◆算法研究与应用◆

动态透镜成像学习人工兔优化算法及应用*

王　伟[1],龙　文[2**]
(1.贵州财经大学管理科学与工程学院,贵州贵阳　550025;2.贵州财经大学数学与统计学院,贵州贵阳　550025)

摘要:针对基本人工兔优化(Artificial Rabbits Optimization,ARO)算法在解决复杂优化问题时存在收敛慢、精度不高和容易陷入局部最优等缺陷,本文提出一种改进的 ARO 算法(记为 IARO 算法)。IARO 算法中的基于正弦函数的非线性递减能量因子能够帮助算法实现从探索阶段到开发阶段的良好过渡,从而提高算法的收敛速度和解的质量。此外,为了提高算法跳出局部最优的概率,IARO 算法引入了一种动态透镜成像学习策略。为了证明 IARO 算法的优越性,首先选取了 6 个基准测试函数进行数值实验,然后用其求解 2 个工程设计优化问题和 1 个包括 15 个数据集的特征选择问题,并与灰狼优化(GWO)算法、鲸鱼优化算法(WOA)、正弦余弦算法(SCA)和基本 ARO 算法进行对比。结果表明,IARO 算法有着比其他对比算法更优越的性能。

关键词:人工兔优化算法;动态透镜成像学习策略;工程优化;特征选择;函数优化

中图分类号:TP18　　**文献标识码**:A　　**文章编号**:1005-9164(2023)04-0735-10
DOI:10.13656/j.cnki.gxkx.20230928.013

在过去的几十年中,元启发式算法在处理各种工程领域中具有挑战性的优化问题方面越来越受欢迎,这是因为它比传统的数值方法更有效。元启发式算法在搜索过程中需要执行探索和开发两个关键任务。在探索任务中进行“宏观搜索”以找到解空间中最有潜力的区域,对应于在问题的可行解空间中进行全局搜索,以搜索全局最优解。在开发任务中执行“微搜索”以提取搜索区域中的有用信息,对应于局部搜索,目的是改进可用的最佳解决方案。由于这两个过程的矛盾性质,所以在任何元启发式算法中都应该保持探索和开发之间的平衡,以执行性能良好的搜索。若未处理好探索和开发的平衡则会导致跳过真解、局部最优停滞和过早收敛等问题。

目前流行的元启发式算法有灰狼优化(Grey Wolf Optimizer,GWO)算法[1]、鲸鱼优化算法(Whale Optimization Algorithm,WOA)[2]、正弦余弦算法(Sine Cosine Algorithm,SCA)[3]、蝗虫优化算法(Grasshopper Optimization Algorithm,

收稿日期:2023-02-10　　修回日期:2023-03-29
*国家自然科学基金项目(12361106),贵州省教育厅创新群体项目(黔教合 KY 字[2021]015)和贵州省自然科学基金重点项目(黔科合基础-ZK[2023]重点 003)资助。
【第一作者简介】
王　伟(1983-),男,硕士,讲师,主要从事智能优化算法及应用研究。
【**通信作者】
龙　文(1977-),男,博士,教授,博士研究生导师,主要从事计算智能研究,E-mail:lw227@mail.gufe.edu.cn。
【引用本文】
王伟,龙文.动态透镜成像学习人工兔优化算法及应用[J].广西科学,2023,30(4):735-744.
WANG W,LONG W. Dynamic Lens Imaging Learning Artificial Rabbits Optimization Algorithm and Its Applications [J]. Guangxi Sciences,2023,30(4):735-744.

GOA)[4]、哈里斯鹰优化(Harris Hawks Optimizer,HHO)算法[5]、海鸥优化算法(Seagull Optimization Algorithm,SOA)[6]、蝴蝶优化算法(Butterfly Optimization Algorithm,BOA)[7]等,它们在解决全局优化问题上显示出一定的有效性。这些算法是基于种群迭代的,并且能够同时使用多个个体来执行搜索。此外,算法中的个体在搜索过程中还能够分享经验,相互交流。这些特性有助于算法在搜索过程中避免局部最优,从而增强了算法的探索能力。

人工兔优化(Artificial Rabbits Optimization,ARO)算法是由 Wang 等[8]于 2022 年提出的一种新型元启发式算法,它的灵感来自自然界中兔子的生存策略,包括迂回觅食和随机隐藏。ARO 算法具有数学模型简单、需调整的参数少、容易编程实现以及不依赖梯度信息等特点,在函数优化和工程优化领域中得到成功应用[8-11]。然而,基本 ARO 算法存在精度差、收敛慢以及容易陷入局部最优等缺点。为了改善 ARO 算法的性能,Wang 等[12]首先在随机隐藏阶段引入了 Lévy 飞行策略,以提高种群的多样性和动态性,种群的多样性加深了全局探索过程,从而提高了算法的收敛精度;然后,通过引入选择性反向策略来提高跟踪效率,防止 ARO 陷入当前的局部解。Wang 等[13]结合天鹰优化(Aquila Optimizer,AO)算法的全局勘探和 ARO 算法的局部开发能力,设计了自适应切换机制,引入混沌反向学习策略,提出一种混合算法。

虽然这些元启发式算法在某些问题上可以找到全局最优,但是没有算法可以找到所有优化问题的全局最优值。无免费午餐(No Free Lunch,NFL)定理指出,不可能存在适用于所有优化问题的理想算法,该定理使得元启发式算法领域的研究非常活跃,并允许对现有算法进行修改以提高其性能[14]。为了改善基本 ARO 算法的搜索能力,本文提出一种改进的 ARO 算法(称为 IARO 算法),不仅设计了一种基于正弦函数的非线性递减能量因子以平衡 IARO 算法的探索和开发能力,还在笔者前期工作的基础上提出了一种动态透镜成像学习策略以避免算法陷入局部最优。数值实验部分选取了 6 个基准测试函数、2 个工程设计优化问题和 1 个包括 15 个数据集的特征选择问题对 IARO 算法进行性能测试。

1 基本 ARO 算法

ARO 算法模拟了兔子的生存策略,通过搜索、觅食和躲避攻击执行优化过程。与其他元启发式算法一样,ARO 算法通过式(1)随机产生一组解组成初始种群个体:

$$P = lb + rand \cdot (ub - lb), \tag{1}$$

其中,ub 和 lb 是解空间的上、下界,$rand$ 是[0,1]间的随机数。

ARO 算法中的优化过程主要包括两个阶段,即全局勘探(迂回觅食)和局部开发(随机隐藏),具体数学模型描述如下:

在迂回觅食阶段,假设种群中的每只兔子都有自己的区域,有一些草和洞穴,兔子总是随机地访问彼此的位置觅食。事实上,在觅食时,兔子很可能会扰乱食物来源以获取足够的食物。因此,ARO 算法的迂回觅食行为意味着每个搜索个体倾向于使用向群中随机选择的其他搜索个体的位置来更新其位置,并添加扰动。迂回觅食策略的数学模型为

$$X_i(t+1) = X_j(t) + R \cdot (X_i(t) - X_j(t)) + \mathrm{round}(0.5 \cdot (0.05 + r_1)) \cdot r_2, \tag{2}$$

$$R = (e - e^{(\frac{t-1}{T})^2}) \cdot \sin(2\pi r_3) \cdot c, \tag{3}$$

式中,X 表示个体的位置;$i, j = 1, 2, \cdots, N$,$i \neq j$,N 为种群规模;r_1、r_2 和 r_3 分别是[0,1]间的随机数;t 为当前迭代次数;T 为最大迭代次数;R 表示移动步长;$k = 1, 2, \cdots, d$,d 是问题的维数;c 的取值为 0 或 1,round 为四舍五入取整函数。

在随机隐藏阶段,为了躲避捕食者,兔子通常会在巢穴周围挖一些不同的洞来躲藏。在 ARO 算法的每次迭代中,兔子总是沿着搜索空间的每个维度在其周围产生若干个洞穴,并且总是从所有洞穴中随机选择一个进行隐藏,以降低被捕食的概率,具体数学模型为

$$X_i(t+1) = X_i(t) + R \cdot (r_4 \cdot H \cdot g \cdot X_i(t) - X_i(t)), \tag{4}$$

$$H = \frac{T - t + 1}{T} \cdot r_5, \tag{5}$$

式中,H 为隐藏参数,在迭代过程中随机从 1 线性减小到 $1/T$,r_4 和 r_5 分别为[0,1]间的随机数,g 的取值为 0 或 1。

在 ARO 算法中,兔子总是倾向于在迭代初始阶段频繁地进行迂回觅食,而在迭代后期阶段频繁地进行随机隐藏,这种搜索机制取决于兔子的能量,它会随着时间的推移而逐渐收缩。因此,设计一个能量因子来模拟从勘探阶段到开发阶段的自动转换,能量因子 A 的定义如下:

$$A(t)=4\cdot(1-\frac{t}{T})\cdot\ln\frac{1}{r_6},\tag{6}$$

式中,r_6 是一个[0,1]间的随机数。

2 改进 ARO 算法(IARO 算法)

2.1 非线性递减能量因子

在搜索过程中平衡好全局勘探和局部开发能力对提高元启发式算法的搜索性能至关重要。一般来说,算法搜索前期的主要目的是在解空间中进行大范围搜索,以找到潜在的可行区域;而在搜索中后期,算法主要执行局部精确搜索,以加快收敛速度。不难发现,基本 ARO 算法设计了一种能量因子 A 以控制其在全局勘探和局部开发之间的转换。能量因子 A 较大表示兔子有足够的能量和体力进行迂回觅食;反之,能量因子 A 较小则表明兔子的体力不足,需要随机隐藏。即,当 $A>1$ 时,兔子在探索阶段随机探索不同兔子的觅食区域,种群进行全局搜索;当 $A\leqslant1$ 时,兔子倾向于在开发阶段随机开发自己的洞穴,进行局部搜索。

由式(6)可知,虽然能量因子 A 的值随迭代次数的增加而随机扰动递减至 0,但这不能真实反映 ARO 算法的优化过程,即前期进行全局搜索需要较大的 A 值,而中后期局部精确搜索则需要较小的 A 值。因此,本文设计了一种基于正弦函数的非线性递减能量因子,其数学公式为

$$A(t)=(A_{\max}-A_{\min})\times(1-\sin((\frac{t}{T})^{\mu}\times\frac{\pi}{2})),\tag{7}$$

式中,$A_{\max}$ 和 $A_{\min}$ 分别为能量因子 A 的最大值和最小值,μ 为缩放因子。图 1 为式(7)描述的能量因子 A 随迭代次数增加的变化曲线。由图 1 可以看出,在算法搜索前期,能量因子 A 取较大值;而在搜索中后期,A 取较小值,符合ARO算法前期进行全局勘探、后期进行局部开发的特点。

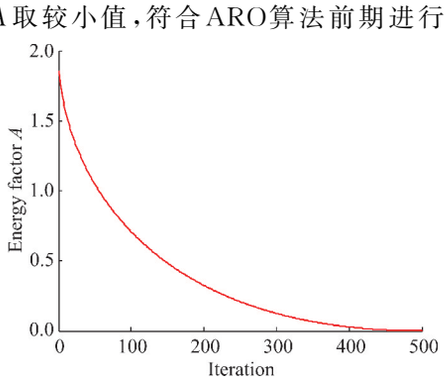


图 1 能量因子 A 的变化曲线

Fig. 1 Variation curve of energy factor A

2.2 动态透镜成像学习策略

在基本 ARO 算法搜索末期,所有个体均向当前群体中最优个体所在区域靠拢,导致群体多样性降低,算法很容易陷入局部最优,这也是元启发式算法的固有缺点。为了克服这个缺点,龙文等[15]提出一种基于凸透镜成像原理的反向学习策略用于改进 GWO 算法的性能。然而,固定的缩放因子 k 无法充分利用透镜成像学习策略的优势。因此,为了进一步增加群体多样性,本文提出一种动态透镜成像学习策略,如图 2 所示。

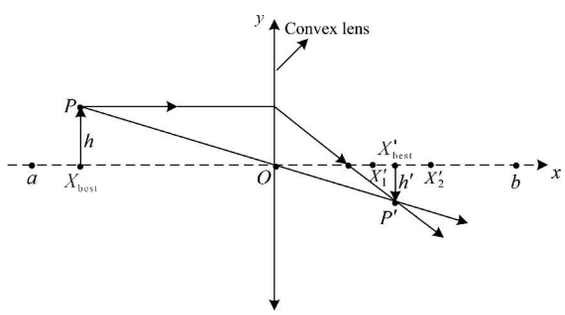


图 2 动态透镜成像学习策略

Fig. 2 Dynamic lens imaging learning strategy

在图 2 所示的动态透镜成像学习策略中,O 是区间[a,b]的中点,h 是光源 P 的高度,h'是光源 P 的像 P'的高度,X_{best} 是当前群体中的最优个体,X'_{best} 为 X_{best} 的反向个体。由图 2 可以得出并推广到 m 维空间中:

$$X'_{best,j}=\frac{a_j+b_j}{2}+\frac{a_j+b_j}{2k}-\frac{X_{best,j}}{k},\tag{8}$$

式中,$k=2\cdot r_7$,r_7 是[0,1]之间的随机数,$j=1,2,\cdots,m$。

由图 2 和式(8)可知,当 k 取不同的值,得到不同的 X'_{best},可以增加种群的多样性,从而避免算法陷入局部最优,即

$$X'_{best}=\begin{cases}X'_1, & 0<k<1\\ a+b-X_{best}, & k=1\\ X'_2, & k>1\end{cases},\tag{9}$$

基于上述两个修改策略,本文提出的 IARO 算法的流程如图 3 所示,迭代步骤如下:

步骤 1:初始化算法参数,如种群规模 N、最大迭代次数 T、$A_{\max}$、$A_{\min}$、μ、k;

步骤 2:在解空间中随机产生一组个体位置作为算法种群初始位置,令 $t=1$;

步骤 3:计算每个个体的目标函数值,并记录当

前群体最优个体位置;

步骤 4:判断 $t<T$,如果是,则转到步骤 5;否则,算法结束,输出最优解;

步骤 5:根据式(7)计算能量因子 A 的值;

步骤 6:判断 $A>1$,如果是,则根据式(2)更新个体位置;否则,由式(4)更新个体位置;

步骤 7:针对当前最优个体,执行动态透镜成像学习策略,以产生新候选个体,令 $t=t+1$,返回步骤 3。

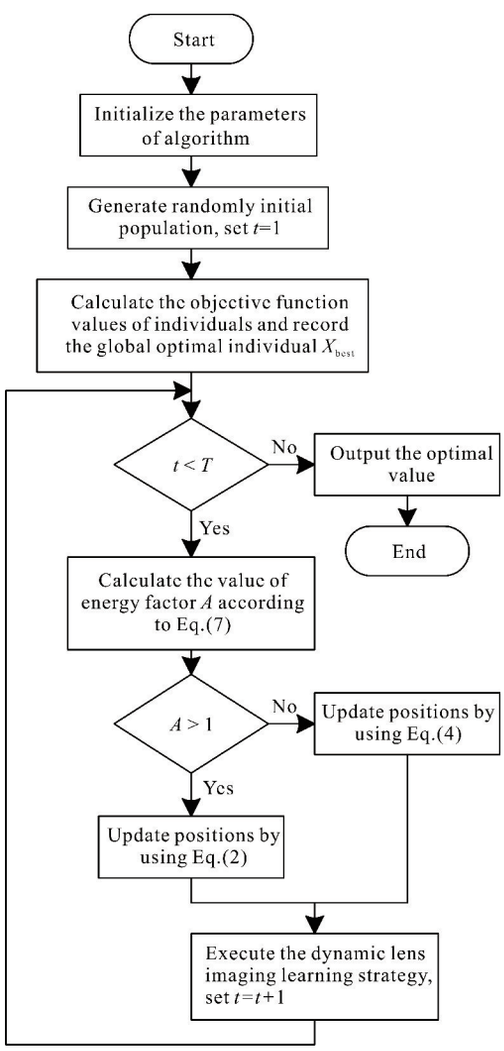


图 3 IARO 算法流程

Fig. 3 Flow chart of the IARO algorithm

3 函数优化问题测试及比较

为了测试 IARO 算法的优化能力,本文选取 6 个基准函数进行数值实验,详细信息如表 1 所示。其中 f_1、f_2 和 f_3 是单峰函数,通常用来测试算法的局部开发能力;多峰函数 f_4、f_5 和 f_6 用于研究算法的全局勘探能力。6 个基准测试函数的全局最优值均为 0。

表 1 6 个基准测试函数

Table 1 Six benchmark test functions

函数表达式 Function expression	搜索区间 Search interval
$f_1(x)=\sum_{i=1}^{D}x_i^2$	[-100,100]
$f_2(x)=\sum_{i=1}^{D}ix_i^2$	[-10,10]
$f_3(x)=\sum_{i=1}^{D}(10^6)^{(i-1)/(D-1)}x_i^2$	[-100,100]
$f_4(x)=\sum_{i=1}^{D}\lvert x_i\cdot\sin(x_i)+0.1\cdot x_i\rvert$	[-10,10]
$f_5(x)=\sum_{i=1}^{D}(0.2x_i^2+0.1x_i^2\cdot\sin(2x_i))$	[-10,10]
$f_6(x)=\sum_{i=1}^{D-1}(x_i^2+2x_{i+1}^2)^{0.25}\cdot((\sin 50(x_i^2+x_{i+1}^2)^{0.1})^2+1)$	[-10,10]

采用 IARO 算法对表 1 中的 6 个基准测试函数进行求解,并与 GWO 算法[1]、WOA[2]、SCA[3] 和基本 ARO 算法[8]进行比较。5 种算法的种群规模设为 30,最大迭代次数设为 500。每个函数的维数均设置为 30。每种算法均对每个基准测试函数独立运行 30 次。

表 2 列出了 5 种算法对 6 个基准测试函数的平均值(Mean)和标准差(Standard deviation,St. dev)的结果。所有算法均在 MATLAB R2014a 软件上实现,操作系统为 Microsoft windows 8 64 位版,处理器为 Intel (R) Core (TM) i5-5200U CPU @ 2.20 GHz,内存为 4 GB RAM。

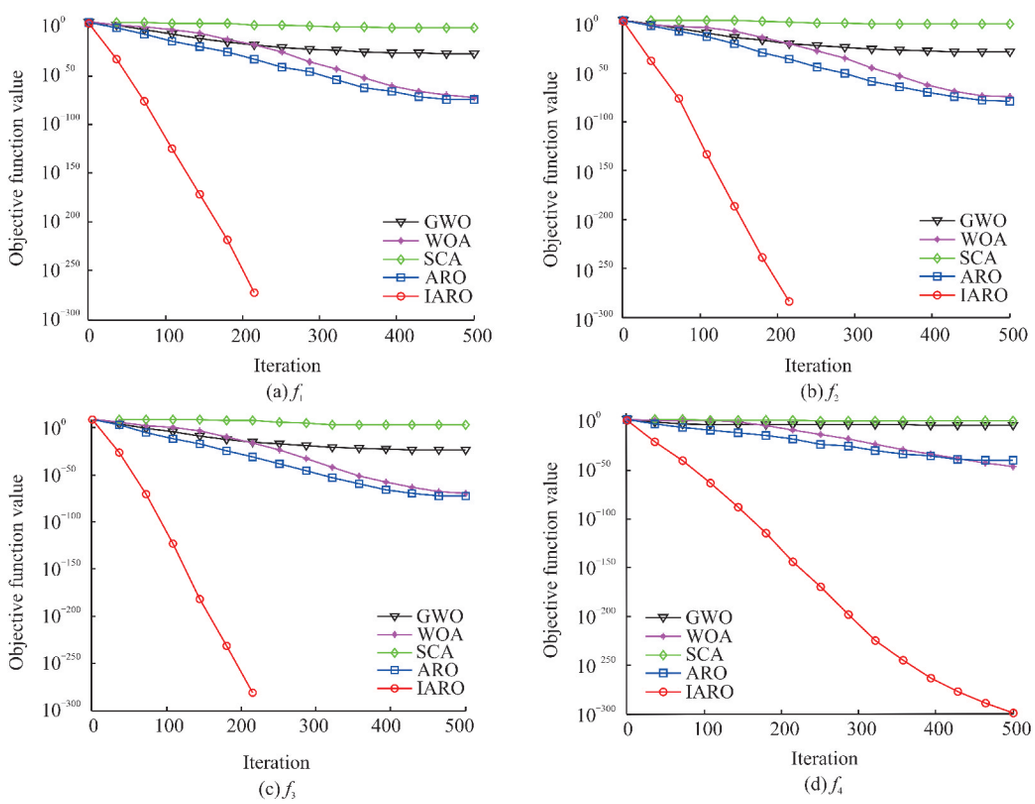
从表 2 可知,除了 f_4 外,IARO 算法在其他 5 个基准测试函数上均获得了理论最优值(0),在 f_4 上的寻优结果也非常接近于 0。在 5 种比较算法中,SCA 的寻优性能最差。与其他 4 种算法相比,IARO 算法在 6 个基准测试函数上均获得了较好的平均值,且具有较强的鲁棒性。关于 5 种算法对 6 个函数的 Friedman 检验结果的排名,IARO 算法在目标函数平均值上排名第一,接下来依次是 ARO 算法、WOA、GWO 算法和 SCA。另外,图 4 是 5 种算法对 6 个基准测试函数的迭代收敛曲线。从图 4 可以看出,与 ARO 算法、WOA、GWO 算法和 SCA 相比,IARO 算法在 6 个基准测试函数上显示出更快的收敛速度和更高的收敛精度。

表 2　5 种算法在 6 个基准测试函数的比较结果

Table 2　Comparison results of five algorithms on six benchmark test functions

函数 Function	GWO		WOA		SCA		ARO		IARO	
	均值 Mean	标准差 St. dev	均值 Mean	标准差 St. dev	均值 Mean	标准差 St. dev	均值 Mean	标准差 St. dev	均值 Mean	标准差 St. dev
f_1	8.52E－28	8.80E－28	5.48E－73	1.24E－72	3.94E－01	2.38E－01	3.66E－75	3.97E－75	**0.00E＋00**	0.00E＋00
f_2	2.46E－28	2.03E－28	2.70E－75	1.73E－75	7.73E－01	6.50E－01	1.57E－79	5.09E－79	**0.00E＋00**	0.00E＋00
f_3	7.96E－25	8.77E－25	2.76E－70	4.12E－70	7.88E＋02	3.76E＋02	1.55E－73	2.78E－73	**0.00E＋00**	0.00E＋00
f_4	4.92E－04	1.28E－03	1.32E－47	7.08E－48	7.12E－01	1.21E＋00	4.59E－41	6.71E－41	**3.99E－300**	0.00E＋00
f_5	1.72E－29	2.21E－29	6.62E－76	6.28E－76	8.42E－02	1.42E－01	2.24E－79	9.85E－79	**0.00E＋00**	0.00E＋00
f_6	6.80E－07	3.84E－07	6.63E－30	5.34E－30	2.20E＋00	2.04E＋00	4.02E－22	4.51E－22	**0.00E＋00**	0.00E＋00
The rank of Friedman test result	4		3		5		2		1	

Note：the best results obtained by five algorithms are marked in bold.



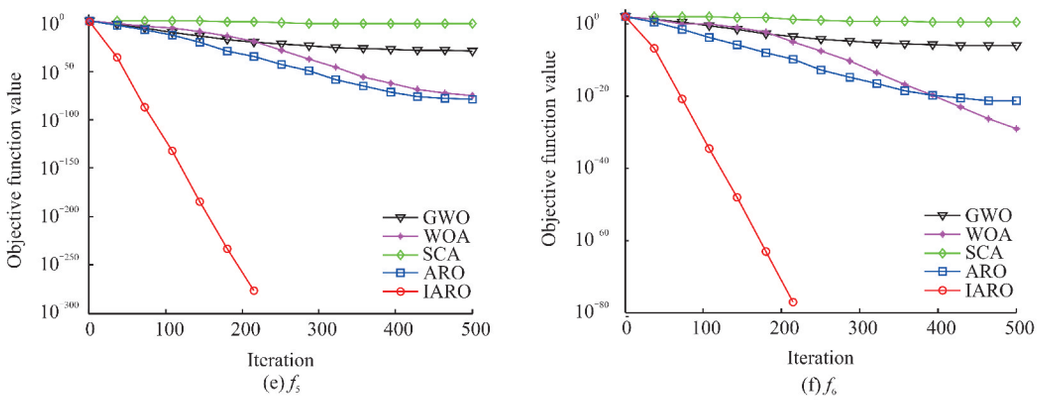


图 4　5 种算法在 6 个函数上的收敛曲线

Fig. 4　Convergence curves of five algorithms on six functions

为了分析两种改进策略的有效性,将基本 ARO 算法与仅采用非线性能量递减因子的 ARO(NARO)算法、仅采用动态透镜成像学习策略的 ARO(DARO)算法和 IARO 算法进行比较,选取表 1 中的 6 个测试函数进行实验,结果如表 3 所示。

表 3　4 种算法的平均结果比较

Table 3　Means of four algorithms

函数 Function	ARO	NARO	DARO	IARO
f_1	3.66E−75	4.29E−80	**0.00E+00**	**0.00E+00**
f_2	1.57E−79	1.82E−83	**0.00E+00**	**0.00E+00**
f_3	1.55E−73	6.01E−79	**0.00E+00**	**0.00E+00**
f_4	4.59E−41	4.11E−48	2.46E−282	**3.99E−300**
f_5	2.24E−79	2.99E−82	**0.00E+00**	**0.00E+00**
f_6	4.02E−22	4.96E−23	**0.00E+00**	**0.00E+00**

Note: the best results obtained by four algorithms are marked in bold.

从表 3 的比较结果可知,仅采用非线性能量递减因子对改进 ARO 算法性能的帮助有限,IARO 算法性能改进的有效算子是引入动态透镜成像学习策略。

4　IARO 算法求解工程设计优化问题

为了进一步验证 IARO 算法的有效性,将其应用于求解两个工程设计优化问题,即压力容器设计优化和拉压弹簧设计优化问题。考虑 IARO 算法在解决这两个工程设计优化问题时需要结合约束处理技术,本文使用惩罚函数法[16]处理约束条件。这里利用 GWO 算法、WOA、SCA、ARO 算法和 IARO 算法对两个工程设计优化问题进行求解,5 种算法的种群规模设置为 50,最大迭代次数设置为 1 000。

4.1　压力容器设计

压力容器设计的结构如图 5 所示,它有 4 个设计变量,即压力容器厚度、封帽厚度、容器内径和容器长度,优化目标是在一定约束条件下使得容器总造价最低,其目标函数和约束条件的数学表达式如下:

$$\begin{aligned}
&\min \quad f = 0.6224T_sRL + 1.7781T_hR^2 + 3.1661T_s^2L + 19.84T_s^2R,\\
&\text{s.t.} \quad g_1 = T_s + 0.0193R \leqslant 0,\\
&\qquad g_2 = T_h + 0.00954R \leqslant 0,\\
&\qquad g_3 = -\pi R^2L + \frac{4}{3\pi R^3 + 1296000} \leqslant 0,\\
&\qquad g_4 = L - 240 \leqslant 0, \qquad (10)
\end{aligned}$$

式中,$0 \leqslant T_s, T_h \leqslant 99, 10 \leqslant R, L \leqslant 200$。

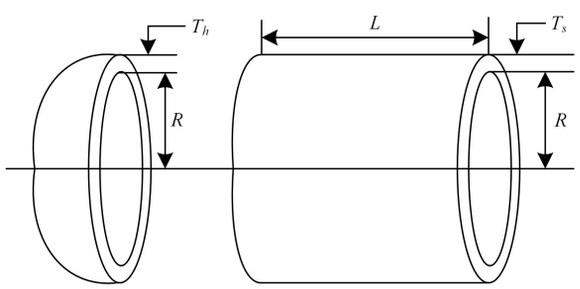


图 5　压力容器设计

Fig. 5　Pressure vessel design

GWO 算法、WOA、SCA、ARO 算法和 IARO 算法在压力容器设计优化问题上获得的最优结果如表 4 所示。从表 4 的比较结果可知,与 GWO 算法、WOA、SCA 和 ARO 算法相比,在相同参数设置的条件下,IARO 算法在压力容器设计优化问题上获得了最低的总造价。

表 4　5 种算法的最优结果比较

Table 4　The optimal results of five algorithms

算法 Algorithm	T_s	T_h	R	L	f
GWO	13.246 5	7.008 4	42.096 6	176.668 0	6 060.144 7
WOA	12.702 4	7.306 8	41.948 5	178.535 0	6 078.796 9
SCA	12.690 3	6.715 9	42.074 6	200.000 0	6 601.644 8
ARO	0.786 4	0.388 7	40.739 3	194.248 0	5 900.114 3
IARO	0.778 2	0.384 7	40.322 2	199.968 0	**5 885.525 7**

Note: the best results obtained by five algorithms are marked in bold.

4.2　拉压管柱弹簧设计

拉压管柱弹簧设计结构如图 6 所示，它有 3 个设计变量如管柱直径、平均线圈直径和有效线圈数。

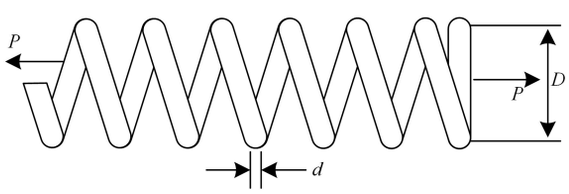


图 6　拉压管柱弹簧设计

Fig. 6　Tension/compression string design

拉压管柱弹簧设计优化问题的目标是在最小挠度、剪切应力、外径限制和设计变量的约束下，使管柱弹簧的重量最小化，其目标函数和约束条件的数学表达式为

$$\begin{aligned}&\min\ f=(P+2)Dd^2,\\&\text{s.t.}\ \ g_1=1-\frac{PD^3}{71785d^4}\leqslant 0,\\&\qquad g_2=\frac{4D^3-dD}{12566(Dd^3-d^4)}+\frac{1}{5108d^2}-1\leqslant 0,\\&\qquad g_3=1-\frac{140.45d}{D^2P}\leqslant 0,\\&\qquad g_4=\frac{d+D}{1.5}-1\leqslant 0。\end{aligned}\tag{11}$$

式中，$0.05\leqslant d\leqslant 2.00$，$0.25\leqslant D\leqslant 1.30$，$2\leqslant P\leqslant 15$。

GWO 算法、WOA、SCA、ARO 算法和 IARO 算法在拉压管柱弹簧设计优化问题上获得的最优结果如表 5 所示。由表 5 的比较结果可以看出，与 GWO 算法、WOA、SCA 和 ARO 算法相比，IARO 算法在拉压管柱弹簧设计优化问题上获得了最小的重量。

表 5　5 种算法的最优结果比较

Table 5　The optimal results of five algorithms

算法 Algorithm	d	D	P	f
GWO	0.050 640	0.331 997	12.910 300	0.012 694
WOA	0.051 490	0.351 878	11.587 000	0.012 675
SCA	0.050 000	0.316 543	14.149 800	0.012 780
ARO	0.051 900	0.361 804	11.000 000	0.012 670
IARO	0.051 890	0.361 749	11.000 000	**0.012 666**

Note: the best results obtained by five algorithms are marked in bold.

5　IARO 算法求解特征选择问题

特征选择是处理高维数据的一个重要环节，它是指从原数据集的 M 个特征中选择出 N 个特征以降低数据集的维数，从而提高机器学习模型的分类效率。特征选择通过数学建模后可转换为求解一个多目标函数优化问题，其目标(适应度)函数可描述为

$$\text{fitness}=(1-\rho)\times E+\rho\times\frac{|l|}{|L|},\tag{12}$$

其中，E 为分类器的分类错误率，ρ 是权重系数，$|l|$ 表示算法所选择的特征数，$|L|$ 为原始数据集的特征数。

为了测试 IARO 算法求解特征选择问题的有效性，本文从 UCI 数据库中选取 15 个数据集进行实验，详细信息如表 6 所示。

表 6　实验中使用的 15 个 UCI 数据集

Table 6　Fifteen UCI datasets used in the experiment

数据集 Dataset	特征数 Number of feature	样本大小 Sample size
BreastEW	30	569
Clean1	166	476
Clean2	166	6 598
Exactly	13	1 000
Exactly2	13	1 000
HeartEW	13	270

续表

Continued table

数据集 Dataset	特征数 Number of feature	样本大小 Sample size
IonosphereEW	34	351
KrvskpEW	36	3 196
M-of-n	13	1 000
PenglungEW	325	73
Semeion	265	1 593
SonarEW	60	208
SpectEW	22	267
Tic-tac-toe	9	958
WaveformEW	40	5 000

利用 GWO 算法、WOA、SCA、ARO 算法和 IARO 算法对表 6 中的 15 个 UCI 数据集进行特征选择并进行比较。5 种算法的种群规模为 10 和最大迭代次数为 20。在每个数据集中,随机选择 90%作为训练集,剩下 10%作为测试集,以 k 最近邻模型为分类模型,$k=3$,各种算法分别对每个数据集单独运行 10 次,得到的平均分类准确率和平均所选特征数分别见表 7 和表 8。

从表 7 的结果可知,IARO 算法在 BreastEW、KrvskpEW、PenglungEW 和 SonarEW 数据集上的平均分类准确率达到100%。与GWO算法相比,

表 7　5 种算法的分类准确率比较

Table 7　Classification accuracy of five algorithms

数据集 Dataset	GWO	WOA	SCA	ARO	IARO
BreastEW	0.975 0	0.957 1	0.982 1	0.975 0	**1.000 0**
Clean1	0.940 4	0.910 6	0.919 1	0.927 7	**0.983 0**
Clean2	**0.986 3**	0.977 2	0.980 6	0.983 3	0.981 8
Exactly	**0.922 0**	0.826 0	0.716 0	0.788 0	0.868 0
Exactly2	**0.790 0**	0.782 0	0.780 0	**0.790 0**	**0.790 0**
HeartEW	0.866 7	0.844 4	0.874 1	0.881 5	**0.897 0**
IonosphereEW	0.960 0	0.942 9	**0.965 7**	**0.965 7**	0.942 9
KrvskpEW	0.971 4	0.928 6	0.914 3	**1.000 0**	**1.000 0**
M-of-n	0.938 0	0.932 0	0.978 0	**0.996 0**	0.950 0
PenglungEW	0.885 7	0.828 6	0.885 7	0.914 3	**1.000 0**
Semeion	**0.989 9**	0.986 2	0.986 2	0.987 4	0.987 4
SonarEW	0.990 0	0.940 0	0.980 0	**1.000 0**	**1.000 0**
SpectEW	**0.923 1**	0.9077	**0.923 1**	0.915 4	**0.923 1**
Tic-tac-toe	**0.827 4**	0.814 7	0.821 1	**0.827 4**	**0.827 4**
WaveformEW	**0.818 4**	0.779 6	0.790 0	0.796 0	0.800 0

Note: the best results obtained by five algorithms are marked in bold.

表 8　5 种算法的所选特征数比较

Table 8　The selected feature number of five algorithms

数据集 Dataset	GWO	WOA	SCA	ARO	IARO
BreastEW	7.8	10.8	**6.6**	10.2	11.0
Clean1	64.2	78.4	**51.0**	63.2	59.8
Clean2	68.6	100.0	**57.2**	80.0	86.4
Exactly	7.4	8.8	**6.0**	7.4	7.0
Exactly2	6.6	5.8	**4.6**	5.2	5.8
HeartEW	5.8	5.6	5.0	5.4	**3.6**
IonosphereEW	6.4	**5.0**	5.6	10.0	10.6
KrvskpEW	16.4	32.6	**15.2**	17.8	19.8
M-of-n	**6.4**	9.2	6.6	7.0	9.0

续表

Continued table

数据集 Dataset	GWO	WOA	SCA	ARO	IARO
PenglungEW	94.6	67.6	69.0	75.8	**29.6**
Semeion	105.6	124.6	**82.6**	121.0	115.4
SonarEW	19.0	17.8	**13.8**	20.4	22.6
SpectEW	4.4	8.2	**4.0**	6.0	6.8
Tic-tac-toe	6.0	7.2	6.0	6.0	**5.0**
WaveformEW	18.6	28.2	**14.0**	23.4	22.0

Note: the best results obtained by five algorithms are marked in bold.

IARO 算法在 8 个数据集上分类更为准确，在 Exactly2、SpectEW 和 Tic-tac-toe 数据集上的分类能力相当，在 Clean2、Exactly、Semeion 和 WaveformEW 数据集上，GWO 算法分类更加准确。除 IonosphereEW 数据集外，IARO 算法在其他 14 个数据集上比 WOA 分类更加准确，并且在 IonosphereEW 数据集上二者的分类准确率相同。与 SCA 相比，IARO 算法在 12 个数据集上的分类能力更强，在 SpectEW 数据集上二者的分类准确率相等；但是，SCA 在 IonosphereEW 和 M-of-n 数据集上的分类能力更强。与 ARO 算法相比，IARO 算法分别在 7 个数据集上分类表现更好，在 Exactly2、KrvskpEW、Semeion、SonarEW 和 Tic-tac-toe 数据集上二者的分类准确率相同；在数据集 Clean2、IonosphereEW 和 M-of-n 上，ARO 算法的分类能力更强。

6 结论

ARO 算法是最近提出的一种元启发式优化技术，它在经典基准测试函数上显示出强大的寻优性能。然而，基本 ARO 算法在求解复杂优化问题上存在容易陷入局部最优、后期收敛速度慢等缺点。为了平衡算法的全局勘探和局部开发能力，本研究不仅提出了一种随正弦函数变化的非线性递减能量因子，还设计了一种动态透镜成像学习策略以增强种群多样性。基于上述两个改进策略，提出了 IARO 算法。数值模拟实验选取了 6 个基准函数以测试 IARO 算法的性能，并与 GWO 算法、WOA、SCA 和 ARO 算法进行比较。数值模拟实验结果表明，IARO 算法在解精度和收敛速度性能上比起对比算法均有较大的提升。此外，将 IARO 算法应用于求解 2 个工程设计优化问题和 1 个含 15 个 UCI 数据集的特征选择问题，结果显示，与其他 4 种算法相比，IARO 算法在求解工程设计优化问题和特征选择问题上具有较强的竞争力。下一步的研究方向是将 IARO 算法应用于多目标优化、动态优化以及其他工程应用领域中。

参考文献

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

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

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

[4] 赵艳玲，王勇，袁磊. 基于 4VA 信息素的蝗虫优化算法 [J]. 广西科学，2022，29(5)：930-939.

[5] HEIDARI A A, MIRJALILI S, FARIS H, et al. Harris hawks optimization: algorithm and applications [J]. Future Generation Computer Systems, 2019, 97: 849-872.

[6] DHIMAN G, KUMAR V. Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems [J]. Knowledge-Based Systems, 2019, 165: 169-196.

[7] 郑洪清，冯文健，周永权. 融合正弦余弦算法的蝴蝶优化算法 [J]. 广西科学，2021，28(2)：152-159.

[8] WANG L Y, CAO Q J, ZHANG Z X, et al. Artificial rabbits optimization: a new bio-inspired meta-heuristic algorithm for solving engineering optimization problems [J]. Engineering Applications of Artificial Intelligence, 2022, 114: 105082.

[9] ELSHAHED M, TOLBA M A, EL-RIFAIE A M, et al. An artificial rabbits' optimization to allocate PVSTATCOM for ancillary service provision in distribution systems [J]. Mathematics, 2023, 11: 339.

[10] RIAD A J, HASANIEN H M, TURKY R A, et al. Identifying the PEM fuel cell parameters using artificial rabbits optimization algorithm [J]. Sustainability, 2023, 15: 4625.

[11] KHALIL A E, BOGHDADY T A, ALHAM M H, et al. Enhancing the conventional controllers for load frequency control of isolated microgrids using proposed multi-objective formulation via artificial rabbits optimization algorithm [J]. IEEE Access, 2023, 11: 3472-3493.

[12] WANG Y Y, HUANG L Q, ZHONG J Y, et al. LARO: opposition-based learning boosted artificial rabbits-inspired optimization algorithm with Lévy flight [J]. Symmetry, 2022, 14: 2282.

[13] WANG Y W, XIAO Y N, GUO Y L, et al. Dynamic chaotic opposition-based learning-driven hybrid Aquila optimization and artificial rabbits optimization algorithm: framework and applications [J]. Processes, 2022, 10: 2703.

[14] WOLPERT D H, MACREADY W G. No free lunch theorems for optimization [J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 67-82.

[15] 龙文，伍铁斌，唐明珠，等. 基于透镜成像学习策略的灰狼优化算法 [J]. 自动化学报，2020，46(10)：2148-2164.

[16] COELLO C A C. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art [J]. Computer Methods in Applied Mechanics and Engineering, 2002, 191: 1245-1287.

Dynamic Lens Imaging Learning Artificial Rabbits Optimization Algorithm and Its Applications

WANG Wei[1],LONG Wen[2**]

(1. School of Management Science and Engineering,Guizhou University of Finance and Economics,Guiyang,Guizhou,550025,China; 2. School of Mathematics and Statistics,Guizhou University of Finance and Economics,Guiyang,Guizhou,550025,China)

Abstract:Aiming at the defects of slow convergence,low precision and easy to fall into local optimum in solving complex optimization problems by Artificial Rabbit Optimization (ARO) algorithm,an improved ARO algorithm (IARO algorithm) is proposed. The nonlinear decreasing energy factor based on sine function in IARO algorithm can help the algorithm to achieve a good transition from the exploration stage to the development stage,so as to improve the convergence speed and solution quality of the algorithm. In order to enhance the probability of helping the algorithm jump out of local optimum,a dynamic lens imaging based learning strategy is introduced. In order to prove the superiority of IARO algorithm,six benchmark functions are selected for numerical experiments,and then it is used to solve two engineering design optimization problems and one feature selection problem including 15 data sets,and compared with Grey Wolf Optimizer (GWO) algorithm Whale Optimization Algorithm (WOA),Sine Cosine Algorithm (SCA) and basic ARO algorithm. The results show that the IARO algorithm has better performance than other comparison algorithms.

Key words:artificial rabbits optimization algorithm;dynamic lens imaging based learning strategy;engineering optimization;feature selection;function optimization

责任编辑:陆　雁,陈少凡