Operations Research and Fuzziology 运筹与模糊学, 2023, 13(4), 3462-3475
Published Online August 2023 in Hans. https://www.hanspub.org/journal/orf
https://doi.org/10.12677/orf.2023.134349

Hans汉斯

基于MATLAB的改进海马算法

赵建萍

贵州大学大数据与信息工程学院，贵州 贵阳

收稿日期：2023年6月20日；录用日期：2023年8月5日；发布日期：2023年8月11日

摘　要

针对海马算法寻优精度不足、易陷入局部最优的问题，本文提出一种基于Singer混沌及失败者放逐的海马算法。在初始化种群阶段，Singer混沌映射被引入用以生成遍历搜索空间的初始海马个体，增强了初始种群的多样性，有利于提高算法的搜索精度；在海马捕食阶段，引入失败者放逐策略，将捕食失败的海马个体放逐到搜索空间内，有利于算法跳出局部最优；在海马繁殖阶段，引入动态繁殖策略，动态影响父本和母本的权重，有利于防止算法过早收敛到局部极值。在算法性能测试实验中选用了10个基准测试函数和10个CEC2013测试函数，实验结果表明本文所提改进海马算法在寻优精度和收敛速度上都有较大提升，是一种优化能力强、鲁棒性好的算法。

关键词

海马算法，Singer混沌映射，失败者放逐策略，动态繁殖策略，CEC2013

Improved Seahorse Algorithm Based on MATLAB

Jianping Zhao

College of Big Data and Information Engineering, Guizhou University, Guiyang Guizhou

Received: Jun. 20th, 2023; accepted: Aug. 5th, 2023; published: Aug. 11th, 2023

Abstract

This paper proposes a multi-strategy improved seahorse algorithm to address the problem that the seahorse algorithm is not sufficiently accurate and easily falls into local optimum. In the initialization phase, Singer chaos mapping is introduced to generate the initial seahorse individuals traversing the search space, which enhances the diversity of the initial population and helps to improve the search accuracy of the algorithm; in the seahorse predation phase, a loser banishment strategy was introduced to banish the seahorse individuals that failed to feed into the search

文章引用: 赵建萍. 基于 MATLAB 的改进海马算法[J]. 运筹与模糊学, 2023, 13(4): 3462-3475.
DOI: 10.12677/orf.2023.134349

space, which is conducive to the algorithm jumping out of the local optimum; in the seahorse reproduction phase, a dynamic reproduction strategy was introduced to dynamically influence the weights of the fathers and the mothers, which is conducive to preventing the algorithm from converging to the local extreme prematurely. Ten benchmark test functions and ten CEC2013 test functions were used to test the performance of the algorithm. The experimental results show that the improved seahorse algorithm proposed in this paper has a large improvement in both the search accuracy and convergence speed, and is an algorithm with strong optimization capability and good robustness.

Keywords

Seahorse Algorithm, Singer Chaos Mapping, Loser Banishment Strategy, Dynamic Reproduction Strategy, CEC2013

1. 引言

作为一种受生物群体行为启发的方法，群智能算法已被成功应用到各个领域，并且仍有大量性能优良的群智能算法被提出以解决不同类型的问题，这其中包含灰狼优化算法(Gray Wolf Optimization, GWO) [1]、蒲公英优化算法(Dandelion Optimization, DO) [2]、海鸥优化算法(Seagull Optimization Algorithm, SOA) [3]、爬行动物搜索算法(Reptile Search Algorithm, RSA) [4]、金豺优化算法(Golden Jackal Optimization, GJO) [5]、蜜獾优化算法(Honey Badger Optimization Algorithm, HBA) [6]等。

海马算法(Sea-horse Optimizer, SHO)是一种 2022 年提出的较新的模拟海马行为的群智能算法[7]。海马作为一种长相奇特、行为也奇特的水生生物，喜欢栖息在多海藻、珊瑚的水域，靠着尾巴缠绕海藻随着波涛在水中浮动，在繁殖行为中，由长有育儿袋的雄性海马进行生育。海马算法在简单优化问题中寻优精度高、稳定性强，但是在面对复杂问题时容易陷入局部最优且收敛速度慢。针对这类群智能算法常见问题，一些学者提出了改进方法，文献[8]中利用二次插值策略生成新的个体，并贪心保留适应度值更优的个体，有效提高算法的寻优精度；文献[9]中利用平均适应度值将种群划分为先进子群和普通子群，并对不同的子群设计不同的位置更新方式，有效提高了算法求解问题的能力；文献[10]中分别对最优个体和劣等个体进行逐维变异优化和混沌干扰，有效提高了算法的寻优性能。

因此，本文针对 SHO 存在的问题，提出了一种改进海马算法(Improved Sea-horse Optimizer, ISHO)。首先利用 Singer 混沌映射生成初始海马种群，提升种群多样性，提高算法的搜索准确度；其次，利用失败者放逐策略，提高算法跳出局部极值的能力；最后，利用动态繁殖策略影响产生子代时亲本的权重，保留前代优秀遗传信息，减小过快陷入局部最优值的可能。本文选用 10 个基准测试函数和 6 个较新的群智能算法进行测试，对结果进行秩和检验，利用 10 个 CEC2013 函数对改进前后算法进行测试，结果验证了改进方法的有效性，证明了 ISHO 是一种寻优精度高、收敛速度快的稳定算法。

2. 标准海马算法

在海马的众多行为中，海马算法主要模拟了海马的运动行为、捕食行为以及繁殖行为。在运动行为中，海马个体的移动分为在搜索空间全局搜索以及在可能最优位置附近局部开发两种情况，为了便于区

分，引入按照正态分布的随机数 r_1，当 r_1 为正数时，海马个体做螺旋运动向精英个体移动，表达式如下，

$$x\_\text{new1}(t+1)=x(t)+\text{Levy}\cdot\left(\text{Elite}(t)-x(t)\right)\cdot x\cdot y\cdot z+\text{Elite}(t) \tag{1}$$

其中 t 是迭代次数，Elite 是精英个体位置，Levy 是莱维飞行函数，表达式如下，

$$\text{Levy}=s\times\frac{w\times\sigma}{|k|^{\frac{1}{\lambda}}} \tag{2}$$

其中 s 是固定常数 0.01，w 和 k 为取值为 $[0,1]$ 的随机数，λ 取值范围是 $[0,2]$，σ 的表达式如下，

$$\sigma=\left(\frac{\Gamma(1+\beta)+\sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right)\times\beta\times2^{\frac{(\beta-1)}{2}}}\right)^{\frac{1}{\beta}} \tag{3}$$

x、y 和 z 分别是螺旋运动下坐标的三维分量，表达式如下，

$$x=\rho\times\cos(\theta) \tag{4}$$

$$y=\rho\times\sin(\theta) \tag{5}$$

$$z=\rho\times\theta \tag{6}$$

其中 $\rho=u\times e^{\theta v}$，$\theta$ 为取值在 $[0,2\pi]$ 的随机数，对数螺旋常数 u 和 v 均为 0.05。

当 r_1 为负数的时候，海马个体随海洋变化进行布朗运动

$$x\_\text{new1}(t+1)=x(t)+\text{rand}\cdot l\cdot\beta_t\cdot\left(x(t)-\beta_t\cdot\text{Elite}(t)\right) \tag{7}$$

rand 是 0 到 1 之间的随机数，参数 l 为 0.05，布朗运动的随机游走系数 β_t 表达式如下，

$$\beta_t=\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right) \tag{8}$$

在海马的捕食行为中，精英个体位置被设定为食物位置，海马捕食成功概率被设定为 90%，按照随机数 r_2 来区分海马捕食成功或失败两种情况。捕食行为描述如下，当 r_2 取值大于 0.1 时，海马捕食成功，此时海马个体的位置更新公式为公式 9。

$$x\_\text{new2}(t+1)=\Delta\cdot\left(\text{Elite}(t)-\text{rand}\cdot x\_\text{new1}(t)\right)+(1-\Delta)\cdot\text{Elite}(t) \tag{9}$$

当 r_2 小于等于 0.1，海马捕食失败的时候，转向全局搜索，即

$$x\_\text{new2}(t+1)=(1-\Delta)\cdot\left(x\_\text{new1}(t)-\text{rand}\cdot\text{Elite}(t)\right)+\Delta\cdot x\_\text{new1}(t) \tag{10}$$

其中调整影响因子的权重 Δ 线性递减，以等式 11 计算。

$$\Delta=\left(1-\frac{t}{\text{max\_iter}}\right)^{\frac{2t}{\text{max\_iter}}} \tag{11}$$

其中 max_iter 为最大迭代次数。

在海马的繁殖行为中，模拟了海马最特殊的一种行为，即由雄性生育下一代，为了更好地继承前一代的寻优成果，根据适应度值排序结果，将排名在前的一半个体设定为雄性，排名在后的一半个体设定为雌性，随机交配产生一个子代海马个体的表达式如下，

$$x\_\text{offspring}=r_3\cdot\text{father}+(1-r_3)\cdot\text{mother} \tag{12}$$

其中 $x\_\text{offspring}$ 是产生的子代的位置，father 为参与繁殖的雄性的位置，mother 为雌性位置，r_3 为取值为 $[0,1]$ 之间的随机数。繁殖行为产生的后代再次进行适应度值排序，按照原先设定种群规模筛选排名靠前的海马个体组成新的种群参与下一次迭代。

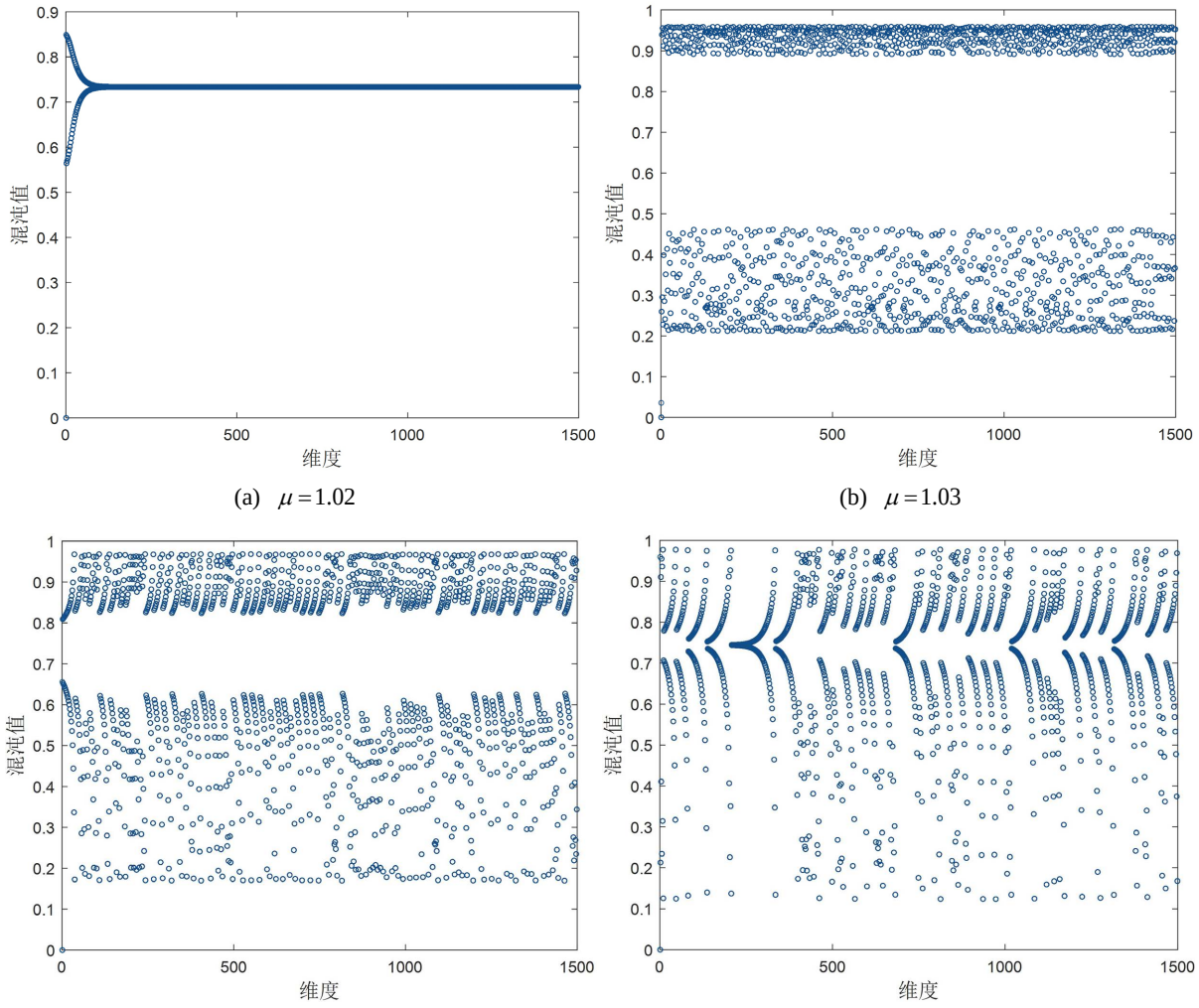
3. 改进海马算法

3.1. Singer 混沌初始化

标准海马算法随机在搜索空间中生成初始的海马个体，随机出现在搜索空间的海马个体位置可能聚集在一处，导致种群多样性不足，容易使算法陷入局部最优，也可能遍历性不够，导致有的算法空间没有被覆盖到，使得算法寻优精度不足。混沌映射初始化是解决这一问题的常用方法，但是不同的混沌映射方式解决问题的效果也有所不同。本文拟采用 Singer 混沌映射来生成初始的海马种群，Singer 混沌映射表达式如下。

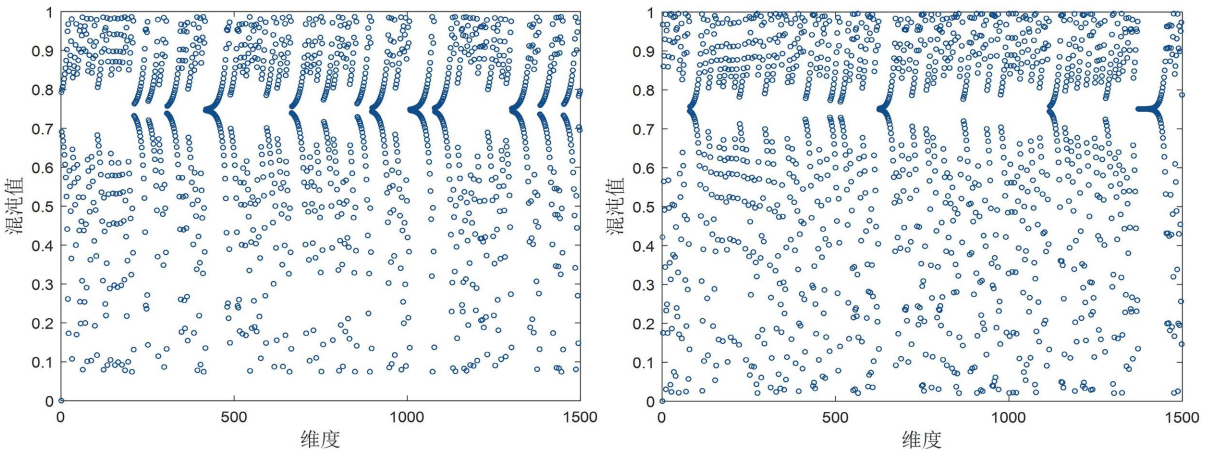
$$x(t+1)=\mu\cdot\left(7.86x(t)-23.31x^2(t)+28.75x^3(t)-13.302875x^4(t)\right) \tag{13}$$

其中，$\mu\in(0.9,1.08)$，在 30 维下取 1500 个散点，将不同 μ 取值下的散点图在图 1 中给出。



(a) $\mu=1.02$　　(b) $\mu=1.03$

(c) $\mu=1.04$　　(d) $\mu=1.05$



(d) $\mu=1.06$ (e) $\mu=1.07$

Figure 1. The scatter plot of the population initialized by different methods
图 1. 不同方法初始化种群散点图

从图 1 中可以看出，随着参数 μ 的增加，Singer 混沌映射生成的散点图趋向覆盖搜索空间，重合的无效散点数量减少，有效的分散散点数量增多。本文选取 μ 为 1.07。将 Singer 混沌生成的序列映射到搜索空间的表达式如下，

$$x=\text{Singer}\left(N,\dim\right)\cdot\left(ub-lb\right)+lb \tag{14}$$

其中 N 为种群规模，dim 为维度，ub 和 lb 分别是搜索空间的上下界。

3.2. 失败者放逐策略

在描述海马捕食行为时，标准 SHO 利用随机数 r_2 来区分捕食的两种结果，当捕食成功的时候，海马个体更靠近精英个体，如果捕食失败，则转为全局搜索，表达式为公式 9 和 10。在海马个体进行捕食的时候，大概率已经处于靠近精英个体的位置，在算法进行到后期的时候，如果捕食失败，再增加精英个体影响力，不仅会加重算法陷入局部极值的情况，还会消耗较多时间在对算法寻优无帮助的行为上。因此为了提高算法寻优精度、增强算法跳出局部最优的能力，引入失败者放逐策略，表达式如下，

$$x\_\text{new}2=x\_\text{new}1+\text{Step}\times x\_\text{new}1 \tag{15}$$

其中 Step 为莱维飞行的步长，即在原位置有一定概率以大步长进行位移。

3.3. 动态繁殖策略

由于海马是由雄性生育的，因此在标准 SHO 中选择种群中适应度值更好的一半个体作为雄性海马，较差的一半个体作为雌性个体，由公式 12 表示海马的繁殖行为。在选取亲本的权重时使用的是随机数，随机数值大就意味着父本的权重大、母本占的权重小，但是这并未考虑到算法实际情况，如果想要种群多样性好、算法寻优精度高，应当增强新生个体在算法前期受到母本影响的权重，想要算法收敛速度快，那么应当增强新生个体在算法后期受到父本影响的权重。除此以外，还要防止算法过快陷入局部极值，因此选用动态繁殖策略，其中变化的边界的表达式为公式 16。

$$p=0.4+0.5\times\left(\frac{\text{max\_iter}-t}{\text{max\_iter}}\right)^{0.6} \tag{16}$$

p 非线性递减，当随机数大于 p 时，在算法后期父本权重占比大的概率更大，当随机数较小的时候，在算法前期父本和母本以相同权重影响产生的后代。使用动态繁殖策略后的海马繁殖行为表达式如下，

$$x\_\text{offspring}=\begin{cases} r_3\cdot\text{father}+(1-r_3)\cdot\text{mother}, & r_3>p \\ \dfrac{\text{father}+\text{mother}}{2}, & r_3\le p \end{cases} \tag{17}$$

其中 father 为父本个体位置，mother 为母本个体位置，$x\_\text{offspring}$ 为产生的后代的位置。改进后的海马繁殖行为增强了算法后期父本作为较优秀个体的影响能力，提高了产生的后代的质量，同时保留了一部分概率使得产生个体受到母本影响，防止算法过早收敛到局部最优。

3.4. ISHO 主要步骤

改进 SHO 算法的主要步骤如下：

第一步：设置种群规模为 N，最大迭代次数为 max_iter，对数螺旋常数 u 和 v 以及常数系数 l。

第二步：利用 Singer 混沌映射生成初始的海马种群。

第三步：计算海马个体的适应度值并进行排序，将适应度值最好的设定为精英个体，同时设定精英个体位置为食物源位置。

第四步：模拟海马的运动行为，根据随机数 r_1 的正负选择运动方式，取正数时进行螺旋运动，反之进行布朗运动。

第五步：模拟海马的捕食行为，根据随机数 r_2 选择海马个体捕食成功或失败两种情况下的位置更新方式，捕食成功就向精英个体靠近，捕食失败就进行莱维放逐。

第六步：模拟海马的繁殖行为，将适应度值靠前的一半个体设定为父本，靠后的一半个体设定为母本，根据随机数 r_3 和动态边界的关系，按照公式 17 对父本和母本的选择不同的影响权重。

第七步：为保证种群不扩张，再次进行适应度值排序，将排名靠前的 N 个海马个体作为新的种群参与下一次算法迭代。

第八步：检查算法是否满足结束条件，若满足就结束循环，输出最优解，否则继续执行循环直至满足结束条件。

改进 SHO 算法的流程图如图 2 所示。

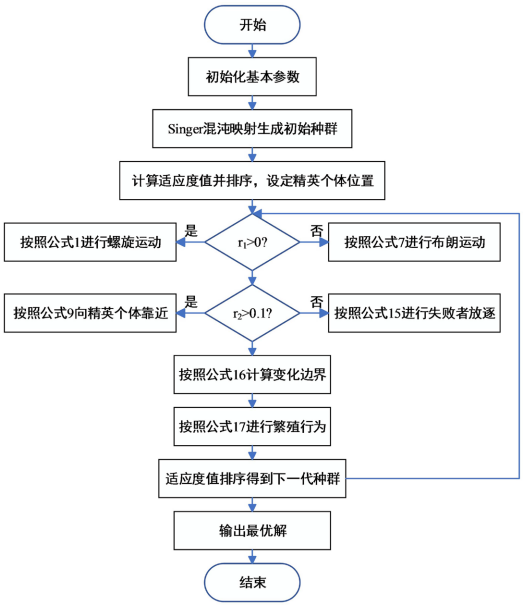


Figure 2. Flow chart of ISHO
图 2. ISHO 流程图

4. 实验结果

4.1. 基准函数测试实验

4.1.1. 基准函数

在验证算法性能的方法中，最为经典的就是 23 个基准测试函数[11]。本文选取 10 个基本函数进行实验，其中 F1~F3 为单峰函数，F4、F5 为多峰函数，F6~F10 为固定维度多峰函数，其求解的难度逐步提升。选取的函数的基本信息在表 1 中列出。

Table 1. System resulting data of standard experiment
表 1. 标准试验系统结果数据

编号	函数	类型	维度	边界	理想值
F1	Sphere	单峰	30	[−100, 100]	0
F2	Schwefel 2.22	单峰	30	[−10, 10]	0
F3	Schwefel 1.2	单峰	30	[−100, 100]	0
F4	Schwefel	多峰	30	[−500, 500]	−12569.487
F5	Shifted Rotated Griewank	多峰	30	[−600, 600]	0
F6	Kowalik	固定维多峰	4	[−5, 5]	0.00030
F7	Goldstein-Price	固定维多峰	2	[−2, 2]	3
F8	Shekel 5	固定维多峰	4	[0, 10]	−10.1532
F9	Shekel 7	固定维多峰	4	[0, 10]	−10.4028
F10	Shekel 10	固定维多峰	4	[0, 10]	−10.5363

4.1.2. 算法性能比较与分析

在算法性能测试实验中，除了标准 SHO 算法，本文选取了 DO、SOA、HBA、RSA、GJO 以及 GWO 作为对比算法，用以验证 ISHO 的有效性和可行性。实验环境为 Windows 11 系统，1.80 GHz CPU，16 GB 内存，编程语言为 MATLAB R2020b。算法参数信息在表 2 中给出。

Table 2. Algorithm parameter settings
表 2. 算法参数设置

参数来源	参数名	设定值
共同参数	种群规模 N	50
	最大迭代次数 max_iter	2000
ISHO	对数螺旋常数 u、v	0.05
	常数系数 l	0.05
HBA	密度因子常系数 C	2
	获取食物能力参数 β	6
RSA	勘探精度控制参数 α	0.1
	精度控制参数 β	0.005

由于算法每次进行实验得到的结果具有偶然性，为了便于对比分析算法性能，将算法独立运行 30 次后得出的结果列在表 3 中。评价指标为平均值、标准差、最优值以及最差值，其中优先级排在首位的是平均值，可以反映出算法的寻优精度。将最优结果加粗表示。

Table 3. The test results of each algorithm under different functions
表 3. 各个算法在不同函数下的测试结果

函数	算法	平均值	标准差	最优值	最差值
F1	**ISHO**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SHO	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	DO	7.13E−13	5.24E−13	1.35E−13	2.27E−12
	SOA	2.66E−63	1.20E−62	2.70E−68	6.61E−62
	HBA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	RSA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	GJO	6.20E−260	0.00E+00	1.30E−267	7.60E−259
	GWO	2.34E−144	7.07E−144	9.58E−148	3.58E−143
F2	**ISHO**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SHO	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	DO	6.33E−07	4.22E−07	1.81E−07	4.22E−07
	SOA	5.56E−39	1.16E−38	1.25E−41	5.67E−38
	HBA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	RSA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	GJO	1.06E−149	1.85E−149	8.29E−153	7.73E−149
	GWO	5.51E−83	1.63E−82	8.17E−85	8.82E−82
F3	**ISHO**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SHO	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	DO	1.15E−03	5.91E−04	2.42E−04	2.53E−03
	SOA	7.51E−33	3.82E−32	3.38E−43	2.09E−31
	HBA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	RSA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	GJO	2.96E−99	1.53E−98	3.96E−112	8.40E−98
	GWO	3.34E−43	1.18E−42	1.77E−51	6.14E−42
F4	**ISHO**	**−1.25E+04**	**9.97E+01**	**−1.26E+04**	**−1.21E+04**
	SHO	−3.47E+03	5.80E+02	−4.53E+03	−2.26E+03
	DO	−1.01E+04	3.89E+02	−1.09E+04	−9.25E+03
	SOA	−6.08E+03	8.47E+02	−8.13E+03	−4.98E+03
	HBA	−9.27E+03	1.05E+03	−1.10E+04	−6.72E+03
	RSA	−5.45E+03	3.29E+02	−5.79E+03	−4.19E+03
	GJO	−4.40E+03	9.47E+02	−5.94E+03	−2.91E+03
	GWO	−6.28E+03	7.27E+02	−7.72E+03	−5.18E+03
F5	**ISHO**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	SHO	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	DO	1.47E−02	1.94E−02	8.50E−13	6.63E−02
	SOA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	HBA	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**

Continued

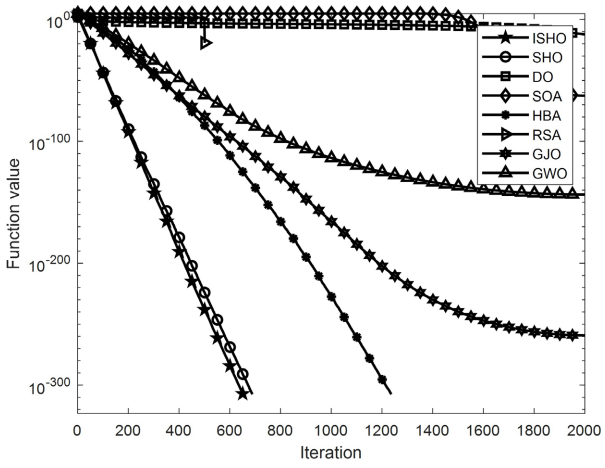
F5	**RSA**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	GJO	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	GWO	1.01E−03	3.84E−03	0.00E+00	1.52E−02
F6	**ISHO**	**3.16E−04**	**6.99E−06**	**3.09E−04**	**3.46E−04**
	SHO	7.09E−03	1.39E−02	4.22E−04	5.04E−02
	DO	3.42E−04	1.68E−04	3.07E−04	1.22E−03
	SOA	1.13E−03	2.79E−04	3.08E−04	1.22E−03
	HBA	3.91E−03	7.71E−03	3.07E−04	2.26E−02
	RSA	9.04E−04	3.38E−04	4.70E−04	2.09E−03
	GJO	3.69E−04	2.32E−04	3.07E−04	1.22E−03
	GWO	1.68E−03	5.08E−03	3.07E−04	2.04E−02
F7	ISHO	3.00E+00	7.05E−07	3.00E+00	3.00E+00
	SHO	5.65E+00	7.68E+00	3.00E+00	3.03E+01
	DO	**3.00E+00**	**2.70E−10**	**3.00E+00**	**3.00E+00**
	SOA	3.00E+00	2.54E−06	3.00E+00	3.00E+00
	HBA	3.90E+00	4.93E+00	3.00E+00	3.00E+01
	RSA	3.00E+00	1.69E−05	3.00E+00	3.00E+00
	GJO	3.00E+00	3.02E−07	3.00E+00	3.00E+00
	GWO	3.00E+00	6.29E−07	3.00E+00	3.00E+00
F8	ISHO	**−9.98E+00**	**9.22E−01**	**−1.02E+01**	**−5.09E+00**
	SHO	−2.69E+00	1.24E+00	−5.48E+00	−8.70E−01
	DO	−6.54E+00	2.93E+00	−1.02E+01	−2.63E+00
	SOA	−4.51E+00	4.36E+00	−1.02E+01	−3.51E−01
	HBA	−9.90E+00	1.37E+00	−1.02E+01	−2.63E+00
	RSA	−5.06E+00	2.96E−07	−5.06E+00	−5.06E+00
	GJO	−8.41E+00	2.53E+00	−1.02E+01	−3.37E+00
	GWO	−9.81E+00	1.29E+00	−1.02E+01	−5.06E+00
F9	ISHO	−9.87E+00	1.62E+00	−1.04E+01	−5.09E+00
	SHO	−3.19E+00	9.99E−01	−5.00E+00	−1.29E+00
	DO	−7.96E+00	3.36E+00	−1.04E+01	−1.84E+00
	SOA	−6.92E+00	4.29E+00	−1.04E+01	−5.21E−01
	HBA	−9.23E+00	2.68E+00	−1.04E+01	−2.77E+00
	RSA	−5.09E+00	9.86E−07	−5.09E+00	−5.09E+00
	GJO	−1.02E+01	9.63E−01	−1.04E+01	−5.13E+00
	GWO	**−1.02E+01**	**9.63E−01**	**−1.04E+01**	**−5.13E+00**
F10	ISHO	−1.04E+01	9.78E−01	−1.05E+01	−5.17E+00
	SHO	−2.82E+00	1.23E+00	−4.92E+00	−7.78E−01
	DO	−9.01E+00	3.15E+00	−1.05E+01	−1.68E+00

Continued

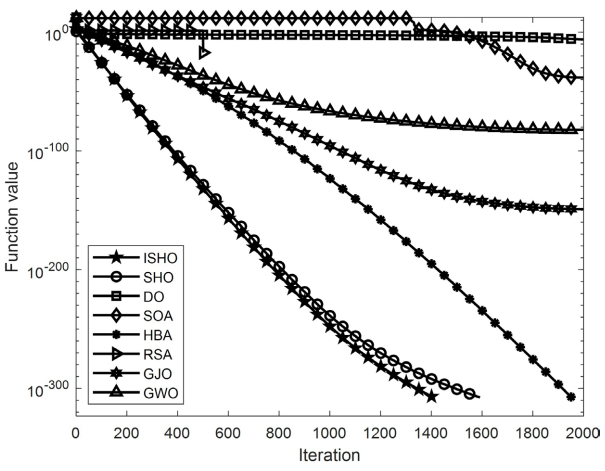
	SOA	−7.48E+00	3.81E+00	−1.05E+01	−5.56E−01
	HBA	−9.37E+00	2.66E+00	−1.05E+01	−2.42E+00
F10	RSA	−5.13E+00	1.70E−06	−5.13E+00	−5.13E+00
	GJO	−1.04E+01	9.79E−01	−1.05E+01	−5.17E+00
	GWO	**−1.05E+01**	**2.97E−05**	**−1.05E+01**	**−1.05E+01**

分析表 3 数据可知，本文改进方法 ISHO 在 F1~F6 以及 F8 这 7 个基准测试函数中排名第一，其中在 F1~F3 和 F5 均得到了理想最优值，在所选 10 个测试函数中均比标准 SHO 有提升，除此以外，在 F7 中排名第四，在 F9 和 F10 中均排名第三，仅次于 GWO 和 GJO。而 GWO 和 GJO 在单峰函数下的寻优效果无法得到理想值。另外，同样在 F1、F2、F3 和 F5 中能找到理想最优值的 HBA 和 RSA 在应对固定维多峰函数时效果没有 ISHO 好。在 F7 中排名第一的 DO 算法在处理单峰函数时同样效果欠佳。综合来看，ISHO 能够在单峰函数、多峰函数、固定维多峰函数中取得不错的结果，不仅算法寻优精度高，且算法稳定性好。

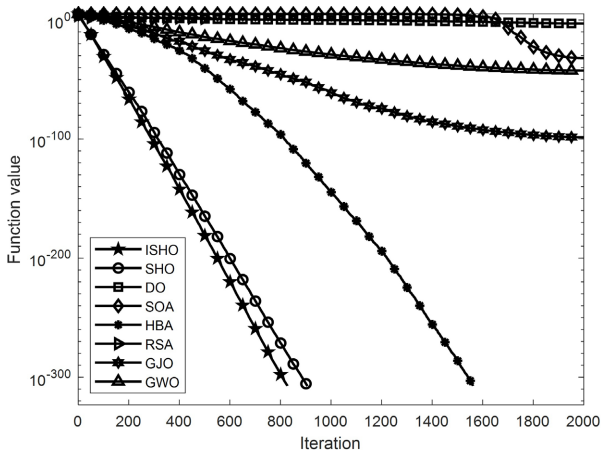
为了更加直观地看出 ISHO 与对比算法的区别，画出各对比算法在所选测试函数下寻优的收敛曲线对比图(图 3)，ISHO 的收敛曲线用标有五角星的曲线表示。



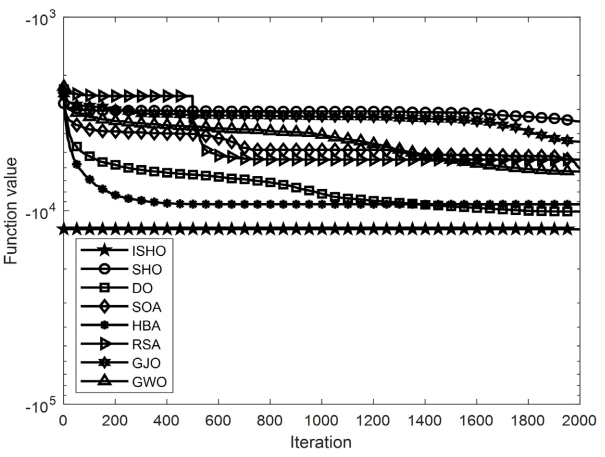
(F1)



(F2)



(F3)



(F4)

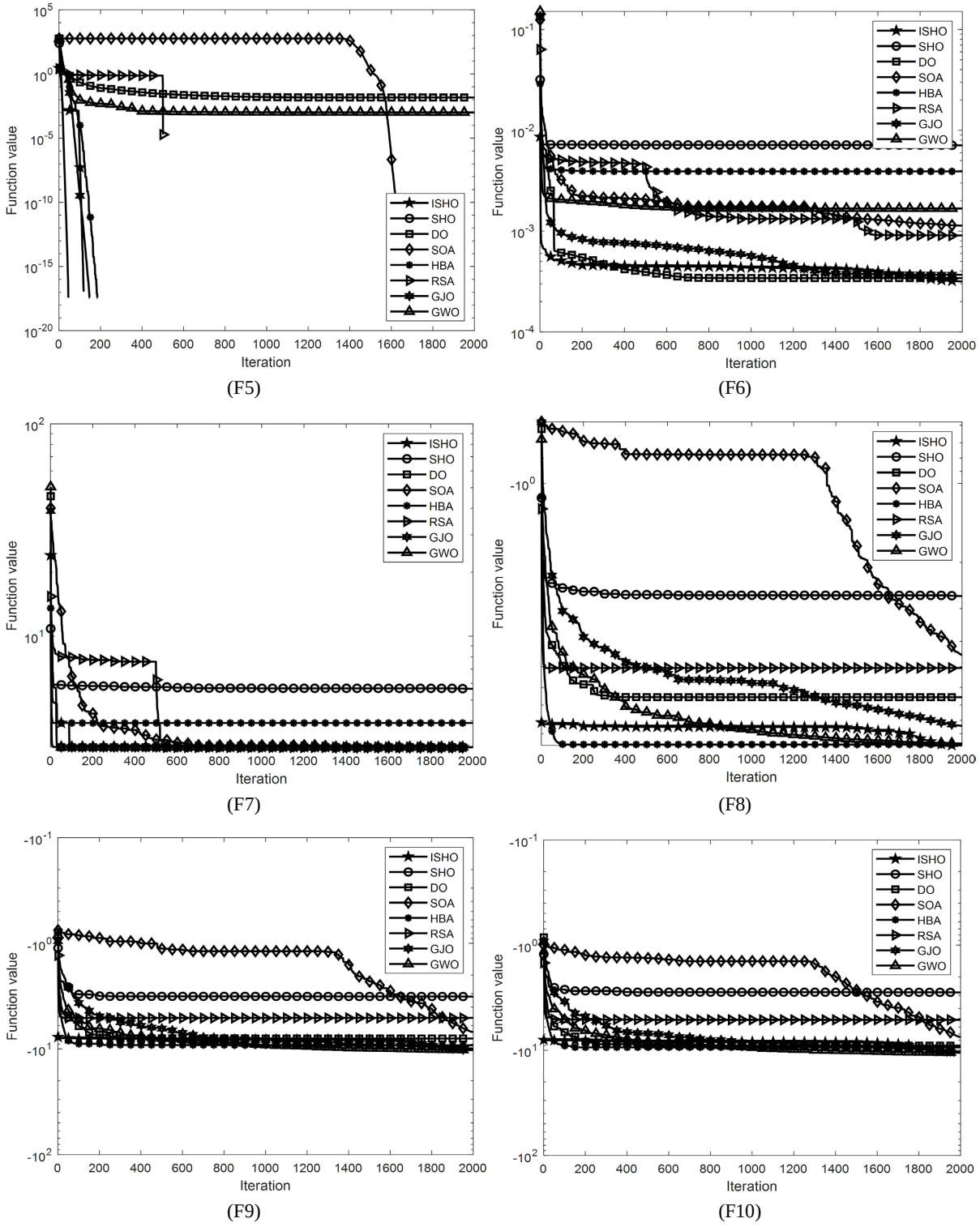


Figure 3. Convergence curves of the comparison algorithms under the test function
图 3. 对比算法在测试函数下的收敛曲线图

在单峰函数 F1~F3 的收敛曲线图中，SHO 和 ISHO 明显在其他对比算法的曲线下方，说明标准 SHO 算法在处理单峰函数时就相对有较快的收敛速度和较高的寻优精度，ISHO 的曲线在 SHO 下方，说明算

法性能又有了提升。在 F6 中，ISHO 率先开始收敛，尽管 DO 算法在第 400 次迭代后有超过 ISHO 的趋势，但是 ISHO 保持较高的寻优精度，在第 1800 次迭代左右赶超 DO，排名第一。在 F7 中，尽管 ISHO 没有排名第一，但是收敛曲线紧挨着排名在前的曲线，与标准 SHO 形成明显对比。在 F8 处，ISHO 最先开始收敛，紧跟着 100 次迭代附近 HBA 收敛到较高精度值附近，在 1900 次迭代附近，ISHO 超过 HBA 成为得到最高寻优精度的算法。在 F9、F10 中，尽管 ISHO 的收敛曲线不是在最下方的，但是仍然接近理想最优值，排名靠前，与标准 SHO 的曲线对比明显。综合来看，ISHO 以较快的收敛速度达到最优值，在个别函数中给与更多的迭代次数能得到更高的寻优精度，与 SHO 收敛曲线对比明显，证明了改进的有效性。

4.1.3. 统计检验

除了可以利用收敛曲线来观察算法的性能，Wilcoxon 秩和检验是一种具有权威性的方法[12]，常用于检验所改进的算法相较于对比算法是否有显著性优势。显著性检验的标准一般设置为 0.05，当检验结果低于 0.05 就说明存在显著性优势，反之不显著或无优势。将各个对比算法在这 10 个所选函数下的寻优结果与本文改进 ISHO 的结果进行 Wilcoxon 秩和检验，将检验结果列在表 4 中。

Table 4. Rank sum test results under different test functions
表 4. 不同测试函数下的秩和检验结果

函数	SHO	DO	SOA	HBA	RSA	GJO	GWO
F1	2.36E−01	0.00E+00	0.00E+00	6.87E−83	7.62E−01	5.8E−308	0.00E+00
F2	1.70E−03	0.00E+00	0.00E+00	1.5E−135	1.25E−60	7.1E−288	0.00E+00
F3	1.90E−02	0.00E+00	0.00E+00	5.7E−139	6.58E−04	0.00E+00	0.00E+00
F8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
F11	3.04E−09	0.00E+00	0.00E+00	7.18E−05	1.08E−70	7.34E−02	0.00E+00
F15	0.00E+00	5.0E−151	0.00E+00	0.00E+00	0.00E+00	4.98E−75	0.00E+00
F18	0.00E+00	0.00E+00	1.5E−215	0.00E+00	6.71E−81	0.00E+00	1.2E−227
F21	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	7.94E−01
F22	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.45E−01	6.05E−65
F23	0.00E+00	1.3E−212	0.00E+00	0.00E+00	0.00E+00	1.59E−02	5.31E−85

观察表 4 可知，在绝大多数情况下，本文所提 ISHO 相比对比算法有着显著性优势，只有极少数情况例外，如 SHO 和 RSA 在 F1 下的检验结果超过 0.05 的标准，结合表 3 可知，SHO 和 RSA 在 F1 时和 ISHO 一样，均达到了理想值，因此 ISHO 在 F1 情况下相比 SHO 和 RSA 没有显著优势，结合在 F1 的收敛曲线图可知，ISHO 以更早收敛到理想值相比 SHO 和 RSA 有较小优势，同样在 F1 达到理想值的 HBA 由于需要较多的迭代次数才收敛到最优，和 ISHO 有一定的差距，因此 ISHO 相比 HBA 有显著性优势，尽管在 F11 处，ISHO 也达到了理想最优值，但是结合收敛曲线来看，GJO 仅仅在 50 代左右就收敛到了最优值，因此 ISHO 相比 GJO 在 F11 处没有体现出显著优势。在 F21 算法运行到后期，ISHO 和 GWO 的收敛曲线越来越近，因此秩和检验结果没有证明 ISHO 有显著性优势，但是从表 3 可知，ISHO 在 F21 处的寻优精度高于 GWO。同理，在 F22 处，GJO 和 ISHO 的收敛曲线逐渐靠近，因此秩和检验结果大于 0.05，但是从排名上来说，ISHO 排名第三，在寻优精度上保持了较高的水平。总的来说，ISHO 相比其他算法在各个函数上有着显著性优势，在极少数情况下，尽管秩和检验结果超过 0.05，但是 ISHO 仍然

保持较快的收敛速度和较高的寻优精度，是一种有较强竞争力的算法。

4.2. CEC 函数测试实验

在测试算法性能的时候，CEC 测试函数集求解极为困难，对于算法来说更具挑战性。本文选取CEC2013 函数集[13]中的 10 个函数进行测试，所选函数的详细信息在表 5 中给出。

Table 5. The selected CEC2013 functions
表 5. 选取的 CEC2013 函数

编号	函数名	类型	理想值
CEC01	Rotated Weierstrass Function	Basic Multimodal Function	−600
CEC02	Rastrigin’s Function	Basic Multimodal Function	−400
CEC03	Rotated Rastrigin’s Function	Basic Multimodal Function	−300
CEC04	Rotated Katsuura Function	Basic Multimodal Function	200
CEC05	Lunacek Bi_Rastrigin Function	Basic Multimodal Function	300
CEC06	Rotated Lunacek Bi_Rastrigin Function	Basic Multimodal Function	400
CEC07	Composition Function 4 (n = 3, Rotated)	Composition Function	1000
CEC08	Composition Function 5 (n = 3, Rotated)	Composition Function	1100
CEC09	Composition Function 6 (n = 5, Rotated)	Composition Function	1200
CEC10	Composition Function 7 (n = 5, Rotated)	Composition Function	1300

为了进一步检验 SHO 算法的改进效果，将 ISHO 与 SHO 在所选 CEC 函数下进行实验，将实验结果的平均值和标准差在表 6 中给出。

Table 6. The test results of SHO and ISHO under CEC functions
表 6. SHO 和 ISHO 在 CEC 函数下测试结果

函数	ISHO		SHO	
	平均值	标准差	平均值	标准差
CEC01	−5.61E+02	2.25E+00	−5.56E+02	2.33E+00
CEC02	−1.07E+01	1.23E+02	3.51E+02	1.30E+02
CEC03	2.82E+02	8.89E+01	4.04E+02	1.07E+02
CEC04	2.00E+02	1.02E+00	2.02E+02	4.89E-01
CEC05	7.35E+02	1.30E+02	1.06E+03	7.64E+01
CEC06	1.12E+03	7.49E+01	1.21E+03	5.88E+01
CEC07	1.35E+03	5.32E+01	1.41E+03	4.03E+01
CEC08	1.44E+03	1.27E+01	1.50E+03	1.96E+01
CEC09	1.50E+03	9.65E+01	1.59E+03	5.51E+01
CEC10	2.67E+03	8.97E+01	2.90E+03	9.57E+01

从表 6 可以看到，在所选的 10 个 CEC 测试函数的实验结果中，ISHO 的平均值均小于 SHO 的结果，尽管没有达到理想最优值，但是仍能够说明 ISHO 的性能相较于 SHO 有了进一步提升，有了更强的能力应对复杂问题的求解。

5. 总结

本文针对标准 SHO 算法寻优精度不高、收敛较慢等问题，提出了改进 ISHO。在生成初始海马种群的时候，利用 Singer 混沌映射，提高初始海马个体在搜索空间的遍历性和均匀性；在捕食阶段，引用失败者放逐策略，将 10%捕食失败的海马个体转向莱维飞行，有几率大步长跳跃的特性增加算法跳出局部最优值的能力；在海马繁殖行为中，利用非线性变化的边界动态设定雄性海马和雌性海马参与繁殖下一代的影响权重，在防止陷入局部极值的同时提高算法的寻优精度。在算法性能测试中分别用 10 个基准测试函数和 10 个 CEC2013 函数进行实验，并分别与 6 个较新的对比算法和改进前的算法对比，实验结果表明 ISHO 相较于 SHO 在复杂函数改进效果明显，在单峰函数、多峰函数、固定维多峰函数和 CEC2013 函数中均表现出良好的性能，证明了 ISHO 是一种有较强寻优能力的算法。

参考文献

[1] 邓飞, 魏祎璇, 刘奕巧, 王统照. 灰狼优化算法的改进及其应用[J]. 统计与决策, 2023, 39(11): 18-24.

[2] 陈秀锋, 郭玉彤, 吴阅晨, 等. 基于蒲公英算法的多目标信号配时优化方法[J/OL]. 吉林大学学报(工学版): 1-9. https://doi.org/10.13229/j.cnki.jdxbgxb20211420, 2023-08-09.

[3] Dhiman, G. and Kumar, V. (2019) Seagull Optimization Algorithm: Theory and Its Applications for Large-Scale Industrial Engineering Problems. *Knowledge-Based Systems*, **165**, 169-196. https://doi.org/10.1016/j.knosys.2018.11.024

[4] Abualigah, L., Elaziz, M.A., Sumari, P., *et al*. (2022) Reptile Search Algorithm (RSA): A Nature-Inspired Meta-Heuristic Optimizer. *Expert Systems with Applications*, **191**, 116158. https://doi.org/10.1016/j.eswa.2021.116158

[5] 谢豪, 李立君, 廖凯, 高自成. 基于金豺优化算法的 PID 参数优化研究[J]. 现代制造工程, 2023(3): 146-151.

[6] 徐碧阳, 覃涛, 魏巍, 等. 基于多策略改进的蜜獾优化算法[J/OL]. 小型微型计算机系统: 1-14. http://kns.cnki.net/kcms/detail/21.1106.TP.20230207.0901.005.html, 2023-06-20.

[7] Zhao, S., Zhang, T., Ma, S., *et al*. (2022) Sea-Horse Optimizer: A Novel Nature-Inspired Meta-Heuristic for Global Optimization Problems. *Applied Intelligence*, **53**, 11833-11860. https://doi.org/10.1007/s10489-022-03994-3

[8] 张希淼, 马宁, 付伟, 等. 融合混沌映射和二次插值的自适应鲸鱼优化算法[J]. 计算机工程与设计, 2023, 44(4): 1088-1096.

[9] 吴迎晨, 肖彪, 赵正彩, 等. 柔性作业车间调度多策略果蝇优化算法研究[J]. 现代制造工程, 2023(5): 22-30+44.

[10] 冯增喜, 李嘉乐, 葛珣, 等. 融合多策略改进鲸鱼优化算法及其应用[J/OL]. 计算机集成制造系统: 1-23. http://kns.cnki.net/kcms/detail/11.5946.tp.20230104.1215.014.html, 2023-06-20.

[11] Ezugwu, A.E., Agushaka, J.O., Abualigah, L., *et al*. (2022) Prairie Dog Optimization Algorithm. *Neural Computing and Applications*, **34**, 20017-20065. https://doi.org/10.1007/s00521-022-07530-9

[12] 李雪利, 杜逆索, 欧阳智, 等. 基于扰动因子和贪心策略的白骨顶优化算法[J]. 智能计算机与应用, 2023, 13(6): 38-49.

[13] 周新宇, 胡建成, 吴艳林, 等. 基于适应度分组的多策略人工蜂群算法[J]. 模式识别与人工智能, 2022, 35(8): 688-700.