文章编号：2095-2163(2020)09-0043-07 中图分类号：TP18 文献标志码：A

# 基于混沌的正余弦鲸鱼优化算法

林 杰，何 庆，王 茜，杨荣莹，宁杰琼
（贵州大学 大数据与信息工程学院，贵阳 550025）

**摘 要：**为了改善鲸鱼优化算法(WOA)的不足，如容易陷入局部最优，收敛速度慢等问题，本文提出了改进鲸鱼优化算法。首先，通过混沌 Tent 映射随机生成算法的初始种群位置，让种群分布更均匀，加快算法的收敛速度；其次，将正余弦算法与鲸鱼优化算法融合，对领导者位置进行筛选，一定程度上避免了算法容易早熟的缺陷；最后，提出自适应策略，保留了鲸鱼优化算法优越性的同时，加入惯性权重，以平衡算法全局探测和局部寻优的能力。实验部分通过对基准函数仿真、使用 Wilcoxon 检验和 MAE 等方法来评价改进算法的性能。仿真结果表明：本文所提改进算法具有良好的有效性和优越性。

**关键词：**鲸鱼优化算法；混沌 Tent 映射；正余弦算法；自适应策略

## Chaos-based sine-cosine whale optimization algorithm

LIN Jie, HE Qing, WANG Qian, YANG Rongying, NING Jieqiong
(College of Big Data and Information Engineering, Guizhou University, Guiyang 550025, China)

**【Abstract】** A chaos-based sine-cosine whale optimization algorithm has been proposed to address the shortcomings of whale optimization algorithms (WOA) such as premature maturation and slow convergence speed. First, the initial population and random parameters in the algorithm are generated by using chaotic Tent mapping to accelerate the algorithm´s convergence speed.Secondly, the sine and cosine algorithm is fused with the whale optimization algorithm, and the leader position is filtered to avoid the shortcomings of the algorithm. In the end, an adaptive weight is added to the algorithm to tune the algorithm´s global search capabilities and local development capabilities while retaining the benefits of the whale optimization algorithm. The performance of the improved whale optimization algorithm is evaluated by simulation experiments on 10 benchmark functions, Wilcoxon tests, MAE and other methods. Experimental results show that the improved algorithms all have good optimization performance, which proves that the proposed improved whale optimization algorithm has certain effectiveness.

**【Key words】** whale optimization algorithm; chaotic Tent mapping; sine and cosine algorithm; adaptive weights

## 0 引 言

为解决各个科学工程中所产生的非线性优化问题，许多自然启发算法越来越受青睐，其实现过程简单且易于执行，而且这些算法绕过局部最优区域的能力也相对较强。其中，鲸鱼优化算法(Whale Optimization Algorithm, WOA)是 Mirjalili 等人 2016 年提出的一种基于群体的智能优化算法，其仿生原理为座头鲸的捕猎行为。由于鲸鱼优化算法表现出优秀的寻优性能，WOA 已经被应用于许多优化问题中：将改进的鲸鱼算法应用在炼钢连铸的调度问题中[1]；在水库优化调度中采用鲸鱼优化算法[2]；在关于电压的传输过程中，通过鲸鱼算法对 LLC 谐振变换器进行建模和优化[3]。但与大多数启发式算法一样，也是从一个随机的种群开始搜索，搜索过程有全局探索阶段和局部开发阶段。由于最优过程的随机性，在勘探和开发之间保持适当的平衡是极其困难的，而且在算法的搜索过程中不断改变领导者的位置，导致收敛结果早熟，即在求解优化问题时可能会迅速收敛到局部最优而不是全局最优，最终使解决方案的质量下降。许多学者对 WOA 的寻优性能进行了一系列的改进，如：基于自适应权重和柯西变异的鲸鱼优化算法具有较好的寻优精度和稳定

**基金项目：**贵州省科技计划项目重大专项项目(黔科合重大专项字[2018] 3002；黔科合重大专项字[2016] 3022)；贵州省公共大数据重点实验室开放课题(2017BDKFJJ004)；贵州省教育厅青年科技人才成长项目(黔科合 KY 字[2016]124)；贵州大学培育项目(黔科合平台人才[2017] 5788)。

**作者简介：**林 杰(1995-)，女，硕士研究生，主要研究方向：智能优化算法、图像处理、数据挖掘；何 庆(1982-)，男，博士，副教授，主要研究方向：计算机应用技术、智能算法、大数据应用等；王 茜(1995-)，女，硕士研究生，主要研究方向：智能优化算法、数据挖掘；杨荣莹(1995-)，女，硕士研究生，主要研究方向：智能优化算法、自然语言处理；宁杰琼(1997-)，女，硕士研究生，主要研究方向：智能优化算法、图像处理。

**通信作者：**何 庆 Email：qhe@ gzu.edu.cn

**收稿日期：**2020-06-26

性[4];混合遗传鲸鱼优化算法来优化频谱利用率和对抗恶意用户模拟授权用户的攻击行为,仿真结果表明,该算法比现有的检测算法具有更高的寻优性能[5]。

为进一步改善 WOA 的寻优性能,本文提出了一种改进鲸鱼算法(Enhanced Whale Optimization Algorithm, EWOA)。首先,将引入混沌理论初始化算法种群位置,并对算法中随机参数进行优化,以加快算法的收敛速度;其次,将正余弦算法与鲸鱼优化算法结合,结合正余弦算法的优点对鲸鱼群领导位置进行筛选,提高算法跳出陷入局部最优的能力,进而增加算法的寻优精度;最后,融合本文提出的自适应策略,引入惯性权重,以便更好的平衡算法的全局搜索和局部开发。算法仿真结果证明,本文提出的算法具有优越性和竞争性。

## 1 标准鲸鱼优化算法

WOA 算法是受启发于座头鲸的捕猎方法,为了捕食海洋表面的小鱼,座头鲸群潜到海面下,在猎物周围制造不同形状的泡泡,然后其会以螺旋路径游回到水面捕食,这种独特的捕猎方法使其不同于世界上的其它生物。

### 1.1 收缩包围

在 WOA 算法中,一头鲸鱼的位置代表一个空间解,随着迭代次数的增加,鲸鱼朝着猎物位置移动,不断更新位置,逐渐将猎物包围起来。这种行为用数学公式(1)表示:

$$X(t+1) = X^*(t) - A \cdot D. \tag{1}$$

其中,$D = |C \cdot X^*(t) - X(t)|$;$X(t)$ 表示鲸鱼在第 $t$ 代的位置;$X^*(t)$ 表示猎物在第 $t$ 代的位置向量;$A$、$C$ 为变量,定义为式(2) 和式(3):

$$A = 2ar_1 - a, \tag{2}$$

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

其中,$r_1$、$r_2$ 是服从[0,1] 之间均匀分布的随机数,$a$ 是线性递减的向量,$a$ 的最大值为 2,最小值为 0,定义为式(4):

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

其中,$T_{\max}$ 为最大迭代次数,本文取值 $T_{\max} = 1\,000$。

### 1.2 位置更新

为了模拟鲸鱼捕食行为,通过两种方式建立数学模型:收缩包围机制和螺旋更新位置。随迭代次数的增加, 通过式(4) 的线性递减模型将 $a$ 从 2 递减到 0,实现收缩包围机制,由式(2) 可知,$A$ 是取值范围为[$-a$, $a$] 的随机值,当 $a = 1$ 时,由式(1) 可知鲸鱼第 $t+1$ 代的位置 $X(t+1)$ 在第 $t$ 代鲸鱼的位置 $X(t)$ 和猎物的位置 $X*(t)$ 之间徘徊,由此实现对猎物的收缩包围。为了模仿鲸鱼的螺旋运动轨迹,猎物和鲸鱼的位置之间的螺旋位置更新方程为(5):

$$X(t+1) = D' \cdot e^{bl} \cdot \cos(2\pi l) + X^*(t). \tag{5}$$

其中,$D' = |X^*(t) - X(t)|$,表示猎物与鲸鱼之间的距离;$b$ 为螺旋常数;$l$ 为随机数,本文 $b$ 取值为 1,$l$ 取值范围是[$-1$,1]。随着迭代次数的变化,鲸鱼绕着猎物包围的圈越来越小,同时沿着螺旋形的路径前进。为了更好的模拟鲸鱼的捕食行为,假设鲸鱼选择收缩包围方式和螺旋模型行进方式进行位置更新的几率各占一半,即选择阈值为 0.5。数学模型为(6):

$$X(t+1) = \begin{cases} X^*(t) - A \cdot D & p < 0.5 \\ D' \cdot e^{bl} \cdot \cos(2\pi l) + X^*(t) & p > 0.5 \end{cases} \tag{6}$$

其中,$p$ 为服从[0,1] 之间均匀分布的概率因子。

### 1.3 随机搜索

为提高搜索猎物的全局勘探能力,鲸鱼可以采用随机搜索的方式搜索猎物。当 $A$ 满足 $|A| \geq 1$ 时,鲸鱼彼此之间的距离 $D_{rand}$ 随机更新,此时鲸鱼会放弃之前的移动方向,随机在向其他方向搜索新的更新位置,避免陷入局部极值。位置更新如公式(7)和公式(8):

$$D_{rand} = |C \cdot X_{rand}(t) - X(t)|, \tag{7}$$

$$X(t+1) = X_{rand}(t) - A \cdot |C \cdot X_{rand}(t) - X(t)|. \tag{8}$$

其中,$X_{rand}$ 表示随机选择的鲸鱼的位置向量。

## 2 改进的鲸鱼优化算法

### 2.1 混沌初始化种群和参数优化

混沌可以定义为这样一种现象:初始状态任何微小的变化都可能导致未来行为的非线性变化,此外,它在数学上被定义为非线性确定性系统产生的半随机行为,依赖于混沌运动具有的遍历性、随机性等特点,可用来显著提高算法的寻优性能[6]。大多数的群智能算法根据不同的概率分布产生一系列(0,1)之间的随机数,而这种随机性可能会导致算法收敛速度变慢,往往会影响目标解的质量。当然,与大多数启发式算法一样,标准 WOA 算法也是从一个随机的种群开始搜索,这种方式让初始化种群缺乏种群多样性,寻优的空间位置不够广泛。为了

改善这个缺点，本文采用混沌序列产生初始种群并替换算法中的种群位置。

Tent 映射初始化种群位置的时候，算法寻到的最优解和收敛速度明显比其他混沌映射要好，同时也证明了鲸鱼优化算法对 Tent 映射的敏感度更高[7]。因此，本文通过式(9)，采用 Tent 映射的方式来生成分布更加均匀的混沌序列，提高了算法的寻优性能。

$$x_{i+1}=\begin{cases}2x_i, & x_i<0.5\\ 2(1-x_i). & x_i\geqslant 0.5\end{cases} \tag{9}$$

### 2.2　正余弦鲸鱼优化算法

正余弦算法(Sine cosine algorithm, SCA)是近年来提出的一种新的全局优化算法，有别于其他受生物启发机理的群体智能优化算法，其结简单、鲁棒性好、容易实现，主要利用正弦函数和余弦函数的数学性质，通过迭代达到寻找最优解的目的。在 SCA 中，假设第 $t$ 代种群中个体 $X_i^t=(x_{i1}^t,x_{i2}^t,\cdots,x_{iD}^t)$，其中，$i=1,2,\cdots,N$，$N$ 为种群大小，$D$ 为个体维度。算法在空间中随机生成 $N$ 个种群的位置，计算每个个体的适应度值，通过对适应度值排序存优，保存最优位置及其对应的适应度值，个体更新位置方式为(10)：

$$X_d^i(t+1)=\begin{cases}X_d^i(t)+a\times\sin(r_3)\times|r_4X^*-X_d^i(t)|, & r_5<0.5\\ X_d^i(t)+a\times\cos(r_3)\times|r_4X^*-X_d^i(t)|. & r_5\geqslant 0.5\end{cases} \tag{10}$$

其中，$X_d^i$ 是第 $i$ 个个体第 $t$ 代第 $d$ 维的位置分量；$X^*$ 是当前最优位置；参数 $a$ 控制搜索方向，变化方式同式(4)，$r_3$ 为[0,2$\pi$]上的随机数，用来控制算法的搜索距离；$r_4$ 为[0,2]上的随机数；$r_5$ 为[0,1]上的随机数，决定第 $t+1$ 代的位置更新方式是正弦方式还是余弦方式。

考虑到在鲸鱼优化算法中，每次迭代时将适应度值最好的位置赋值给鲸鱼群领导者，这样的方式导致算法容易陷入局部最优区域，往往导致寻优精度较低。而在 SCA 中，算法可以随机选取正余弦交叉寻优，使得二者位置更新方式相互补充，更好的协调全局探索和局部开发，使 SCA 逐步收缩并徘徊在目标解附近。因此，在本文所提的正余弦鲸鱼优化算法中，经过排序存优的领导者不是直接进行下一次迭代过程，而是记录当前领导者位置，同时对种群中每个个体的位置根据式(10)进行正弦(余弦)位置更新，再计算每个个体的适应度值，通过比较适应度筛选出一个新的领导者位置，最后引入贪婪机制，比较正余弦操作之前的领导者和新的领导者之间的适应度值大小，更新全局最优位置，再进入到下一次的迭代过程。值得注意的是，对于在 SCA 中出现的随机参数 $r_3$、$r_4$、$r_5$，本文都通过 Tent 混沌映射产生，一定程度上加快了算法的收敛速度。

### 2.3　混沌自适应的惯性权重

在当前大多数群智能优化算法中，权重对算法的全局探索和局部开发能力的平衡扮演着重要的角色。重反映的是后一个追随者摆脱前一个位置束缚的能力，通常较大的惯性权重能够让算法具有较好的全局探索能力，而较小的惯性权重会使算法具有较好的局部开发能力。在算法迭代早期应使用较大的权重，让算法以较大步伐快速到达目标值附近；而在迭代后期，应使用较小的权重，让鲸鱼较小的步伐移动，以便在目标解的附近精确搜索，更好的局部寻优。标准 WOA 算法在搜索包围和位置更新时，权重是定值，受 PSO 的启发[8-9]，本文提出一种基于混沌自适应的惯性权重为式(11)：

$$\omega=\omega_s+(\omega_e-\omega_s)\times\log_{10}\left(1+\frac{10t}{T_{max}}\right). \tag{11}$$

其中，$T_{max}$ 表示最大迭代次数；$\omega_s$ 表示惯性权重初始值；$\omega_e$ 是最大迭代次数的惯性权重。实验表明，当 $\omega$ 在[0.4,0.9]之间变化时，算法具有较好的寻优性能。随着迭代的变化，将惯性权重从 0.9 非线性递减至 0.4，实现了权重的动态变化，使算法的全局搜索和局部开发得到较好的平衡，一定程度上也增加了算法跳出局部最优区域的概率。考虑到搜索猎物阶段选取的随机鲸鱼位置向量，本文 EWOA 位置更新为公式(12)~(14)：

$$X(t+1)=\omega\cdot X^*(t)-A\cdot D,\ |A|<1,\ p^{Tent}<0.5 \tag{12}$$

$$X(t+1)=\omega\cdot X_{rand}-A\cdot D_{rand},\ |A|\geqslant 1,\ p^{Tent}<0.5 \tag{13}$$

$$X(t+1)=D'\cdot e^{bl}\cdot\cos(2\pi l)+\omega X^*(t).\ p^{Tent}\geqslant 0.5 \tag{14}$$

### 2.4　EWOA 算法流程

综上所述，本文所提 EWOA 算法的流程如图 1 所示。

**Step 1**　根据混沌 Tent 映射生成 $N$ 个初始化个体位置，$N$ 为种群大小，并设置其它参数；

**Step 2**　由目标函数 $f(x)$ 计算出每头鲸鱼的适应度，并对最优值存优；

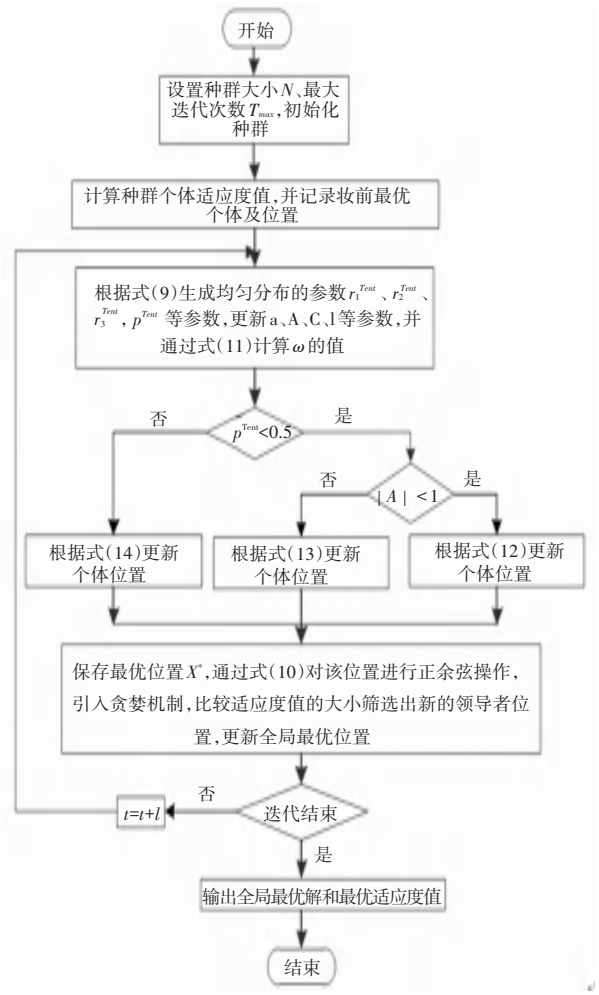



**图 1 算法流程图**

**Fig. 1 Algorithm flow chart**

**Step 3** 通过式(9)生成均匀分布的 $r_1^{Tent}$、$r_2^{Tent}$、$r_3^{Tent}$、$p^{Tent}$ 等参数,同时更新 $a$、$A$、$C$、$l$ 等,通过式(11) 更新参数;

**Step 4** 比较 $|A|$ 的大小,用 *Tent* 映射生成概率因子 $p^{Tent}$ 值与0.5作比较,选择相应的位置更新公式:若 $p^{Tent} < 0.5$,当 $|A| < 1$ 时,通过式(12) 包围收缩;当 $|A| \geqslant 1$ 时,按照式(13) 随机搜索;若 $p^{Tent} \geqslant 0.5$,按照式(14) 的螺旋机制更新当前位置;

**Step 5** 将存下来的最优位置 $X^*$ 按照式(10)进行位置更新,引入贪婪机制,决定是否更新全局最优位置;

**Step 6** 判断算法是否满足停止循环条件,若满足,则跳出主循环,输出目标位置和目标值,否则返回 Step3~5。

## 3 仿真实验和结果分析

本文实验的运行环境均为 64 位的 Window10 系统,处理器类型为 Intel Core i5-7500,编程软件为 MATLAB R2014b,为了验证 EWOA 的性能,将改进的鲸鱼优化算法(EWOA)、标准的鲸鱼优化算法(WOA)、加入正余弦变化策略的鲸鱼优化算法(SCWOA)、加入 Tent 映射的鲸鱼优化算法(TWOA)、加入混沌自适应权重的鲸鱼优化算法(WWOA)、灰狼算法(GWO)和正余弦算法(SCA)同时在表 1 的 10 个基准测试函数下进行 30 次对比实验。

由于不同维度会影响算法的寻优效果,所以将表 1 所示的 10 个测试函数 $f_1 \sim f_{10}$ 维度设置从 10 维随机递增到 200 维,其中 $f_1 \sim f_5$ 为连续单峰函数,用来验证算法收敛速度,$f_6 \sim f_{10}$ 是多峰函数,具有多个局部极值,用来验证算法跳出局部最优的能力和全局探测能力。同时,在低维和高维的情况下验证算法的求解能力,从而更加全面地验证算法的有效性。

为保证实验的公平性,本次实验所有算法的共同参数统一设置:种群大小为 30,算法最大迭代次数为1 000 次,各算法的主要参数设置见表 2。

**表 1 基准测试函数**

**Tab. 1 Benchmark test functions**

| 编号 | 函数名 | 维度 | 搜索区域 | 理论值 |
|---|---|---|---|---|
| $f_1$ | Sphere | 80 | [-100,100] | 0 |
| $f_2$ | Schwefel 2.22 | 60 | [-10,10] | 0 |
| $f_3$ | Schwefel 1.2 | 100 | [-100,100] | 0 |
| $f_4$ | Schwefel 2.21 | 40 | [-100,100] | 0 |
| $f_5$ | Step | 30 | [-100,100] | 0 |
| $f_6$ | Rastrigin | 90 | [-5.12,5.12] | 0 |
| $f_7$ | Ackley | 60 | [-32,32] | 0 |
| $f_8$ | Griewank | 200 | [-600,600] | 0 |
| $f_9$ | Penalized1 | 10 | [-50,50] | 0 |
| $f_{10}$ | Penalized2 | 30 | [-50,50] | 0 |

**表 2 算法主要参数**

**Tab. 2 The main parameters of the algorithm**

| 算法 | 主要参数 |
|---|---|
| WOA | $a_{max} = 2$, $a_{min} = 0$ |
| EWOA | $\omega_{max} = 0.9$, $\omega_{min} = 0.4$ ,$k = 0.1$ |
| TWOA | $r_1 = r_1^{Tent}$、$r_2 = r_2^{Tent}$、$p = p^{Tent}$ |
| WWOA | $\omega_{max} = 0.9$, $\omega_{min} = 0.4$ |
| GWO | $a_{max} = 2$, $a_{min} = 0$ |

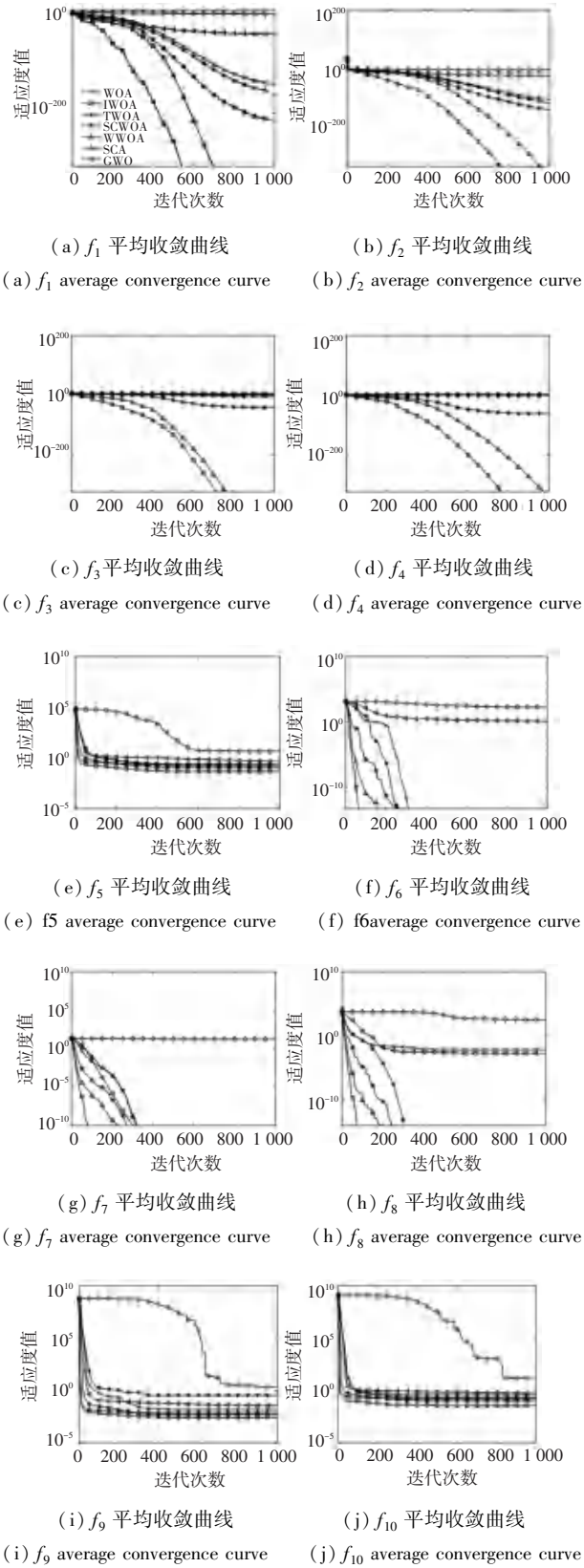
表 3 包含 10 个基准测试函数的优化结果,包括 5 个评估性能。通常最优值、最差值和平均值等几项指标能反映算法的寻优能力和收敛精度,由表 3 可知,单峰函数 $f_1 \sim f_5$ 的求解结果大部分都能达到

理论值 0，虽然 $f_5$ 并没有达到理论值，但综合几个改进算法的评价指标仍然比其它算法要好，值得注意的是 WWOA 在求解其中几个单峰函数时同样达到了理论效果，这是因为 WOA 算法容易陷入局部极值，在算法中加入了混沌自适应权重，一定程度上使得算法拥有跳出局部最优区域的能力，而伴随维度的增加，EWOA 的精度仍然是最高的，说明不同的改进策略提高了算法的寻优能力，从而达到了良好的寻优效果；对于复杂的非线性多峰态函数 $f_6$ ~ $f_{10}$，EWOA 的最优值、最差值等评价指标都比其他算法要好，证明了 EWOA 的全局寻优能力，在求解高维函数时，EWOA 达到的求解精度和稳定性明显优于其它对比算法，证明了本文所提改进算法的竞争性。

其次，标准差可以衡量算法的稳定性和跳出局部最优的能力，表 3 中不管是单峰函数还是多峰函数，EWOA 的标准差都比其它算法要好，其中部分还达到理论值，从而说明 EWOA 在求解过程中得到的寻优结果并非偶然；如求解多峰函数 $f_8$ 时，虽然其中几个改进算法也达到了理论值，但是如图 2 所示，EWOA 算法的收敛精度和收敛速度明显比其他几种算法要好，值得注意的是 WWOA 的部分标准差都和 EWOA 相似，说明鲸鱼优化算法对于权重比较敏感，导致加入权重后的算法有较好的寻优效果，但由图 2 可知，综合几个改进点的 EWOA 在所有的求解中都表现出较好的效果，说明本文算法相比于其它算法更稳定。

为了直观分析算法的寻优效果，图 2(a)~(j)给出了各算法在 10 个基准测试函数上独立运行 30 次寻优的平均收敛曲线，其中各算法图例同(a)所示。其中图 2(a)~(e)为函数 $f_1$ ~ $f_5$ 的平均收敛曲线；图 2(f)~(j)为多峰函数 $f_6$ ~ $f_{10}$ 的平均收敛曲线。由图 2 可知，EWOA 的收敛速度和收敛精度都明显比其它算法更优，这是由于在算法中加入了混沌 Tent 映射、正余弦鲸鱼算法和混沌自适应权重，所以 EWOA 算法收敛速度和求解精度都比其它算法要好。

表 3 中 Mnci 表示每个算法 30 次的平均收敛代数(Average number of convergent iterations)，其中 $f_1$ ~ $f_4$ 设置收敛精度为 $1 \times 10^{-100}$，由表 3 中 Mnci 值可知，10 个测试函数中，EWOA 算法的平均收敛代数比其他算法快，其中‘--’表示迭代到1 000次时算法均未收敛。相比之下，加入一个改进点的算法比原始鲸鱼优化算法、正余弦算法和灰狼算法要快，说明本文改进算法相比于其它算法，具有更快的收敛速度。




(a) $f_1$ 平均收敛曲线　(b) $f_2$ 平均收敛曲线

(a) $f_1$ average convergence curve　(b) $f_2$ average convergence curve

(c) $f_3$平均收敛曲线　(d) $f_4$ 平均收敛曲线

(c) $f_3$ average convergence curve　(d) $f_4$ average convergence curve

(e) $f_5$ 平均收敛曲线　(f) $f_6$ 平均收敛曲线

(e) f5 average convergence curve　(f) f6average convergence curve

(g) $f_7$ 平均收敛曲线　(h) $f_8$ 平均收敛曲线

(g) $f_7$ average convergence curve　(h) $f_8$ average convergence curve

(i) $f_9$ 平均收敛曲线　(j) $f_{10}$ 平均收敛曲线

(i) $f_9$ average convergence curve　(j) $f_{10}$ average convergence curve

**图 2　测试函数平均收敛曲线**

**Fig. 2　Average convergence curve of test function**

表 3 算法寻优结果对比

Tab. 3 Comparison of algorithm optimization results

| | 算法 | 最优值 | 最差值 | 均值 | 标准差 | Mnci |
|---|---|---|---|---|---|---|
| | WOA | 3.06E-166 | 1.19E-148 | 6.96E-150 | 2.65E-149 | 660 |
| | EWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 231 |
| | TWOA | 4.82E-177 | 1.28E-144 | 4.25E-146 | 2.33E-145 | 657 |
| $f_1$ | WWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 446 |
| | SCWOA | 3.00E-244 | 1.82E-215 | 6.06E-217 | 0.00E+00 | 509 |
| | SCA | 6.63E-41 | 3.60E-38 | 2.79E-39 | 6.80E-39 | - |
| | GWO | 5.90E+00 | 3.82E+03 | 5.66E+02 | 1.06E+03 | - |
| | WOA | 3.34E-114 | 2.20E-100 | 1.35E-101 | 4.86E-101 | 951 |
| | EWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 372 |
| | TWOA | 1.89E-117 | 3.96E-103 | 1.45E-104 | 7.24E-104 | 923 |
| $f_2$ | WWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 573 |
| | SCWOA | 8.86E-150 | 7.41E-133 | 2.56E-134 | 1.35E-133 | 732 |
| | SCA | 7.91E-21 | 1.05E-19 | 3.85E-20 | 2.43E-20 | - |
| | GWO | 1.25E-02 | 5.26E+00 | 4.99E-01 | 9.77E-01 | - |
| | WOA | 5.33E+05 | 1.15E+06 | 9.02E+05 | 1.47E+05 | - |
| | EWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 405 |
| | TWOA | 2.85E-03 | 4.00E+05 | 3.04E+04 | 7.83E+04 | - |
| $f_3$ | WWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 468 |
| | SCWOA | 1.73E-64 | 6.46E-11 | 2.15E-12 | 1.18E-11 | - |
| | SCA | 3.69E-03 | 2.59E+02 | 1.73E+01 | 5.05E+01 | - |
| | GWO | 1.09E+05 | 3.19E+05 | 1.89E+05 | 4.71E+04 | - |
| | WOA | 5.29E-02 | 9.67E+01 | 7.39E+01 | 2.72E+01 | - |
| | EWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 383 |
| | TWOA | 6.48E+01 | 1.00E+02 | 9.69E+01 | 8.14E+00 | - |
| $f_4$ | WWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 575 |
| | SCWOA | 3.29E-78 | 2.39E-60 | 8.68E-62 | 4.35E-61 | - |
| | SCA | 1.30E-01 | 8.13E+00 | 1.24E+00 | 1.61E+00 | - |
| | GWO | 8.78E+01 | 9.48E+01 | 9.22E+01 | 1.85E+00 | - |
| | WOA | 8.18E-02 | 4.30E-01 | 2.07E-01 | 1.12E-01 | 943 |
| | EWOA | 2.91E-02 | 6.56E-02 | 4.49E-02 | 1.55E-02 | 597 |
| | TWOA | 3.68E-02 | 4.04E-01 | 2.41E-01 | 1.35E-01 | 942 |
| $f_5$ | WWOA | 1.30E-01 | 2.78E-01 | 1.82E-01 | 5.72E-02 | 668 |
| | SCWOA | 3.06E-02 | 2.45E-01 | 1.04E-01 | 7.46E-02 | 936 |
| | SCA | 2.50E-01 | 7.59E-01 | 4.67E-01 | 1.75E-01 | - |
| | GWO | 3.91E+00 | 5.24E+00 | 4.50E+00 | 4.87E-01 | 788 |
| | WOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 281 |
| | EWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 49 |
| | TWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 304 |
| $f_6$ | WWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 106 |
| | SCWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 190 |
| | SCA | 0.00E+00 | 9.34E+00 | 1.33E+00 | 3.07E+00 | 504 |
| | GWO | 5.18E+00 | 4.43E+02 | 1.84E+02 | 1.05E+02 | 797 |
| | WOA | 8.88E-16 | 7.99E-15 | 4.56E-15 | 2.72E-15 | 373 |
| | EWOA | 8.88E-16 | 8.88E-16 | 8.88E-16 | 0.00E+00 | 97 |
| | TWOA | 8.88E-16 | 7.99E-15 | 4.44E-15 | 2.64E-15 | 374 |
| $f_7$ | WWOA | 8.88E-16 | 8.88E-16 | 8.88E-16 | 0.00E+00 | 317 |
| | SCWOA | 8.88E-16 | 8.88E-16 | 8.88E-16 | 0.00E+00 | 329 |
| | SCA | 4.00E-14 | 5.06E-14 | 4.14E-14 | 2.40E-15 | 446 |
| | GWO | 1.26E+00 | 2.06E+01 | 1.76E+01 | 6.58E+00 | 260 |
| | WOA | 0.00E+00 | 1.62E-01 | 5.40E-03 | 2.96E-02 | 331 |
| | EWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 59 |
| | TWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 300 |
| $f_8$ | WWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 150 |
| | SCWOA | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 227 |
| | SCA | 1.11E-16 | 4.36E-02 | 1.45E-03 | 7.96E-03 | 575 |
| | GWO | 5.90E+01 | 8.20E+02 | 2.96E+02 | 1.82E+02 | 820 |
| | WOA | 1.42E-03 | 1.11E-01 | 1.48E-02 | 2.30E-02 | 933 |
| | EWOA | 4.69E-04 | 5.94E-03 | 2.66E-03 | 1.36E-03 | 630 |
| | TWOA | 1.96E-03 | 9.48E+00 | 3.31E-01 | 1.73E+00 | 921 |
| $f_9$ | WWOA | 3.40E-03 | 1.03E-02 | 6.67E-03 | 1.80E-03 | 651 |
| | SCWOA | 1.06E-03 | 1.14E-02 | 4.99E-03 | 2.83E-03 | 913 |
| | SCA | 6.27E-03 | 7.98E-02 | 3.44E-02 | 1.47E-02 | - |
| | GWO | 3.45E-01 | 1.31E+01 | 2.08E+00 | 2.74E+00 | 809 |
| | WOA | 3.32E-02 | 6.43E-01 | 2.54E-01 | 1.44E-01 | 928 |
| | EWOA | 6.79E-03 | 1.05E-01 | 5.07E-02 | 2.50E-02 | 680 |
| | TWOA | 8.35E-02 | 7.74E-01 | 3.05E-01 | 1.77E-01 | 926 |
| $f_{10}$ | WWOA | 6.73E-02 | 5.21E-01 | 1.53E-01 | 9.12E-02 | 648 |
| | SCWOA | 1.77E-02 | 2.22E-01 | 1.02E-01 | 4.85E-02 | 944 |
| | SCA | 1.95E-01 | 8.91E-01 | 5.29E-01 | 1.95E-01 | - |
| | GWO | 2.22E+00 | 2.71E+02 | 1.54E+01 | 4.89E+01 | 812 |

另外,平均误差(Mean Absolute Error, MAE)是在基于 10 个测试函数,通过对算法进行排序,完成对算法的定量分析,MAE 可以有效证明算法优越性的性能指标[10],表 4 是在给定基准测试函数情况下算法的 MAE 排序,MAE 计算公式为(15):

$$\mathrm{MAE} = \frac{\sum_{i=1}^{N_f} |m_i - o_i|}{N_f}. \tag{15}$$

其中,$m_i$ 表示算法寻优结果的平均值;$o_i$ 表示每个基准测试函数的理论值;$N_f$ 表示所采用测试函数个数。由表 4 可知,EWOA 具有最小的平均误差,即相对于其他对比算法,MAE 的最优算法是 EWOA 算法,进一步验证了本文所提 EWOA 改进的有效性。

表 4 算法 MAE 排名

Tab. 4 Algorithm MAE ranking

| 算法 | MAE | 排名 |
|---|---|---|
| EWOA | 0.009 817 | 1 |
| SCWOA | 0.021 114 | 2 |
| WWOA | 0.034 144 | 3 |
| SCA | 2.095 101 | 4 |
| TWOA | 3 051.874 | 5 |
| GWO | 18 997.54 | 6 |
| WOA | 90 253.61 | 7 |

对运行 30 次后的算法结果仅基于均值和标准差评估算法性能并不可靠。为体现改进算法的鲁棒性和公平性,通常使用 Wilcoxon 秩和检验来评估所提出的改进算法。因此,在 5%的显著水平下使用统计检验,来验证本文所提改进算法每次运行结果是否与其他算法在统计上存在显著性差异。通常,当 $p < 5\%$时,可以被认为是拒绝零假设的有力验证,说明两种对比算法具有显著性差异[11]。

表 5 列出了在 10 个测试函数下,本文算法与其他 6 个对比算法在 Wilcoxon 检验中的 $p$ 值,因为无法将 EWOA 与自身比较,所以将 EWOA 与 WOA、TWOA、WWOA、SCWOA、SCA 和 GWO 进行比较。

表 5 Wilcoxon 秩和检验 $p$ 值

Tab. 5 Wilcoxon rank sum test $p$ value

| 函数 | $p_1$ | $p_2$ | $p_3$ | $p_4$ | $p_5$ | $p_6$ |
|---|---|---|---|---|---|---|
| $f_1$ | 1.21E-12 | 1.21E-12 | Na | 1.21E-12 | 1.21E-12 | 1.21E-12 |
| $f_2$ | 1.21E-12 | 1.21E-12 | Na | 1.21E-12 | 1.21E-12 | 1.21E-12 |
| $f_3$ | 1.21E-12 | 1.21E-12 | Na | 1.21E-12 | 1.21E-12 | 1.21E-12 |
| $f_4$ | 1.21E-12 | 1.20E-13 | Na | 1.21E-12 | 1.21E-12 | 1.21E-12 |
| $f_5$ | 5.83E-04 | 4.08E-03 | 5.83E-04 | 1.75E-02 | 5.83E-04 | 5.83E-04 |
| $f_6$ | Na | Na | Na | Na | 4.68E-10 | 1.21E-12 |
| $f_7$ | 1.26E-08 | 1.22E-08 | Na | Na | 3.50E-13 | 1.21E-12 |
| $f_8$ | 3.34E-01 | Na | Na | Na | 4.16E-14 | 1.21E-12 |
| $f_9$ | 8.48E-09 | 7.77E-09 | 8.89E-10 | 1.11E-03 | 3.02E-11 | 3.02E-11 |
| $f_{10}$ | 5.46E-09 | 6.07E-11 | 5.57E-10 | 2.13E-05 | 3.02E-11 | 3.02E-11 |
| +/=/- | 8/1/1 | 8/2/0 | 3/7/0 | 7/3/0 | 10/0/0 | 10/0/0 |

(下转第 52 页)