计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.181102197

基于 Logistic 模型和随机差分变异的正弦余弦算法

徐 明[1] 焦建军[1] 龙 文[2]
1 贵州财经大学数学与统计学院 贵阳 550025
2 贵州财经大学贵州省经济系统仿真重点实验室 贵阳 550025
(201601138@mail.gufe.edu.cn)

摘 要 针对标准正弦余弦算法(Sine Cosine Algorithm,SCA)处理全局优化问题时存在收敛速度慢、易陷入局部最优和求解精度低的缺点,文中提出了一种基于非线性转换参数和随机差分变异策略的改进正弦余弦算法(LS-SCA)。首先,设计一种基于 Logistic 模型的非线性转换参数策略以平衡算法的全局搜索和局部开发能力;其次,引入随机差分变异策略以增强种群的多样性与避免算法陷入局部最优;最后,将非线性转换参数和随机差分变异策略进行融合。一方面,选取 12 个标准测试函数进行全局寻优的仿真实验。结果表明,与其他 SCA 类算法和最新智能算法相比,LS-SCA 在收敛精度和收敛速度指标上均能达到较优的效果。其中,随机差分变异策略对 LS-SCA 全局寻优能力的提升尤为明显。另一方面,利用 LS-SCA 优化神经网络参数解决了两类经典分类问题。实验结果表明,与传统的 BP 算法和其他智能算法相比,基于 LS-SCA 的神经网络能达到较高的分类准确率。

关键词:正弦余弦算法;非线性转换参数;随机差分变异;Logistic 模型;神经网络
中图法分类号 TP301.6

Sine Cosine Algorithm Based on Logistic Model and Stochastic Differential Mutation

XU Ming[1],JIAO Jian-jun[1] and LONG Wen[2]
1 School of Mathematics & Statistics,Guizhou University of Finance and Economics,Guiyang 550025,China
2 Guizhou Key Laboratory of Economics System Simulation,Guizhou University of Finance and Economics,Guiyang 550025,China

Abstract In view of the slow convergence speed,easy to fall into local optimum and low precision of the standard sine cosine algorithm,an improved sine cosine algorithm (LS-SCA) with the nonlinear conversion parameter and the stochastic differential mutation strategy was proposed to solve global optimization problems. Firstly,a nonlinear conversion parameter based on Logistic model is designed to balance between global exploration and local exploitation. Secondly,a stochastic differential mutation strategy is introduced to maintain the diversity of population and avoid falling into the optimal value. Finally,the nonlinear conversion parameter and stochastic differential mutation strategies are fused. On the one hand,12 standard test functions are selected for global optimization experiments. The results show that LS-SCA is superior to the other SCAs and comparison latest algorithms in convergence accuracy and convergence speed with the same number of fitness function evaluations. Stochastic differential mutation strategy can improve LS-SCA's global optimization ability especially. On the other hand,LS-SCA is used to optimize the parameters of neural network to solve two classical classification problems. Compared with the traditional BP algorithm and the other intelligent algorithms,the neural network based on LS-SCA can achieve higher classification accuracy.

Keywords Sine cosine algorithm, Nonlinear conversion parameter, Stochastic differential mutation, Logistic model, Neural network

到稿日期:2018-11-28 返修日期:2019-04-26 本文已加入开放科学计划(OSID),请扫描上方二维码获取补充信息。
基金项目:国家自然科学基金(61463009,11761019,11361014);贵州省高校科技拔尖人才支持计划(黔教合 KY 字[2017]070);贵州省微分-差分动力系统应用科技创新人才团队(20175658)
This work was supported by the National Natural Science Foundation of China (61463009,11761019,11361014), Program for the Science and Technology Top Talents of Higher Learning Institutions of Guizhou (KY[2017]070) and Guizhou Differential-Differential Dynamic System Innovation Talents Team (20175658).
通信作者:龙文(lw227@mail.gufe.edu.cn)

1　引言

优化问题普遍存在于科学研究、经济管理、工程设计等众多领域中。近年来，元启发式算法被广泛应用于求解优化问题[1-3]。受数学中三角函数概念的启发，澳大利亚学者 Mirjalili[4]提出了一种新型元启发式算法——SCA。研究结果表明[4]：SCA 在函数优化方面比遗传算法（Genetic Algorithm，GA）、粒子群优化（Particle Swarm Optimization，PSO）、蝙蝠算法（Bat Algorithm，BA）和花朵授粉算法（Flower Pollination Algorithm，FPA）更有竞争力。该算法由于原理简单、容易编程实现、需调节的参数少，自提出以来，受到了众多研究学者的广泛关注，在生物局部序列比对[5]、电力系统的短期热液调度[6]、特征选择[7]、高维全局优化[8]、参数优化[9]、电力潮流优化[10]等领域中有着成功的应用。

但是，标准 SCA 存在收敛速度慢、易陷入局部最优和收敛精度低的缺点。对此，研究学者提出了许多策略来改善标准 SCA 的性能。Issa 等[5]提出一种自适应 SCA（ASCA-PSO），用于求解生物局部序列比对问题，ASCA-PSO 算法利用 PSO 增强了局部搜索能力；Elaziz 等[11]提出了一种基于反向学习策略的改进 SCA（OBSCA），用于求解全局优化问题，OBSCA 能有效地改善标准 SCA 的全局搜索能力；刘勇等[12]基于抛物线函数和指数函数，提出了转换参数非线性递减的 SCA，该算法能较好地平衡全局搜索和局部开发能力；张校非等[13]提出一种改进的 SCA（DIW-SCA），用于求解函数优化问题，该方法引入动态惯性权重以平衡全局与局部搜索能力，引入自适应变异算子以增强种群的多样性；Rizk-Allah[14]提出了一种基于多正交搜索策略的 SCA（MOSCA），该算法分为两个阶段，SCA 阶段的作用是提高全局勘探能力，而多正交搜索阶段以增强局部开采能力为主；Nenavath 等[15]提出了一种基于 SCA 和差分进化的混合算法，用于求解全局优化和目标检测，该混合算法具有较好的跳出局部最优和快速收敛的能力。

在标准 SCA 中，转换参数 r_1 在平衡算法全局搜索能力和局部开发能力方面起着关键作用[4]。然而算法中转换参数 r_1 随搜索迭代次数的增加从 a 线性递减至 0。在实际优化过程中，由于算法搜索过程极复杂，将转换参数 r_1 设置为线性递减策略不能适应实际搜索情况，在解决多峰值函数优化问题时尤甚。本文提出了一种改进的正弦余弦算法（LS-SCA）。标准测试函数的实验结果表明，LS-SCA 的性能具有较强的竞争力。本文的主要贡献如下：

（1）提出一种改进的 SCA，它保留了基本 SCA 的框架。

（2）分析了 SCA 的搜索过程中转换参数 r_1 的变化趋势，其符合 Logistic 模型，并设计了一种基于 Logistic 模型的非线性转换参数 r_1 策略。

（3）提出一种随机差分变异策略，以降低算法陷入局部最优的概率。

2　正弦余弦算法

在基本 SCA 中，首先在搜索空间中随机产生一些个体组成初始种群；然后在每次迭代中，群体中的个体均按式（1）更新位置[4]：

$$X(t+1)=\begin{cases}X(t)+r_1\sin(r_2)\cdot|r_3X^*-X(t)|, & r_4<0.5\\ X(t)+r_1\cos(r_2)\cdot|r_3X^*-X(t)|, & r_4\geqslant 0.5\end{cases}\tag{1}$$

其中，t 为当前迭代次数；X^* 为当前最优个体位置；r_2，r_3 和 r_4 为 3 个随机参数，r_1 称为转换参数，随迭代次数的增加从 a 线性递减至 0，即：

$$r_1=a-a\frac{t}{t_{\max}}\tag{2}$$

其中，$a>0$ 为常数，$t_{\max}$ 为最大迭代次数。最后，循环迭代至算法满足终止条件。

SCA 的迭代原理如图 1 所示。可以看出，当 $r_1\sin(r_2)$ 或 $r_1\cos(r_2)$ 的值在区间（1，2］或［−2，−1）之内时，SCA 进行全局搜索；当 $r_1\sin(r_2)$ 或 $r_1\cos(r_2)$ 的值在区间［−1，1］之内时，SCA 进行局部开发。

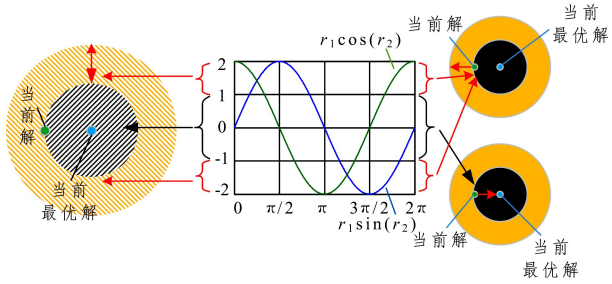


图 1　SCA 优化原理示意图

Fig. 1　Optimization principle of SCA

3　改进的正弦余弦算法

3.1　基于 Logistic 模型的转换参数策略

SCA 是基于种群进化的元启发式算法，具有较好性能的关键是平衡全局搜索和局部开发能力。由图 1 可知，转换参数 r_1 的设置直接影响了 SCA 的全局搜索能力和局部开发能力的平衡。在基本 SCA 中，转换参数 r_1 被设置为随迭代次数增加从 a 线性递减至 0。这种线性递减方式导致算法易陷入局部最优，在求解多峰值函数优化问题时尤甚。

对基于种群进化的元启发式优化算法来说，性能较好的算法要求在搜索初期具有较强的全局搜索能力，能在解空间中搜索出更多的潜在全局最优点；而在搜索末期则应具有较强的局部开发能力，以提高算法的求解精度并加快收敛速度。由 SCA 的原理可知，在进化初期，较大的转换参数 r_1 值使得搜索步长较大，群体搜索范围较广，可维持群体中个体的多样性，使算法的全局搜索能力增强，易于找到全局最优值，但收敛速度较慢。为了加快收敛，此时转换参数 r_1 值减小的速度应较大。在进化末期，较小的转换参数 r_1 值使得搜索步长较小，群体集中在某个较小区域搜索，使算法局部开发能力增强，但群体多样性较差，易陷入局部最优。为了降低算法陷入局部最优的概率，此时转换参数 r_1 值减小的速度应趋缓。

假设转换参数 r_1 的最大值和最小值分别为 $r_{1,\max}$ 和 $r_{1,\min}$。如果进化搜索开始时，r_1 的衰减率为 b，随着算法迭代次数的不断增加，衰减率逐渐减小，当 r_1 值减小到 $r_{1,\min}$ 时，r_1 停止减小，即衰减率为 0。因此，转换参数 r_1 值的变化规律符合 Logistic 模型，其数学表达式为：

$$\begin{cases}\dfrac{\mathrm{d}r_1(t)}{\mathrm{d}t}=b\cdot\left(1-\dfrac{r_1(t)}{r_{1,\min}}\right)\cdot r_1(t)\\ r_1(0)=r_{1,\max}\end{cases}\tag{3}$$

使用分离变量法对式(3)进行求解，可得到转换参数 r_1 的动态调整公式：

$$r_1(t)=\frac{r_{1,\min}}{1+\left(\dfrac{r_{1,\min}}{r_{1,\max}}-1\right)\cdot \mathrm{e}^{-bt}}\tag{4}$$

其中，t 为当前迭代次数，参数 b 为初始衰减率，调整参数 b 值可调整曲线的曲率，即调整转换参数 r_1 的下降速度。初始衰减率 b 的值越大，在搜索初期 b 值的下降速度越快；反之，b 值的下降速度较慢。当 $t=0$ 时，$r_1=r_{1,\max}$；而当 $t\to\infty$ 时，易证 $r_1=r_{1,\min}$。

3.2 随机差分变异策略

在基本 SCA 中，由式(1)可知，群体中个体的更新方式是在当前个体和当前群体最优解 X^* 附近产生新个体。也就是说，群体中个体由当前最优个体 X^* 引导向最优解靠拢。如果 X^* 是局部最优解，随着进化搜索的进行，个体均聚集在局部最优解 X^* 附近，将损失群体多样性，从而使得算法陷入局部最优，收敛精度不高。解决此问题的一般方法是加入变异操作，以增强群体的多样性。常用的变异算子有均匀变异、高斯变异、柯西变异、多样性变异等。

本文提出一种新的变异策略，即随机差分变异策略。其利用当前个体、当前最优个体和在群体中随机选择的个体进行随机差分得到新的个体，具体表达式为：

$$X(t+1)=r\times(X^*-X(t))+r\times(X'(t)-X(t))\tag{5}$$

其中，r 为[0,1]的随机数，t 为当前迭代次数，X^* 为当前最优个体位置，X'为从群体中随机选取的个体位置。每次迭代都采用随机差分变异策略(即式(5))对群体进行扰动，产生新的个体，从而帮助群体降低陷入局部最优的概率，防止早熟现象的发生。

LS-SCA 的流程图如图 2 所示。

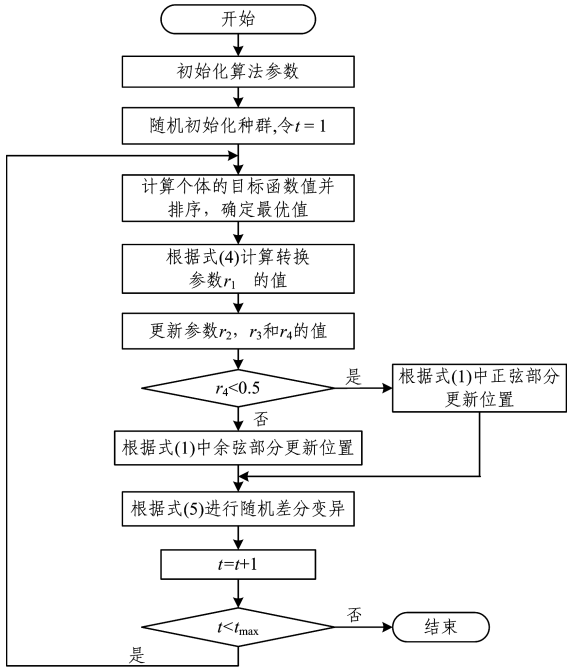


图 2 LS-SCA 的流程图

Fig. 2 Flow chart of LS-SCA

4 仿真实验及分析

4.1 标准测试函数及参数设置

为了验证 LS-SCA 的寻优性能，本文选取 12 个国际通用的标准测试函数进行数值实验，并将其与标准 SCA、改进 SCA 和其他元启发式算法进行比较。12 个测试函数的具体表达式及搜索区间如表 1 所列。在 12 个测试函数中，f_1-f_6 是单峰函数，f_7-f_{12} 为多峰函数，所有测试函数的全局最优值均为 0。

表 1 标准测试函数

Table 1 Benchmark test functions

函数表达式	搜索区间
$f_1(x)=\sum_{i=1}^{D}x_i^2$	[−100,100]
$f_2(x)=\sum_{i=1}^{D}\lvert x_i\rvert+\prod_{i=1}^{D}\lvert x_i\rvert$	[−10,10]
$f_3(x)=\sum_{i=1}^{D}\lvert x_i\rvert^{(i+1)}$	[−1,1]
$f_4(x)=\sum_{i=1}^{D}ix_i^4$	[−1.28,1.28]
$f_5(x)=\sum_{i=1}^{D}ix_i^2$	[−10,10]
$f_6(x)=\sum_{i=1}^{D}(10^6)^{(i-1)/(D-1)}x_i^2$	[−100,100]
$f_7(x)=\sum_{i=1}^{D}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]+\sin^2(3\pi x_1)+\lvert x_D-1\rvert[1+\sin^2(3\pi x_D)]$	[−10,10]
$f_8(x)=\sum_{i=1}^{D-1}(x_i^2+2x_{i+1}^2)^{0.25}\cdot((\sin 50(x_i^2+x_{i+1}^2)^{0.1})^2+1)$	[−10,10]
$f_9(x)=\frac{1}{4\,000}\sum_{i=1}^{D}x_i^2-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[−600,600]
$f_{10}(x)=\sum_{i=1}^{D}\lvert x_i\cdot\sin(x_i)+0.1\cdot x_i\rvert$	[−10,10]
$f_{11}(x)=0.1D-(0.1\sum_{i=1}^{D}\cos(5\pi x_i)-\sum_{i=1}^{D}x_i^2)$	[−1,1]
$f_{12}(x)=0.1(\sin^2(3\pi x_1)+\sum_{i=1}^{D-1}(x_i-1)^2(1+\sin^2(3\pi x_{i+1}))+(x_D-1)^2(1+\sin^2(2\pi x_D)))$	[−100,100]

基本 SCA 和 LS-SCA 的参数设置如下：种群规模 $N=30$，最大迭代次数为 $t_{max}=500$；在基本 SCA 中，常数 $a=2$；在 LS-SCA 中，转换参数的最大值和最小值分别设置为 $r_{1,max}=2$ 和 $r_{1,min}=1$，衰减系数 $b=10$。所有仿真实验均在 Intel Core Quad，CPU：Q8300、2 GB 内存、2.50 GHz 主频的计算机上实现，程序采用 MATLAB 2014a 语言实现。

4.2 LS-SCA 与基本 SCA 和其他改进 SCA 的比较

本文采用所提的 LS-SCA 对表 1 中的 12 个标准测试函数进行求解，并与基本 SCA、转换参数抛物线递减 SCA（PSCA）[12] 和动态惯性权重 SCA（DIW-SCA）[13] 进行比较。为了公平起见，4 种算法采用相同的适应度函数评价次数，即 15000 次（种群规模为 30，最大迭代次数为 500），PSCA 和 DIW-SCA 的其他参数详见各自参考文献。12 个测试函数的维数均设置为 $D=30$。4 种算法对 12 个标准测试函数单独运行 30 次，表 2 比较了它们的平均值、标准差和基于平均值的综合排名。

表 2　LS-SCA，SCA，PSCA 和 DIW-SCA 求解 12 个测试函数（$D=30$）的比较结果

Table 2　Comparison results of LS-SCA，SCA，PSCA and DIW-SCA for solving 12 test functions ($D=30$)

函数	SCA		PSCA		DIW-SCA		LS-SCA	
	平均值	标准差	平均值	标准差	平均值	标准差	平均值	标准差
$f_1(x)$	4.34×10^{-3}	5.43×10^{-3}	3.28×10^{-2}	5.89×10^{-2}	0	0	0	0
$f_2(x)$	1.50×10^{-5}	1.78×10^{-5}	9.29×10^{-5}	5.56×10^{-5}	1.42×10^{-263}	0	0	0
$f_3(x)$	2.21×10^{-6}	1.47×10^{-6}	4.68×10^{-6}	4.30×10^{-6}	0	0	0	0
$f_4(x)$	2.48×10^{-5}	4.12×10^{-5}	6.37×10^{-5}	1.08×10^{-4}	0	0	0	0
$f_5(x)$	6.86×10^{-5}	1.05×10^{-4}	1.64×10^{-5}	1.58×10^{-5}	0	0	0	0
$f_6(x)$	2.27×10^{-2}	1.98×10^{-2}	3.04×10^{-2}	3.78×10^{-2}	0	0	0	0
$f_7(x)$	4.52×10^{-3}	4.91×10^{-3}	7.78×10^{-3}	5.02×10^{-3}	0	0	0	0
$f_8(x)$	1.07×10^{-1}	1.30×10^{-1}	1.15×10^{-1}	9.65×10^{-2}	0	0	0	0
$f_9(x)$	3.15×10^{-1}	3.53×10^{-1}	5.57×10^{-1}	3.20×10^{-1}	0	0	0	0
$f_{10}(x)$	3.61×10^{-2}	2.94×10^{-2}	3.93×10^{-2}	2.05×10^{-2}	3.05×10^{-262}	0	0	0
$f_{11}(x)$	1.19×10^{-6}	2.72×10^{-6}	1.08×10^{-5}	3.76×10^{-5}	0	0	0	0
$f_{12}(x)$	1.25×10^{-6}	2.87×10^{-6}	2.45×10^{-6}	2.55×10^{-6}	0	0	0	0
平均排名	3.0833		3.9167		1.1667		1.0000	
最终排名	3		4		2		1	

从表 2 中结果可知，LS-SCA 对 12 个测试函数的求解结果明显优于标准 SCA 和 PSCA。与 DIW-SCA 相比，LS-SCA 在两个函数（f_2 和 f_{10}）上获得了较好的结果，而在其他 10 个函数上得到了相似的寻优结果。基于表 2 中平均值数据对各算法进行综合排名可以看出，LS-SCA 性能最优，DIW-SCA 其次，而 PSCA 最差。

图 3 给出了 LS-SCA 与标准 SCA，PSCA 和 DIW-SCA 求解 12 个测试函数的收敛曲线比较。

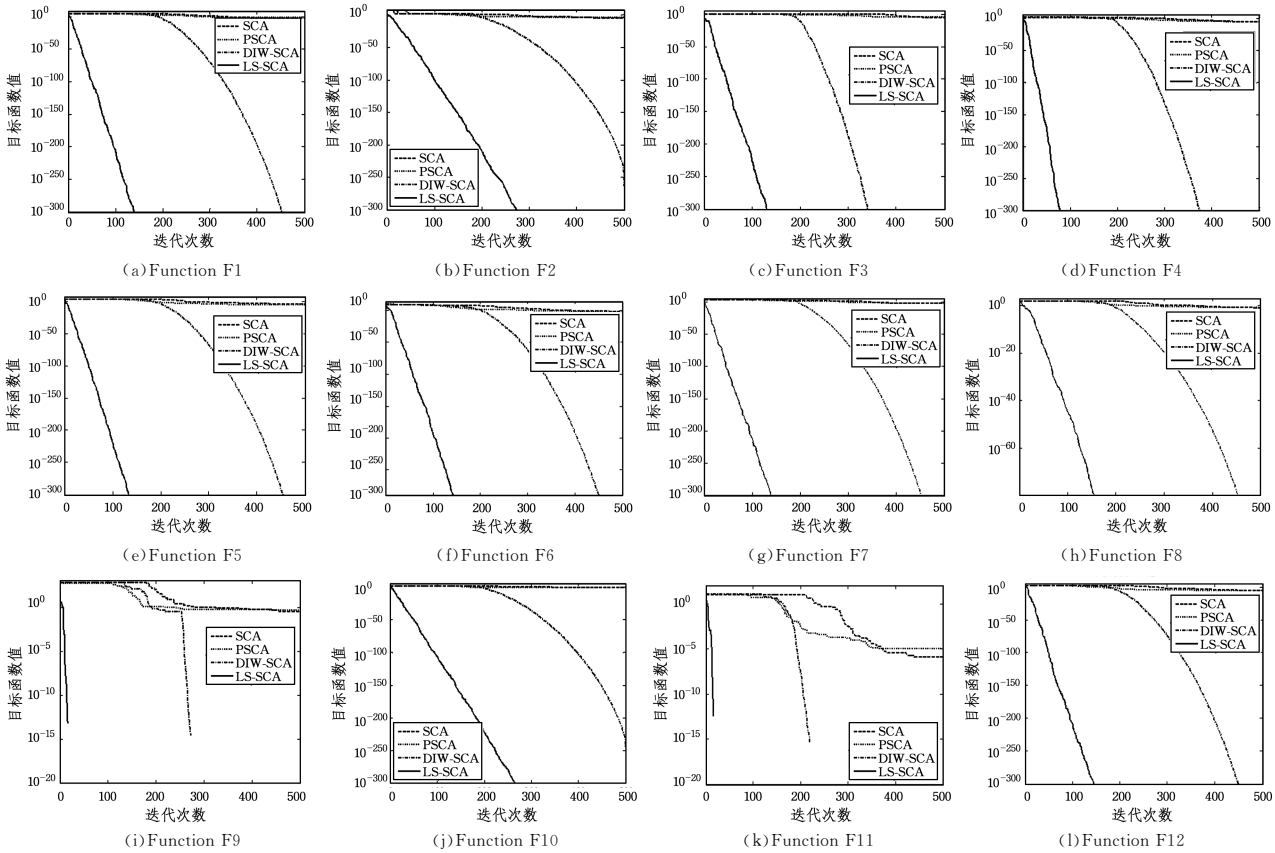


图 3　LS-SCA，SCA，PSCA 和 DIW-SCA 求解 12 个测试函数的收敛曲线（$D=30$）

Fig. 3　Convergence curves of LS-SCA，SCA，PSCA and DIW-SCA for solving 12 test functions ($D=30$)

从图 3 可以清晰地看出，LS-SCA 比标准 SCA 和 PSCA 具有更快的收敛速度和更高的收敛精度。虽然 DIW-SCA 和 LS-SCA 在大部分函数上获得了相似的实验结果，但 LS-SCA 的收敛速度明显更快。

4.3 与其他群体智能算法的比较

为了进一步验证算法的有效性，将 LS-SCA 与 7 种其他群体智能优化算法进行比较，即协方差矩阵自适应进化策略(Covariance Matrix Adaptation Evolution Strategy，CMA-ES)[16]、非线性权重粒子群优化(Nonlinear Weight PSO，NW-PSO)算法[17]、全局指导搜索人工蜂群(Gbest-Guided Artificial Bee Colony，GABC)算法[18]、精英教与学优化(Elitist Teaching-Learning-Based OPtimization，ETLBO)算法[19]、反向差分进化(Opposite-Based Differential Evolution，ODE)算法[20]、修改灰狼优化(Modified Grey Wolf Optimizer，MGWO)算法[21]和改进鲸鱼优化算法(Improved Whale Optimization Algorithm，IWOA)[22]。

为了比较的客观性，8 种算法采用相同的适应度评价次数 15000，即种群规模为 30，最大迭代次数为 500。表 3 比较了 8 种算法分别对 12 个测试函数独立实验 30 次的平均值、标准差值和基于平均值的综合排名；基于以上实验，表 4 进一步给出了 LS-SCA 与其他 7 种算法进行比较的 Wilcoxon 秩和检验结果。

表 3 8 种算法求解 12 个测试函数($D=30$)的比较结果

Table 3 Comparison results of eight algorithms for solving 12 test functions ($D=30$)

函数	统计结果	CMA-ES	NW-PSO	GABC	ETLBO	ODE	MGWO	IWOA	LS-SCA
$f_1(x)$	平均值	8.64×10^{-11}	2.82×10^{-16}	4.00×10^{-16}	2.70×10^{-119}	2.68×10^{-49}	9.64×10^{-43}	0	0
	标准差	3.83×10^{-11}	5.07×10^{-16}	3.76×10^{-16}	4.29×10^{-119}	2.50×10^{-49}	1.24×10^{-42}	0	0
$f_2(x)$	平均值	2.03×10^{-5}	4.03×10^{-3}	2.59×10^{-7}	1.19×10^{-60}	3.86×10^{-31}	4.11×10^{-25}	2.77×10^{-267}	0
	标准差	1.15×10^{-5}	8.05×10^{-3}	1.98×10^{-8}	5.87×10^{-61}	4.00×10^{-31}	2.65×10^{-25}	0	0
$f_3(x)$	平均值	3.91×10^{-10}	2.17×10^{-34}	4.02×10^{-42}	3.26×10^{-278}	8.51×10^{-149}	1.81×10^{-150}	0	0
	标准差	4.00×10^{-10}	4.21×10^{-34}	6.81×10^{-42}	0	8.47×10^{-149}	3.00×10^{-150}	0	0
$f_4(x)$	平均值	7.31×10^{-27}	1.87×10^{-17}	8.86×10^{-29}	3.91×10^{-235}	1.57×10^{-77}	3.60×10^{-79}	0	0
	标准差	1.00×10^{-26}	2.99×10^{-17}	6.19×10^{-29}	0	3.50×10^{-77}	4.47×10^{-79}	0	0
$f_5(x)$	平均值	1.32×10^{-11}	1.73×10^{-12}	6.89×10^{-17}	1.76×10^{-120}	5.11×10^{-49}	4.06×10^{-43}	0	0
	标准差	6.16×10^{-12}	1.34×10^{-12}	6.75×10^{-17}	1.63×10^{-120}	1.12×10^{-48}	7.81×10^{-43}	0	0
$f_6(x)$	平均值	2.89×10^{-3}	5.28×10^{-8}	1.86×10^{-12}	1.18×10^{-115}	1.91×10^{-44}	1.51×10^{-39}	0	0
	标准差	2.58×10^{-3}	6.99×10^{-8}	1.02×10^{-12}	1.30×10^{-115}	2.23×10^{-44}	2.05×10^{-39}	0	0
$f_7(x)$	平均值	4.59×10^{-5}	1.19×10	4.58×10^{-15}	2.62×10^{-120}	2.38×10^{-49}	3.27×10^{-43}	0	0
	标准差	4.27×10^{-5}	8.43	6.63×10^{-15}	3.04×10^{-120}	3.10×10^{-49}	3.71×10^{-43}	0	0
$f_8(x)$	平均值	1.49×10^{-1}	1.53×10	8.06×10^{-2}	2.08×10^{-28}	3.51×10^{-15}	1.04×10^{-11}	0	0
	标准差	5.87×10^{-3}	6.60	1.85×10^{-2}	1.35×10^{-28}	2.45×10^{-15}	6.24×10^{-12}	0	0
$f_9(x)$	平均值	6.60×10^{-11}	2.33×10^{-1}	6.21×10^{-16}	0	2.44×10^{-16}	0	0	0
	标准差	1.32×10^{-11}	1.30×10^{-1}	9.63×10^{-16}	0	1.45×10^{-16}	0	0	0
$f_{10}(x)$	平均值	9.71×10^{-6}	6.32×10^{-5}	1.35×10^{-6}	2.21×10^{-61}	9.53	2.61×10^{-22}	7.28×10^{-262}	0
	标准差	2.94×10^{-6}	7.25×10^{-5}	2.42×10^{-6}	2.97×10^{-61}	7.79	5.75×10^{-22}	0	0
$f_{11}(x)$	平均值	3.95×10^{-13}	3.84×10^{-1}	9.94×10^{-19}	3.59×10^{-123}	1.14×10^{-51}	0	0	0
	标准差	6.63×10^{-14}	8.09×10^{-2}	8.56×10^{-19}	6.83×10^{-123}	9.16×10^{-52}	0	0	0
$f_{12}(x)$	平均值	2.01×10^{-1}	8.79×10^{-3}	3.13×10^{-19}	6.66×10^{-121}	8.83	5.23×10^{-46}	0	0
	标准差	1.79×10^{-1}	4.91×10^{-3}	2.69×10^{-19}	1.47×10^{-120}	1.23×10	7.56×10^{-46}	0	0
平均排名		7.2500	7.3333	5.9167	2.9167	5.0000	4.1667	1.1667	1.00
最终排名		7	8	6	3	5	4	2	1

表 4 LS-SCA 与其他 7 种算法的统计检验结果比较

Table 4 Statistical test results of LS-SCA and other seven algorithms

算法	R^+	R^-	p-value	$\alpha=0.05$	$\alpha=0.1$
LS-SCA vs CMA-ES	78.0	0.0	1.0269×10^{-5}	Yes	Yes
LS-SCA vs NW-PSO	78.0	0.0	1.0269×10^{-5}	Yes	Yes
LS-SCA vs GABC	78.0	0.0	1.0269×10^{-5}	Yes	Yes
LS-SCA vs ETLBO	72.0	6.0	3.7585×10^{-5}	Yes	Yes
LS-SCA vs ODE	78.0	0.0	1.0269×10^{-5}	Yes	Yes
LS-SCA vs MGWO	66.5	11.5	1.2531×10^{-4}	Yes	Yes
LS-SCA vs IWOA	40.5	37.5	0.1662	No	No

从表 3 可知，与 CMA-ES，NW-PSO，GABC 和 ODE 算法相比，LS-SCA 在 12 个测试函数上均获得了较好的平均值和标准差值。与 ETLBO 和 MGWO 算法相比，LS-SCA 分别在 11 个和 10 个测试函数上获得了较好的结果；然而，ETLBO 和 MGWO 算法分别在 1 个测试函数(f_9)和 2 个测试函数(f_9 和 f_{11})上取得了较好的平均值和标准差值。与 IWOA 相比，LS-SCA 在两个测试函数(f_2 和 f_{10})上获得了较好的实验结果，在其余 10 个测试函数上得到了相似的平均值和标准差值。基于表 3 中平均值数据对各算法进行综合排名可以看出，LS-SCA 总体性能上最优，IWOA 排名第二，NW-PSO 算法性能最差。

从表 4 中多问题 Wilcoxon 秩和检验结果可知，在所有情况下，LS-SCA 所获得的 R^+ 值均大于 R^- 值。除了 LS-SCA vs IWOA 情形外，其他情形的 p 值均小于 0.05，说明 LS-SCA 的性能明显优于 CMA-ES，NW-PSO，GABC，ETLBO，ODE 和 MGWO。另外，LS-SCA vs IWOA 情形的 p 值大于 0.1，说明 LS-SCA 和 IWOA 的算法性能没有明显的差异性。

从表 3、表 4 不难看出，与其他几类最近提出的改进算法相比，无论对于单峰函数(f_1-f_6)还是多峰函数(f_7-f_{12})，LS-SCA 在寻优计算上均有较好的表现。

4.4 优化 MLP 网络参数

多层感知器(Multilayer Perceptron，MLP)是一种最流行和最常见的神经网络类型，其结构如图 4 所示。

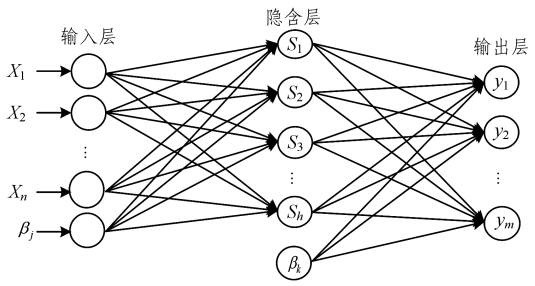


图 4　MLP 神经网络的结构

Fig. 4　Structure of neural network

利用式(6)计算输入层到隐含层的权重和：

$$s_j=\sum_{i=1}^{n}(w_{ij}\cdot X_i)+\beta_j,j=1,2,\cdots,h \tag{6}$$

其中，n 是输入节点的数量，w_{ij} 是连接输入层中第 i 个节点到隐含层中第 j 个节点的权重，β_j 是第 j 个隐含节点的偏差，h 是隐含节点的数量。

每个隐含节点的输出计算式如下：

$$S_j=sigmoid(s_j)=\frac{1}{1+\exp(-s_j)} \tag{7}$$

网络的最终输出可表示为：

$$y_k=\frac{1}{1+\exp(-\sum_{j=1}^{h}(W_{jk}\cdot S_j)+\beta_k)},k=1,2,\cdots,m \tag{8}$$

其中，y_k 表示第 k 个输出，m 为输出节点的数量；W_{jk} 表示连接隐含层中第 j 个节点到输出层中第 k 个节点的权重；β_k 为第 k 个输出节点的偏差。

由式(6)－式(8)可以看出，权重和偏差根据给定的输入获得 MLP 网络的最终输出。因此，应找到一组合适的权重和偏差值，以提高 MLP 网络的分类或预测准确度。本节采用 LS-SCA 优化 MLP 网络的权重和偏差值，利用 LS-SCA 优化 MLP 网络的参数，其适应度函数通常选取均方误差(Mean Square Error，MSE)：

$$MSE=\frac{1}{M}\sum_{p=1}^{M}(y-\hat{y})^2 \tag{9}$$

其中，y 表示实际值，$\hat{y}$ 表示 MLP 网络的预测值，M 表示训练数据集的样本数。

利用 LS-SCA 优化 MLP 参数得到的优化后的网络，称为 LS-SCA-MLP 网络。利用 LS-SCA-MLP 对 Wine 和 Ionosphere 分类数据集进行实验，并将其与其他 7 种优化算法进行比较，即反向传播法(Back Propagation，BP)、GA、PSO、差分进化(Differential Evolution，DE)、进化策略(Evolution Strategy，ES)、鲸鱼优化算法(Whale Optimization Algorithm，WOA)和 SCA。为了比较的客观性，所有算法的迭代次数为 250，种群规模为 50。Wine 数据集有 13 个特征，MLP 网络结构设置为 13-27-1；Ionosphere 数据集有 33 个特征，MLP 网络结构设置为 33-67-1[23]。

表 5 列出了 8 种算法对 Wine 和 Ionosphere 数据集的分类准确率的最优值、平均值和标准差。需要注意的是，除了 SCA 和 LS-SCA，其他 6 种算法的结果直接来源于文献[23]。

表 5　8 种算法对 Wine 和 Ionosphere 数据集的分类准确率

Table 5　Classification accuracy of eight algorithms for Wine and Ionosphere data sets

算法	Wine 数据集			Ionosphere 数据集		
	最优值	平均值	标准差	最优值	平均值	标准差
BP	0.9091	0.7697	0.1153	0.7833	0.7367	0.00385
GA	0.9394	**0.8894**	0.0580	0.8250	0.8025	0.0157
PSO	0.8939	0.8227	0.0474	0.7917	0.7600	0.0242
DE	0.8788	0.7576	0.0763	0.8417	0.7767	0.0340
ES	0.8333	0.7515	0.0436	0.7917	0.7358	0.0281
WOA	0.9545	**0.8894**	0.0335	0.8667	0.7942	0.0429
SCA	0.8333	0.8000	0.0203	0.8205	0.7385	0.0627
LS-SCA	**0.9630**	0.8852	0.0843	**0.8718**	**0.8085**	0.0367

由表 7 的比较结果可知，对于 Wine 数据集，LS-SCA 获得了最高的分类准确率(0.9630)，而 GA 和 WOA 得到了较好的平均值(0.8894)。与 GA 和 WOA 相比，LS-SCA 的分类准确率平均值(0.8852)稍差；与其他算法相比，LS-SCA 具有较高的分类准确率。对于 Ionosphere 数据集，与其他 7 种算法相比，LS-SCA 获得了最高的分类准确率(0.8718)和平均值(0.8085)。

结束语　本文提出了一种 LS-SCA，用于求解全局优化问题。首先，分析了转换参数对 SCA 性能的影响，基于 Logistic 模型设计了一种非线性转换参数策略，用于协调算法的全局搜索能力和局部开发能力；然后，利用随机差分变异策略产生较好的多样性个体，以降低算法陷入局部最优的概率。对 12 个标准测试函数进行仿真实验，结果表明，LS-SCA 能克服标准 SCA 的缺点。本文还分析了两个改进策略的有效性，即基于 Logistic 模型的非线性转换参数策略对 SCA 改进的帮助有限，而随机差分变异策略能帮助 SCA 跳出局部最优。最后，利用 LS-SCA 优化 MLP 神经网络参数并对 Wine 和 Ionosphere 数据集进行分类，获得了较好的结果。下一步的研究方向是将 LS-SCA 应用于约束优化、多目标优化及工程应用中。

参 考 文 献

[1] ZHANG M，TIAN N，PALADE V，et al. Cellular artificial bee colony algorithm with Gaussian distribution[J]. Information Sciences，2018，462：374-401.

[2] LI J，LUO Y，LI B，et al. Differential hybrid particle swarm optimization algorithm based on different dimensional variation[J]. Computer Science，2018，45(5)：208-214.

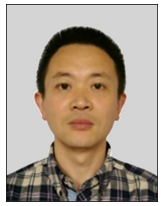
[3] PAROUHA R P，DAS K N. A memory based differential evolution algorithm for unconstrained optimization[J]. Applied Soft Computing，2016，38：501-517.

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

[5] 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.

[6] DAS S，BHATTACHARYA A，CHAKRABOTY A K. Solution of short-term hydrothermal scheduling using sine cosine algo-

rithm[J]. Soft Computing, 2018, 22(19): 6409-6427.
[7] SINDHU R, NGADIRAN R, YACOB Y M, et al. Sine-cosine algorithm for feature selection with elitism strategy and new updating mechanism[J]. Neural Computing & Applications, 2017, 28(10): 2947-2958.
[8] LONG W, WU T, LIANG X, et al. Solving high-dimensional global optimization problems using an improved sine cosine algorithm[J]. Expert Systems with Applications, 2019, 123: 108-126.
[9] LI S, FANG H, LIU X. Parameter optimization of support vector regression based on sine cosine algorithm[J]. Expert Systems with Applications, 2018, 91: 63-77.
[10] ATTIA A F, SEHIEMY R A E, HASANIEN H M. Optimal power flow solution in power systems using a novel sine-cosine algorithm[J]. Journal of Electrical Power & Energy Systems, 2018, 99: 331-343.
[11] ELAZIZ M A, OLIVA D, XIONG S. An improved opposition-based sine cosine algorithm for global optimization[J]. Expert Systems with Applications, 2017, 90: 484-500.
[12] LIU Y F, MA L P. Sine cosine algorithm with nonlinear decreasing conversion parameter[J]. Computer Engineering and Applications, 2017, 53(2): 1-5.
[13] ZHANG X F, BAI Y P, HAO Y, et al. Research of improved sine cosine algorithm in function optimization[J]. Journal of Chongqing University of Technology (Natural Science), 2017, 31(2): 146-152.
[14] RIZK-ALLAH R M. Hybridizing sine cosine algorithm with multi-orthogonal search strategy for engineering design problems[J]. Journal of Computational Design and Engineering, 2018, 5(2): 249-273.
[15] NENAVAH H, JATOTH R K. Hybridizing sine cosine algorithm with differential evolution for global optimization and object tracking[J]. Applied Soft Computing, 2018, 62: 1019-1043.
[16] HANSEN N, MÜLLER S D, KOUMOUTSAKOS P. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation[J]. Evolutionary Computation, 2014, 11(1): 1-18.
[17] CHATTERJEE A, SIARRY P. Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization[J]. Computers & Operations Research, 2006, 33(3): 859-871.
[18] ZHU G, KWONG S. Gbest-guided artificial bee colony algorithm for numerical function optimization[J]. Applied Mathematics and Computation, 2010, 217(7): 3166-3173.
[19] RAO R V, PATEL V. An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems[J]. Journal of Industrial Engineering Computations, 2012, 3(4): 535-560.
[20] RAHNAMAYAN S, TIZHOOSH H R, SALAMA M M A. Opposition based differential evolution[J]. IEEE Transactions on Evolutionary Computation, 2008, 12(1): 64-79.
[21] RODRÍGUEZ L, CASTILLO O, SORIA J, et al. A fuzzy hierarchical operator in the grey wolf optimizer algorithm [J]. Applied Soft Computing, 2017, 57: 315-328.
[22] HU H, BAI Y, XU T. Improved whale optimization algorithm based on inertia weights and theirs applications [J]. International Journal of Circuits, Systems and Signal Processing, 2017, 11: 12-26.
[23] ALJARAH I, FARIS H, MIRJALILI S. Optimizing connection weights in neural networks using the whale optimization algorithm[J]. Soft Computing, 2018, 22(1): 1-15.



XU Ming, born in 1976, Ph. D, professor, is member of China Computer Federation (CCF). His main research interests include machine learning and intelligent computing.

LONG Wen, born in 1977, Ph.D, professor, Ph.D supervisor. His main research interests include intelligent computing and data mining.