqslant x_1\leqslant 60\\12\leqslant x_2\leqslant 60\\12\leqslant x_3\leqslant 60\\12\leqslant x_4\leqslant 60\end{cases} \tag{16}$$

目标函数为:

$$\min\ f(x)=\left(\frac{1}{6.931}-\frac{x_2x_3}{x_1x_4}\right)^2 \tag{17}$$

仍采用 4.1 节选用的其他 5 种算法与 MSCZOA 算法进行对比分析。6 种算法对齿轮设计问题的实验结果如表 6 所示。

根据表 6 数据可知,MSCZOA 算法在 $T_a=48.527$、$T_b=19.391$、$T_d=16.104$、$T_f=43.223$ 处寻得最低成本 $f=2.70\times10^{-12}$,数据均优于与其他 4 个算法所得数据,表明 MSCZOA 算法能够有效处理齿轮系设计问题,在搜索过程中凭借良好的搜索能力寻得最优解。

根据上述两个工程设计问题,再次验证了 TZOA 所具有的优越性能,能够解决一些复杂的实际工程问题。特别的,根据在基准测试函数与对于无约束条件的实际优化问题的实验中,与其他算法相比,MSCZOA 算法表现出较优秀的寻优能力与寻优速度,在未来该算法可以进一步考虑应用于该类工程优化领域。

表 6 齿轮系设计问题结果

Table 6 Results of gear train design problems

算法	x_1	x_2	x_3	x_4	$f(x)$
MSCZOA	48.527	19.391	16.104	43.223	2.70×10^{-12}
ZOA	52.923	25.999	14.690	51.437	2.31×10^{-11}
GWO	22.842	12.000	13.239	46.804	9.92×10^{-10}
WOA	34.196	20.390	13.071	52.675	2.31×10^{-11}
IZOA	52.849	15.015	25.721	50.832	2.31×10^{-11}
CZOA	27.505	12.331	12.453	29.832	0.000 737

5 结 论

本文通过对 ZOA 算法的研究,针对 ZOA 算法难以跳出局部最优解、鲁棒性差等缺点,提出了一种多策略融合的斑马优化算法。引入基于 Logistics 混沌收敛因子,平衡了算法迭代前后期的全局与局部搜索能力;引入位置扰动策略,提高了算法迭代过程的多样性;引入记忆更新策略,提高了算法的收敛速度。本文基于 14 种基准测试函数,将改进后算法与其他算法进行对比实验并通过消融实验及时间复杂度分析,证明改进后就算法收敛精度、速度更优,并且具有跳出局部最优的能力,并通过统计检验验证了 MSCZOA 算法与其他算法的差异性。并将 MSCZOA 算法应用于工程应用问题,进一步验证改进策略的有效性与算法的实用性,对未来 MSCZOA 算法在更多工程应用领域提供了研究扩展性。

参 考 文 献

[1] 李雅丽,王淑琴,陈倩茹,等. 若干新型群智能优化算法的对比研究[J]. 计算机工程与应用,2020,56(22):1-12.
LI Y L, WANG SH Q, CHEN Q R, et al. Comparative study of several new swarm intelligence optimization algorithms [J]. Computer Engineering and Applications, 2020, 56(22): 1-12.

[2] RAO R V, SAVSANI V J, VAKHARIA D P. Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems[J]. Computer-Aided Design, 2011, 43: 303-315.

[3] TROJOVSKÝ P. A new human-based metaheuristic algorithm for solving optimization problems based on preschool education[J]. Scientific Reports, 2023,13: 21472-21503.

[4] LAITH A, ALI D, SEYEDALI M, et al. The arithmetic optimization algorithm [J]. Computer

Methods in Applied Mechanics and Engineering, 2021,376:113609-113648.

[5] MIRJALILI S, SEYED M M, ANDREW L. LEWIS. Grey wolf optimizer [J]. Advancs in Engineering Software,2014,69: 46-61.

[6] XUE J, SHEN B. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization[J]. The Journal of Supercomputing, 2023, 79(7): 7305-7336.

[7] 万怡华,张雪梅.混合多策略改进蜣螂算法的避障路径规划[J].电子测量技术,2024,47(2):69-78.
WAN Y H, ZHANG X M. Obstacleavoidancepath planning of hybrid multi-strategy improved dung beetle optimizer [J]. Electronic Measurement Technology,2024,47(2):69-78.

[8] 陈敏,陈晔,牛兴龙,等.求解全局优化问题的多策略改进灰狼算法[J].国外电子测量技术,2022,41(11):22-29.
CHEN M, CHEN Y, NIU X L, et al. Multi-strategy improved grey wolf algorithm for solving global optimization problems [J]. Foreign Electronic Measurement Technology, 2022, 41(11): 22-29.

[9] 马学森,谈杰,陈树友,等.云计算多目标任务调度的优化粒子群算法研究[J].电子测量与仪器学报,2020,34(8):133-143.
MA X S, TAN J, CHEN SH Y, et al. Research on optimization particle swarm algorithm for multi-objective task scheduling in cloud computing [J]. Journal of Electronic Measurement and Instrumentation, 2020, 34(8): 133-143.

[10] 李珺,徐秦.面向特征选择任务的改进蜣螂优化算法[J]电子测量技术,2024,47(1):79-86.
LI J, XU Q. An improved scarab optimization algorithm for feature selection task [J]. Electronic Measurement Technology, 2024, 47(1): 79-86.

[11] 邢燕好,于昊,张佳,等.基于粒子群参数优化的O-VMD数据处理方法研究[J].仪器仪表学报,2023,44(4):304-313.
XING Y H, YU H, ZHANG J, et al. Research on O-VMD data processing method based on parameter optimization of particle subgroups [J]. Chinese Journal of Scientific Instrument, 2019, 44(4): 304-313.

[12] TROJOVSKÁ E, DEHGHANI M, TROJOVSKY P. Zebra optimization algorithm: A new bio-inspired optimization algorithm for solving optimization algorithm[J]. IEEE Access, 2022,10:49445-49473.

[13] ZARE P, DAVOUDKHANI I F, ZARE R, et al. Maiden application of zebra optimization algorithm for design pidn-tidf controller for frequency control in offshore fixed platforms microgrid in the presence of tidal energy [C]. 8th International Conference on Technology and Energy Management (ICTEM), 2023:1-7.

[14] HE W M, MA S Y, WANG C S, et al. Improve zebra optimization algorithm with adaptive oscillation weight and golden sine operator[DB/OL]. (2024-02-04)[2024-06-11].

[15] 苏晨,王防修,黄淄博.融合正切搜索与竞争交配的斑马优化算法及应用[J/OL].计算机科学与探索,1-21[2024-08-29].
SU CH, WANG F X, HUANG Z B. Zebra optimization algorithm of fusion tangent search and competitive mating and its application [J/OL]. Journal of Computer Science and Exploration, 1-21[2024-08-29].

[16] DAMA A, KHALAF O I, CHANDRA G R. Enhancing the zebra optimization algorithm with chaotic sinusoidal map for versatile optimization[J]. Iraqi Journal For Computer Science and Mathematics, 2024, 5(1): 307-319.

[17] ZHANG X Q, ZHANG Y Y, MING Z F. Improved ynamic grey wolf optimizer [J]. Frontiers of Information Technology & Electronic Engineering, 2021,22(6):877-891.

[18] MIRJALILI S, LEWIS A. The whale optimization algorithm [J]. Advances in Engineering Software, 2016,95: 51-67.

作者简介

黄子介,硕士研究生,主要研究方向为智能优化算法及应用。
E-mail:2376023117@qq.com

代永强(通信作者),博士,教授,硕士生导师,主要研究方向为数据挖掘与机器学习、生物信息计算与数字健康。
E-mail:dyq@gsau.edu.cn