

◆ 计算科学 ◆

融合疯狂秃鹰搜索的混沌正余弦算法^{*}周 甜, 麦雄发, 刘利斌^{**}, 郑贵林

(南宁师范大学广西应用数学中心, 广西南宁 530100)

摘要:针对正余弦算法(Sine Cosine Algorithm, SCA)在解决优化问题时存在收敛速度慢、计算精度低等缺陷, 本文提出一种融合疯狂秃鹰搜索算法的混沌正余弦算法(Chaotic Sine Cosine Algorithm based on Crazy Bald-eagle Search, CSCA-CBS)。CSCA-CBS 采用结合 Logistic 与 Tent 的混合混沌映射进行种群初始化, 从而获得更加均匀和多样的初始种群; 受秃鹰搜索算法所启发, CSCA-CBS 采用带有疯狂算子的秃鹰搜索策略, 该策略能够提升 CSCA-CBS 的全局探索能力; 为了在迭代后期避免陷入局部最优区域, CSCA-CBS 使用逐维反向柯西变异策略对种群进行有规律的扰动, 极大地集成了反向学习和柯西变异的优势。在 15 个基准函数上进行的仿真实验结果表明, CSCA-CBS 在计算代价和可靠性、解的质量分析以及收敛性能等方面优于多种先进的 SCA 变体和非 SCA 基准算法。此外, 土壤水分特征曲线的参数反演实验进一步验证了 CSCA-CBS 的实用性和有效性。

关键词:正余弦算法; Logistic-Tent 混沌映射; 秃鹰搜索算法; 疯狂算子; 参数反演

中图分类号: TP301 文献标识码: A 文章编号: 1005-9164(2025)01-0106-15

DOI: 10.13656/j.cnki.gxkx.20241106.001

在数学领域中, 优化是指在给定问题的解集合中依据合适的方法(如牛顿法、单纯形法等)来搜寻全局最优解或近似最优解。然而, 由于在实际工程问题中普遍存在测量误差、外部干扰等不规则性因素, 以及系统的复杂性, 这些因素导致理论模型与实际之间存在较大误差, 从而使得传统基于梯度的优化算法^[1]难以准确地找到全局最优解。因此, 基于群体智能的元启发式优化算法开始被人们所关注, 其通过模拟自然

界中各类生物的复杂行为进行迭代寻优。近些年已被成功应用于图像处理^[2]、电力系统^[3]、机器学习^[4]、工程设计^[5]等领域。

正余弦算法(Sine Cosine Algorithm, SCA)是一种模拟正余弦函数来回振荡行为的群体智能优化算法^[6]。和其他群体智能优化算法一样, SCA 在搜索全局最优时经过两个阶段: 全局探测和局部挖掘。因为 SCA 具有结构简单、实现方便等优点, 同时其随机

收稿日期: 2024-06-25

修回日期: 2024-09-09

^{*} 国家自然科学基金项目(12361087), 广西科技计划项目(桂科 AD23023003)和广西研究生教育创新计划项目(JGY2024267)资助。

【第一作者简介】

周 甜(1998—), 女, 在读硕士研究生, 主要从事群体智能算法及其应用研究。

【**通信作者简介】

刘利斌(1982—), 男, 博士, 教授, 主要从事群体智能算法与微分方程数值解等研究, E-mail: liulibin969@163.com。

【引用本文】

周甜, 麦雄发, 刘利斌, 等. 融合疯狂秃鹰搜索的混沌正余弦算法[J]. 广西科学, 2025, 32(1): 106-120.

ZHOU T, MAI X F, LIU L B, et al. A Chaotic Sine Cosine Algorithm Based on Crazy Bald-Eagle Search [J]. Guangxi Sciences, 2025, 32(1): 106-120.

参数可以很好地平衡算法的探索 and 开发能力,所以被广泛应用于解决各类工程应用问题,例如经济排放调度^[7]、路径规划^[8]、文本特征选择^[9]、光纤放大器的设计^[10]等。Rizk-Allah 等^[11]对 SCA 的最新研究进展进行了全面综述,并将 SCA 变体算法的改进思路大致分为以下 3 类。①种群多样性的加强。一般来说,SCA 的随机初始化方式会因群体分布不均,从而降低种群多样性,最终导致局部收敛甚至搜索停滞。为算法迭代提供一个较优的初始种群位置,可以显著提高算法的搜索能力,因此各种混沌学理论被用于替代随机方法,如 Singer 映射^[12]和 Logistic 映射^[13]。此外,Feng 等^[14]通过对立学习生成对立解来扩大搜索范围,避免算法陷入局部最优。王华秋等^[15]在初始化阶段引入拉丁超立方体,通过均匀划分搜索空间并采样,使初始种群分布更均匀。Chen 等^[16]在 SCA 中引入正交学习,扩大了算法的邻域搜索能力。②位置更新方程的设计。在 SCA 中,种群位置的更新机制主要依赖正弦和余弦函数,通过二者在进化过程中平衡探索与开发,从而提升算法收敛性能。但近年来随着人工智能、大数据、物联网、5G、元宇宙的发展,优化问题正在向大规模、多任务、多目标、多参数等方向演化,给 SCA 带来了巨大挑战。如何避免算法在面对复杂优化问题时陷入局部最优成为了该领域重点关注的研究热点。Sindhu 等^[17]提出一种具有精英策略引导最佳解机制的正余弦算法用于特征选择,成功提升了算法的全局搜索能力。Raut 等^[18]为了最大限度提高种群的搜索效率,在更新机制中充分结合莱维飞行策略并将其用于解决网络配置和分布式发电机同时分配的问题。但是,这种单一改进策略在许多问题中依然很难发挥算法的全部性能。因此,Liu 等^[19]分别设计了增强的精英领导力、交叉和最优选择结合以及逐维更新这 3 种新颖策略来修正 SCA。③与先进算法的混合。两种或两种以上算法的更新机制组合可以有效增强原始 SCA 的搜索能力,Issa 等^[20]提出粒子群优化与 SCA 的混合算法,通过共享两种算法的进化机制,在生物局部序列对比上得到成功应用。Gupta 等^[21]结合差分进化与 SCA 来解决多维背包问题。Ansari 等^[22]提出将海豚群优化算法与 SCA 混合来研究视频内容自动检测。

尽管 SCA 已经在各个方面取得了巨大进展,但是仍需要进一步提升其性能。因此,本文提出一种融合疯狂秃鹰搜索的混沌正余弦算法 (Chaotic Sine Cosine Algorithm based on Crazy Bald - eagle

Search, CSCA-CBS)。为了验证 CSCA-CBS 的优化能力,在 15 个基准函数上进行测试,并使用非参数检验方法进行统计学分析,然后通过求解土壤水分特征曲线参数反演问题,进一步验证 CSCA-CBS 的实用性和有效性。

为了优化 SCA,本文提出 3 种改进策略:①引入 Logistic-Tent 映射策略初始化种群,融合 Tent 映射的均匀分布特性和 Logistic 映射的非线性特征,提高初始种群多样性;②采用“疯狂算子”扰动种群,增强全局探索能力并避免陷入局部最优,同时将秃鹰搜索策略与“疯狂算子”相融合,进一步提升全局探索性能和效率;③提出逐维反向变异策略,通过反向扩大搜索空间,允许种群在陷入局部最优时接受部分较差解以跳出局部最优,并增强变异策略效率,减少维度间干扰,从而提升算法寻优能力。

1 基本正余弦算法

考虑一个 n 维最小化问题:

$$\min_{x_i \in \Omega} f(x_1, x_2, \dots, x_n), \quad (1)$$

其中, x_i 为决策变量, $\Omega = \{lb_i \leq x_i \leq ub_i, i = 1, 2, \dots, n\}$ 是解空间。 ub_i 和 lb_i 分别为 x_i 在解空间中的上、下边界。

SCA 利用正余弦函数构建描述全局探索与局部开发的数学模型:

$$x_{i,j}^{iter+1} = \begin{cases} x_{i,j}^{iter} + r_1 \times \sin(r_2) \times |r_3 \times p_{best,j}^{iter} - x_{i,j}^{iter}|, & r_4 < 0.5 \\ x_{i,j}^{iter} + r_1 \times \cos(r_2) \times |r_3 \times p_{best,j}^{iter} - x_{i,j}^{iter}|, & \text{else} \end{cases}, \quad (2)$$

$$r_1 = a - a \times \frac{iter}{Max_iter}, \quad (3)$$

式中, $x_{i,j}^{iter}$ 和 $p_{best,j}^{iter}$ 分别表示个体 i 和最优个体在第 j 维经过第 $iter$ 次迭代后的种群位置。 r_1 为探索($r_1 > 1$)与挖掘($r_1 < 1$)的控制参数, r_2 、 r_3 和 r_4 分别是 $[0, 2\pi]$ 、 $[0, 2]$ 和 $[0, 1]$ 间的随机数,常数 $a = 2$, Max_iter 为最大迭代次数。

令个体上一代的最佳位置 $x_*^{iter} = \operatorname{argmin} f(x_{i,j}^{iter})$, 且 $y_*^{iter+1} = \operatorname{argmin} f(x_{i,j}^{iter+1})$, 通过贪心选择策略来获取当前全局最优位置 x_*^{iter+1} , 即:

$$x_*^{iter+1} = \begin{cases} y_*^{iter+1}, & f(y_*^{iter+1}) < f(x_*^{iter}) \\ x_*^{iter}, & \text{otherwise} \end{cases}. \quad (4)$$

2 融合疯狂秃鹰搜索的混沌正弦算法

2.1 Logistic-Tent 映射初始化种群

SCA 中的随机初始解方式会导致种群分布不均匀, 多样性降低, 进而限制算法的优化能力。混沌理论凭借其独特的遍历性和规律性, 与 SCA 的初始化需求高度契合, 有助于种群更充分地探索整个搜索空间。目前使用广泛的混沌映射主要是 Logistic 映射^[23]和 Tent 映射^[24], 且大量文献已证明这两种方法在种群分布上优于随机函数^[25-26]。Logistic 映射[图 1(a)]和 Tent 映射[图 1(b)]的直方图均反映了两种映射不同的局限性, 因此它们难以很好地指导种群的高效探索。值得注意的是, Logistic 映射使 $[0.0, 0.1]$ 和 $[0.9, 1.0]$ 的种群分布密集, 而 Tent 映射可以大致认为是在整个搜索空间具有分布, 但分布均匀度稍差。

Logistic 映射和 Tent 映射的公式分别如式(5)和式(6)所示:

$$z_{n+1} = \mu z_n (1 - z_n), \quad (5)$$

$$z_{n+1} = \begin{cases} kz_n/2, & z_n < 0.5 \\ k(1 - z_n)/2, & z_n \geq 0.5 \end{cases}, \quad (6)$$

其中, μ 和 k 均为控制参数, $\mu \in [3.57, 4.00]$ ^[27], $k \in [2, 4]$, z_n 为状态值。

基于此, 本文将两种映射结合, 提出一种 Logistic-Tent 映射策略用于初始化种群, Logistic-Tent 映射直方图[图 1(c)]既整体均匀, 又在某些区域有起伏, 能更好地兼顾全局和局部搜索, 从而提升算法的综合性能。具体公式如下:

$$z_{n+1} = \begin{cases} (rz_n(1 - z_n) + (4 - r)z_n/2) \bmod 1, & z_n < 0.5 \\ (rz_n(1 - z_n) + (4 - r)(1 - z_n)/2) \bmod 1, & \text{else} \end{cases}, \quad (7)$$

其中, r 为混沌参数, $r \in (0, 4]$ ^[27]。接着, 将产生的混沌序列值映射到搜索空间:

$$x_{i,j} = lb_j + z_{i,j} \cdot (ub_j - lb_j), \quad (8)$$

其中, $x_{i,j}$ 表示第 i 个个体在第 j 维的位置, $z_{i,j}$ 是第 i 个个体的状态, ub_j 和 lb_j 为 x_i 取值的上、下边界。

2.2 疯狂秃鹰搜索策略

SCA 由于缺乏对历史信息与当前信息的整合, 仅依赖正余弦函数进行交替迭代寻优难以平衡收敛速度与寻优精度, 在诸多复杂优化问题中表现乏力。作为一种新型的全局优化算法, 秃鹰搜索算法^[28]会充分根据历史迭代中的最佳信息和平均化信息来选

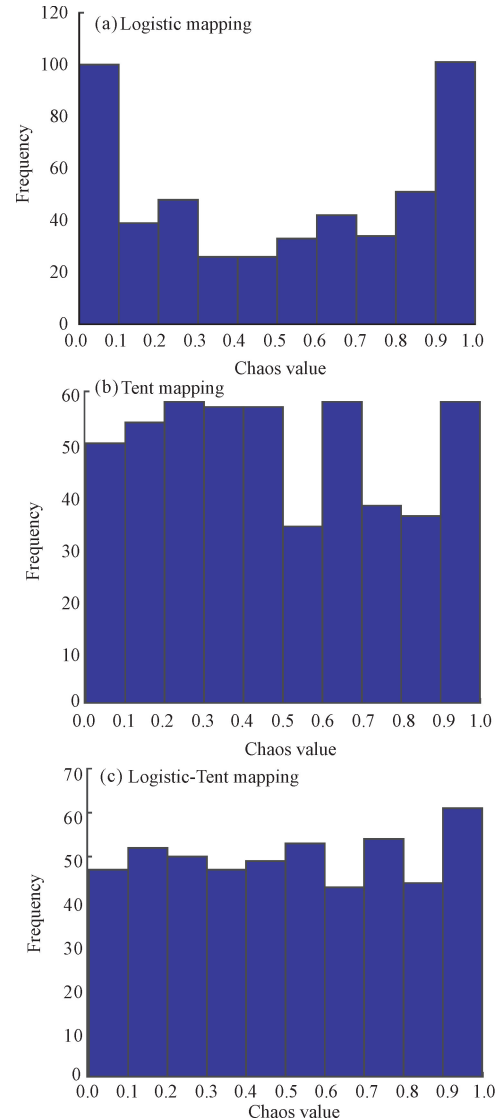


图 1 混沌映射的分布情况对比图

Fig. 1 Comparison of chaotic mapping

择当前最佳搜索区域, 其搜索行为如式(9)所示:

$$x_{i,j}^{iter+1} = x_{best,j}^{iter} + \alpha \times r \times (x_{mean}^{iter} - x_{i,j}^{iter}), \quad (9)$$

其中, $x_{best,j}^{iter}$ 是秃鹰搜索当前的最佳位置, x_{mean}^{iter} 是搜索前后的平均位置, $\alpha \in [1.5, 2.0]$ 表示控制秃鹰搜索范围的参数^[28], $r \in (0, 1)$ 是伪随机数。

虽然加入秃鹰搜索策略能够有效弥补 SCA 的不足, 并使得种群一直朝着最优区域的方向进行移动, 但难免会出现种群的聚集现象, 从而大大降低种群多样性, 增加陷入局部最优的概率。于是, 本文引入疯狂算子对种群进行微小扰动, 使得种群可以在充分搜索的过程中维持其多样性。综上, 融合疯狂秃鹰搜索策略的 SCA 更新公式如下:

$$x_{i,j}^{iter+1} = \begin{cases} \tilde{x}_{best,j}^{iter} + r_1 \times \sin(r_2) \times |r_3 \times x_{mean}^{iter} - x_{i,j}^{iter}|, \\ r_4 < 0.5 \\ \tilde{x}_{best,j}^{iter} + r_1 \times \cos(r_2) \times |r_3 \times x_{mean}^{iter} - x_{i,j}^{iter}|, \text{else} \end{cases}, \quad (10)$$

$$\tilde{x}_{best,j}^{iter} = x_{best,j}^{iter} \times (1 + P(c_4) \times \text{sign}(c_4) \times x_{craziness}), \quad (11)$$

其中, $\tilde{x}_{best,j}^{iter}$ 表示通过疯狂算子进行微小扰动后的种群最佳位置, $x_{craziness}$ 是疯狂因子。 $P(c_4)$ 和 $\text{sign}(c_4)$ 的表达式如下:

$$P(c_4) = \begin{cases} 1, c_4 \leq P_c \\ 0, \text{otherwise} \end{cases}, \quad (12)$$

$$\text{sign}(c_4) = \begin{cases} -1, c_4 \geq 0.5 \\ 1, \text{otherwise} \end{cases}, \quad (13)$$

其中, P_c 表示疯狂概率, c_4 为一个随机数。

2.3 逐维反向柯西变异

与高斯函数相比,柯西函数能够在短时间内对种群进行大幅度扰动,使算法在陷入局部最优区域时快速逃离^[29]。然而,柯西变异缺乏明确的逃离方向,导致种群逃离行为杂乱无章,甚至可能抛弃处于较优区域的个体。针对这个问题,本文引入透镜成像反向学习策略将搜索空间范围进行反向扩大,一是使得种群在陷入局部最优时能够适当接受个别较差的解作为踏板跳出局部最优区域;二是能够在柯西扰动时指明一条明确的逃离方向。另外,为了进一步提高变异策略的效率以及维度之间的相互干扰,通过逐维变异可以更好地得到变异解。整个变异过程的数学表达式如下:

$$x_{best,j}^{iter} = \frac{lb_j + ub_j}{2} + \frac{lb_j + ub_j}{2k'} - \frac{x_{best,j}^{iter}}{k'}, \quad (14)$$

$$x_{best,j}^{iter+1} = x_{best,j}^{iter} + x_{best,j}^{iter} \otimes \text{cauchy}(0,1), \quad (15)$$

其中, k' 为缩放因子, $\otimes$ 表示逐维乘积操作。

综上, CSCA-CBS 的算法实现步骤如下:

算法 1 CSCA-CBS 算法

初始化: Logistic-Tent 混沌序列初始化种群。

- ① 评估种群位置并排序, 并记录个体最优和全局最优。
- ② $iter = 1$
- ③ while ($iter < Max_iter$)
- ④ for $i = 1 : N$
- ⑤ for $j = 1 : D$

- ⑥ 根据公式(3), (12), (13) 计算 $r_1, P(c_4), \text{sign}(c_4)$;
- ⑦ 产生 $[0, 2\pi], [0, 2], [0, 1]$ 之间的随机数 r_2, r_3, r_4 ;
- ⑧ 根据公式(10)更新个体位置;
- ⑨ end for
- ⑩ 更新个体最优和全局最优;
- ⑪ 根据公式(14)产生种群的反向解;
- ⑫ 根据公式(15)对种群进行扰动, 确定全局最优解;
- ⑬ end for
- ⑭ $iter = iter + 1$
- ⑮ end while
- ⑯ 输出全局最优解和最优适应度。

3 实验与结果分析

3.1 实验设置

为了合理评估 CSCA-CBS 的优化性能, 选取 15 个常用的基准测试函数^[30](表 1), 并设置种群规模 N 为 50, 最大迭代次数 Max_iter 为 1 000, 维度分别为 30 和 100, 以确保实验的可靠性和公平性。其中, CSCA-CBS 中的控制参数 $\alpha = 2$, 疯狂因子 $x_{craziness} = 0.0001$, 缩放因子 $k' = 10\,000$ 。为了全面评估算法性能, 本文采用“平均值±标准差”的形式呈现结果, 并记录各函数收敛的平均迭代次数(AI)和优化成功率(SR), 以衡量算法的计算代价和可靠性, 保障数值结果的可靠性。

当求解最优值 $f_{\min}(x)$ 与理论最优值 f^* 之间的距离 $|f_{\min}(x) - f^*| \leq \theta$ 时, 即代表优化成功, 并记录此时的迭代次数。其中, θ 表示精度阈值, 根据文献[31-32], 设置见表 1 所示。整个数值仿真实验的工作硬件环境: 操作系统为 Windows 11, CPU 为 Intel(R) Core(TM) i5-1155G7, 主频为 2.50 GHz, 内存为 8 GB; 编程软件为 MATLAB 2021a。

3.2 不同改进策略的有效性分析

为了分析不同改进策略对 SCA 优化性能的影响, 本节将原始 SCA、基于 Logistic-Tent 序列的 SCA(简称 CSCA)、基于疯狂秃鹰搜索策略的 SCA(简称 CBSSCA)、结合逐维反向柯西变异策略的 SCA(简称 DRCSCA)同时用于求解表 1 中 15 个基准函数。

表 1 基准测试函数

Table 1 Benchmark functions

函数名称 Function name	取值范围 Scope	理论 最优值 Optimum value	精度 阈值 Accuracy
f_1 : Sphere	$[-100, 100]$	0	$1e-6$
f_2 : Generalized rosenbrock	$[-100, 100]$	0	$1e+2$
f_3 : Ackley	$[-32, 32]$	0	$1e-6$
f_4 : Generalized griewank	$[-600, 600]$	0	$1e-6$
f_5 : Generalized rastrigin	$[-5.12, 5.12]$	0	$5e+1$
f_6 : Non-continuous rastrigin	$[-500, 500]$	0	$5e+1$
f_7 : Schwefel's function 2.26	$[-500, 500]$	0	$2e+3$
f_8 : Schwefel's function 1.2	$[-100, 100]$	0	$1e-6$
f_9 : Schwefel's function 1.2.1	$[-100, 100]$	0	$1e-6$
f_{10} : High conditioned elliptic	$[-100, 100]$	0	$1e-6$
f_{11} : Weierstrass	$[-0.5, 0.5]$	0	$1e-6$
f_{12} : Schwefel's function 2.22	$[-10, 10]$	0	$1e-6$
f_{13} : Schwefel's function 2.21	$[-100, 100]$	0	$1e-6$
f_{14} : Generalized penalized 1	$[-50, 50]$	0	$1e-6$
f_{15} : Generalized penalized 2	$[-50, 50]$	0	$1e-6$

表 2 各个改进策略在 SCA 中的有效性分析结果 (30 维)

Table 2 Effectiveness analysis results of various improvement strategies in SCA (30 D)

函数 Function	SCA	CSCA	CBSSCA	DRCSA	CSCA-CBS
f_1	$1.41E+04 \pm 3.92E-04$	$7.79E-06 \pm 1.37E-05$	$7.56E-186 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_2	$1.24E+02 \pm 3.54E+02$	$6.57E+01 \pm 1.03E+02$	$2.66E+01 \pm 8.13E-01$	$2.87E+01 \pm 1.41E-01$	$2.88E+01 \pm 1.77E-01$
f_3	$1.29E+01 \pm 9.23E+00$	$1.27E+01 \pm 8.36E+00$	$1.40E+01 \pm 9.67E+00$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_4	$1.09E-01 \pm 2.03E-01$	$1.87E-01 \pm 1.74E-01$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_5	$1.30E+01 \pm 3.12E+01$	$8.82E-01 \pm 1.49E+00$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_6	$7.32E+01 \pm 4.67E+01$	$6.10E+01 \pm 5.05E+01$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_7	$8.26E+03 \pm 3.25E+02$	$8.37E+03 \pm 2.41E+02$	$5.80E+03 \pm 6.02E+02$	$9.72E+03 \pm 2.95E+02$	$9.28E+03 \pm 3.42E+02$
f_8	$9.65E+02 \pm 1.17E+03$	$1.29E+02 \pm 2.04E+02$	$1.17E-58 \pm 1.35E-58$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_9	$2.16E+05 \pm 2.58E+05$	$3.06E+05 \pm 2.89E+05$	$3.52E+05 \pm 6.10E+05$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_{10}	$1.03E-04 \pm 2.99E-04$	$7.10E-05 \pm 1.38E-04$	$1.09E-185 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_{11}	$1.29E-05 \pm 2.61E-05$	$1.47E-05 \pm 2.97E-05$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_{12}	$4.34E-07 \pm 5.97E-07$	$1.94E-08 \pm 3.74E-07$	$5.18E-96 \pm 8.53E-96$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_{13}	$7.54E+00 \pm 5.80E+00$	$9.50E+00 \pm 1.01E+01$	$1.70E-30 \pm 4.12E-30$	$0.00E+00 \pm 0.00E+00$	$0.00E+00 \pm 0.00E+00$
f_{14}	$1.56E+03 \pm 3.62E+03$	$4.91E+03 \pm 1.19E+04$	$3.77E-01 \pm 5.60E-02$	$1.12E+00 \pm 3.66E-02$	$9.68E-01 \pm 1.30E-01$
f_{15}	$2.16E+00 \pm 1.88E-01$	$2.09E+00 \pm 2.34E-01$	$1.19E+00 \pm 3.49E-01$	$2.80E+00 \pm 3.27E-02$	$2.83E+00 \pm 1.03E-01$

Note; data in bold are the optimal results.

3.3 对比实验

选择 ASCA^[33]、IWCCSCA^[34]、OBSCA^[35] 以及

由表 2 可知, 由 Logistic-Tent 映射组合而成的 CSCA 在增加种群多样性的同时保证了种群的均匀分布, 因此在 SCA 的基础上使收敛效果明显提升 1 至 3 个数量级。引入疯狂算子对种群进行微小扰动后, 能使种群在充分搜索的过程中维持多样性。对于较难收敛的 f_2 、 f_7 、 f_{14} 、 f_{15} , CBSSCA 的性能相较于 SCA 提升了 0 至 4 个数量级, 而对于其余测试函数, CBSSCA 在 4 个函数上达到了理论最优值。尽管 Logistic-Tent 映射和疯狂算子的加入使 SCA 在收敛速度和精度上均取得满意结果, 但 SCA 在迭代过程中仍会逐渐收敛到局部最优。为此, 在迭代后期提出了逐维反向柯西变异策略。从表 2 可以看出, DRCS-CA 相较于 SCA 收敛性显著增强, 这表明逐维反向柯西变异策略的应用使算法性能得到了大幅提升。实验结果表明, 3 种策略的引入均对算法性能产生积极影响, 且逐维反向柯西变异策略的效果最为显著。将 3 种策略融合后, CSCA-CBS 在各个阶段都表现出更优异的性能。具体而言, 融合策略在算法的初始阶段、收敛阶段以及避免陷入局部最优方面均展现出明显的优势。

IM-CSA^[36] 等 4 种对比算法来展现本文所提出的 CSCA-CBS 的竞争力。本文将从计算代价和可靠性、

解的质量分析以及收敛性能这 3 个方面进行详细比较。对比算法的各参数均按照对应文献进行设置,所有测试函数结果均独立执行 30 维。

3.3.1 计算代价和可靠程度

本文采用 AI 和 SR 这两个指标来评价各算法的计算代价和可靠性。30 维独立运行的 AI 和 SR 结果见表 3。本文所提出的 CSCA-CBS 由于在初始化、全局探测以及后期开发阶段都对种群多样性进行了

有效提升,大大加快了算法的搜索速度。这使得 CS-CA-CBS 在优化的 12 个函数中均实现了 100% 的成功率。另外,由于将透镜成像反向学习策略引入到后期种群的大扰动阶段,CSCA-CBS 极度地减小了计算代价,平均仅迭代几次就可以达到所需精度。综合来看,本文所提出的 CSCA-CBS 在计算代价和可靠程度方面表现最佳。

表 3 平均迭代次数和优化成功率的据对比 (30 维)

Table 3 Comparison of average iteration times and optimization success rate (30 D)

函数 Function	SCA		CSCA		CBSSCA		DRCSCA		CSCA-CBS	
	AI	SR	AI	SR	AI	SR	AI	SR	AI	SR
f_1	N. A.	0%	76.43	100%	51.81	100%	51.81	100%	1.00	100%
f_2	341.60	100%	32.77	100%	33.44	100%	33.44	100%	0.00	100%
f_3	N. A.	0%	108.17	100%	73.67	100%	73.67	100%	1.53	100%
f_4	968.97	6.67%	74.17	100%	62.00	100%	62.00	100%	1.00	100%
f_5	23.43	100%	29.37	100%	17.40	100%	17.40	100%	0.00	100%
f_6	617.53	73.33%	48.57	100%	37.60	100%	37.60	100%	0.20	100%
f_7	9.23	100%	21.73	100%	N. A.	0%	N. A.	0%	N. A.	0%
f_8	N. A.	0%	79.23	100%	295.00	100%	295.00	100%	0.97	100%
f_9	N. A.	0%	111.70	100%	197.00	100%	197.00	100%	1.10	100%
f_{10}	978.87	3.33%	68.80	100%	52.61	100%	52.61	100%	1.00	100%
f_{11}	N. A.	0%	150.53	100%	783.42	40%	783.42	40%	2.13	100%
f_{12}	984.30	3.33%	102.27	100%	69.20	100%	69.20	100%	1.73	100%
f_{13}	N. A.	0%	118.63	100%	103.01	100%	103.01	100%	1.73	100%
f_{14}	N. A.	0%	N. A.	0%	N. A.	0%	N. A.	0%	N. A.	0%
f_{15}	N. A.	0%	976.70	3.33%	N. A.	0%	N. A.	0%	N. A.	0%

Note: data in bold are the optimal results. "N. A." indicates a solution failure, i. e., the result still does not reach the threshold accuracy at the end of 1 000 iterations θ .

3.3.2 解的质量分析

5 种不同 SCA 变体求解 15 个基准函数在 30 维下的质量分析见表 4。本文通过 Wilcoxon 秩和检验进行统计,显著性水平为 0.05。由表 4 可知,与 AS-CA、IWCCSCA、OBSCA 和 IM-SCA 相比,CSCA-CBS 能够在 11 个函数上获得更优的结果,并且这 11 个函数均获得理论最优值。对比 ASCA,CSCA-CBS 在除 f_2 、 f_7 、 f_{14} 、 f_{15} 以外均取得了更优结果。与 IWCCSCA 相比,CSCA-CBS 分别在 6 个函数上获得最优结果以及在 5 个函数上取得相似结果,但在 f_2 上略逊一筹。与 OBSCA 相比,CSCA-CBS 在 6 个函数上表现更优。而与 IM-SCA 相比,CSCA-CBS 有 4

个较优和 8 个同等结果。综上,本文所提的 CSCA-CBS 虽然在 f_2 、 f_7 、 f_{14} 、 f_{15} 这 4 个函数上与其他 SCA 变体一样陷入局部最优,但在剩下的 11 个函数均取得了理论最优,说明 CSCA-CBS 能够在大多数函数上充分对搜索空间进行探索,并具备较强的全局优化能力。为了观察和分析维度变化对算法性能的影响,表 5 列出了在其他参数不变、问题维度为 100 维的条件下 5 种 SCA 变体的求解精度,每个算法分别独立运行 30 次并取平均值。由表 5 可知,CSCA-CBS 的性能依然是最好的,对于 f_1 、 f_3 — f_6 、 f_8 — f_{13} ,CSCA-CBS 在 30 次实验中均能找到理论最优解。

表 4 ASCA、IWCCSCA、OBSCA、IM-SCA、CSCA-CBS 的优化结果 (30 维)

Table 4 Optimization results of ASCA, IWCCSCA, OBSCA, IM-SCA, and CSCA-CBS (30 D)

函数 Function	ASCA	IWCCSCA	OBSCA	IM-SCA	CSCA-CBS
f_1	$1.28\text{E}-01 \pm 2.87\text{E}-01$	$7.19\text{E}-128 \pm 3.94\text{E}-127$	$1.78\text{E}-247 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_2	$9.15\text{E}+00 \pm 1.34\text{E}+01$	$2.83\text{E}+01 \pm 4.82\text{E}-01$	$2.83\text{E}+01 \pm 4.84\text{E}-01$	$2.78\text{E}+01 \pm 7.01\text{E}-01$	$2.88\text{E}+01 \pm 1.77\text{E}-01$
f_3	$3.10\text{E}-02 \pm 4.59\text{E}-02$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$3.55\text{E}-15 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_4	$1.31\text{E}-01 \pm 2.40\text{E}-01$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_5	$5.65\text{E}-02 \pm 1.29\text{E}-01$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_6	$3.83\text{E}+01 \pm 4.87\text{E}+01$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_7	$1.36\text{E}-02 \pm 1.71\text{E}-02$	$5.81\text{E}+02 \pm 6.66\text{E}+01$	$8.42\text{E}+03 \pm 2.00\text{E}+02$	$1.07\text{E}+04 \pm 2.06\text{E}+02$	$9.28\text{E}+03 \pm 3.42\text{E}+02$
f_8	$4.29\text{E}+01 \pm 7.77\text{E}+01$	$4.20\text{E}-127 \pm 2.17\text{E}-126$	$4.60\text{E}-53 \pm 1.03\text{E}-52$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_9	$1.58\text{E}+04 \pm 3.84\text{E}+04$	$6.29\text{E}-112 \pm 3.44\text{E}-111$	$2.24\text{E}-84 \pm 3.10\text{E}-84$	$6.83\text{E}-292 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{10}	$5.12\text{E}-02 \pm 2.00\text{E}-01$	$2.80\text{E}-133 \pm 1.53\text{E}-132$	$1.65\text{E}-247 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{11}	$9.11\text{E}-03 \pm 2.60\text{E}-02$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$2.11\text{E}+01 \pm 2.63\text{E}+01$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{12}	$1.04\text{E}-04 \pm 1.57\text{E}-04$	$1.29\text{E}-68 \pm 6.37\text{E}-68$	$4.04\text{E}-149 \pm 8.89\text{E}-149$	$3.42\text{E}-191 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{13}	$1.83\text{E}-02 \pm 1.51\text{E}-02$	$1.95\text{E}-64 \pm 1.06\text{E}-63$	$1.35\text{E}-72 \pm 2.98\text{E}-72$	$2.94\text{E}-168 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{14}	$7.80\text{E}-02 \pm 2.63\text{E}-02$	$3.01\text{E}-01 \pm 2.66\text{E}-01$	$3.62\text{E}+01 \pm 5.09\text{E}+00$	$7.40\text{E}-01 \pm 4.30\text{E}-02$	$9.68\text{E}-01 \pm 1.30\text{E}-01$
f_{15}	$5.11\text{E}-03 \pm 1.58\text{E}-02$	$1.12\text{E}+00 \pm 1.10\text{E}+00$	$1.72\text{E}+00 \pm 1.91\text{E}-01$	$9.48\text{E}-01 \pm 1.06\text{E}-01$	$2.83\text{E}+00 \pm 1.03\text{E}-01$
$+/\approx/-$	4/0/11	1/5/9	2/3/10	3/8/4	-/-/-

Note: data in bold are the optimal results. -, +, and $\approx$ indicate that the compared algorithms are significantly inferior, superior, and approximately equal to CSCA-CBS, respectively.

表 5 ASCA、IWCCSCA、OBSCA、IM-SCA、CSCA-CBS 的优化结果 (100 维)

Table 5 Optimization results of ASCA, IWCCSCA, OBSCA, IM-SCA, and CSCA-CBS (100 D)

函数 Function	ASCA	IWCCSCA	OBSCA	IM-SCA	CSCA-CBS
f_1	$9.02\text{E}-02 \pm 1.59\text{E}-01$	$2.41\text{E}-127 \pm 1.15\text{E}-126$	$5.94\text{E}-160 \pm 1.25\text{E}-159$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_2	$7.26\text{E}+00 \pm 1.14\text{E}+01$	$9.82\text{E}+01 \pm 2.59\text{E}-01$	$1.39\text{E}+02 \pm 9.03\text{E}+01$	$9.87\text{E}+01 \pm 2.42\text{E}-01$	$9.89\text{E}+01 \pm 4.81\text{E}-02$
f_3	$3.80\text{E}-02 \pm 5.52\text{E}-02$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$3.55\text{E}-15 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_4	$7.53\text{E}-02 \pm 1.48\text{E}-01$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_5	$2.93\text{E}-02 \pm 4.34\text{E}-02$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_6	$2.65\text{E}+02 \pm 23.37\text{E}+02$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_7	$3.56\text{E}-02 \pm 5.24\text{E}-02$	$2.30\text{E}+03 \pm 1.51\text{E}+02$	$3.37\text{E}+04 \pm 5.63\text{E}+02$	$3.92\text{E}+04 \pm 3.87\text{E}+02$	$3.58\text{E}+04 \pm 7.61\text{E}+02$
f_8	$6.46\text{E}+03 \pm 5.07\text{E}+03$	$5.70\text{E}-119 \pm 3.12\text{E}-118$	$4.90\text{E}+03 \pm 6.74\text{E}+03$	$9.71\text{E}-258 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_9	$2.12\text{E}+12 \pm 1.38\text{E}+12$	$9.24\text{E}-41 \pm 4.94\text{E}-40$	$9.09\text{E}-02 \pm 2.00\text{E}-01$	$6.23\text{E}-167 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{10}	$8.20\text{E}-02 \pm 1.41\text{E}-01$	$3.33\text{E}-115 \pm 1.82\text{E}-114$	$1.71\text{E}-161 \pm 3.79\text{E}-161$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{11}	$2.30\text{E}+00 \pm 1.81\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$1.40\text{E}+02 \pm 7.85\text{E}+01$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{12}	$1.89\text{E}-01 \pm 1.74\text{E}-01$	$8.08\text{E}-63 \pm 4.43\text{E}-62$	$5.99\text{E}-94 \pm 1.23\text{E}-93$	$2.10\text{E}-172 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{13}	$2.17\text{E}-02 \pm 1.93\text{E}-02$	$4.64\text{E}-62 \pm 2.52\text{E}-61$	$3.34\text{E}+01 \pm 1.66\text{E}+01$	$1.84\text{E}-116 \pm 9.98\text{E}-116$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{14}	$4.00\text{E}-02 \pm 1.03\text{E}-01$	$1.70\text{E}-01 \pm 2.52\text{E}-01$	$2.13\text{E}+01 \pm 4.52\text{E}+00$	$1.07\text{E}+00 \pm 6.77\text{E}-02$	$1.14\text{E}+00 \pm 6.30\text{E}-02$
f_{15}	$2.55\text{E}-03 \pm 3.17\text{E}-03$	$3.27\text{E}+00 \pm 3.77\text{E}+00$	$7.33\text{E}+01 \pm 6.61\text{E}+01$	$3.97\text{E}+00 \pm 1.78\text{E}-01$	$9.88\text{E}+00 \pm 3.01\text{E}-02$
$+/\approx/-$	4/0/11	0/5/10	0/3/12	0/6/9	-/-/-

Note: data in bold are the optimal results. -, +, and $\approx$ indicate that the compared algorithms are significantly inferior, superior, and approximately equal to CSCA-CBS, respectively.

3.3.3 收敛性能

图 2 和图 3 分别展示了 CSCA-CBS 以及其他几种改进的 SCA 变体基准算法在维度为 30 和 100 时对 9 个基准函数的收敛曲线,以直观地展示算法的收敛速度和全局搜索能力。从图 2、3 可以看出,无论在低维(维度 30)还是高维(维度 100)的情况下,当迭代次数不到 100 次时,本文所提出的 CSCA-CBS 可以找到 9 个函数的全局最优值,而对比算法则需要迭代

到接近 1 000 次才能接近或达到最优值,有的算法甚至会在迭代过程中陷入局部最优。因此,无论函数特征如何,CSCA-CBS 均展现出强大的全局最优搜索能力。综上,相对于 ASCA、IWCCSCA、OBSCA 和 IM-SCA,CSCA-CBS 在计算代价和可靠程度、解的质量分析、收敛性能方面都具有显著优势。

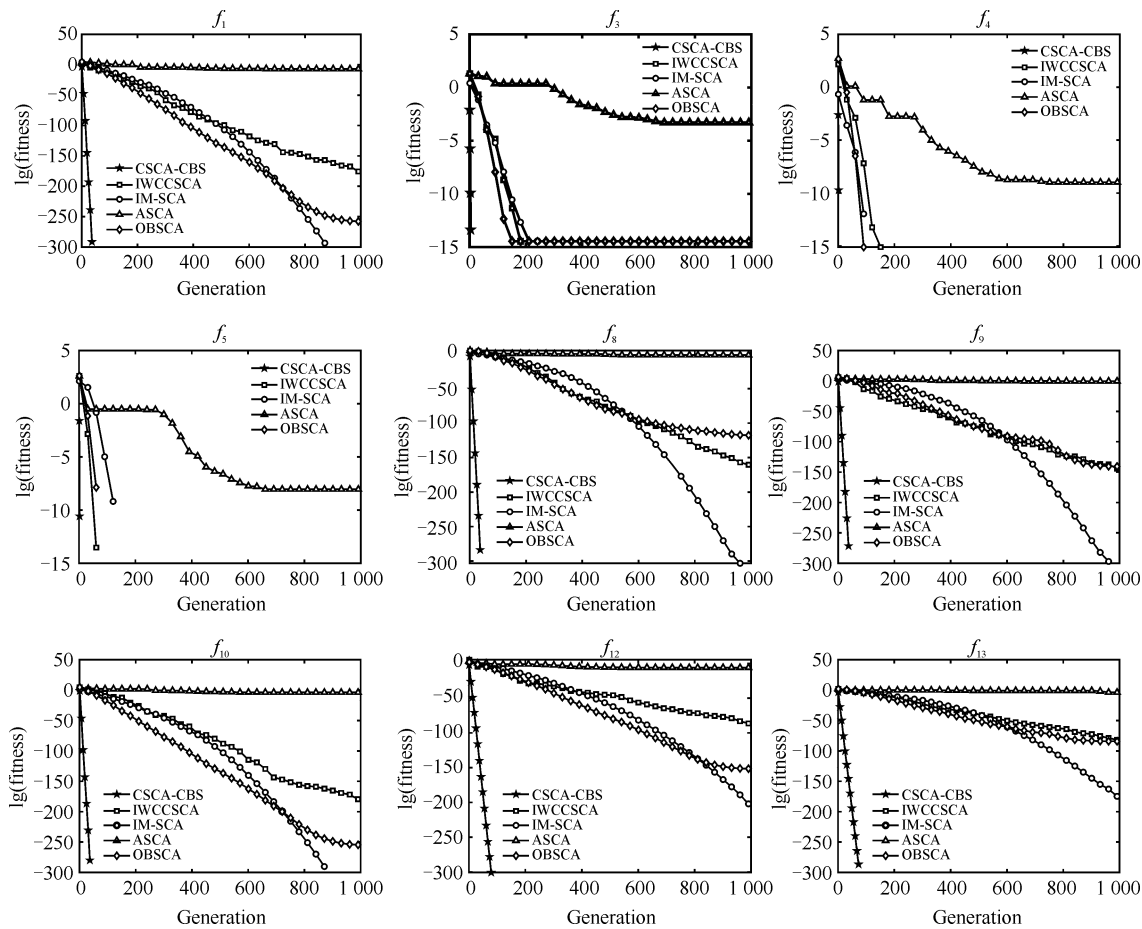


图 2 CSCA-CBS、ASCA、IWCCSCA、OBSCA、IM-SCA 的收敛曲线(30 维)
Fig. 2 Convergence curves of CSCA-CBS, ASCA, IWCCSCA, OBSCA, and IM-SCA (30 D)

3.4 与非 SCA 比较

为了进一步测试 CSCA-CBS 在其他非 SCA 中的显著优势,选用了近 3 年内开发的 4 种优秀算法进行比较,包括 IAGA^[30]、SOGWO^[37]、ICS-FG^[38]、EF-

PA-G^[39]。表 6 列出 IAGA、SOGWO、ICS-FG、EF-PA-G 和 CSCA-CBS 在 30 维下独立运行 30 次后获得的平均值和标准差,其中算法参数设置来源于其对应文献。

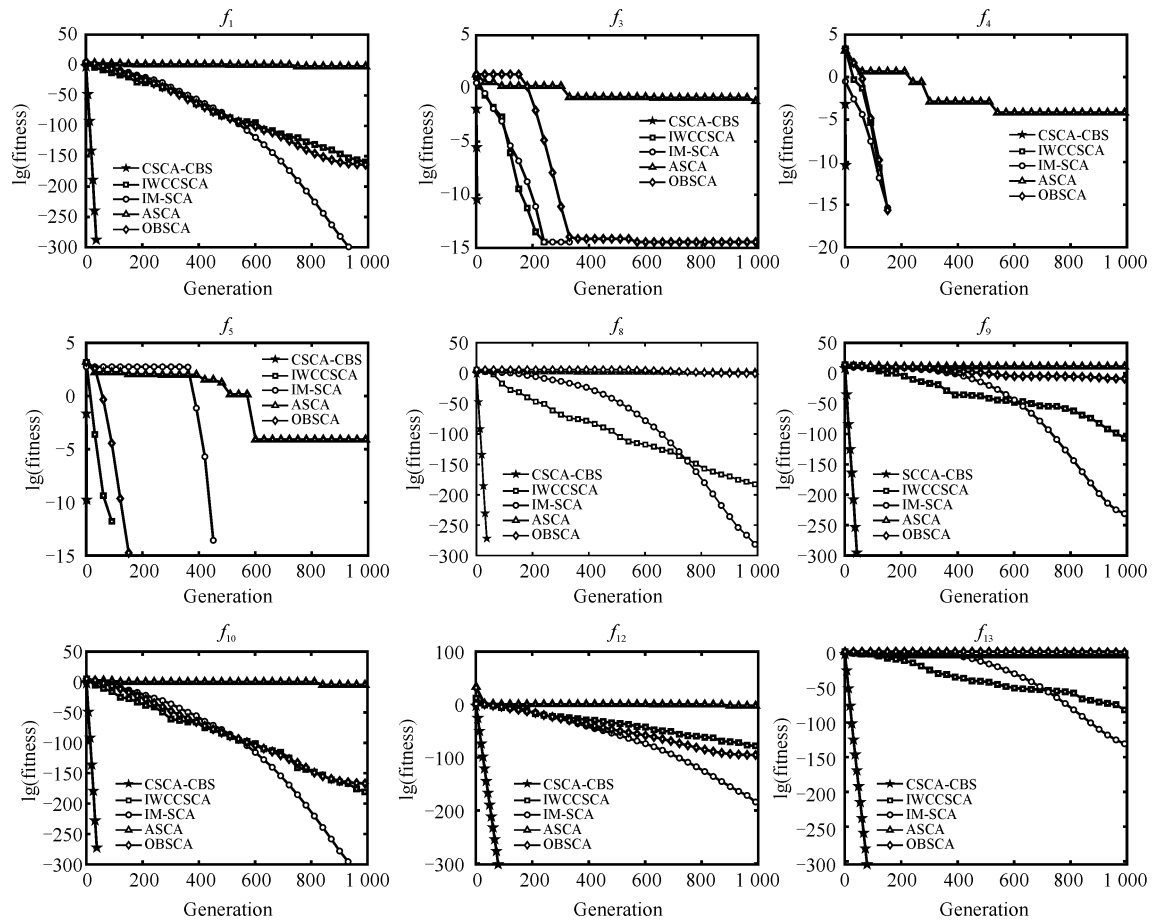


图 3 CSCA-CBS, ASCA, IWCCSCA, OBSCA, IM-SCA 的收敛曲线(100 维)

Fig. 3 Convergence curves of CSCA-CBS, ASCA, IWCCSCA, OBSCA, and IM-SCA (100 D)

表 6 IAGA, SOGWO, ICS-FG, EFPA-G, CSCA-CBS 的优化结果(30 维)

Table 6 Optimization results of IAGA, SOGWO, ICS-FG, EFPA-G, and CSCA-CBS (30 D)

函数 Function	IAGA	SOGWO	ICS-FG	EFPA-G	CSCA-CBS
f_1	$1.54\text{E}-32 \pm 6.44\text{E}-32$	$4.92\text{E}-85 \pm 1.79\text{E}-84$	$2.02\text{E}-88 \pm 1.10\text{E}-87$	$2.82\text{E}-269 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_2	$2.44\text{E}+01 \pm 9.75\text{E}+00$	$2.61\text{E}+01 \pm 6.00\text{E}-01$	$2.83\text{E}+01 \pm 3.80\text{E}-01$	$2.87\text{E}+01 \pm 6.90\text{E}-02$	$2.88\text{E}+01 \pm 1.77\text{E}-01$
f_3	$2.96\text{E}-15 \pm 1.35\text{E}-15$	$9.59\text{E}-15 \pm 2.66\text{E}-15$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_4	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$1.01\text{E}-03 \pm 4.07\text{E}-03$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_5	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$1.78\text{E}-01 \pm 9.74\text{E}-01$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_6	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$2.50\text{E}+00 \pm 4.57\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_7	$8.94\text{E}-02 \pm 4.22\text{E}-02$	$6.10\text{E}+03 \pm 7.61\text{E}+02$	$4.47\text{E}+03 \pm 1.37\text{E}+02$	$6.91\text{E}+03 \pm 6.56\text{E}+02$	$9.28\text{E}+03 \pm 3.42\text{E}+02$
f_8	$3.51\text{E}-35 \pm 8.14\text{E}-35$	$3.21\text{E}-27 \pm 1.30\text{E}-26$	$8.51\text{E}-08 \pm 4.66\text{E}-07$	$1.85\text{E}-266 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_9	$4.51\text{E}-20 \pm 9.39\text{E}-20$	$3.98\text{E}+02 \pm 2.12\text{E}+03$	$2.34\text{E}-61 \pm 1.06\text{E}-60$	$8.17\text{E}-225 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{10}	$1.08\text{E}-32 \pm 2.19\text{E}-32$	$6.82\text{E}-85 \pm 3.21\text{E}-84$	$7.26\text{E}-100 \pm 3.98\text{E}-99$	$3.67\text{E}-267 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{11}	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{12}	$2.14\text{E}-22 \pm 3.57\text{E}-22$	$2.93\text{E}-49 \pm 2.65\text{E}-49$	$2.32\text{E}-34 \pm 1.25\text{E}-33$	$6.22\text{E}-135 \pm 2.86\text{E}-134$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{13}	$4.83\text{E}-13 \pm 5.86\text{E}-13$	$2.36\text{E}-21 \pm 6.80\text{E}-21$	$8.87\text{E}-35 \pm 4.86\text{E}-34$	$3.98\text{E}-135 \pm 8.68\text{E}-135$	$0.00\text{E}+00 \pm 0.00\text{E}+00$
f_{14}	$4.67\text{E}-01 \pm 6.36\text{E}-01$	$1.39\text{E}+00 \pm 4.50\text{E}+00$	$3.62\text{E}-01 \pm 1.29\text{E}-01$	$1.22\text{E}-01 \pm 5.20\text{E}-02$	$9.68\text{E}-01 \pm 1.30\text{E}-01$
f_{15}	$5.74\text{E}-01 \pm 8.26\text{E}-01$	$1.72\text{E}-01 \pm 1.18\text{E}-01$	$2.13\text{E}+00 \pm 8.53\text{E}-02$	$2.21\text{E}-02 \pm 1.76\text{E}-02$	$2.83\text{E}+00 \pm 1.03\text{E}-01$
+ / $\approx$ / -	4 / 4 / 7	3 / 1 / 11	3 / 5 / 7	4 / 5 / 6	- / - / -

Note: data in bold are the optimal results. -, +, and $\approx$ indicate that the compared algorithms are significantly inferior, superior, and approximately equal to CSCA-CBS, respectively.

由表 6 可知,相比于 IAGA,CSCA-CBS 对函数 f_2 、 f_7 、 f_{14} 、 f_{15} 的优化结果较差,而对于 f_4 、 f_5 、 f_6 、 f_{11} 这 4 个函数均能够搜索到全局最优解;CSCA-CBS 对函数 f_2 、 f_7 、 f_{15} 的优化结果同样差于 SOGWO、ICS-FG 和 EFPA-G,但所有算法对 f_{11} 有着一样的优化效果;对于 ICS-FG 和 EFPA-G 来说,对函数 f_3 、 f_4 、 f_5 、 f_6 、 f_{11} 的优化效果与 CSCA-CBS 相近。综上,CSCA-CBS 在数值优化结果、稳定性等方面都比 IAGA、SOGWO、ICS-FG、EFPA-G 优异。

3.5 非参数检验

为了进一步证明本文所提出的 CSCA-CBS 的显著性能,本文采用两种非参数统计检验方法(包括 Friedman 检验和 Kruskal-Wallis 检验)来对上述所有比较算法进行分析和排名。

表 7 列出了对所有基准测试函数在 30 维下的求解性能排名统计情况。每个算法及其排名均按升序列出(得分越低越好)。如表 7 所示,10 种算法在两次非参数统计检验中的排名几乎一致,本文所提算法 CSCA-CBS 在优化问题中的综合实力为第一名,其中 OBSCA 和 ICS-FG 的排名略有不同。Friedman 检验和 Kruskal-Wallis 检验方法表明,与其他 9 种先进算法相比,CSCA-CBS 明显更好,相对原始 SCA 来说有着显著的性能改进。

表 7 所有对比算法的平均排名

Table 7 Average ranking of all comparison algorithms

排名 Rank	Friedman		Kruskal-Wallis	
	算法 Algorithm	得分 Score	算法 Algorithm	得分 Score
1	CSCA-CBS	2.60	CSCA-CBS	54.33
2	IM-SCA	2.73	IM-SCA	62.17
3	EFPA-G	2.80	EFPA-G	66.21
4	IWCCSA	3.27	IWCCSA	77.02
5	ICS-FG	4.33	OBSCA	84.57
6	OBSCA	4.47	ICS-FG	85.02
7	IAGA	4.53	IAGA	90.75
8	SOGWO	6.40	SOGWO	109.99
9	ASCA	8.20	ASCA	130.54
10	SCA	9.27	SCA	150.16

4 土壤水分特征曲线参数反演问题

为了验证本文所提出的 CSCA-CBS 在实际应用上的实用性和有效性,将其应用于土壤水分特征曲线

的参数反演问题上。

4.1 问题描述

土壤水分特征曲线是获得土壤学传导率的常用手段,也是定量分析土壤水分含量并用于预测的重要理论依据。Van Genuchten (VG)方程是目前来说使用最为广泛的模型,具体表达式如下:

$$\theta(h) = \frac{\theta_s - \theta_r}{[1 + |\alpha h|^N]^M} + \theta_r, \tag{16}$$

式中, $\theta(h)$ 是不同土壤吸水力 h 下的土壤含水率; θ_s 是土壤饱和含水率; θ_r 是土壤残余含水率; α 、 N 、 M 为土壤水分特征曲线的形状参数, $M = 1 - 1/N$, $N > 1$ 。由式(16)可知, $[\theta_s, \theta_r, \alpha, N]$ 是需要根据实测数据待反演的 4 个参数。

本文使用绝对值误差来构成目标函数,反演的最优参数如公式(17)所示:

$$[\theta_s, \theta_r, \alpha, N] = \operatorname{argmin} \left\{ \sum_{i=1}^m |\theta(h) - \theta_i| \right\}, \tag{17}$$

其中, θ_i 为实测土壤含水率, m 为实测数据组数。

4.2 数值实验

在 VG 方程参数反演的数值实验中,土壤数据来源于对西北某干旱地区的实测采集^[40]。本节主要分为两个部分:①比较不同方法对粉壤土的求解结果,主要将 SCA、CSCA-CBS 与非线性单纯形法(SM)、阻尼最小二乘法(DLSM)、遗传算法(GA)、混合遗传算法(HGA)、随机粒子群算法(PSO)、混合粒子群算法(HPSO)的求解结果进行比较;②比较不同改进方法对 3 种类型土壤 VG 方程的求解精度和优化性能,主要包括 HPSO^[41]、改进 PSO (IPSO)^[42] 以及 SCA,并将反演结果与实测值进行比较。VG 方程参数的取值范围是 $\theta_s \in [0.2, 0.7]$, $\theta_r \in [0.0, 0.2]$, $\alpha \in [0, 1]$, $n \in [1, 10]$ 。种群规模 $N = 200$, 最大迭代次数为 1 000。整个数值实验均独立运行 40 次。

表 8 列出了使用 SM、DLSM、GA、HGA、SPSO、HPSO、SCA 和 CSCA-CBS 对粉壤土 VG 方程进行参数反演的结果,其中 SM、DLSM、GA、HGA、SPSO、HPSO 的求解数据来源于文献^[41]。由表 8 可知,GA 和 HPSO 的反演结果较为相似,其函数值误差仅相差 0.000 1,于是可将粉壤土 VG 方程的参数 θ_r 和 θ_s 分别锁定为 0.057 和 0.363;其次,将 CSCA-CBS 与 GA 进行比较,可进一步将参数 α 锁定为 0.014,并且 CSCA-CBS 的最优函数误差值达到最小,证明了其在粉壤土中的应用效果。

表 8 不同方法对土壤含水率方程的参数反演结果
Table 8 Parameter inversion results of different methods for soil water content equation

算法 Algorithm	θ_r	θ_s	α	N	函数值 Function value
SM	0.053	0.363	0.013	4.241	0.108 2
DLSM	0.053	0.363	0.014	4.197	0.147 3
GA	0.057	0.363	0.014	2.945	0.063 8
HGA	0.066	0.358	0.014	4.073	0.077 9
SPSO	0.066	0.359	0.013	4.044	0.077 4
HPSO	0.057	0.363	0.013	3.167	0.063 9
SCA	0.061	0.364	0.012	3.267	0.065 1
CSCA-CBS	0.057	0.363	0.014	2.949	0.061 7

Note;data in bold are the optimal results.

表 9 给出了 CSCA-CBS 与 SCA、IPSO、HPSO

表 9 不同方法在 3 类土壤下的参数反演结果

Table 9 Parameter inversion results of different methods for three soil types

土壤类型 Soil type	算法 Algorithm	θ_r	θ_s	α	n	函数值 Function value
Dehumidification of silty loam	HPSO	0.057	0.362	0.014	2.983	0.062 5
	IPSO	0.064	0.360	0.013	3.323	0.069 1
	SCA	0.058	0.365	0.014	3.176	0.065 1
	CSCA-CBS	0.057	0.363	0.014	2.949	0.061 7
Hygroscopic of silt loam	HPSO	0.061	0.347	0.013	5.779	0.058 2
	IPSO	0.059	0.354	0.013	3.625	0.061 6
	SCA	0.000	0.368	0.015	2.111	0.061 2
	CSCA-CBS	0.061	0.361	0.013	5.501	0.057 8
Dehumidification of fine sand	HPSO	0.083	0.377	0.015	4.442	0.039 4
	IPSO	0.050	0.700	0.330	1.542	0.076 7
	SCA	0.000	0.700	0.550	1.349	0.039 5
	CSCA-CBS	0.083	0.377	0.01	4.622	0.036 9
Hygroscopic of fine sand	HPSO	0.081	0.374	0.015	5.012	0.025 9
	IPSO	0.059	0.700	0.271	1.648	0.035 2
	SCA	0.083	0.371	0.015	5.170	0.033 6
	CSCA-CBS	0.081	0.374	0.015	5.013	0.025 4
Dehumidification of gravel	HPSO	0.060	0.275	0.024	2.801	0.016 7
	IPSO	0.048	0.700	0.417	1.707	0.025 2
	SCA	0.058	0.273	0.023	3.033	0.017 7
	CSCA-CBS	0.061	0.273	0.023	3.223	0.015 6
Hygroscopic of gravel	HPSO	0.045	0.589	0.322	1.718	0.020 9
	IPSO	0.043	0.700	0.465	1.674	0.063 4
	SCA	0.056	0.268	0.022	3.593	0.022 1
	CSCA-CBS	0.056	0.269	0.022	3.630	0.020 3

Note;data in bold are the optimal results.

在 3 类土壤(粉壤土、细砂和砾石)下的参数反演结果,其中每种土壤具有脱湿和细湿两种情况。从计算的最优函数值来看,CSCA-CBS 无论在何种土壤下都能获得最优的函数值。由图 4 可以看出,CSCA-CBS 在迭代搜索过程中均表现出比其他方法更好的优化性能,特别是对于细砂和砾石这两种土壤。另外,CSCA-CBS 也能够不同土壤环境下计算出与实际测量值十分吻合的反演参数,如图 5 所示。综上所述,本文提出的 CSCA-CBS 可用于求解 VG 方程的参数反演问题,并能提供一种比较可靠的解决方案。

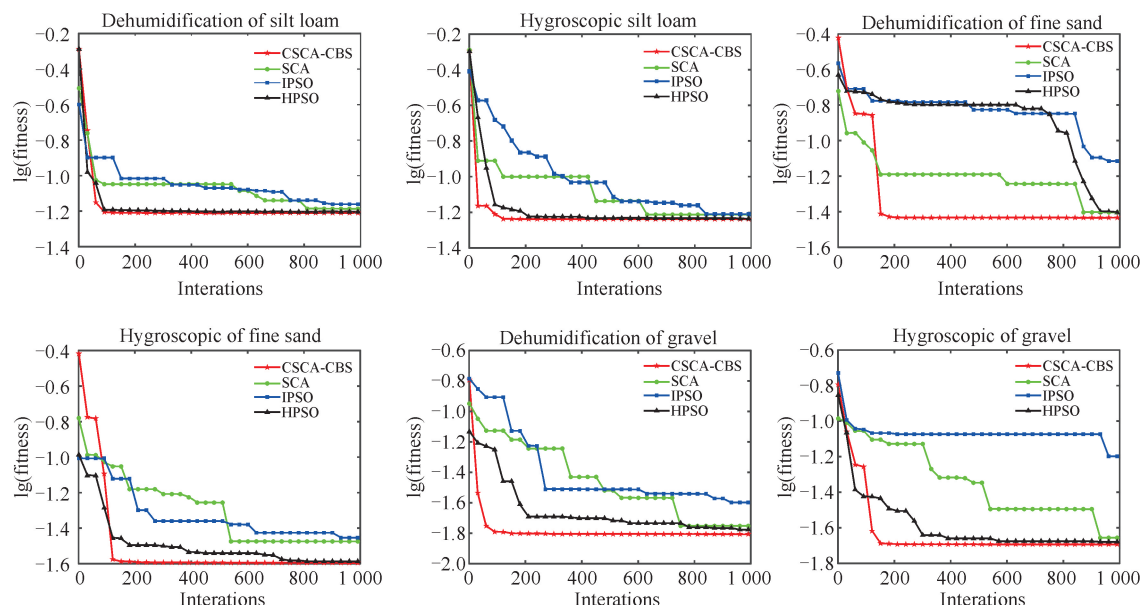


图4 VG方程参数反演平均迭代曲线

Fig. 4 Average iteration curves for parametric inversion of the VG equations

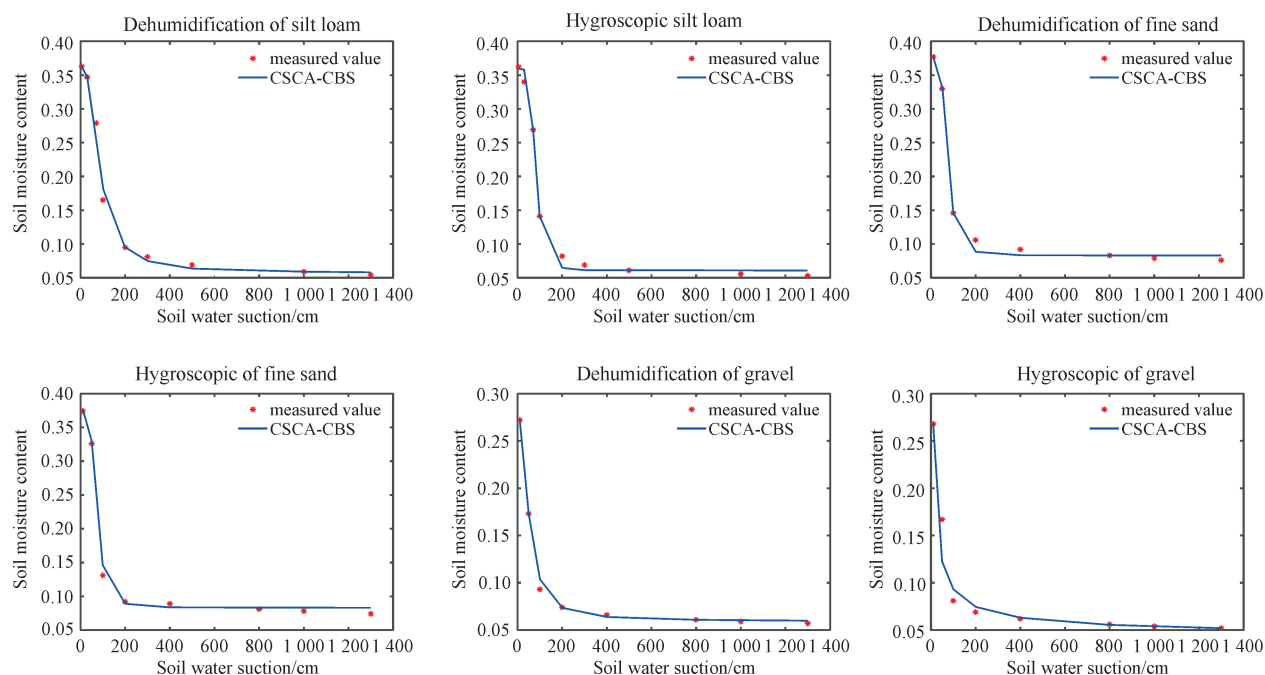


图5 CSCA-CBS对不同土壤水分特征曲线的反演结果

Fig. 5 Inversion results of CSCA-CBS for different soil moisture characteristic curves

5 结论

本文提出了一种融合疯狂秃鹰搜索算法的混沌正余弦算法 CSCA-CBS, 主要从 3 个方面进行改进: ①引入一种混合的混沌映射策略, 以提升初始种群的多样性; ②采用疯狂秃鹰搜索策略, 以增强原始 SCA 的全局探索能力; ③提出逐维反向变异策略, 以帮助种群跳出局部最优。为检验 CSCA-CBS 的可靠性, 本文在不同维度和不同测试函数下进行了广泛的对

比实验。结果表明, 与先进的 SCA 变体以及非 SCA 相比, CSCA-CBS 在收敛速度和精度上均优于对比算法。随后, 利用 CSCA-CBS 求解土壤水分特征曲线参数的结果表明, CSCA-CBS 具有更快的计算速度和更高的求解精度, 极大地提高了参数优化的普适性, 这为解决 VG 方程问题提供了有效方法。

参考文献

- [1] 郑立辉, 郭亚军, 潘德惠. 一种基于梯度搜索的全局优化

- 新算法[J]. 控制理论与应用, 1997, 14(3): 343-348.
- [2] WU M, OU X F, LU Y L, et al. Heterogeneous Cuckoo search-based unsupervised band selection for hyperspectral image classification [J]. IEEE Transactions on Geoscience and Remote Sensing, 2023, 62: 1-16.
- [3] MEHEDI I M, AHMADIPOUR M, SALAM Z, et al. Optimal feature selection using modified cuckoo search for classification of power quality disturbances [J]. Applied Soft Computing, 2021, 113: 107897.
- [4] XIONG Y B, WU J, WANG Q, et al. SAPNN: self-adaptive probabilistic neural network for medical diagnosis [J]. International Journal of Computational Science and Engineering, 2024, 27(1): 68-77.
- [5] 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.
- [6] MIRJALILI S. SCA: a sine cosine algorithm for solving optimization problems [J]. Knowledge-Based Systems, 2016, 96: 120-133.
- [7] LIU J S, ZHAO F Y, LI Y, et al. A new global sine cosine algorithm for solving economic emission dispatch problem [J]. Information Sciences, 2023, 648: 119569.
- [8] AKAY R, YILDIRIM M Y. Multi-strategy and self-adaptive differential sine-cosine algorithm for multi-robot path planning [J]. Expert Systems with Applications, 2023, 232: 120849.
- [9] 文武, 万玉辉, 文志云. 基于正余弦算法的文本特征选择 [J]. 计算机工程与科学, 2022, 44(8): 1467-1473.
- [10] 蒋杰伟, 金库, 刘尚辉, 等. 基于正余弦扰动的海洋捕食者算法优化 RFA 设计 [J]. 光电子 · 激光, 2023, 34(7): 673-681.
- [11] RIZK-ALLAH R M, HASSANIEN A E. A comprehensive survey on the sine-cosine optimization algorithm [J]. Artificial Intelligence Review, 2023, 56(6): 4801-4858.
- [12] 王庆, 李玉琛, 蒙飞, 等. 混沌正余弦算法在含风能电力系统经济排放调度中的应用 [J]. 电力系统保护与控制, 2022, 50(24): 172-177.
- [13] JIAO S B, WANG C, GAO R, et al. A novel hybrid Harris hawk sine cosine optimization algorithm for reactive power optimization problem [J]. Journal of Experimental & Theoretical Artificial Intelligence, 2024, 36(6): 901-937.
- [14] FENG Z K, DUAN J F, NIU W J, et al. Enhanced sine cosine algorithm using opposition learning, adaptive evolution and neighborhood search strategies for multi-variable parameter optimization problems [J]. Applied Soft Computing, 2022, 119: 108562.
- [15] 王华秋, 熊维双. 采用聚类的变异正余弦优化算法研究 [J]. 重庆理工大学学报(自然科学), 2022, 36(11): 292-302.
- [16] CHEN H, HEIDARI A A, ZHAO X H, et al. Advanced orthogonal learning-driven multi-swarm sine cosine optimization: framework and case studies [J]. Expert Systems with Applications, 2020, 144: 113113.
- [17] SINDHU R, NGADIRAN R, YACOB Y M, et al. Retraction note: sine-cosine algorithm for feature selection with elitism strategy and new updating mechanism [J]. Neural Computing and Applications, 2024, 36(18): 11019.
- [18] RAUT U, MISHRA S. An improved sine-cosine algorithm for simultaneous network reconfiguration and DG allocation in power distribution systems [J]. Applied Soft Computing, 2020, 92: 106293.
- [19] LIU J S, ZHAO F Y, LI Y, et al. A new global sine cosine algorithm for solving economic emission dispatch problem [J]. Information Sciences, 2023, 648: 119569.
- [20] ISSA M, HASSANIEN A E, OLIVA D, et al. ASCA-PSO: adaptive sine cosine optimization algorithm integrated with particle swarm for pairwise local sequence alignment [J]. Expert Systems with Applications, 2018, 99: 56-70.
- [21] GUPTA S, SU R, SINGH S. Diversified sine-cosine algorithm based on differential evolution for multidimensional knapsack problem [J]. Applied Soft Computing, 2022, 130: 109682.
- [22] ANSARI S A, ZAFAR A. A fusion of dolphin swarm optimization and improved sine cosine algorithm for automatic detection and classification of objects from surveillance videos [J]. Measurement, 2022, 192: 110921.
- [23] PHATAK S C, RAO S S. Logistic map: a possible random-number generator [J]. Physical Review E, 1995, 51(4): 3670-3678.
- [24] 李大海, 刘庆腾, 王振东. 融合 Tent 混沌和维度学习的阴阳对算法 [J]. 计算机工程与设计, 2023, 44(8): 2320-2328.
- [25] HERBADJI D, HERBADJI A, HADDAD I, et al. An enhanced logistic chaotic map based tweakable speech encryption algorithm [J]. Integration, 2024, 97: 102192.
- [26] ZHANG Q Z, LIU H S, GUO J, et al. Improved GW-OMCSVM algorithm based on nonlinear convergence factor and tent chaotic mapping and its application in transformer condition assessment [J]. Electric Power

- Systems Research, 2023, 224: 109754.
- [27] 高毅. 基于 Logistic 混沌算法的伪随机序列 IP 设计 [D]. 哈尔滨: 黑龙江大学, 2021.
- [28] ALSATTAR H A, ZAIDAN A A, ZAIDAN B B. Novel meta-heuristic bald eagle search optimisation algorithm [J]. Artificial Intelligence Review, 2020, 53(3): 2237-2264.
- [29] KALASHNIKOV V, KREINOVICH V, FLORESMUNIZ J G, et al. Structure of filled functions: why Gaussian and Cauchy templates are most efficient [J]. International Journal of Combinatorial Optimization Problems and Informatics, 2016, 7(3): 87-93.
- [30] OUYANG A J, LU Y S, LIU Y M, et al. An improved adaptive genetic algorithm based on DV-Hop for locating nodes in wireless sensor networks [J]. Neurocomputing, 2021, 458: 500-510.
- [31] ZHAN Z H, ZHANG J, LI Y, et al. Orthogonal learning particle swarm optimization [J]. IEEE Transactions on Evolutionary Computation, 2011, 15(6): 832-847.
- [32] ZENG N Y, WANG Z D, LIU W B, et al. A dynamic neighborhood-based switching particle swarm optimization algorithm [J]. IEEE transactions on cybernetics, 2020, 52(9): 9290-9301.
- [33] JI Y T, TU J Z, ZHOU H F, et al. An adaptive chaotic sine cosine algorithm for constrained and unconstrained optimization [J]. Complexity, 2020, 2020: 6084917.
- [34] 陈亮, 汤显峰. 改进正余弦算法优化特征选择及数据分类[J]. 计算机应用, 2022, 42(6): 1852-1861.
- [35] ABD ELAZIZ M, OLIVA D, XIONG S. An improved opposition-based sine cosine algorithm for global optimization [J]. Expert Systems with Applications, 2017, 90: 484-500.
- [36] 刘丽娟, 刘定一, 刘婷婷. 改进的正余弦优化算法在 WSN 覆盖中的应用[J]. 数学的实践与认识, 2021, 51(11): 129-137.
- [37] DHARGUPTA S, GHOSH M, MIRJALILI S, et al. Selective opposition based grey wolf optimization [J]. Expert Systems with Applications, 2020, 151: 113389.
- [38] OU X F, WU M, PU Y Y, et al. Cuckoo search algorithm with fuzzy logic and Gauss-Cauchy for minimizing localization error of WSN [J]. Applied Soft Computing, 2022, 125: 109211.
- [39] ZHENG J, YUAN T, XIE W W, et al. An enhanced flower pollination algorithm with Gaussian perturbation for node location of a WSN [J]. Sensors, 2023, 23(14): 6463.
- [40] 王金生, 杨志峰, 陈家军, 等. 包气带土壤水分滞留特征研究[J]. 水利学报, 2000, 31(2): 1-6.
- [41] 刘利斌, 欧阳艾嘉, 乐光学, 等. 基于混合粒子群的土壤水分特征曲线参数优化[J]. 计算机工程与应用, 2011, 47(35): 218-221.
- [42] 孙晓敏, 孙秋红. 基于改进的基本粒子群算法的 Van Genuchten 方程参数优化[J]. 水利与建筑工程学报, 2017, 15(5): 88-93.

A Chaotic Sine Cosine Algorithm Based on Crazy Bald - Eagle Search

ZHOU Tian, MAI Xiongfa, LIU Libin^{**}, ZHENG Guilin

(Guangxi Applied Mathematics Center, Nanning Normal University, Nanning, Guangxi, 530100, China)

Abstract: Aiming at the slow convergence and low computational accuracy of the Sine Cosine Algorithm (SCA) in optimization problems, a Chaotic Sine Cosine Algorithm based on Crazy Bald-eagle Search (CSCA-CBS) is proposed. CSCA-CBS employs Logistic-Tent chaotic mapping for population initialization, thereby obtaining a more uniform and diverse initial population. Inspired by the bald-eagle search algorithm, CSCA-CBS adopts a bald-eagle search strategy with a crazy operator to enhance its global exploration capability. To avoid falling into local optima during later iterations, CSCA-CBS utilizes a dimension-by-dimension reverse

Cauchy mutation strategy to systematically perturb the population, effectively integrating the advantages of opposition-based learning and Cauchy mutation. Simulation experiments with 15 benchmark functions show that the CSCA-CBS can outperform state-of-the-art SCA variants and four non-SCA in terms of computational cost and reliability, solution quality analysis, and convergence performance. Additionally, parameter inversion experiments on soil moisture characteristic curves further validate the excellent practicality and effectiveness of CSCA-CBS.

Key words: sine cosine algorithm; Logistic-Tent chaotic mapping; bald-eagle search algorithm; crazy operator; parameter inversion

责任编辑: 陆 雁



微信公众号投稿更便捷

联系电话: 0771-2503923

邮箱: gxkx@gxas.cn

投稿系统网址: <http://gxkx.ijournal.cn/gxkx/ch>