DOI:10.19651/j.cnki.emt.2518688

改进金豺算法的多目标约束问题研究

徐静萍 王文杰 张 鑫
(西安邮电大学电子工程学院 西安 710121)

摘 要:针对金豺优化算法在求解有约束优化问题时面临的种群质量差、收敛速度慢和易陷入局部极值等问题,提出一种基于多策略的金豺优化算法。首先,为了增加群体的多样性和改善初始解的品质,使用了混沌精英初始化策略来产生精英群体;然后引入能量调节机制,对全局搜索和局部优化进行协调;最后,针对群体中的个体差异,设计了一种融合突变的方法以防止出现局部极值问题。通过标准测试函数的比较试验,证明了改进后的算法具有较好的收敛性能和较快的收敛速度。此外,在 CEC2021 测试函数和压力容器设计优化问题上进行实验,通过收敛性分析、鲁棒性检测和 Wilcoxon 秩和统计的验证进一步证明了改进的金豺优化算法在单目标约束和多目标约束问题中的有效性和实用性。

关键词:金豺优化算法;混沌精英初始化策略;能量调控机制;融合突变;压力容器设计
中图分类号:TP301.6;TN02 **文献标识码**:A **国家标准学科分类代码**:510

Improved golden jackal algorithm for multi-objective constrained problems

Xu Jingping Wang Wenjie Zhang Xin
(School of Electronic Engineering, Xi′an University of Posts and Telecommunications, Xi′an 710121, China)

Abstract: A multi-strategy based golden jackal optimization algorithm is proposed to address the problems of poor population quality, slow convergence speed and easy to fall into local extremes faced by the golden jackal optimization algorithm in solving constrained optimization problems. First, in order to increase the diversity of the population and improve the quality of the initial solution, a chaotic elite collaborative initialization strategy is used to generate an elite population; then, an energy regulation mechanism is introduced to coordinate the global search and local optimization; finally, a fusion mutation method is designed for the individual differences in the population in order to prevent the problem of local extremes. The improved algorithm is proved to have better convergence performance and faster convergence speed through the comparison test of standard test functions. In addition, experiments on the CEC2021 test function and the pressure vessel design optimization problem further demonstrate the effectiveness and practicality of the improved golden jackal optimization algorithm in single-objective constraints and multi-objective constraints problems through convergence analysis, robustness test, and validation of Wilcoxon′s rank sum statistics.

Keywords: gold jackal optimization algorithm; chaotic elite initialization strategy; energy regulation mechanism; fusion mutation; pressure vessel design

0 引 言

在工程领域,优化问题广泛存在于各种复杂系统中,如工程设计、机器学习、资源分配等。传统的优化方法,如梯度下降法、线性规划等,在处理高维、非线性、多目标等复杂优化问题时,往往面临收敛速度慢、易陷入局部最优等问题。研究者们逐渐转向元启发式优化算法,如粒子群优化算法(particle swarm optimization,PSO)[1];灰狼优化算法(grey wolf optimizer,GWO)[2];鲸鱼优化算法(whale optimization algorithm,WOA)[3]。这些优化算法通过模拟自然界的生物行为或物理现象,能够在全局搜索和局部优化之间取得良好的平衡。

金豺优化算法(golden jackal optimization,GJO)[4]是一种模拟金豺协作捕猎行为的种群智能优化算法,因其简单易实现而在解决复杂问题时具有广阔的应用前景,如癌症生物标志物分类[5]和配电网优化[6]等。尽管 GJO 已被广泛应用于多个领域,但其仍存在易陷入局部最优、种群多样性差和收敛速度慢等问题。Wang 等[7]采用类似克隆的

收稿日期:2025-04-25

策略进行种群更新,并在选择阶段采用模拟退火方法,但由于过度依赖于最优解,算法容易陷入局部极值,针对这一问题,本文提出自适应 t-差分融合变异策略利用种群个体差异进行位置更新,避免过度依赖最优解。Mohapatra 等[8]提出基于对立面学习增强金豺优化算法(opposition-based learning golden jackal optimization,OGJO),通过基于对立面学习策略帮助算法摆脱局部最优值,但 OBL 过度干扰局部搜索,可能导致探索与开发的失衡,为此本文使用一种分阶段混合逃逸能量调控机制来协调算法的全局探索与局部开发过程。Wang 等[9]提出一种基于多策略混合的金豺算法(multi-strategy hybrid based golden jackal algorithm,LGJO),首先使用混沌映射策略来初始化种群生成初始解,其次提出一种基于余弦变化的动态惯量权重以及基于高斯突变的位置更新策略,利用最优个体来提高种群多样性,然而单一混沌映射容易造成重复个体生成风险和维度适应性不足,因此本文利用 Logistic 映射对群体进行初始化后,再通过纵横交叉机制与透镜成像逆向学习策略结合形成混合精英策略优化种群多样性与质量。Yang 等[10]提出基于 Rosenbrock 直接旋转策略的改进的金豺优化算法,使算法的局部搜索能力显著提升,但在解决高维度复杂问题时,算法个体适应度值低,易达到局部极值,本文一方面通过在搜索阶段通过调控机制利用柯西分布进行扰动以及自适应 t 分布扰动以此来跳出局部最优;另一方面通过混沌精英初始化策略提高种群质量与多样性,避免提前陷入局部最优。

通过对已有研究中还存在的问题针对性改进,本文提出了一种基于多策略改进的金豺优化算法(multi-strategy improved golden jackal optimization,MSIGJO)。

1 金豺优化算法

1.1 初始化种群

金豺在捕猎时,第一步就是寻找猎物。在初始化时,将在搜索空间中随机产生候选解,使其分布均匀:

$$\boldsymbol{Y}_0 = Y_{\min} + rand(Y_{\max} - Y_{\min}) \tag{1}$$

式中:$\boldsymbol{Y}_0$ 为猎物初始种群的位置;$rand$ 是[0,1]的随机数;$Y_{\max}$ 是解的上界,$Y_{\min}$ 为下界。

首先构建初始矩阵,并将猎物解作为初始解存储其中:

$$\boldsymbol{P}_{rey} = \begin{bmatrix} Y_{1,1} & Y_{1,2} & \cdots & Y_{1,d} \\ Y_{2,1} & Y_{2,2} & \cdots & Y_{2,d} \\ \vdots & \vdots & & \vdots \\ Y_{n,1} & Y_{n,2} & \cdots & Y_{n,d} \end{bmatrix} \tag{2}$$

式中:n 表示猎物种群规模;d 表示问题纬度。

计算每一个目标函数值并存入矩阵 $\boldsymbol{F}_{OA}$:

$$\boldsymbol{F}_{OA} = \begin{bmatrix} f(Y_{1,1};Y_{1,2};\cdots;Y_{1,d}) \\ f(Y_{2,1};Y_{2,2};\cdots;Y_{2,d}) \\ \vdots \\ f(Y_{n,1};Y_{n,2};\cdots;Y_{n,d}) \end{bmatrix} \tag{3}$$

式中:$\boldsymbol{F}_{OA}$ 表示目标函数值矩阵;f 表示目标函数。将最优目标函数值记为雄性金豺 $Y_M(t)$,第二优的目标函数值记为雌性金豺 $Y_{FM}(t)$,将它们组成新的金豺对进行追踪或抓捕阶段求解。

1.2 追踪阶段

金豺在其领地内搜索猎物,并逐步向猎物靠近。金豺与猎物之间的相对位置计算方法如式(4)、(5)所示。

$$\boldsymbol{Y}_1(t) = \boldsymbol{Y}_M(t) - E \cdot |\boldsymbol{Y}_M(t) - \boldsymbol{rl} \times \boldsymbol{Prey}(t)| \tag{4}$$

$$\boldsymbol{Y}_2(t) = \boldsymbol{Y}_{FM}(t) - E \cdot |\boldsymbol{Y}_{FM}(t) - \boldsymbol{rl} \times \boldsymbol{Prey}(t)| \tag{5}$$

式中:$\boldsymbol{Prey}(t)$ 表示猎物位置向量;$\boldsymbol{rl}$ 表示基于 Levy 分布的任意向量;t 是迭代数;$\boldsymbol{Y}_M(t)$ 是雄金豺的初始位置分布,$\boldsymbol{Y}_{FM}(t)$ 是雌金豺初始位置分布;$\boldsymbol{Y}_1(t)$、$\boldsymbol{Y}_2(t)$ 分别是位置更新后的雄雌金豺的位置分布。$\boldsymbol{rl}$ 是莱维飞行函数,其公式如(6)所示。

$$rl = 0.05 \times LF(y) \tag{6}$$

$$LF(y) = 0.01 \times \frac{\mu \times \sigma}{|v^{\frac{1}{\beta}}|} \tag{7}$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin(\frac{\pi\beta}{2})}{\Gamma(\frac{1+\beta}{2}) \times \beta \times 2^{\frac{\beta-1}{2}}} \right)^{1/\beta} \tag{8}$$

式中:μ、v 是[0,1]内均匀分布的随机值;β 是设置为 1.5 的常数。在金豺优化算法中,猎物的逃脱能量作为关键参数,决定雄雌个体的搜索与捕获策略,其计算公式如(9)~(11)所示。

$$E = E_1 \times E_0 \tag{9}$$

$$E_0 = 2r - 1 \tag{10}$$

$$E_1 = c_1(1 - \frac{t}{T}) \tag{11}$$

其中,E_1 表示猎物能量的下降过程,在逃脱过程中,猎物的能量值呈线性递减,由初始值 1.5 逐步降至 0;E_0 表示初始能量值;r 表示[0,1]内的随机值;T 为最大迭代数,$c_1 = 1.5$。

通过式(12)计算金豺位置平均值来更新位置:

$$Y(t+1) = \frac{Y_1(t) + Y_2(t)}{2} \tag{12}$$

1.3 抓捕阶段

在捕猎过程中,猎物的逃脱能量随豺狼的攻击而逐渐递减。豺狼对会先基于前期探测信息对猎物实施包围,待包围完成后,再发起最终攻击。在抓捕阶段,金豺的位置更新计算公式如式(13)、(14)所示。

$$\boldsymbol{Y}_1(t) = \boldsymbol{Y}_M(t) - E \times |\boldsymbol{rl} \times \boldsymbol{Y}_M(t) - \boldsymbol{Prey}(t)| \tag{13}$$

$$\boldsymbol{Y}_2(t) = \boldsymbol{Y}_{FM}(t) - E \times |\boldsymbol{rl} \times \boldsymbol{Y}_{FM}(t) - \boldsymbol{Prey}(t)| \tag{14}$$

1.4 探索与开发的转换

在 GJO 算法设计中,猎物逃脱能量是调控雄雌金豺追踪和捕猎策略的关键参数。猎物的逃脱能量 E 随着迭代次数的增加而显著降低。当 $|E|>1$ 时,金豺进行全局搜

索，寻找猎物的位置；当 $|E|<1$ 时，金豺在局部进行开发，进行围攻猎物。

2　改进的金豺优化算法

2.1　混沌精英初始化策略

研究表明，初始种群的质量显著影响算法的求解精度与收敛速率[11]。为提升初始解质量，MSIGJO 结合 Logistic 混沌映射与混合精英策略进行种群初始化：通过 Logistic 映射生成具有遍历性的混沌序列，同时利用混合精英策略产生多样性解，二者协同构建初始种群。这一机制既提供了优质的初始解，又通过增强全局探索能力，弥补了基础 GJO 算法在初始化阶段的局限性。

1）Logistic 映射初始化种群

混沌映射作为一种非线性系统的复杂动力学方法，其产生的混沌变量具有遍历性、伪随机性、初值敏感性和非周期性等特性[12]，通过遍历性在解空间均匀采样，避免早熟收敛。在优化算法领域，混沌映射常被用于替代传统伪随机数生成机制，通过产生[0，1]区间内的非周期序列来改善搜索过程。MSIGJO 算法采用 Logistic 混沌映射生成具有遍历特性的混沌序列[13]，通过非线性变换将混沌变量映射至解空间维度，并对种群个体施加混沌扰动。Logistic 混沌映射定义为：

$$x_{i+1}=ax_i(1-x_i) \tag{15}$$

式中：a 为常数，$a=4$。

2）混合精英策略

混合精英策略将透镜成像反向学习（LOBL）与纵横交叉机制相结合。该方法旨在通过透镜成像生成多样化候选解，并利用交叉算子促进个体间信息交换，从而实现探索与开发的平衡。

透镜成像反向学习（LOBL）是一种受到光学透镜成像原理启发的反向学习策略[14]，用于全局优化问题中增强种群多样性和提高搜索效率。传统的反向学习方法通常简单地计算一个解相对于搜索空间中心的反向位置，而透镜成像反向学习则引入了一个缩放因子，该因子通过透镜成像模型调整“像”的位置与尺度。对于种群中的每个候选解 x_i，利用 LOBL 策略生成新的候选解 x_i'：

$$x_i'=\frac{lb+ub}{2}+\frac{lb+ub}{2k}-\frac{x_i}{k} \tag{16}$$

式中：lb 和 ub 分别为搜索空间的下界和上界，k 为缩放因子，其定义为：

$$k=\left(1+\sqrt{\frac{t}{T}}\right)^{10} \tag{17}$$

式中：t 表示当前迭代次数，T 为最大迭代次数。生成后对 x_i' 进行边界控制，保证：

$$x_i'=\min(\max(x_i',lb),ub) \tag{18}$$

在 LOBL 更新之后，采用纵横交叉策略进一步增强解的多样性并重组信息[15]。该过程分为两个部分：

横向交叉是对于 $i=1,2,\cdots,N$（其中 N 为候选解的数量），利用相邻个体进行交叉更新：

对于奇数索引的个体，更新公式如式（19）所示。

$$x_i'(j)=rx_i(j)+(1-r)x_{i+1}(j)+c(x_i(j)-x_{i+1}(j)) \tag{19}$$

对于偶数索引的个体，更新公式如式（20）所示。

$$x_i'(j)=rx_{i-1}(j)+(1-r)x_i(j)+c(x_{i-1}(j)-x_i(j)) \tag{20}$$

式中：j 表示维度索引，r 是服从均匀分布[0，1]的随机数，$c=2\times rand()-1$ 为服从区间[−1，1]的系数。更新后同样进行边界控制。

纵向交叉是在横向交叉的基础上，对每个候选解的各个维度进行纵向交叉。对每个维度 j，随机选取该解中的两个维度 d_1 和 d_2 根据式（21）进行更新：

$$x_i'(j)=r_1x_i(d_1)+(1-r_1)x_i(d_2) \tag{21}$$

式中：r_1 同样为[0，1]均匀分布的随机数。该机制有助于在个体内部不同维度之间交换信息，进一步提高解的多样性。在经过 LOBL 和交叉操作后，计算新候选解 x_i' 的适应度 $f(x_i')$。精英更新规则如式（22）、（23）所示。

如果 $f(x_i')<f(\text{MaleJackal})$，则更新精英解：

$$\text{MaleJackal}=x_i' \tag{22}$$

否则若 $f(\text{MaleJackal})<f(\mathbf{x}_i')<f(\text{FemaleJackal})$，则更新次优解：

$$\text{FemaleJackal}=x_i' \tag{23}$$

2.2　分阶段混合逃逸能量调控机制

在 GJO 算法中，逃逸能量 E 作为核心控制参数，由确定性分量和随机分量构成。然而，随机分量的波动导致 E 呈现非平稳变化，影响算法在全局探索与局部开发间的平稳转换。为此，MSIGJO 提出分阶段混合调控机制：初期采用较大扰动结合柯西分布[16]增强探索能力；后期运用余弦退火与高斯扰动[17]优化局部搜索；同时引入 Lévy 扰动提升逃逸能力。该机制使逃逸能量随迭代次数自适应调整，确保各阶段具有最优搜索步长，既避免早熟收敛又提高搜索效率。在迭代的早期阶段，为了充分探索搜索空间，采用较大的扰动如式（24）所示。

$$E_1=1.5\cdot e^{-2\cdot\frac{t}{T}} \tag{24}$$

$$E_0\sim\text{Cauchy}(0,1) \tag{25}$$

式中：扰动项 E_0 采用柯西分布生成，这种设置确保了在初期阶段，逃逸能量较大，从而使个体有更大概率跳出局部最优区域。当迭代进入中后期，搜索重心转向局部开发，此时需要降低扰动幅度以实现细致搜索。调控公式如式（26）所示。

$$E_1=0.8\cdot0.5\cdot\left(1+\cos\left(\pi\cdot\frac{t}{T}\right)\right) \tag{26}$$

$$E_0\sim\mathcal{N}(0,0.5^2) \tag{27}$$

式中：扰动项 E_0 采用高斯分布生成，这样在后期，逃逸能量逐步减小，有助于算法精细开发和快速收敛。

为进一步增强个体跳出局部最优的能力,在计算最终逃逸能量时,还引入 Lévy 扰动。最终逃逸能量 E 的计算公式为:

$$E = E_1 \cdot E_0 + LF \cdot \left(1 - \frac{t}{T}\right) \tag{28}$$

式中:$LF \cdot \left(1 - \frac{t}{T}\right)$ 保证了 Lévy 扰动在迭代初期贡献较大,随着迭代进程逐渐减弱。

2.3 自适应 t-差分融合变异

在标准 GJO 算法中,最终解通过雄雌金豺个体位置的平均值获得,然而这种处理方式未能充分考虑种群个体适应度差异所表征的捕猎能力异质性,导致搜索精度和收敛速度受限。为此,MSIGJO 根据算法各阶段求解的特点,引入了自适应 t-差分融合变异策略:基于自适应 t 分布的长尾特性实施扰动,能够帮助算法跳出局部最优,增强算法的探索能力[18];而差分变异则会利用全局最优和随机个体的信息,通过计算个体间的差异,能够捕捉种群中分布的信息,从而引导搜索向更优区域前进,进一步提高搜索效率。

针对金豺优化算法不同阶段的搜索特性,提出一种基于适应度分级的混合变异策略:首先计算种群平均适应度值作为分类阈值,将个体划分为两类:对于适应度低于平均值的个体,采用自适应 t 分布扰动以增强全局探索能力;对于适应度高于平均值的个体,则实施差分变异操作以提升局部开发效率。每次变异后,通过精英保留策略比较新旧个体的适应度,择优保留至下一代种群中。该机制有效平衡了算法在不同进化阶段的探索与开发需求。自适应 t 分布扰动变异公式为:

$$freen = e^{4\left(\frac{t}{T}\right)^2} \tag{29}$$

$$x_i' = x_i + \boldsymbol{trnd}(freen) \cdot x_i \tag{30}$$

式中:$freen$ 为自适应 t 分布的自由度参数,$\boldsymbol{trnd}(freen)$ 表示自由度为 $freen$ 的 t 分布随机变量。否则从种群中随机选取一个个体 a(且 $a \neq i$),同时利用全局最优个体 x_M 的信息,对当前个体 x_i 进行差分变异更新:

$$x_i' = r \cdot (x_M - x_i) - r' \cdot (x_a - x_i) \tag{31}$$

式中:r 和 r' 均为服从均匀分布 U(0,1) 中产生的随机数。

2.4 MSIGJO 算法的流程

MSIGJO 算法的流程图如图 1 所示。算法流程包括 8 个步骤:

1)用 Logistic 映射生成金豺种群位置,设定种群规模 n、维度 d、最大迭代次数 T;

2)计算个体适应度,选取最优雄豺、雌豺个体;

3)通过混合精英策略更新种群和雌、雄豺;

4)根据式(24)~(28)计算逃脱能量 E;

5)如 $|E|>1$,使用式(4)、(5)、(12)更新位置;反之使用式(13)、(14)、(12)更新位置;

6)对低于平均适应度的个体实施 t 分布扰动,高于平均的个体执行差分变异;

7)对比扰动前后个体,保留更优解更新种群;

8)达到迭代次数则输出最优解,否则返回步骤 3)迭代。

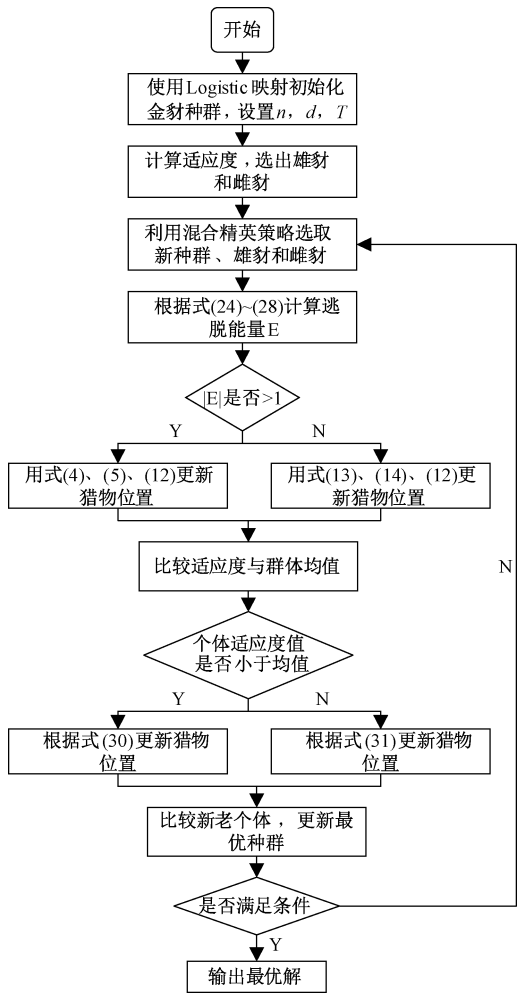

图 1 MSIGJO 算法流程

Fig. 1 Algorithm flow chart of MSIGJO

3 实验结果与分析

为了充分验证 MSIGJO 算法在解决优化问题方面的有效性,分别将 MSIGJO 与粒子群优化算法(PSO)、灰狼优化算法(GWO)、鲸鱼优化算法(WOA)、基本金豺优化算法(GJO)和改进的金豺优化算法 OGJO、LGJO 在基准测试函数和 CEC2021 测试函数上进行对比验证。所有算法均在具有 16 GB 运行内存和 64 位系统的 Windows11 平台上运行,且使用 Matlab2024a 进行实验。

3.1 参数设置与测试函数说明

为确保实验结果的可靠性与可比性,本研究采用严格的参数控制方案:所有对比算法均按照表 1 所示的统一参数标准进行初始化;MSIGJO 算法的运行参数与其他算法保持完全一致,均设置种群规模 $N=30$、最大迭代次数 $T=1\,000$,每个算法均执行 20 次独立重复实验以消除随机性影响。

表 1　参数设置

Table 1　Parameter settings

算法	参数
MSIGJO	$a=4$；$c_1=1.5$
LGJO	$c_1=1.5$
OGJO	$c_1=1.5$
GJO	$c_1=1.5$
PSO	$c_1=c_2=2$；$V_{max}=1$
WOA	$b=1$
GWO	$a\in[0,2]$

本研究采用 13 个特征各异的基准测试函数进行算法性能评估，具体包括两类典型函数集：单峰函数组（F1～F7），这类函数具有唯一的全局最优解，能够有效评估算法的收敛速度与求解精度；多峰函数组（F8～F13），这类函数包含多个局部极值点和一个全局最优解，可全面测试算法跳出局部最优和平衡全局探索与局部开发的能力。所有测试函数的数学表达式、搜索范围及理论最优值等详细信息如表 2 所示。

3.2　收敛精度分析

为验证 MSIGJO 算法的优化性能，本研究将其与多种经典优化算法进行了对比实验，结果如表 3 所示。在单峰函数测试集中，MSIGJO 表现出显著优势，在 F1～F4 函数优化中，该算法均能精确收敛至理论最优解，且求解速率更快以及标准差波动范围更小。在 F6、F7 中，MSIGJO 性能明显优于其他对比算法，且在 F5 中与 LGJO 最优值相差不大。在 F8、F9 和 F11 函数优化中，该算法均能够精确收敛至理论最优解；且在 F10、F12 与 F13 函数中，所有对比的算法中 MSIGJO 算法各项数据均排第一。由此可见，MSIGJO 算法从收敛精度，稳定性到平衡全局搜索与局部开发的能力都是非常出色的。

表 2　基准测试函数

Table 2　Benchmark test function

函数	名称	纬度	范围	最优值
F1	Sphere	30	[−100,100]	0
F2	Schwefel 2.22	30	[−10,10]	0
F3	Schwefel 1.2	30	[−100,100]	0
F4	Schwefel 2.21	30	[−100,100]	0
F5	Rosenbrock	30	[−30,30]	0
F6	Step	30	[−100,100]	0
F7	Quartic with noise	30	[−1.28,1.28]	0
F8	Schwefel 2.26	30	[−500,500]	−12,569.5
F9	Rastrigin	30	[−5.12,5.12]	0
F10	Ackley	30	[−32,32]	0
F11	Griewank	30	[−600,600]	0
F12	Penalized1	30	[−50,50]	0
F13	Penalized2	30	[−50,50]	0

为更直观评估算法性能，图 2 展示了关键收敛曲线对比结果：在单峰函数 F1～F4 测试中，MSIGJO 仅需约 100 次迭代即收敛至理论最优值，这可能因为混沌精英初始化策略通过 Logistic 混沌映射提升种群多样性以及混合精英策略提供优质初始解，从而使算法收敛精度与速度得到较大提升；而从图 2 中可以看到 MSIGJO 算法的初次迭代都非常优秀，说明混合精英策略所提供初始解具有很高的质量。从 F7、F8 和 F10 函数收敛曲线可以看出 MSIGJO 算法寻优精度远远高于其他函数，说明 MSIGJO 算法采用的分阶段混合逃逸能量调控机制通过动态调整搜索策略，实现了全局探索与局部开发的优化平衡。从 F6、F12 和 F13 函数收敛曲线可以看出，MSIGJO 算法多次跳出局部最优且收敛精度与速度更快，说明将自适应 t-差分融合变异用于改进位置更新有助于算法跳出局部最优并且能够提高寻求最优解的概率。本实验通过对比 MSIGJO 算法与多种经典优化算法在单峰和多峰测试函数上的表现，验证了其具有高质量的初始解，优秀的收敛速度与精度以及跳出局部最优的能力。

3.3　CEC2021 测试函数

为全面评估 MSIGJO 算法处理单目标约束问题的性能，本研究采用 CEC2021 基准测试函数集进行验证。该测试集包含 10 个具有维度可扩展特性的复杂函数，专注于单目标有约束优化问题，这类问题在工程、经济、科学等领域广泛存在。各算法依据表 1 的参数进行配置，本研究在 20 维空间下对 MSIGJO 算法与对比算法进行了系统测试，实验结果如表 4 所示。为更清晰地展示算法性能差异，图 3 选取了典型测试函数的收敛曲线对比图。

表 3 基准测试函数优化结果对比

Table 3 Comparison of optimization results for benchmarking functions

函数	指标	GJO	PSO	WOA	GWO	LGJO	OGJO	MSIGJO
F1	最优值	1.71×10^{-114}	3.04×10^{-6}	9.37×10^{-159}	4.37×10^{-60}	1.17×10^{-136}	1.09×10^{-124}	**0**
	平均值	2.08×10^{-114}	2.35×10^{-4}	1.98×10^{-149}	6.67×10^{-59}	3.62×10^{-135}	2.33×10^{-119}	**0**
	标准差	5.32×10^{-115}	3.28×10^{-4}	2.81×10^{-149}	8.82×10^{-59}	3.96×10^{-135}	3.30×10^{-119}	**0**
F2	最优值	8.00×10^{-67}	1.01×10^{-4}	4.18×10^{-105}	5.60×10^{-35}	3.07×10^{-92}	1.23×10^{-70}	**0**
	平均值	8.02×10^{-67}	2.39×10^{-4}	1.49×10^{-102}	7.98×10^{-35}	1.54×10^{-89}	$1,26\times10^{-70}$	**0**
	标准差	2.74×10^{-69}	1.95×10^{-4}	2.10×10^{-102}	3.37×10^{-35}	2.18×10^{-89}	3.45×10^{-72}	**0**
F3	最优值	1.84×10^{-44}	5.79×10^{2}	1.19×10^{4}	2.04×10^{-15}	1.29×10^{-44}	8.47×10^{-49}	**0**
	平均值	3.17×10^{-42}	5.81×10^{2}	1.58×10^{4}	7.70×10^{-15}	2.48×10^{-41}	1.50×10^{-46}	**0**
	标准差	4.46×10^{-42}	2.48	5.58×10^{3}	8.00×10^{-15}	3.51×10^{-41}	2.10×10^{-46}	**0**
F4	最优值	2.35×10^{-35}	2.87	2.43	1.52×10^{-15}	1.37×10^{-41}	4.75×10^{-35}	**0**
	平均值	1.10×10^{-34}	4.17	1.17×10	3.73×10^{-15}	6.60×10^{-38}	6.55×10^{-33}	**0**
	标准差	1.23×10^{-34}	1.83	1.31×10	3.12×10^{-15}	9.33×10^{-38}	9.20×10^{-33}	**0**
F5	最优值	2.71×10	8.40×10	2.69×10	2.70×10	2.62×10	2.80×10	2.62×10
	平均值	2.72×10	8.52×10	2.69×10	2.70×10	**2.67×10**	2.80×10	2.70×10
	标准差	1.74×10^{-2}	1.57	5.72×10^{-2}	1.20	6.50×10^{-1}	3.56×10^{-2}	**3.36×10^{-3}**
F6	最优值	2.68	1.87×10^{-4}	2.45×10^{-2}	2.47×10^{-1}	1.24	1.76	**1.18×10^{-6}**
	平均值	2.83	2.24×10^{-4}	2.99×10^{-2}	3.70×10^{-1}	1.87	2.25	**6.04×10^{-6}**
	标准差	2.26×10^{-1}	5.15×10^{-5}	7.65×10^{-3}	1.73×10^{-1}	8.83×10^{-1}	6.91×10^{-1}	**6.88×10^{-6}**
F7	最优值	4.16×10^{-4}	2.97×10^{-2}	3.43×10^{-4}	4.15×10^{-4}	1.25×10^{-4}	4.25×10^{-5}	**1.49×10^{-6}**
	平均值	4.89×10^{-4}	3.10×10^{-2}	1.92×10^{-3}	5.18×10^{-4}	2.12×10^{-4}	1.21×10^{-4}	**1.91×10^{-6}**
	标准差	1.02×10^{-4}	1.93×10^{-3}	2.23×10^{-3}	1.45×10^{-4}	1.24×10^{-4}	1.11×10^{-4}	**6.01×10^{-6}**
F8	最优值	-6.57×10^{3}	-9.31×10^{3}	-1.15×10^{4}	-7.35×10^{3}	-1.25×10^{4}	-7.33×10^{3}	**-1.26×10^{4}**
	平均值	-4.79×10^{3}	-7.76×10^{3}	-1.03×10^{4}	-6.80×10^{3}	-9.56×10^{3}	-5.56×10^{3}	**-1.26×10^{4}**
	标准差	2.51×10^{3}	2.38×10^{2}	1.80×10^{3}	7.68×10^{2}	3.45×10^{2}	3.68×10	**2.35×10^{-1}**
F9	最优值	0	4.38×10	0	0	0	0	**0**
	平均值	0	5.58×10	5.60×10^{-15}	5.68×10^{-14}	0	0	**0**
	标准差	0	1.69×10	1.79×10^{-14}	8.04×10^{-14}	0	0	**0**
F10	最优值	4.00×10^{-15}	3.63×10^{-4}	4.44×10^{-16}	1.46×10^{-14}	3.99×10^{-15}	4.00×10^{-15}	**4.44×10^{-16}**
	平均值	4.32×10^{-15}	1.10×10^{-2}	3.28×10^{-15}	1.78×10^{-14}	3.99×10^{-15}	4.35×10^{-15}	**4.44×10^{-16}**
	标准差	1.12×10^{-15}	1.56×10^{-2}	2.24×10^{-15}	4.86×10^{-15}	0	1.12×10^{-15}	**0**
F11	最优值	0	3.31×10^{-5}	0	0	0	0	**0**
	平均值	0	1.33×10^{-2}	0	3.26×10^{-3}	0	0	**0**
	标准差	0	1.56×10^{-2}	0	5.25×10^{-3}	0	0	**0**
F12	最优值	1.40×10^{-1}	7.50×10^{-5}	6.28×10^{-4}	1.27×10^{-2}	2.86×10^{-2}	7.71×10^{-2}	**7.64×10^{-8}**
	平均值	2.61×10^{-1}	3.11×10^{-2}	8.26×10^{-3}	2.42×10^{-2}	8.10×10^{-2}	1.63×10^{-2}	**4.63×10^{-7}**
	标准差	1.79×10^{-1}	6.99×10^{-2}	6.11×10^{-3}	1.43×10^{-2}	3.81×10^{-2}	7.04×10^{-2}	**5.47×10^{-7}**
F13	最优值	1.24	1.59×10^{-5}	8.47×10^{-2}	3.01×10^{-1}	3.40×10^{-1}	1.24	**2.22×10^{-7}**
	平均值	1.57	2.71×10^{-3}	2.40×10^{-1}	5.94×10^{-1}	9.55×10^{-1}	1.49	**4.06×10^{-6}**
	标准差	2.02×10^{-1}	5.13×10^{-3}	1.58×10^{-1}	2.16×10^{-1}	3.33×10^{-1}	1.29×10^{-1}	**2.69×10^{-6}**

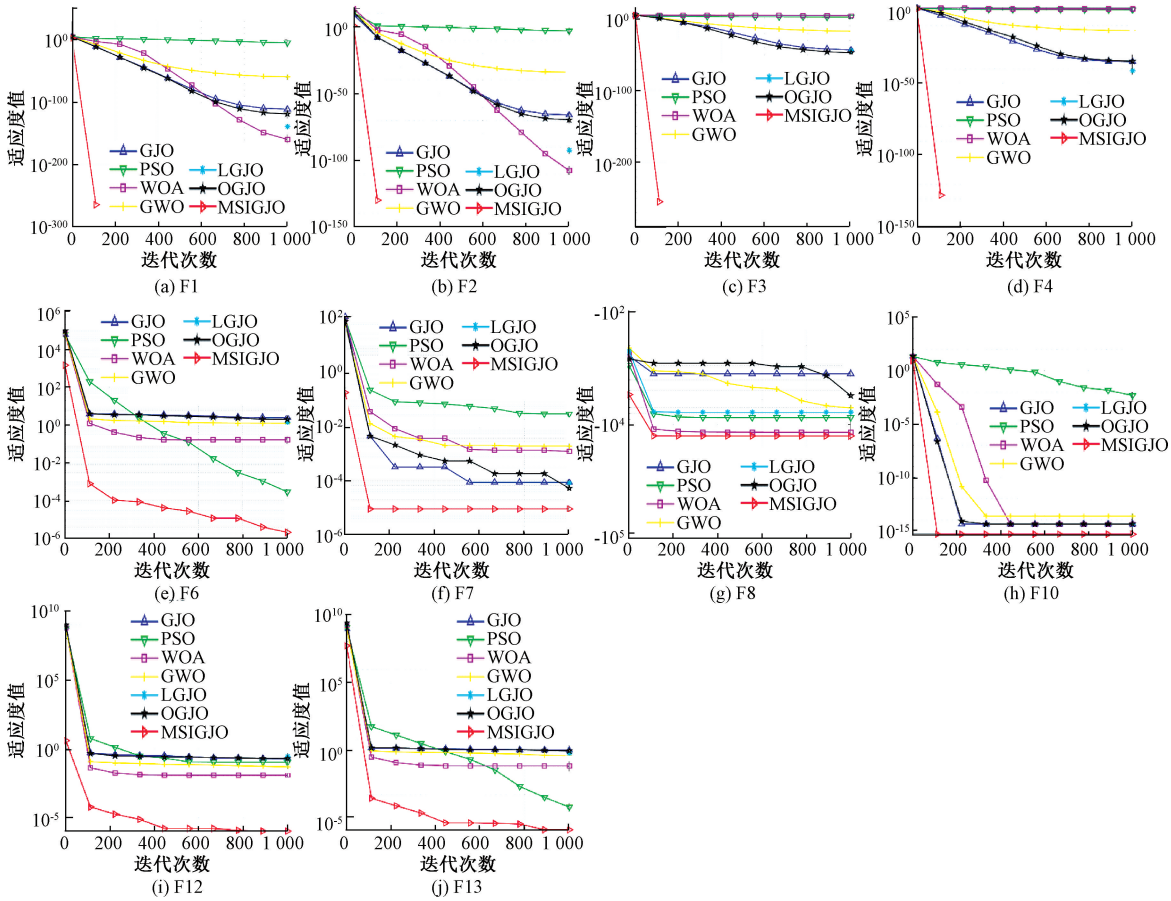

图 2　收敛曲线对比图

Fig. 2　Convergence curve of the algorithm

表 4　CEC2021 测试函数优化结果

Table 4　CEC2021 test function optimization results

函数	指标	GJO	PSO	WOA	GWO	LGJO	OGJO	MSIGJO
	最优值	1.11×10^{-161}	3.44×10^{-7}	3.78×10^{-179}	6.87×10^{-87}	1.29×10^{-229}	2.53×10^{-174}	**0**
F1	平均值	4.08×10^{-157}	1.00×10^{3}	1.41×10^{-168}	5.73×10^{-85}	1.40×10^{-229}	3.46×10^{-166}	**0**
	标准差	1.28×10^{-156}	3.16×10^{3}	0	1.19×10^{-84}	0	0	**0**
	最优值	0	1.22×10^{2}	0	0	0	0	**0**
F2	平均值	0	5.07×10^{2}	5.45×10^{-13}	9.33×10^{-1}	0	0	**0**
	标准差	0	3.50×10^{2}	8.78×10^{-13}	2.55	0	0	**0**
	最优值	0	3.97	0	0	0	0	**0**
F3	平均值	0	2.19×10	0	7.34×10	0	0	**0**
	标准差	0	1.24×10	0	3.78×10	0	0	**0**
	最优值	0	1.47	0	0	0	0	**0**
F4	平均值	0	2.14	2.35×10^{-1}	6.55×10^{-2}	0	0	**0**
	标准差	0	3.75×10^{-1}	4.81×10^{-1}	1.99×10^{-1}	0	0	**0**
	最优值	1.14×10^{-240}	2.41	4.12×10^{-165}	1.87×10^{-36}	0	1.13×10^{-234}	**0**
F5	平均值	2.09×10^{-200}	3.94×10^{2}	4.56×10^{-138}	1.45	2.01×10^{-309}	1.78×10^{-192}	**0**
	标准差	0	3.53×10^{2}	1.44×10^{-137}	4.17	0	0	**0**
	最优值	8.52×10^{-5}	4.59	5.06×10^{-4}	2.31×10^{-2}	1.11×10^{-4}	2.14×10^{-4}	**0**

表 4(续)

Table 4 (continued)

函数	指标	GJO	PSO	WOA	GWO	LGJO	OGJO	MSIGJO
F6	平均值	4.88×10^{-2}	6.81×10	3.14×10^{-1}	9.73×10^{-1}	1.68×10^{-2}	1.87×10^{-1}	**0**
	标准差	1.12×10^{-2}	6.36×10	7.93×10^{-1}	1.72	4.44×10^{-2}	4.23×10^{-1}	**0**
	最优值	1.09×10^{-4}	1.81	7.62×10^{-3}	1.57×10^{-2}	4.80×10^{-5}	2.78×10^{-5}	$\mathbf{2.22\times10^{-16}}$
F7	平均值	2.96×10^{-2}	2.99×10^{2}	5.34×10^{-2}	7.63×10^{-2}	7.22×10^{-3}	5.06×10^{-3}	$\mathbf{8.88\times10^{-17}}$
	标准差	2.24×10^{-2}	3.51×10^{2}	4.72×10^{-2}	1.01×10^{-1}	1.08×10^{-2}	7.73×10^{-3}	$\mathbf{1.17\times10^{-16}}$
	最优值	0	3.32×10	0	0	0	0	**0**
F8	平均值	0	6.25×10	0	0	0	0	**0**
	标准差	0	2.01×10	0	0	0	0	**0**
	最优值	8.88×10^{-15}	2.35×10^{-6}	1.84×10^{-176}	8.88×10^{-15}	3.39×10^{-230}	8.88×10^{-15}	**0**
F9	平均值	8.88×10^{-15}	6.85×10^{-6}	7.99×10^{-15}	1.15×10^{-14}	3.55×10^{-15}	8.88×10^{-15}	$\mathbf{9.34\times10^{-316}}$
	标准差	0	4.34×10^{-6}	6.55×10^{-15}	4.29×10^{-15}	4.59×10^{-15}	0	**0**
	最优值	9.22×10^{-4}	4.94×10	2.20×10^{-2}	5.77×10	1.29×10^{-3}	7.74×10^{-4}	**0**
F10	平均值	2.39×10	6.02×10	1.01×10^{-1}	7.64×10	3.95	3.20×10	$\mathbf{8.90\times10^{-316}}$
	标准差	3.88×10	1.67×10	4.92×10^{-2}	1.24×10	1.25×10	4.17×10	**0**

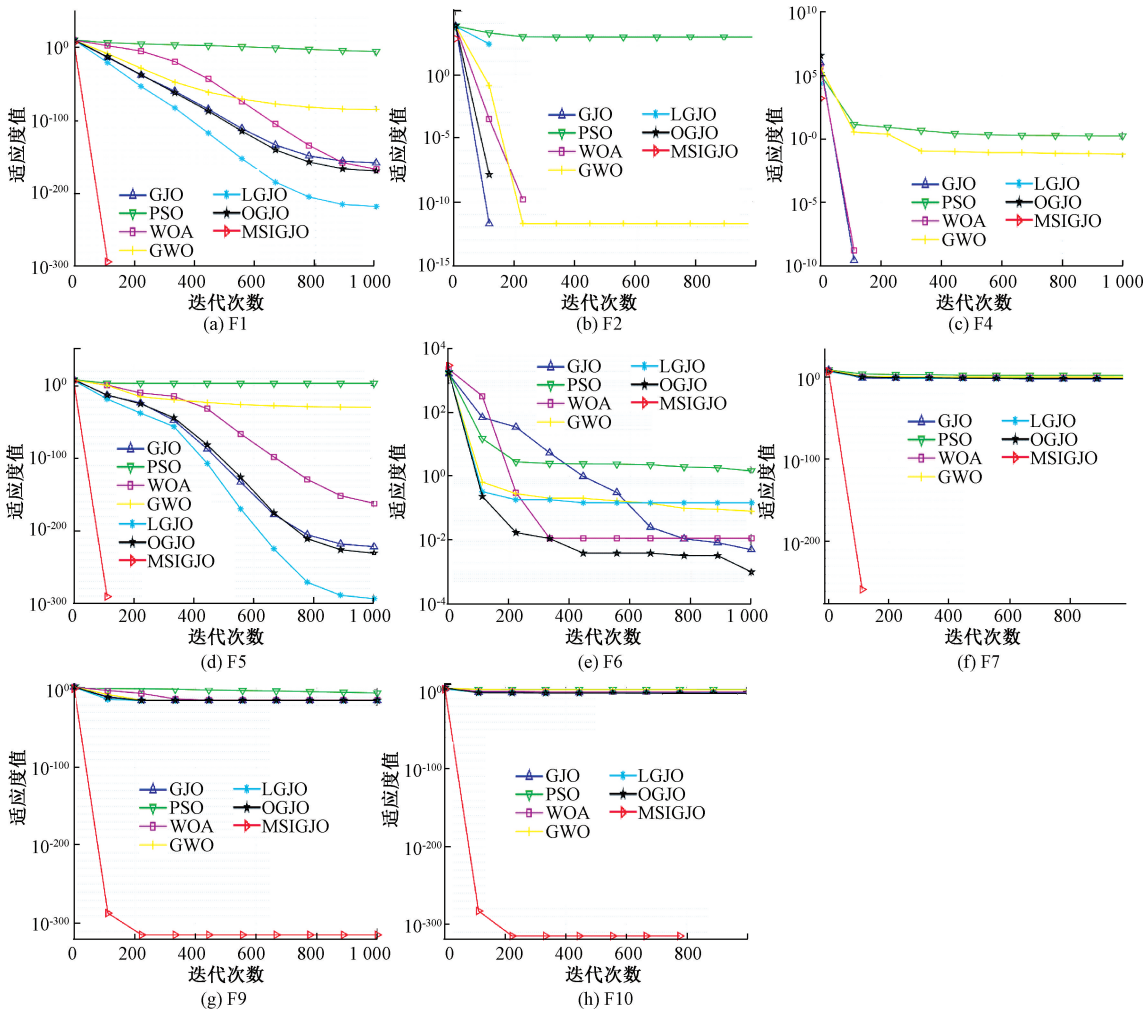

图 3 CEC2021 测试函数收敛曲线

Fig. 3 CEC2021 test function convergence curve

表 4 的实验数据表明，MSIGJO 算法在 F1～F6 和 F8 函数测试中展现出显著的优化性能：不仅所有测试函数的平均值均收敛至理论最优值，且 20 次独立运行的标准差均为 0，验证了算法具有优异的收敛稳定性和鲁棒性。F7 虽未取得理想值，但在所有对比算法中排名第一，且稳定性优越。通过对收敛曲线的对比分析，可以更加直观地体现 MSIGJO 算法的性能优化：在 F2、F4、F6 函数测试中，该算法在初次迭代即获得最优解，如图 3 纵坐标轴上三角星标记所示；从 F9、F10 的收敛曲线可以看到 MSIGJO 的寻优范围远远超过其他算法。由图 3 可以看出：MSIGJO 在所有测试函数中的收敛速度都明显优于其他算法，经过 100 次左右迭代均可以取得最优值，说明 MSIGJO 算法不仅可以跳出局部最优进行更大范围的寻优，而且由于初始生成高质量解，导致收敛精度与速度明显优于其他算法。基于 CEC2021 测试集的实验结果表明，MSIGJO 算法在单目标约束优化问题中展现出显著优势，不仅收敛速度更快、求解精度更高，同时具备更强的跳出局部最优能力，并在全局探索与局部开发之间实现了更优的平衡，综合性能显著优于其他对比算法。

3.4　Wilcoxon 秩和检验

为验证 MSIGJO 算法的性能优势，本研究采用 Wilcoxon 秩和检验对 CEC2021 测试集上的优化结果进行统计分析。P 值的大小可以反映两种算法之间差异的显著性，一般当 P 值＜5%时，表示两者存在显著差异。如表 5 所示，检验结果表明 MSIGJO 算法与对比算法在大多数测试函数上的性能具有明显差异。这一统计结果不仅证实了 MSIGJO 算法在寻优能力上的显著优势，同时也验证了该算法在求解单目标约束优化问题时的可靠性和有效性。

表 5　Wilcoxon 秩和检验实验结果

Table 5　Wilcoxon rank sum test experimental results

P 值	GJO	PSO	WOA	GWO	LGJO	OGJO
F1	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}
F2	1	6.34×10^{-5}	3.44×10^{-2}	1.98×10^{-3}	1	1
F3	1	6.39×10^{-5}	1	2.21×10^{-3}	1	3.68×10^{-1}
F4	1	6.39×10^{-5}	1	6.39×10^{-5}	1	1
F5	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	1	5.97×10^{-3}
F6	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}
F7	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}	6.39×10^{-5}
F8	1	6.39×10^{-5}	1	1	1	1
F9	1.69×10^{-4}	1.73×10^{-4}	1.41×10^{-4}	8.17×10^{-5}	1.73×10^{-4}	1.63×10^{-4}
F10	1.83×10^{-4}	1.83×10^{-4}	1.83×10^{-4}	1.83×10^{-4}	1.83×10^{-4}	1.83×10^{-4}

3.5　压力容器设计优化问题

压力容器设计优化问题是经典的工程设计多目标优化问题，由容器厚度 T_s、封头厚度 T_h、内半径 R 和圆柱形长度 L 共 4 个设计变量构成，其目标是在满足一定的约束条件下优化这 4 个设计变量来最小化总成本（材料、成型和焊接的成本），相当于求有约束条件的多目标函数最小值问题。压力容器设计问题的目标函数为：

$$\text{Variable } \boldsymbol{x}=[T_s,T_h,R,L];$$

$$\min f(x)=0.622\,4x_1x_3x_4+1.778\,1x_2x_3^2+3.166\,1x_1^2x_4+19.84x_1^2x_3;$$

$$\begin{cases}\text{s.t.}\\ g_1(x)=-x_1+0.0193x_3\leqslant 0;\\ g_2(x)=-x_2+0.00954x_3\leqslant 0;\\ g_3(x)=-\pi x_3^2x_4-\dfrac{4\pi x_3^3}{3}+1\,296\,000\leqslant 0;\\ g_4(x)=x_4-240\leqslant 0;\end{cases}$$

$$\begin{aligned}&\textit{Variable range}\\ &0\leqslant x_i\leqslant 100,i=1,2;\\ &10\leqslant x_i\leqslant 200,i=3,4\end{aligned}\tag{32}$$

为了验证算法在压力容器设计优化问题中的有效性，采用 GJO、PSO、WOA、GWO、LGJO、OGJO 及 MSIGJO 等 7 种优化算法进行对比实验。实验设置如下：算法的参数设置同表 1，种群规模为 30，最大迭代次数 1 000 次，每种算法独立运行 20 次。实验结果如表 6 所示，MSIGJO 算法在压力容器总成本优化方面表现最优，最低成本达 6 059.71。相较于标准 GJO 算法，MSIGJO 的最优成本和平均成本分别降低了 882.81 和 1 370.65。实验结果表明，MSIGJO 算法在解决压力容器设计优化问题中具有显著优势，能够获得更优的设计方案，充分验证了该算法在面对多个目标且有多个约束存在的情况下仍然具有出色的解决问题的能力，也证明了 MSIGJO 算法在实际工程应用中的有效性。

表 6 压力容器设计的优化结果

Table 6 Optimization results of pressure vessel design

算法	T_s	T_h	R	L	best	mean
GJO	18.28	9.10	56.97	51.60	6 942.52	7 448.25
PSO	13.00	6.78	42.09	176.63	6 063.92	6 704.49
WOA	15.66	8.10	49.64	101.16	6 622.79	9 141.63
GWO	12.86	7.36	42.08	176.80	6 061.79	6 366.44
LGJO	15.88	7.56	51.37	87.93	6 455.53	7 016.48
OGJO	13.40	6.89	40.32	200.00	6 290.35	6 780.45
MSIGJO	12.96	7.34	42.08	176.94	**6 059.71**	**6 077.60**

4 结 论

针对金豺优化算法在求解有约束优化问题时面临的种群质量差、收敛速度慢和易陷入局部最优等问题,本文提出一种基于多策略改进的金豺优化算法:采用 Logistic 混沌映射和混合精英策略初始化种群,以提高初始解的质量和多样性;设计分阶段混合逃逸能量调控机制,动态平衡算法的全局探索与局部开发能力;同时利用自适应 t-差分融合变异根据不同个体间的差异进行选择性位置更新,以增强算法的随机搜索能力,扩大寻优范围,加快收敛速度并提升跳出局部最优的能力。为验证 MSIGJO 的性能,选取了 13 个标准基准测试函数和 CEC2021 测试函数集进行实验。实验数据表明,MSIGJO 在收敛速度、精度和鲁棒性等关键性能指标上均展现出显著优势,Wilcoxon 秩和检验进一步验证了其优越性。通过 CEC2021 测试函数与压力容器设计优化问题的求解,证明了 MSIGJO 在单目标有约束问题和多目标有约束的优化问题上的性能同样表现良好。未来研究将聚焦于复杂多目标优化问题及实际工程案例的拓展应用。

参考文献

[1] WANG D SH, TAN D P, LIU L. Particle swarm optimization algorithm: An overview [J]. Soft Computing, 2018, 22: 387-408.

[2] MIRJALILI S, MIRJALILI S M, LEWIS A. Grey wolf optimizer[J]. Advances in Engineering Software, 2014, 69(3): 46-61.

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

[4] CHOPRA N, ANSARI M M. Golden jackal optimization: A novel nature-inspired optimizer for engineering applications [J]. Expert Systems with Applications, 2022, 198(7): 116924-116970.

[5] KUMAR P J S, PANDURANGAN R, BAPU B R T, et al. Cancer miRNA biomarker classification based on syntax-guided hierarchical attention network optimized with golden jackal optimization algorithm [J]. Biomedical Signal Processing and Control, 2024, 95: 106303.

[6] ELSEIFY M A, HASHIM F A, HUSSIEN A G, et al. Single and multi-objectives based on an improved golden jackal optimization algorithm for simultaneous integration of multiple capacitors and multi-type DGs in distribution systems[J]. Applied Energy, 2024, 353: 122054.

[7] WANG B, JIN Q B, ZHAO R L, et al. A new optimization idea: parallel search-based golden jackal algorithm[J]. IEEE Access, 2023, 11: 102115-102131.

[8] MOHAPATRA S, MOHAPATRA P. An improved golden jackal optimization algorithm using opposition-based learning for global optimization and engineering problems[J]. International Journal of Computational Intelligence Systems, 2023, 16(1): 147.

[9] WANG J, WANG W CH, CHAU K W, et al. An improved golden jackal optimization algorithm based on multi-strategy mixing for solving engineering optimization problems [J]. Journal of Bionic Engineering, 2024, 21(2): 1092-1115.

[10] YANG J, XIONG J L, CHEN Y L, et al. Improved golden jackal optimization for optimal allocation and scheduling of wind turbine and electric vehicles parking lots in electrical distribution network using rosenbrock's direct rotation strategy [J]. Mathematics, 2023, 11(6): 1415.

[11] ROSHANI R, MOTAMENI H, MOHAMADI H. A decentralized method for initial populations of genetic algorithms[J]. The Journal of Supercomputing, 2023, 79(9): 10232-10251.

[12] 宋宇,高岗,梁超,等.基于多策略改进灰狼算法的无人机路径规划[J].电子测量技术, 2025, 48(1): 84-91.
SONG Y, GAO G, LIANG CH, et al. UAV path planning based on multi-strategy improved gray wolf algorithm[J]. Electronic Measurement Technology,

2025，48(1)：84-91.

[13] BAO H，HUA ZH Y，WANG N，et al. Initials-Boosted coexisting chaos in a 2-D sine map and its hardware implementation[J]. IEEE Transactions on Industrial Informatics，2020，17(2)：1132-1140.

[14] LI L G，HUANG X W，QIAN SH Q，et al. Fuzzy hybrid coyote optimization algorithm for image thresholding[J]. Computers，Materials & Continua，2022，72(2)：3073-3090.

[15] YIN B，MO L P，MIN W，et al. An improved beetle antennae search algorithm and its application in coverage of wireless sensor networks[J]. Scientific Reports，2024，14(1)：29372.

[16] WANG X，LIU Q，ZHANG L. An adaptive sand cat swarm algorithm based on cauchy mutation and optimal neighborhood disturbance strategy[J]. 2023，8(2)：191.

[17] MU SH D，LIU B Y，GU J J，et al. Analyzing vehicle path optimization using an improved genetic algorithm in the presence of stochastic perturbation matter[J]. Scientific Reports，2024，14(1)：26105.

[18] LIU K ZH，DAI Y Q，LIU H. Improvement of dung beetle optimization algorithm application to robot path planning[J]. Applied Sciences，2025，15(1)：396.

作者简介

徐静萍，硕士，高级实验师，主要研究方向为嵌入式系统及模拟集成电路设计。

E-mail：xjp0706@163.com

王文杰(通信作者)，硕士研究生，主要研究方向为嵌入式系统及无人机系统与航迹规划。

E-mail：1372163130@qq.com

张鑫，硕士研究生，主要研究方向为无人机航迹控制和路径规划。

E-mail：1820736715@qq.com