DOI:10.19651/j.cnki.emt.2519799

增强型蜣螂优化算法及其应用*

蔡春雷 刘 微 杨迪雅
(沈阳理工大学信息科学与工程学院 沈阳 110159)

摘 要:针对蜣螂优化算法全局搜索能力弱、收敛速度慢、易陷于局部最优的不足,创新性地提出了一种基于黄金正弦算法的增强型蜣螂优化算法名为 GSDBO 算法。首先,采用 Tent 混沌映射与透镜成像反向学习策略优化种群初始化,以生成高质量初始解;其次,采用改进的黄金正弦算法替代滚球蜣螂的原有位置更新机制,以提升算法的全局搜索精度与收敛速度;最后,结合 t 分布变异策略与自适应调节机制,动态平衡全局探索与局部开发能力。实验采用 CEC2005 与 CEC2020 测试函数,并结合 Wilcoxon 秩和检验,验证所提算法的有效性和可行性,结果表明 GSDBO 算法在收敛速度与求解精度均呈现显著提升。在悬臂梁设计、焊接梁设计和机器人路径规划 3 个工程优化问题中,所提算法均获得最优解,进一步验证了其解决复杂实际问题的有效性。

关键词:蜣螂优化算法;黄金正弦算法;自适应 t 分布;混沌映射
中图分类号:TP301.6;TN03 **文献标识码**:A **国家标准学科分类代码**:520.1040

Enhanced dung beetle optimization algorithm and its application

Cai Chunlei Liu Wei Yang Diya
(School of Information Science and Engineering, Shenyang Ligong University, Shenyang 110159, China)

Abstract: Addressing the limitations of the dung beetle optimization algorithm, such as weak global search capability, slow convergence rate and susceptibility to local optima, this paper innovatively proposes an enhanced dung beetle optimization algorithm based on the golden sine algorithm, named GSDBO algorithm. Firstly, the population initialization is optimized using Tent chaotic mapping and lens imaging reverse learning strategy to generate high-quality initial solutions; secondly, an improved golden sine algorithm is adopted to replace the original position update mechanism of the rolling ball beetle, in order to improve the global search accuracy and convergence speed of the algorithm; finally, by combining the t-distribution mutation strategy and adaptive adjustment mechanism, a dynamic balance is achieved between global exploration and local development capabilities. The experiment used CEC2005 and CEC2020 test functions, combined with Wilcoxon rank sum test, to verify the effectiveness and feasibility of the proposed algorithm. The results showed that the GSDBO algorithm exhibited significant improvements in convergence speed and solution accuracy. In the three engineering optimization problems of cantilever beam design, welding beam design, and robot path planning, this algorithm has obtained the optimal solution, further verifying its effectiveness in solving complex practical problems.

Keywords: dung beetle optimization algorithm; golden sine algorithm; adaptive t-distribution; chaotic mapping

0 引 言

优化问题的核心是在约束条件下寻求目标函数的最优解,这一挑战广泛存在于科学研究与工程实践中。随着计算技术的快速发展,优化方法已成为解决复杂系统设计、资源调配和决策分析等问题的关键工具。实际工程问题常呈现非线性、多模态及黑箱特性,具体表现为目标函数解析式缺失、梯度信息不可导等特征,这导致传统梯度类算法难以保证收敛性[1]。在此背景下,不依赖于梯度信息的元启发式算法,通过模拟自然规律或群体智能,为破解此类复杂优化难题开辟了新路径。

在众多元启发式算法中,群智能优化算法因其自组织、自适应和鲁棒性强的特点备受关注。常见算法如灰狼优化(grey wolf optimizer, GWO)[2]、鲸鱼优化算法(whale

收稿日期:2025-09-07
* 基金项目:辽宁省教育厅高等学校基本科研项目(JYTMS20230189)、沈阳理工大学引进高层次人才科研支持计划项目(1010147001131)资助

optimization algorithm, WOA)[3] 黄金正弦优化算法(golden sine algorithm, GSA)[4] 和金豺优化算法(golden jackal optimization, GJO)[5],通过模拟生物群体的协作机制,在解决复杂优化问题上展现出独特优势。特别是近年来提出的蜣螂优化算法(dung beetle optimizer, DBO)[6],通过模拟蜣螂种群的滚球、产卵、觅食与偷窃行为,构建了独特的4阶段优化框架。在23个基准函数和29个CEC2017测试函数中表现出优于传统算法的收敛精度和稳定性。该算法已成功应用于充电桩故障检测[7]、无人机航迹规划[8]、风电功率密度预测[9]和空气质量预测[10]等领域。

然而,依据"没有免费午餐"定理可知,任何优化算法都存在固有的局限性,本研究的DBO算法作为一种新提出算法,亦面临着所有群智能优化算法在创立初期所共有的问题,其收敛速度与精度仍具备进一步提升的空间。针对上述问题,学术界已提出多种改进策略。余荣川等[11]通过Circle混沌映射、改进正余弦算法融合柯西变异及Levy飞行策略对DBO算法进行改进,在基准函数测试中展现出较好的收敛精度与寻优性能。但Circle混沌映射存在遍历性随维度增加而衰减的问题,在高维搜索空间中易导致初始种群分布不均匀,进而对算法的全局探索能力产生不利影响。蒋翱徽等[12]利用交叉策略、次优引导策略与偷窃行为增强策略,有效提升了算法整体性能,实现了变分模态分解参数的自适应优化。该方法在管道泄漏信号分解与有效模态筛选重构方面表现出较好的效果,获得了理想的降噪性能。然而,其增强策略未设置对迭代后期的收敛约束机制,虽有助于避免陷入局部最优,但存在一定概率影响算法收敛效率。Wang等[13]提出了一种基于正余弦函数的DBO算法。该算法利用sinh和cosh函数扰乱初始种群分布,并平衡滚球蜣螂的开发行为,通过非线性机制提升搜索效率与全局探索能力。此改进在机器人夹爪优化、压力容器设计和无人机路径规划3类工程问题上验证了其有效性。但非线性函数扰动在改善初始分布多样性的同时,可能对解空间的局部结构产生影响,其对算法早期搜索行为的作用机制有待进一步分析。雒明世等[14]引入拉丁超立方抽样、变螺旋搜索、准反向学习与柯西变异策略改进DBO算法,用于求解无线传感网络节点定位目标函数最优值,有效提升了定位精度。然而,在算法中准反向学习与柯西变异策略的执行方式上,缺乏动态调整机制,对种群探索与开发能力平衡的影响仍有优化空间。刘微等[15]则采用Logistic-tent混沌映射、改进正余弦算法、自适应t分布扰动以及Levy-柯西变异策略对DBO进行改进,在压力容器设计等3个工程优化问题中取得了更低的优化成本。另一方面,该算法所引入的变异策略在增强跳出局部极值能力的同时,由于Levy飞行的大步长特性和柯西分布的宽尾部性质引入较强的随机扰动,从而在高维复杂问题中对收敛稳定性和精度产生影响。

针对上述改进的DBO算法依旧存在的问题,本文提出了一种融合黄金正弦算法的增强型蜣螂优化(gold sine algorithm based adaptive DBO, GSDBO)算法,该算法的改进体现在以下3个层面:

1)在种群初始化阶段,引入Tent混沌映射与透镜成像反向学习策略。Tent映射具备优于其他混沌映射的遍历一致性与分布均匀性,有助于生成分布更为均匀的初始解集[16];进一步结合透镜成像反向学习策略,可有效克服传统优化算法中初始种群易陷于局部区域、难以全面覆盖解空间的问题。该策略借鉴光学透镜成像原理,将当前解映射至其反向位置以生成反向解,从而协同Tent混沌映射扩展初始种群的勘探范围,有效提升初始解集质量。

2)在探索阶段采用改进的黄金正弦算法重构种群位置更新机制。相较于正余弦算法因三角函数周期性导致的搜索步长不稳定,黄金正弦算法利用黄金分割系数构建具有数学导向性的搜索机制,该系数凭借其强约束性,能有效抑制无意义的随机波动。同时,引入自适应权重因子,通过动态调节个体历史位置的惯性权重,在迭代初期强化全局探索,在后期促进局部开发。改进后的位置更新机制确保了种群能更精准地向最优解区域收缩,从而在增强搜索过程的精度与稳定性的同时,也提升了算法的收敛速度。

3)引入t分布策略与自适应调节机制,本方法将自适应t分布变异应用于蜣螂种群。这种全局性的扰动能更全面地提升种群多样性,结合自适应调节机制,在迭代前期,通过较大幅度的t分布变异强化全局探索能力、覆盖更广泛解空间,在后期,自适应缩小变异幅度、聚焦优质解周围进行精细化局部开发,以实现全局探索与局部开发的动态平衡。

为验证改进效果,将所提算法与多种群智能优化算法展开对比;其次,运用Wilcoxon秩和检验对统计显著性进行评估;最后,在悬臂梁设计、焊接梁设计和机器人路径规划3个工程优化问题中开展应用验证。结果表明,GSDBO算法展现出卓越的性能,具备强劲的竞争力。

1 DBO算法

本节描述了DBO算法的基本原理。DBO算法是一种受蜣螂群体社会行为启发的新颖群智能优化算法。该算法使用式(1)~(7)来根据蜣螂的行为,如滚球、舞蹈、繁殖、觅食和偷窃,更新每个蜣螂的位置。$x_i(t)$表示第i只蜣螂在第t次迭代中的位置,L_b和U_b分别表示算法搜索的下限和上限。

1.1 滚球蜣螂

蜣螂能够借助太阳、月亮及偏振光等自然光源进行轨迹导航以移动粪球,其运动轨迹如图1所示。图1中展示了蜣螂基于太阳导航的场景,右箭头指向蜣螂的翻滚方向。在无障碍模式下,其滚动时位置会随光源强度变化。

蜣螂滚球的数学表达式如式(1)所示。

图 1　蜣螂运动轨迹模型

Fig. 1　Model of dung beetle movement trajectory

$$\begin{cases} \boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + a \times k \times \boldsymbol{x}_i(t-1) + b \times \Delta \boldsymbol{x} \\ \Delta \boldsymbol{x} = | \boldsymbol{x}_i(t) - \boldsymbol{X}^{\mathrm{w}} | \end{cases} \tag{1}$$

其中，$\boldsymbol{x}_i(t+1)$ 表示第 i 只蜣螂在第 $t+1$ 次迭代中的位置；a 为 1 或 -1；k 表示偏转系数，取值在 $(0, 0.2]$；b 为 $(0,1)$ 之间的随机数；$\boldsymbol{X}^{\mathrm{w}}$ 为蜣螂种群中最差的位置。

若蜣螂滚球受阻，会以跳舞方式决定新滚动方向。新的行进方向利用切线函数模仿其跳舞行为确定。位置更新表达式如式(2)所示。

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + \tan(\theta) | \boldsymbol{x}_i(t) - \boldsymbol{x}_i(t-1) | \tag{2}$$

其中，θ 是属于 $[0, \pi]$ 的偏转角。

1.2　繁殖蜣螂

为了给后代提供一个安全的环境，蜣螂会在安全区域产卵，而安全产卵区域被定义为如式(3)所示的边界选择策略。

$$\begin{cases} L_{\mathrm{b}}^{*} = \max(\boldsymbol{X}^{*} \times (1-R), L_b) \\ U_{\mathrm{b}}^{*} = \min(\boldsymbol{X}^{*} \times (1+R), U_b) \\ R = 1 - \dfrac{t}{T_{\max}} \end{cases} \tag{3}$$

其中，R 表示收敛因子；$\boldsymbol{X}^{*}$ 表示当前局部最优位置；L_{b}^{*} 和 U_{b}^{*} 分别代表产卵区的下界和上界；$T_{\max}$ 表示最大迭代次数。

由式(3)可知，产卵区随 R 值动态变化。因此，产卵蜣螂的位置也随 R 值动态变化，数学表达式如式(4)所示。

$$\boldsymbol{x}_i(t+1) = \boldsymbol{X}^{*} + \boldsymbol{b}_1 \times (\boldsymbol{x}_i(t) - L_{\mathrm{b}}^{*}) + \boldsymbol{b}_2 \times (\boldsymbol{x}_i(t) - U_{\mathrm{b}}^{*}) \tag{4}$$

其中，$\boldsymbol{b}_1$ 和 $\boldsymbol{b}_2$ 为两个大小为 $1 \times D$ 随机向量；D 表示优化问题维度。

1.3　觅食蜣螂

当幼年蜣螂觅食时，它们也需要确定最佳觅食区来引导它们觅食，觅食区在式(5)中定义。

$$\begin{cases} L_{\mathrm{b}}^{\mathrm{b}} = \max(\boldsymbol{X}^{\mathrm{b}} \times (1-R), L_{\mathrm{b}}) \\ U_{\mathrm{b}}^{\mathrm{b}} = \min(\boldsymbol{X}^{\mathrm{b}} \times (1+R), U_{\mathrm{b}}) \end{cases} \tag{5}$$

$L_{\mathrm{b}}^{\mathrm{b}}$ 和 $U_{\mathrm{b}}^{\mathrm{b}}$ 表示最佳觅食区域的下限和上限。$\boldsymbol{X}^{\mathrm{b}}$ 表示全局最优位置。所以，小蜣螂的位置更新表达式如式(6)所示。

$$\boldsymbol{x}_i(t+1) = \boldsymbol{x}_i(t) + C_1 \times (\boldsymbol{x}_i(t) - L_{\mathrm{b}}^{\mathrm{b}}) + \boldsymbol{C}_2 \times (\boldsymbol{x}_i(t) - U_{\mathrm{b}}^{\mathrm{b}}) \tag{6}$$

其中，C_1 是遵循正态分布的随机数；$\boldsymbol{C}_2$ 为 $(0,1)$ 范围内的随机向量。

1.4　偷窃蜣螂

在蜣螂种群中，存在蜣螂窃取其他同类粪球的现象，称为偷窃蜣螂。根据式(5)，$\boldsymbol{X}^{\mathrm{b}}$ 是最佳觅食区域，因此可以将 $\boldsymbol{X}^{\mathrm{b}}$ 周围的区域设定为争夺食物的最佳区域。偷粪蜣螂的位置更新表达式如式(7)所示。

$$\boldsymbol{x}_i(t+1) = \boldsymbol{X}^{\mathrm{b}} + S \times \boldsymbol{g} \times (| \boldsymbol{x}_i(t) - \boldsymbol{X}^{*} | + | \boldsymbol{x}_i(t) - \boldsymbol{X}^{\mathrm{b}} |) \tag{7}$$

其中，S 为常数；$\boldsymbol{g}$ 为 $1 \times D$ 的随机向量。

2　增强型蜣螂优化算法 GSDBO

本文针对原始 DBO 算法存在的收敛精度不足和易陷入局部最优等问题，提出多个策略融合改进的方法。首先引入 Tent 混沌映射和透镜成像反向学习策略优化种群初始化过程，其次采用改进的黄金正弦算法重构滚球蜣螂的位置更新机制，最后结合自适应 t 分布策略增强算法跳出局部最优的能力，实现全局探索与局部开发能力的动态平衡。

2.1　Tent 混沌映射-透镜成像反向学习策略

在原始 DBO 中，种群初始化通常采用随机生成策略，该方法生成的初始解在空间中分布均匀性较差，易导致个体聚集于局部区域，从而削弱算法的全局探索能力，甚至引发早熟收敛。为改善初始种群质量，本文引入 Tent 混沌映射对种群初始化过程进行改进。已有研究表明，Tent 混沌映射在遍历性和分布均匀性方面优于 Logistic、Cubic 等常用混沌映射，能够有效克服随机初始化所导致的解分布稀疏或重叠问题，进而提升初始种群的多样性[17]。Tent 混沌映射的数学表达式如式(8)所示。

$$\boldsymbol{x}_i(t+1) = \mathrm{Tent}(\boldsymbol{x}_i(t)) = \begin{cases} \dfrac{\boldsymbol{x}_i(t)}{u}, & 0 \leqslant x < u \\ \dfrac{1 - \boldsymbol{x}_i(t)}{1-u}, & u \leqslant x \leqslant 1 \end{cases} \tag{8}$$

其中，$\boldsymbol{x}_i(t+1)$ 和 $\boldsymbol{x}_i(t)$ 分别为第 i 个蜣螂个体在 t 和 $t+1$ 次迭代时的位置；u 为映射参数，本文中为 0.5。

为进一步提升种群多样性并拓展勘探范围，引入透镜成像反向学习策略对混沌映射种群进行处理[18]。从混沌映射种群及其反向种群里挑选优良个体作为算法初始种群，以此提高初始解质量，增加 DBO 算法寻得最优解的概率。透镜成像反向学习策略的原理如图 2 所示。

$$\frac{(u_{\mathrm{b}} + l_{\mathrm{b}})/2 - x}{x' - (u_{\mathrm{b}} + l_{\mathrm{b}})/2} = \frac{h}{h'} \tag{9}$$

在式(9)中,(l_b,u_b) 表示 X 轴的范围,h 和 x 分别表示物体 B 的高度以及在 X 轴上的投影;x' 和 h' 分别表示经透镜成像后的物体的高度以及在 X 轴上的投影。

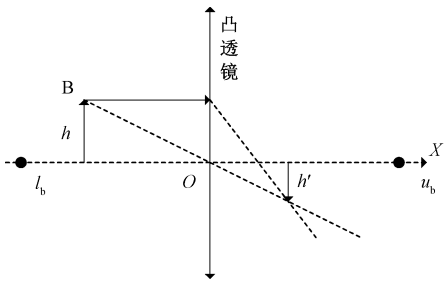


图 2 透镜成像反向学习策略

Fig. 2 Schematic of lens imaging reverse learning strategy

令 $h/h'=k$,通过变换可得:

$$x'=\frac{u_b+l_b}{2}+\frac{u_b+l_b}{2k}-\frac{x}{k} \tag{10}$$

Tent 混沌映射与透镜成像反向学习策略相结合,前者提供分布均匀的初始解,后者则拓展解空间以提升多样性,两者优势互补,为后续全局探索构建了高质量的初始种群。

2.2 改进的黄金正弦算法

在 DBO 算法中,滚球蜣螂遇到障碍物时,传统方法通过正切函数模拟其舞蹈行为以调整运动方向。然而,该方法在复杂搜索空间中存在求解精度不足、收敛速度慢的问题。针对这一不足,本文引入 GSA 算法对位置更新机制进行改进。

GSA 算法基于正弦函数的周期性特征与黄金分割率的空间压缩特性,具有出色的收敛速度和开发能力。因此,本文将其核心思想引入 DBO 算法,重构滚球蜣螂的位置更新公式。具体而言,当蜣螂个体遇到障碍时,算法通过黄金比例系数动态缩小解空间范围,利用正弦函数的遍历性生成方向调节因子,使蜣螂个体以黄金正弦优化的方式向全局最优位置逼近,减少个体搜索的盲目性。

此外,为在全局探索与局部开发之间取得更佳平衡,引入自适应权重系数 w,该系数随迭代次数动态调整当前个体位置对新位置的影响权重[19]。迭代初期赋予较高权重以强化全局搜索,后期逐步降低权重以聚焦局部开发。改进后黄金正弦算法的更新表达式如式(11)所示。

$$\begin{cases}\boldsymbol{x}_i(t+1)=w\boldsymbol{x}_i(t)\mid\sin(r_1)\mid-r_2\sin(r_1)\mid y_1\boldsymbol{X}-\\ \quad wy_2\boldsymbol{x}_i(t)\mid\\ y_1=a\times(1-\tau)+b\times\tau\\ y_2=a\times\tau+b\times(1-\tau)\\ w=3^{-\frac{16t^2}{T^2}}\end{cases} \tag{11}$$

其中,y_1 和 y_2 为黄金比例系数;a 和 b 的初始值分别为 $-\pi$ 和 π;τ 是定义为 $\frac{1-\sqrt{5}}{2}$ 的黄金数;r_1 是在 $[0,2\pi]$ 范围内的随机数;r_2 是在 $[0,\pi]$ 范围内的随机数;t 和 T 分别为当前迭代次数和总迭代次数。

通过融入改进的黄金正弦算法,滚球蜣螂在遭遇障碍时,自适应权重因子先对当前位置信息占比进行动态调节,避免了因过度依赖当前区域而陷入局部最优;黄金正弦因子再同步生成具有数学导向性的方向与步长参数,确保滚球蜣螂的位置更新既不脱离有效解空间区域,方向调节过程中也不会丢失关键位置信息,从而有效提升算法的全局搜索精度与收敛速度。

2.3 自适应 t 分布策略

t 分布又称为学生分布[20],主要用于估计基于小样本的具有未知方差的正态分布总体的平均值。本文提出一种自适应 t 分布策略,通过动态调节自由度参数实现算法全局探索与局部开发能力的平衡优化。该策略中,自由度参数随迭代进程平滑递增,使得变异操作在搜索初期具有较强的随机扰动以强化个体全局探索能力,有利于跳出局部最优;随着迭代进行,扰动强度逐渐减弱,有助于后期在最优解附近进行精细搜索。具体位置更新表达式如式(12)所示。

$$\begin{cases}\boldsymbol{x}_{\text{new}}^t=\boldsymbol{x}_i^t+\text{trnd}(C_{\text{iter}})\times\boldsymbol{x}_i^t\\ C_{\text{iter}}=\max(1,\log(1+\exp(4\times(t/T)^2)))\end{cases} \tag{12}$$

其中,$\boldsymbol{x}_{\text{new}}^t$ 为更新后蜣螂的位置;trnd 为 t 分布函数;C_{iter} 为自适应自由度。

自适应 t 分布变异策略通过对蜣螂个体进行变异操作,为种群引入了更多的变化,增加了找到全局最优解的可能性。然而,变异操作本身具有随机性,其产生的新解未必优于原解,因此本文采用贪婪规则,在每次位置更新后对比新旧解的适应度值,仅保留更优解,确保算法迭代过程的单调收敛性。贪婪策略公式为:

$$\boldsymbol{x}_{\text{new}}^t=\begin{cases}\boldsymbol{x}_i^t, & f(x_{\text{new}}^t)>f(\boldsymbol{x}_i^t)\\ \boldsymbol{x}_{\text{new}}^t, & f(x_{\text{new}}^t)\leqslant f(\boldsymbol{x}_i^t)\end{cases} \tag{13}$$

2.4 GSDBO 算法改进流程

GSDBO 算法改进流程如图 3 所示。

3 仿真实验及结果分析

本实验在 Intel ® Core™ i5-13490F 处理器、8 GB 内存的硬件环境下进行,操作系统为 Windows 10 专业版(64 位),软件平台采用 MATLAB R2022b。实验采用 CEC2005 标准测试集的 23 个基准函数以及 CEC2020 测试集的 10 个测试函数。对比算法包括 DBO[6]、WOA[3]、GJO[5]、海马优化算法(seahorse optimization algorithm,SHO)[21]4 种原始算法以及 2 种改进的 DBO 算法,由于 2 种改进的算法名称相同,故分别命名为 SCDBO1[13] 和 SCDBO2[15]实验统一设置种群规模为 30、最大迭代次数

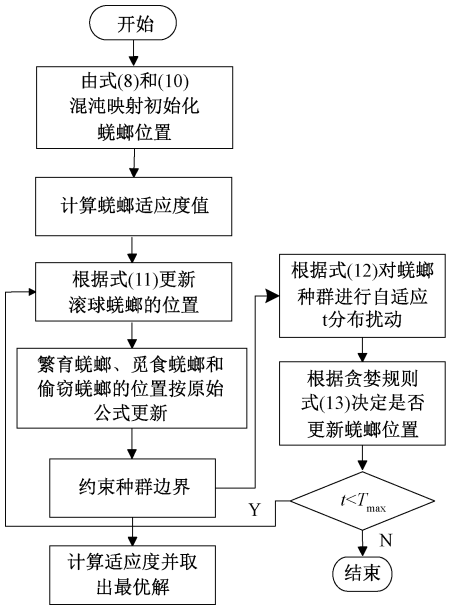


图 3　GSDBO 算法流程

Fig. 3　Flowchart of GSDBO

500，每个算法独立运行 30 次以消除随机性影响。

3.1　GSDBO 算法在 CEC2005 测试函数中实验结果分析

CEC2005 标准测试集的 23 个基准函数涵盖单峰测试函数(F1～F7)、多峰测试函数(F8～F13)、固定维度多峰测试函数(F14～F23)。GSDBO 算法与其余 6 种算法在 CEC2005 测试函数上的统计结果如表 1 所示，其中涵盖了平均值(Avg)和标准差(Std)，7 种算法的最优值均以粗体标注，收敛曲线如图 4 所示。

在单峰测试函数方面，GSDBO 算法表现突出。在 F1～F4 函数测试中，其不仅结果达到理论最优，收敛速度也明显快于其他算法，体现卓越的局部开发能力；在 F5 函数测试时，GSDBO 算法虽未达到理论最优值，但性能仍优于其他算法；在 F6 函数测试中，该算法略差于 SCDBO2 算法但也位处第 2。多峰测试函数测试中，GSDBO 算法在 F9 和 F11 函数上达到理论最优值。尽管其他算法也能找到这些函数的最优值，但收敛速度均不及 GSDBO 算法。在 F8、F10、F12 和 F13 函数测试中，GSDBO 算法虽未达到理论最优，但表现依旧优于其他算法。在固定维度多峰测试函数测试中，GSDBO 算法展现出良好的性能，在 10 个函

表 1　CEC2005 测试函数实验数据

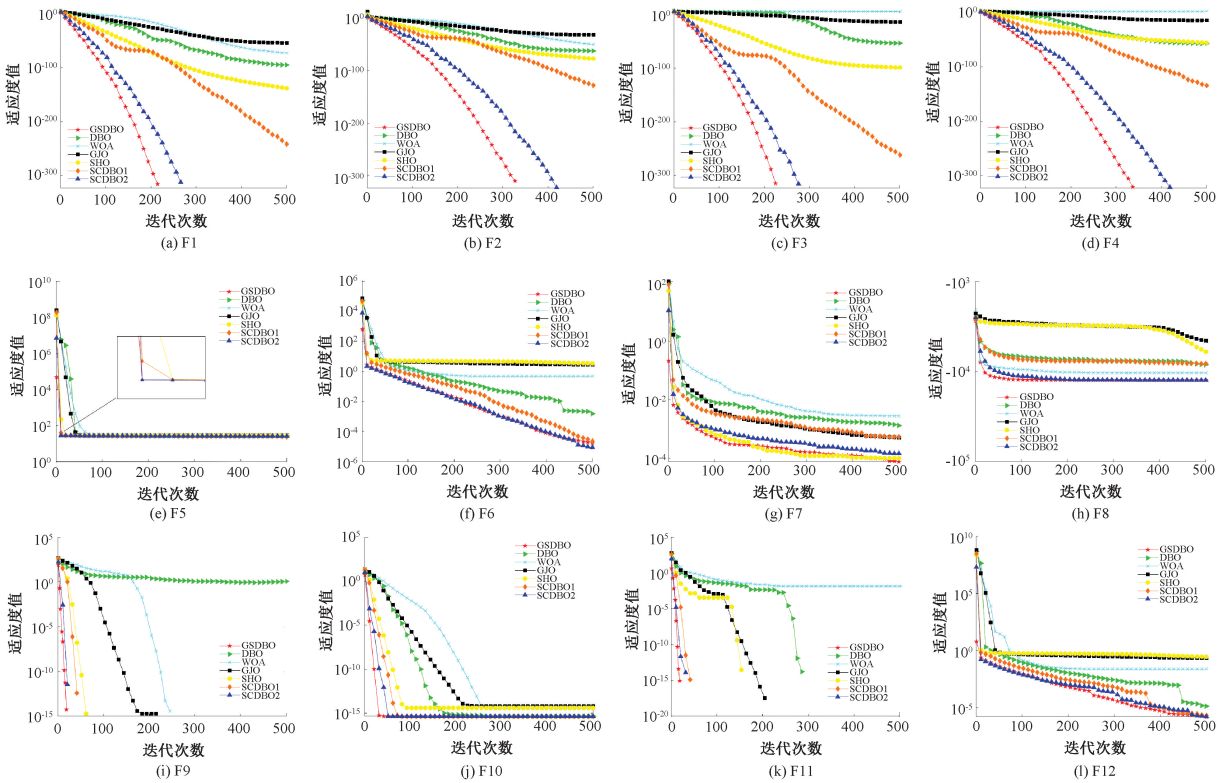
Table 1　CEC2005 test function experimental data

函数	统计量	GSDBO	DBO	WOA	GJO	SHO	SCDBO1	SCDBO2
F1	平均值 Avg	**0**	2.18×10^{-95}	2.41×10^{-73}	3.07×10^{-54}	1.50×10^{-138}	3.85×10^{-242}	**0**
	标准差 Std	**0**	1.19×10^{-94}	1.13×10^{-72}	7.41×10^{-54}	1.04×10^{-138}	**0**	**0**
F2	平均值 Avg	**0**	3.57×10^{-63}	9.57×10^{-51}	2.12×10^{-32}	3.94×10^{-78}	2.31×10^{-128}	**0**
	标准差 Std	**0**	1.81×10^{-62}	2.73×10^{-50}	2.27×10^{-32}	1.04×10^{-77}	1.04×10^{-127}	**0**
F3	平均值 Avg	**0**	1.41×10^{-54}	4.54×10^{4}	2.41×10^{-15}	5.70×10^{-100}	1.41×10^{-262}	**0**
	标准差 Std	**0**	7.65×10^{-54}	9.63×10^{3}	9.21×10^{-15}	1.72×10^{-99}	**0**	**0**
F4	平均值 Avg	**0**	4.07×10^{-58}	4.48×10^{1}	2.44×10^{-15}	1.89×10^{-56}	2.94×10^{-135}	**0**
	标准差 Std	**0**	1.93×10^{-57}	2.81×10^{1}	1.01×10^{-14}	5.07×10^{-56}	1.02×10^{-134}	**0**
F5	平均值 Avg	$\mathbf{2.51\times10^{1}}$	2.57×10^{1}	2.80×10^{1}	2.80×10^{1}	2.80×10^{1}	2.52×10^{1}	2.53×10^{1}
	标准差 Std	2.04×10^{-1}	$\mathbf{1.42\times10^{-1}}$	4.43×10^{-1}	5.93×10^{-1}	5.71×10^{-1}	2.95×10^{-1}	2.13×10^{-1}
F6	平均值 Avg	1.69×10^{-5}	1.53×10^{-3}	4.75×10^{-1}	2.81	3.39	6.58×10^{-5}	$\mathbf{8.99\times10^{-6}}$
	标准差 Std	3.18×10^{-5}	4.79×10^{-2}	2.37×10^{-1}	4.28×10^{-1}	6.44×10^{-1}	2.25×10^{-5}	$\mathbf{1.47\times10^{-5}}$
F7	平均值 Avg	$\mathbf{7.45\times10^{-5}}$	1.31×10^{-3}	2.82×10^{-3}	5.05×10^{-4}	9.83×10^{-5}	5.30×10^{-4}	1.43×10^{-4}
	标准差 Std	$\mathbf{5.34\times10^{-5}}$	1.17×10^{-3}	2.29×10^{-3}	3.42×10^{-5}	7.04×10^{-5}	3.27×10^{-4}	1.33×10^{-4}
F8	平均值 Avg	$\mathbf{-1.25\times10^{4}}$	-8.42×10^{3}	-1.03×10^{4}	-4.55×10^{3}	-6.03×10^{3}	-8.20×10^{3}	$\mathbf{-1.25\times10^{4}}$
	标准差 Std	$\mathbf{1.90\times10^{2}}$	2.27×10^{3}	1.71×10^{3}	1.45×10^{3}	5.29×10^{2}	9.72×10^{2}	2.35×10^{2}
F9	平均值 Avg	**0**	1.19	**0**	**0**	**0**	**0**	**0**
	标准差 Std	**0**	6.53	**0**	**0**	**0**	**0**	**0**
F10	平均值 Avg	$\mathbf{4.44\times10^{-16}}$	$\mathbf{4.44\times10^{-16}}$	3.52×10^{-15}	6.60×10^{-15}	3.99×10^{-15}	$\mathbf{4.44\times10^{-16}}$	$\mathbf{4.44\times10^{-16}}$
	标准差 Std	**0**	**0**	2.42×10^{-15}	1.59×10^{-15}	**0**	**0**	**0**
F11	平均值 Avg	**0**	**0**	1.47×10^{-2}	**0**	**0**	**0**	**0**
	标准差 Std	**0**	**0**	4.62×10^{-2}	**0**	**0**	**0**	**0**
F12	平均值 Avg	$\mathbf{1.62\times10^{-6}}$	1.36×10^{-5}	2.46×10^{-2}	2.27×10^{-1}	2.86×10^{-1}	2.07×10^{-6}	1.68×10^{-6}
	标准差 Std	5.46×10^{-6}	3.42×10^{-5}	2.24×10^{-2}	8.393×10^{-2}	1.20×10^{-2}	3.74×10^{-6}	$\mathbf{3.27\times10^{-6}}$

表 1(续)

Table 1 (continued)

函数	统计量	GSDBO	DBO	WOA	GJO	SHO	SCDBO1	SCDBO2
F13	平均值 Avg	**1.33×10^{-2}**	7.03×10^{-1}	5.29×10^{-1}	1.65	2.11	9.58×10^{-1}	2.33×10^{-2}
	标准差 Std	**1.98×10^{-2}**	4.49×10^{-1}	2.70×10^{-1}	2.42×10^{-1}	3.15×10^{-1}	5.28×10^{-1}	2.93×10^{-2}
F14	平均值 Avg	**1.13**	1.72	3.48	5.49	4.64	2.07	1.38
	标准差 Std	**5.03×10^{-1}**	1.21	3.51	4.42	4.59	2.57	1.80
F15	平均值 Avg	**3.42×10^{-4}**	9.45×10^{-4}	6.39×10^{-4}	2.57×10^{-3}	2.39×10^{-3}	6.87×10^{-4}	4.26×10^{-4}
	标准差 Std	**9.52×10^{-5}**	5.08×10^{-4}	3.42×10^{-4}	6.03×10^{-3}	6.10×10^{-3}	3.27×10^{-4}	1.86×10^{-4}
F16	平均值 Avg	**−1.03**	**−1.03**	**−1.03**	**−1.03**	**−1.03**	**−1.03**	**−1.03**
	标准差 Std	6.11×10^{-16}	6.45×10^{-16}	1.15×10^{-9}	1.54×10^{-7}	1.61×10^{-8}	5.97×10^{-16}	**5.90×10^{-16}**
F17	平均值 Avg	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**	**3.98×10^{-1}**
	标准差 Std	**0**	**0**	2.76×10^{-5}	2.09×10^{-4}	3.52×10^{-3}	**0**	**0**
F18	平均值 Avg	**3**	**3**	**3**	**3**	**3**	**3**	**3**
	标准差 Std	3.04×10^{-15}	**2.36×10^{-15}**	1.97×10^{-4}	5.21×10^{-6}	4.01×10^{-9}	2.55×10^{-15}	2.40×10^{-15}
F19	平均值 Avg	**−3.86**	**−3.86**	−3.85	**−3.86**	**−3.86**	**−3.86**	**−3.86**
	标准差 Std	**2.52×10^{-15}**	2.40×10^{-3}	8.16×10^{-3}	3.63×10^{-3}	4.02×10^{-3}	1.99×10^{-3}	1.99×10^{-3}
F20	平均值 Avg	**−3.27**	−3.19	−3.18	−3.10	−2.99	−3.24	−3.23
	标准差 Std	**6.01×10^{-2}**	1.14×10^{-1}	1.74×10^{-1}	2.35×10^{-1}	3.11×10^{-1}	8.44×10^{-2}	6.35×10^{-2}
F21	平均值 Avg	**−10.15**	−6.87	−7.67	−8.36	−6.18	−9.43	**−10.15**
	标准差 Std	**8.28×10^{-4}**	2.56	2.91	2.83	2.09	2.07	1.17×10^{-2}
F22	平均值 Avg	**−10.40**	−8.28	−7.84	−10.03	−5.87	−8.99	**−10.40**
	标准差 Std	**3.61×10^{-3}**	2.63	3.01	1.34	2.23	2.90	4.13×10^{-3}
F23	平均值 Avg	**−10.53**	−9.28	−7.51	−9.60	−5.64	−9.52	**−10.53**
	标准差 Std	**1.80×10^{-4}**	2.30	3.36	2.42	2.28	2.36	9.98×10^{-3}



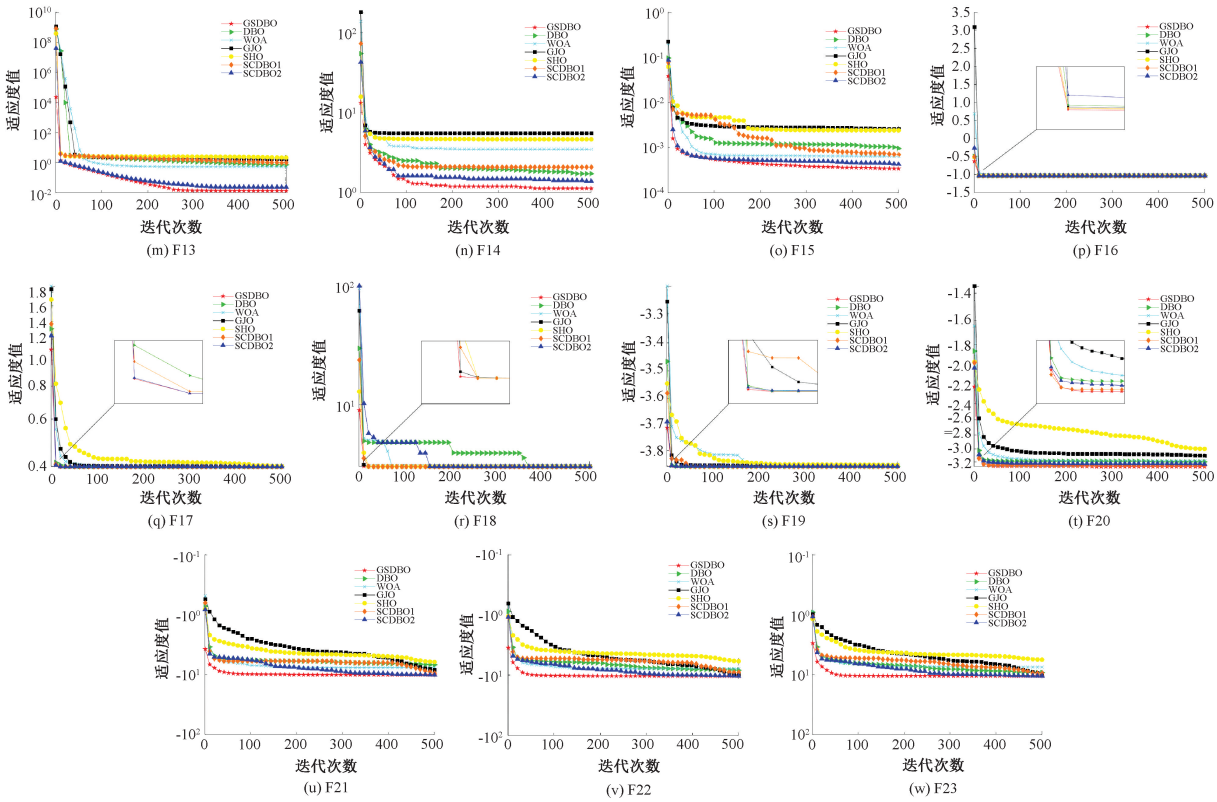


图 4　各算法在 CEC2005 测试函数中的收敛图

Fig. 4　Convergence plots of various algorithms in the CEC2005 test function

数中有 7 个获得理论最优解，收敛速度也较为出色。即便在 F16～F18 函数中其他算法也能达到最优解，GSDBO 算法的收敛速度仍位居第 1。

综上分析可得，GSDBO 算法在 CEC2005 测试函数集上展现出显著的性能优势。

3.2　GSDBO 算法在 CEC2020 测试函数中实验结果分析

为进一步验证 GSDBO 算法的性能表现，将其置于 CEC2020 测试函数集环境下开展测试。CEC2020 测试集由 10 个复杂的函数组成，即复合单峰函数(F1)、多峰函数(F2～F4)、混合函数(F5～F7)和复合函数(F8～F10)。GSDBO 算法与其他 6 种算法在 CEC2020 测试函数上的实验统计结果如表 2 所示。结果显示，GSDBO 算法在该测试函数集上依然展现出优异性能。

从表 2 的数据分析可以看出，在 CEC2020 测试集的 10 个基准函数上，GSDBO 算法在 9 个函数中均取得了最优的收敛精度，展现出绝对优势。尽管在 F10 函数上的表现略逊于 SCDBO1 算法，但这并不影响 GSDBO 算法整体性能的优越性。

3.3　Wilcoxon 秩和检验

Wilcoxon 符号秩检验作为一种经典的非参数统计方法，适用于评估不同算法在复杂数据分布下的性能显著性差异。为科学验证 GSDBO 算法的整体有效性，本研究采用该方法对 GSDBO 算法与其余 6 种对比算法的仿真结果进行统计分析。将 GSDBO 算法与其余 6 种算法的运行结果开展 Wilcoxon 符号秩检验。P 代表显著性水平，其判断准则如下：当 P 值<0.05 时，说明两种算法在性能表现上存在显著差异；若 P 值>0.05，则意味着两种算法之间无显著差异；若 P 值显示为 N/A，表明这两种算法在寻优能力方面表现相当。Wilcoxon 符号秩检验结果如表 3 所示。分析可知，在 CEC2005 测试函数集上，GSDBO 算法在大多数函数上的表现均显著优于对比算法，这证明了其改进策略的有效性和优越性。

4　工程应用

为验证 GSDBO 算法的实用性与工程可靠性，本节选取悬臂梁设计、焊接梁设计及机器人路径规划 3 类典型工程优化问题进行测试分析。上述问题可根据搜索空间特征划分为连续变量优化问题与离散组合优化问题两类典型场景。其中，悬臂梁设计与焊接梁设计属于连续优化问题，机器人路径规划问题则归于离散组合优化范畴。值得注意的是，不同类型优化问题的解空间结构与求解难度存在差异。连续空间优化中，算法性能主要取决于局部搜索精度与收敛速度；而在离散空间优化中，算法更依赖其全局搜索能力及跳出局部最优的能力。因此，通过在上述两类代表性问题上的系统测试，可系统地评估 GSDBO 算法在不同变量类型、约束强度及搜索空间结构下的优化性

表 2 CEC2020 测试函数实验数据

Table 2 CEC2020 test function experimental data

函数	统计量	GSDBO	DBO	WOA	GJO	SHO	SCDBO1	SCDBO2
F1	平均值 Avg	**4.00×10^{3}**	1.18×10^{6}	5.96×10^{7}	2.84×10^{8}	6.74×10^{8}	4.09×10^{3}	4.65×10^{3}
	标准差 Std	3.69×10^{3}	3.42×10^{6}	9.16×10^{7}	3.86×10^{8}	8.70×10^{8}	3.40×10^{3}	**3.33×10^{3}**
F2	平均值 Avg	**1.84×10^{3}**	2.03×10^{3}	2.25×10^{3}	2.09×10^{3}	1.93×10^{3}	1.99×10^{3}	1.96×10^{3}
	标准差 Std	2.93×10^{2}	4.06×10^{2}	3.90×10^{2}	3.69×10^{2}	**2.16×10^{2}**	3.77×10^{2}	2.82×10^{2}
F3	平均值 Avg	**7.40×10^{2}**	7.47×10^{2}	7.86×10^{2}	7.59×10^{2}	7.65×10^{2}	7.61×10^{2}	7.56×10^{2}
	标准差 Std	**1.09×10^{1}**	1.45×10^{1}	2.46×10^{1}	1.44×10^{1}	1.37×10^{1}	1.71×10^{1}	1.78×10^{1}
F4	平均值 Avg	**1.90×10^{3}**	1.91×10^{3}	1.91×10^{3}	2.09×10^{3}	3.70×10^{3}	**1.90×10^{3}**	**1.90×10^{3}**
	标准差 Std	**1.13**	7.36	9.09	9.85×10^{2}	4.04×10^{4}	1.33	1.31
F5	平均值 Avg	**5.88×10^{3}**	1.65×10^{4}	2.66×10^{5}	5.23×10^{4}	2.28×10^{5}	1.35×10^{4}	1.56×10^{4}
	标准差 Std	**2.69×10^{3}**	1.65×10^{4}	5.56×10^{5}	1.16×10^{5}	1.81×10^{5}	2.73×10^{4}	3.48×10^{4}
F6	平均值 Avg	**1.75×10^{3}**	1.79×10^{3}	1.82×10^{3}	1.83×10^{3}	1.78×10^{3}	1.80×10^{3}	1.76×10^{3}
	标准差 Std	9.04×10^{1}	9.98×10^{1}	**8.11×10^{1}**	1.01×10^{2}	1.08×10^{2}	1.36×10^{2}	1.13×10^{2}
F7	平均值 Avg	**6.62×10^{3}**	1.09×10^{4}	5.81×10^{5}	9.65×10^{3}	1.28×10^{4}	1.02×10^{4}	6.87×10^{3}
	标准差 Std	**6.55×10^{3}**	1.18×10^{4}	1.27×10^{6}	4.33×10^{3}	9.27×10^{3}	1.06×10^{4}	7.37×10^{3}
F8	平均值 Avg	**2.30×10^{3}**	2.31×10^{3}	2.32×10^{3}	2.35×10^{3}	2.42×10^{3}	**2.30×10^{3}**	**2.30×10^{3}**
	标准差 Std	**2.52**	9.88	7.19	4.71×10^{1}	1.86×10^{2}	4.6	1.45×10^{1}
F9	平均值 Avg	**2.68×10^{3}**	2.75×10^{3}	2.78×10^{3}	2.77×10^{3}	2.77×10^{3}	2.70×10^{3}	2.69×10^{3}
	标准差 Std	1.09×10^{2}	6.33×10^{1}	4.95×10^{1}	**2.23×10^{1}**	6.42×10^{1}	1.05×10^{2}	1.04×10^{2}
F10	平均值 Avg	2.93×10^{3}	2.94×10^{3}	2.96×10^{3}	2.95×10^{3}	2.94×10^{3}	**2.92×10^{3}**	2.93×10^{3}
	标准差 Std	2.53×10^{1}	2.70×10^{1}	3.34×10^{1}	4.04×10^{1}	4.61×10^{1}	**2.52×10^{1}**	2.53×10^{1}

表 3 Wilcoxon 秩和检验统计结果

Table 3 Wilcoxon rank sum test statistical results

函数	GSDBOvsDBO	GSDBOvsWOA	GSDBOvsGJO	GSDBOvsSHO	GSDBOvsSCDBO1	GSDBOvsSCDBO2
F1	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	N/A
F2	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	N/A
F3	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	N/A
F4	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	1.21×10^{-12}	N/A
F5	1.20×10^{-10}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	1.41×10^{-1}	2.15×10^{-6}
F6	5.57×10^{-10}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	1.05×10^{-3}	2.75×10^{-3}
F7	9.75×10^{-10}	3.01×10^{-11}	2.60×10^{-10}	1.98×10^{-1}	7.38×10^{-10}	7.97×10^{-2}
F8	9.91×10^{-11}	1.20×10^{-8}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	6.37×10^{-3}
F9	3.33×10^{-1}	1.61×10^{-1}	N/A	N/A	N/A	N/A
F10	N/A	1.23×10^{-9}	2.48×10^{-13}	1.68×10^{-14}	N/A	N/A
F11	N/A	1.61×10^{-1}	N/A	N/A	N/A	N/A
F12	2.43×10^{-9}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	1.27×10^{-2}	3.51×10^{-2}
F13	5.07×10^{-10}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	3.01×10^{-11}	4.51×10^{-2}
F14	6.84×10^{-11}	6.84×10^{-11}	2.47×10^{-11}	3.36×10^{-11}	8.31×10^{-2}	3.01×10^{-2}
F15	5.61×10^{-5}	4.11×10^{-7}	1.25×10^{-7}	6.35×10^{-5}	2.01×10^{-4}	1.76×10^{-2}
F16	3.08×10^{-1}	1.44×10^{-11}	1.44×10^{-11}	1.44×10^{-11}	8.04×10^{-1}	3.24×10^{-3}
F17	3.33×10^{-1}	9.10×10^{-12}	1.72×10^{-12}	2.69×10^{-12}	3.33×10^{-1}	3.33×10^{-1}
F18	1.52×10^{-1}	2.59×10^{-11}	2.59×10^{-11}	2.59×10^{-11}	1.28×10^{-1}	2.17×10^{-2}
F19	8.04×10^{-3}	1.20×10^{-11}	2.27×10^{-11}	1.34×10^{-11}	1.38×10^{-2}	1.76×10^{-2}
F20	1.68×10^{-1}	1.63×10^{-3}	4.93×10^{-8}	7.61×10^{-7}	6.96×10^{-1}	4.40×10^{-2}
F21	3.73×10^{-6}	7.26×10^{-11}	1.08×10^{-10}	1.43×10^{-11}	4.93×10^{-5}	4.80×10^{-9}
F22	6.74×10^{-4}	4.40×10^{-11}	1.95×10^{-11}	1.76×10^{-11}	2.64×10^{-3}	8.48×10^{-6}
F23	5.40×10^{-5}	2.27×10^{-10}	2.27×10^{-10}	2.27×10^{-10}	2.84×10^{-5}	3.39×10^{-7}

能，为其工程应用提供有力的验证依据。在实验设置中，保持统一参数配置：种群数量设为 30，最大迭代次数设为 500。

4.1　悬臂梁设计问题

悬臂梁设计问题的核心目标在于满足特定约束条件下实现悬臂总重量的最小化。以 x_j $(j=1,2,3,4,5)$ 表示悬臂块的长度，悬臂梁设计问题数学模型描述如式(14)所示。

$$\begin{cases}\min f(x)=0.062\,24(x_1+x_2+x_3+x_4+x_5)\\ \text{s.t.}\begin{cases}\dfrac{61}{x_1^3}+\dfrac{37}{x_2^3}+\dfrac{19}{x_3^3}+\dfrac{7}{x_4^3}+\dfrac{1}{x_5^3}-1\leqslant 0\\ 0.01\leqslant x_1,x_2,x_3,x_4,x_5\leqslant 100\end{cases}\end{cases}\tag{14}$$

各算法优化结果如表 4 所示，GSDBO 算法在全部参与测试的算法中以 1.339 968 的最优成本指标脱颖而出，其对应的四维设计参数组合为(6.016 565，5.302 865，4.503 619，3.490 883，2.159 925)。

表 4　悬臂梁设计问题各算法优化结果

Table 4　Optimization results of various algorithms for cantilever beam design problem

算法	x_1	x_2	x_3	x_4	x_5	最优结果
GSDBO	6.016 565	5.302 865	4.503 619	3.490 883	2.159 925	**1.339 968**
DBO	5.990 518	5.322 224	4.492 736	3.516 773	2.152 129	1.340 001
WOA	5.984 946	4.937 222	4.370 197	4.794 383	2.045 389	1.381 045
GJO	6.032 467	5.326 509	4.475 923	3.488 408	2.127 655	1.340 075
SHO	6.048 887	5.257 228	4.455 210	3.500 194	2.217 995	1.340 321
SCDBO1	6.012 060	5.323 133	4.515 474	3.482 932	2.140 679	1.339 995
SCDBO2	6.005 941	5.304 943	4.505 115	3.505 244	2.152 664	1.339 971

4.2　焊接梁设计问题

焊接梁设计优化问题作为机械工程领域的经典约束优化案例，通过调整结构参数实现制造成本最小化。焊接梁设计问题数学模型描述如式(15)～(18)所示，其中，x_1、x_2、x_3 和 x_4 分别表示焊接梁的焊缝宽度 h、横梁的长度 l、高度 t 和厚度 b。

$$\begin{cases}x=[x_1\quad x_2\quad x_3\quad x_4]=[h\quad l\quad t\quad b]\\ \min f(x)=1.104\,71x_1^2x_2+0.048\,11x_3x_4(14+x_2)\end{cases}\tag{15}$$

约束条件：

$$\text{s.t.}\begin{cases}g_1(x)=\tau(x)+\tau_{\max}\leqslant 0\\ g_2(x)=\sigma(x)+\sigma_{\max}\leqslant 0\\ g_3(x)=\delta(x)+\delta_{\max}\leqslant 0\\ g_4(x)=x_1-x_4\leqslant 0\\ g_5(x)=P-P_c(x)\leqslant 0\\ g_6(x)=0.125-x_1\leqslant 0\\ g_7(x)=0.104\,71x_1^2+0.048\,11x_3x_4(14+x_2)-5\leqslant 0\\ 0.1\leqslant x_1,x_4\leqslant 2.0,0.1\leqslant x_2,x_3\leqslant 10.0\end{cases}\tag{16}$$

式中：

$$\begin{cases}\tau(x)=\sqrt{(\tau')^2+2\tau'\tau''\dfrac{x_2^2}{2R}+(\tau'')^2}\\ \tau'=\dfrac{P}{\sqrt{2}x_1x_2}\\ \tau''=\dfrac{MR}{J}\end{cases}\tag{17}$$

$$\begin{cases}M=P\left(L+\dfrac{x_2}{2}\right)\\ R=\sqrt{\dfrac{x_2^2}{4}+\left(\dfrac{x_1+x_3}{2}\right)^2}\\ J=2\left\{\sqrt{2}x_1x_2\left[\dfrac{x_2^2}{4}+\left(\dfrac{x_1+x_3}{2}\right)^2\right]\right\}\end{cases}\tag{18}$$

$$\begin{cases}\sigma(x)=\dfrac{6PL}{x_4x_3^2}\\ \delta(x)=\dfrac{4PL^3}{Ex_4x_2^2}\\ P_c(x)=\dfrac{4.013E\sqrt{x_3^2x_4^5}}{L^2}\left(1-\dfrac{x_3}{2L}\sqrt{\dfrac{E}{4G}}\right)\end{cases}\tag{19}$$

各算法优化结果如表 5 所示，GSDBO 算法在所有测试算法中取得最小成本值 1.692 771，其最优变量组合为(0.205 728，3.234 940，9.036 613，0.205 730)。

4.3　机器人路径规划

在路径规划领域，栅格法是一种常用的环境建模方法，其核心思想是将任务环境离散化为均匀分布的矩形栅格单元[22]。本研究采用栅格地图法进行环境建模，其中每个栅格代表一个可搜索空间：白色区域标记为可通行区域，黑色栅格表示随机分布的障碍物。在栅格化环境建模中，各栅格单元(不含起止点)遵循以下空间拓扑约束：

$$cell(x,y)=\begin{cases}1, & \text{障碍栅格}\\ 0, & \text{可通行栅格}\end{cases}\tag{20}$$

路径长度的代价函数为：

$$\sum_{i=2}^{n}\sqrt{(x_i-x_{i-1})^2+(y_i-y_{i-1})^2}\tag{21}$$

表 5 焊接梁设计问题各算法优化结果

Table 5 Optimization results of various algorithms for welding beam design problems

算法	H	L	T	B	最优结果
GSDBO	0.205 728	3.234 940	9.036 613	0.205 730	**1.692 771**
DBO	0.205 706	3.235 522	9.036 672	0.205 729	1.692 823
WOA	0.188 355	3.815 664	9.015 408	0.206 699	1.746 753
GJO	0.205 443	3.245 458	9.036 305	0.206 083	1.696 378
SHO	0.204 068	3.290 352	9.020 356	0.206 472	1.700 636
SCDBO1	0.205 730	3.234 904	9.036 619	0.205 730	1.692 773
SCDBO2	0.205 729	3.234 894	9.036 692	0.205 729	1.692 774

式中:n 为生成路径中栅格的个数。

在 20×20 栅格环境中,将 GSDBO 算法与其余算法在平均路径长度和最优路径长度中进行对比。种群数量均设为 30,最大迭代次数为 500,为消除随机结果带来的影响,每个算法独立运行 30 次,各算法优化结果如表 6 所示。

表 6 机器人路径规划问题各算法优化结果

Table 6 Optimization results of various algorithms for robot path planning

算法	最优路径	平均路径	标准差
GSDBO	**27.56**	**29.92**	**1.33**
DBO	28.68	32.33	2.46
WOA	30.13	34.30	2.47
GJO	28.42	31.24	1.79
SHO	30.81	32.56	2.26
SCDBO1	28.42	31.57	2.05
SCDBO2	27.82	31.54	1.41

由表 6 可知,本文算法在最优路径、平均路径和标准差 3 个指标中均为最优结果。各算法的迭代曲线如图 5 所示。7 种算法的最优路径对比情况如图 6 和 7 所示,其中图 6 为 GSDBO 算法与其他群智能优化算法的最优路径对比,图 7 为 GSDBO 算法与原始 DBO 及其改进算法的最优路径对比。

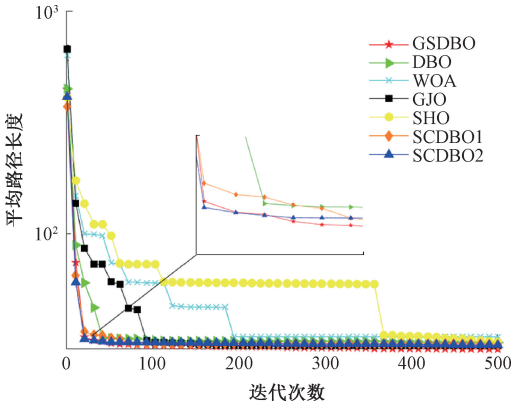


图 5 各算法迭代曲线

Fig. 5 Iteration curves of various algorithms

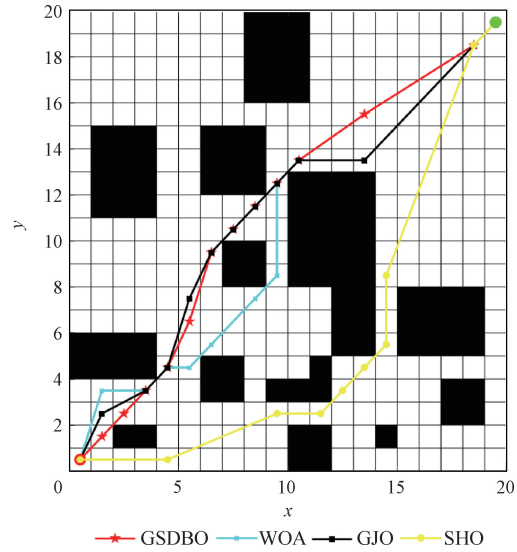


图 6 GSDBO 算法与其他群智能优化算法的最优路径对比

Fig. 6 Comparison of optimal paths between GSDBO algorithm and other swarm intelligence optimization algorithms

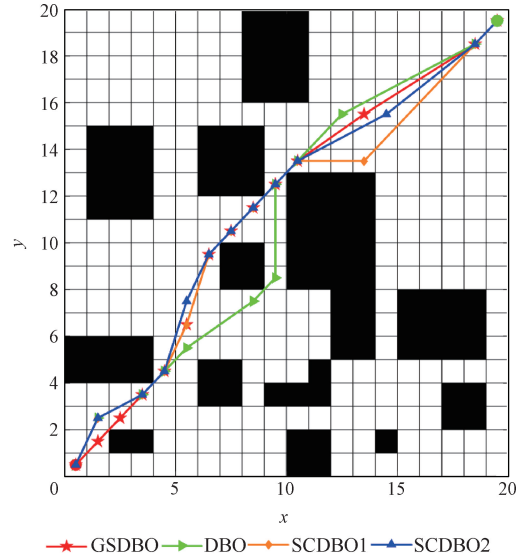


图 7 GSDBO 算法与原始 DBO 及其改进算法的最优路径对比

Fig. 7 Comparison of optimal paths between GSDBO algorithm, the original DBO and its improved algorithms

5　结　　论

本研究针对 DBO 算法 DBO 全局搜索能力弱、收敛慢、易陷局部最优等不足，提出一种基于黄金正弦算法的增强型蜣螂优化算法 GSDBO。在改进措施上，采用 Tent 混沌映射与透镜成像反向学习策略优化种群初始化过程，有效改善了初始解分布均匀性与多样性；用改进的黄金正弦算法重构滚球蜣螂位置更新公式，提升全局搜索精度与收敛速度；结合自适应 t 分布策略平衡全局探索与局部开发能力。实验选用 CEC2005 与 CEC2020 测试函数，将 GSDBO 算法与其余 6 种算法进行对比，并结合 Wilcoxon 秩和检验，验证了 GSDBO 算法在收敛速度和求解精度上的显著提升。最后，将该算法应用到在 3 个经典工程案例中均获最优解，验证了其解决复杂实际问题的有效性。

在未来的研究工作中，将着重对算法性能进行深度优化，全方位提升其收敛速度、求解精度以及稳定性等关键指标。同时，计划将优化后的算法拓展至通信工程领域，如无线传感器网络布局、信号处理等方面，希望为该领域提供更高效的解决方案。

参考文献

[1] ZHU F, LI G SH, TANG H, et al. Dung beetle optimization algorithm based on quantum computing and multi-strategy fusion for solving engineering problems[J]. Expert Systems with Applications, 2024, 236: 121219.

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

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

[4] TANYILDIZI E, DEMIR G. Golden sine algorithm: A novel math-inspired algorithm[J]. Advances in Electrical & Computer Engineering, 2017, 17(2): 71-78.

[5] CHOPRA N, ANSARI M M. Golden jackal optimization: A novel nature-inspired optimizer for engineering applications[J]. Expert Systems with Applications, 2022, 198: 116924.

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

[7] 陈庆斌，杨耿煌，耿丽清，等. 基于合作博弈策略和 DBO-BiLSTM-Attention 的电动汽车充电桩故障预测[J]. 电子测量与仪器学报，2025，39(4)：163-171.
CHEN Q B, YANG G H, GENG L Q, et al. Fault prediction of electric vehicle charging stations based on cooperative game strategy and DBO-BiLSTM-Attention[J]. Journal of Electronic Measurement and Instrumentation, 2025, 39(4): 163-171.

[8] 梅雨琳，曲良东，饶爽. 多策略改进蜣螂优化算法的无人机航迹规划[J]. 电子测量技术，2025，48(11)：67-77.
MEI Y L, QU L D, RAO SH. Multi-strategy improved dung beetle optimization algorithm for UAV path planning[J]. Electronic Measurement Technology, 2025, 48(11): 67-77.

[9] LUO B X, ZUO P, ZHU L J, et al. A wind power density forecasting model based on RF-DBO-VMD feature selection and BiGRU optimized by the attention mechanism[J]. Atmosphere, 2025, 16(3): 266.

[10] 张诗云，朱菊香，张涛，等. 基于 VMD-DBO-LSTM 的空气质量预测[J]. 国外电子测量技术，2024，43(3)：58-66.
ZHANG SH Y, ZHU J X, ZHANG T, et al. Air quality predication based on VMD-DBO-LSTM[J]. Foreign Electronic Measurement Technology, 2024, 43(3): 58-66.

[11] 余荣川，徐子娟，陈小梅，等. 改进蜣螂优化算法的研究[J]. 南宁师范大学学报(自然科学版)，2024，41(2)：51-61.
YU R CH, XU Z J, CHEN X M, et al. Research on multi-strategy improvement of dung beetle optimization algorithm[J]. Journal of Nanning Normal University (Natural Science Edition), 2024, 41(2): 51-61.

[12] 蒋翱徽，刘文红. 改进 DBO 优化 VMD 算法的管道泄漏信号降噪研究[J]. 上海电机学院学报，2025，28(2)：100-106.
JIANG AO H, LIU W H. Research on pipeline leakage signal denoising by improving DBO optimized VMD algorithm[J]. Journal of Shanghai Dianji University, 2025, 28(2): 100-106.

[13] WANG X, WEI Y X, GUO Z H, et al. A sinh-cosh-enhanced DBO algorithm applied to global optimization problems[J]. Biomimetics, 2024, 9(5): 271.

[14] 雒明世，陈利军，金浩. 基于测距修正与改进蜣螂优化的 DV-Hop 定位算法[J]. 无线通信技术，2024，33(4)：32-38.
LUO M SH, CHEN L J, JIN H. DV-Hop localization algorithm based on ranging correction and improved dung beetle optimizer[J]. Wireless Communication Technology, 2024, 33(4): 32-38.

[15] 刘微，任腾腾，韩广雨，等. 多策略改进蜣螂优化算法及其应用[J]. 电子测量技术，2024，47(12)：109-121.

LIU W, REN T T, HAN G Y, et al. Multi-strategy improvement of dung beetle optimization algorithm and its application[J]. Electronic Measurement Technology, 2024, 47(12): 109-121.

[16] 杨赫然,李帅,孙兴伟,等.基于改进松鼠搜索算法优化神经网络的数控机床进给系统热误差预测[J].仪器仪表学报,2024,45(1):60-69.
YANG H R,LI SH,SUN X W, et al. Thermal error prediction of CNC machine tool feed system based on neural network optimized by improved squirrel search algorithm[J]. Chinese Journal of Scientific Instrument, 2024, 45(1): 60-69.

[17] KAUR G, ARORA S. Chaotic whale optimization algorithm[J]. Journal of Computational Design and Engineering, 2018, 5(3): 275-284.

[18] XIAO CH, YANG H L, ZHANG B. Multi-unmanned aerial vehicle path planning based on improved nutcracker optimization algorithm [J]. Drones, 2025, 9(2): 116.

[19] LIU W, YAN W L, LI T, et al. A multi-strategy improved grasshopper optimization algorithm for solving global optimization and engineering problems [J]. International Journal of Computational Intelligence Systems, 2024, 17(1): 182.

[20] YIN SH H, LUO Q F, DU Y L, et al. DTSMA: Dominant swarm with adaptive t-distribution mutation-based slime mould algorithm[J]. Mathematical Biosciences and Engineering, 2022, 19(3): 2240-2285.

[21] ZHAO SH J, ZHANG T R, MA SH L, et al. Seahorse optimizer: A novel nature-inspired meta-heuristic for global optimization problems[J]. Applied Intelligence, 2023, 53(10): 11833-11860.

[22] 张彪,李永强.基于动态寻优蚁群算法的移动机器人路径规划[J].仪器仪表学报,2025,46(3):74-85.
ZHANG B, LI Y Q. Path planning of mobile robot based on the dynamic optimization ant colony algorithm [J]. Chinese Journal of Scientific Instrument, 2025, 46(3): 74-85.

作者简介

蔡春雷,硕士研究生,主要研究方向为智能优化算法及其应用。

E-mail:553464750@qq.com

刘微(通信作者),博士,教授,主要研究方向为智能优化算法及其应用。

E-mail:LiuWei19781020@126.com

杨迪雅,硕士研究生,主要研究方向为智能优化算法及其应用。

E-mail:1596306960@qq.com