控制与决策
Control and Decision

多尺度正余弦优化算法

申元霞, 张学锋, 方馨, 汪小燕

引用本文:
申元霞,张学锋,方馨,汪小燕. 多尺度正余弦优化算法[J]. 控制与决策, 2022, 37(11): 2860-2868.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2021.0513

多尺度正余弦优化算法

申元霞†, 张学锋, 方 馨, 汪小燕

(安徽工业大学 计算机科学与技术学院，安徽 马鞍山 243000)

摘 要: 针对标准正余弦算法进化后期的停滞问题,对进化过程中的种群多样性进行分析,得出标准正余弦算法的种群多样性受控制因子的直接影响,且种群多样性表达式中控制因子指数随迭代次数的增加而下降. 为了改善标准正余弦算法进化后期的探索和开采,提出多尺度正余弦优化算法. 该算法通过自适应的多尺度控制因子调节群体多样性从而实现多层次的搜索;同时设计协助种群实施局部搜索,其种群独立进化,个体可以直接学习主群或协助种群中的最优个体,以加快收敛速度和提高解的质量. 将所提出算法与改进的正余弦算法和新型群智能算法进行对比实验,实验结果表明,所提出算法能够较好地平衡进化过程中的探索和开采,提高全局优化能力.

关键词: 正余弦算法；停滞问题；种群多样性；探索和开采；控制因子；全局优化

中图分类号: TP273 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2021.0513

引用格式: 申元霞,张学锋,方馨,等. 多尺度正余弦优化算法[J]. 控制与决策, 2022, 37(11): 2860-2868.

A multi-scale sine cosine algorithm for optimization problems

SHEN Yuan-xia†, ZHANG Xue-feng, FANG Xin, WANG Xiao-yan

(School of Computer Science and Technology，Anhui University of Technology，Ma'anshan 243000，China)

Abstract: In order to address the stagnation problem in the late stage of evolution of the standard sine cosine algorithm (SCA), this paper makes the analysis of population diversity. The analysis results show that the control factor affects directly population diversity and is decreased exponentially with increase of iterations in the expression of population diversity. In order to improve the ability of exploration and exploitation in the late stage of evolution of the SCA, a multi-scale sine cosine algorithm (MSCA) is presented. In the MSCA, an adaptive multi-scale control factor is designed to regulate population diversity for achieving the search with different layers. Meanwhile, an assisted swarm is developed to coordinate the local search for accelerating the convergence speed and improving calculation accuracy. The assisted swarm evolves independently and each individual can learn from the best experience of the mast swarm or the assisted swarm. The MSCA is evaluated on 23 benchmark functions and compared with the improved versions of the SCA and new swarm intelligence algorithms. The numerical results show that the MSCA can better coordinate the exploitation and exploration capabilities and improve the global optimization ability.

Keywords: SCA；stagnation problem；population diversity；exploration and exploitation；control factor；global optimization

0 引 言

群体智能算法是目前求解复杂优化问题的有效方法之一,特别是近20年,多种群体智能范式相继被提出,有模拟动物行为的范式,如人工蜂群(artificial bee colony, ABC)算法、灰狼优化(grey wolf optimization, GWO)算法等;有受自然界现象启发的范式,如教与学(teaching-learning-based optimization, TLBO)算法等;有受人类心理学和社会行为启发,如粒子群优化(particle swarm optimization, PSO)算法等.

2016年,Mirjalili[1]通过正弦和余弦函数的震荡性模拟群体趋于最优解的过程,提出正余弦算法(sine cosine algorithm, SCA). SCA具有收敛速度快、解的质量高等特点,已被证明在给定迭代次数下其收敛精度优于多种经典群智能算法[2],并广泛应用于目标跟踪[3]、特征选择[4]等工程领域. 但是,SCA在求解复杂问题时会陷入局部最优,特别是在进化后期会出现进化停滞现象. 为了改善该现象,文献[5]将反向学

收稿日期: 2021-03-29；**录用日期:** 2021-07-30.

基金项目: 安徽高校自然科学研究项目(KJ2019A0063)；安徽省自然科学基金项目(1808085MF196).

责任编委: 林崇.

†通讯作者. E-mail: yuanxiashen@163.com.

习引入SCA以缓解后期的进化停滞;文献[6]提出了结合PSO与Levy飞行的SCA,算法中个体有条件地选择按PSO或SCA更新方程以更新自身位置,同时对个体采用Levy飞行方式进行随机游走,使其跳出局部最优;文献[7]提出了一种混合自适应的SCA算法,其采用反向个体和在更新方程中添加扰动项的方式保持群体多样性;文献[8]提出了精英混沌搜索策略的交替正余弦算法,通过非线性对数曲线的控制参数作为调整策略平衡算法的开发和探索能力,同时利用精英个体的混沌搜索策略增强算法的开发能力;文献[9]提出了基于记忆引领SCA算法,该算法中个体不再只学习群体最优解,而是依据给定的条件可以学习个体保留的个体最优解;文献[10]提出了多策略SCA算法,该算法将突变操作、交叉操作、混沌搜索与反向学习策略融合到算法中以提高算法的全局搜索能力. 上述改进算法中采用了Levy飞行、混沌搜索、反向学习、多榜样学习等策略改善SCA进化后期的搜索能力,虽然辅助算子从一定程度上增加了群体进化的能力,但是并没有对导致SCA进化过程中出现停滞现象的原因进行理论分析.

为了探究SCA的进化停滞现象,本文首先对SCA种群多样性的进化方程进行分析,指出影响种群多样性的因素. 在理论分析的基础上,提出多尺度SCA,利用多尺度的控制因子控制群体的多样性,从而实现协调算法的探索和利用. 同时设计一个协助种群实施局部搜索,以提高算法的收敛速度和精度. 与4种改进的SCA和4种群智能算法的对比实验结果表明,多尺度SCA能够兼顾收敛速度和收敛精度,统计结果优于对比算法.

1　标准正余弦算法(SCA)

SCA利用正弦和余弦函数的数学特性实现群体的寻优过程. 与其他群体智能算法类似,SCA对下一代位置的更新也是通过对当前最优解的信息进行加工. 假设X_t为第t代种群的个体,则第$t+1$代种群的个体更新为

$$X_{t+1}(i)=\begin{cases}X_t(i)+a(t)\sin(r_1)|r_2X_t^*(i)-X_t(i)|,\\ \quad r_3<0.5;\\ X_t(i)+a(t)\cos(r_1)|r_2X_t^*(i)-X_t(i)|,\\ \quad r_3\geqslant 0.5.\end{cases}\tag{1}$$

其中: X_t^*为当前群体最优个体; r_1为服从$(0,2\pi)$均匀分布的随机数; r_2为服从$(0,2)$均匀分布的随机数,是赋予当前群体最优个体X_t^*的随机权重,用于调节X_t^*对X_t更新的影响; r_3为服从$(0,1)$均匀分布的随机数,用于选择sin函数或cos函数作为更新策略; $a(t)$为控制因子,用于控制X_t更新步长,并随着迭代次数变化,其更新方程为

$$a(t)=\lambda(1-t/T),\tag{2}$$

T为最大迭代次数. 随着迭代次数的增加,$a(t)$的值从λ递减至0, $a(t)$起着平衡全局探索和局部搜索能力的作用. 当$|a(t)\sin(r_1)|<1$或$|a(t)\cos(r_1)|<1$时,有助于下一代个体X_{t+1}趋向X_t^*运动,强调算法对最优解的利用能力;当$|a(t)\sin(r_1)|\geqslant 1$或$|a(t)\cos(r_1)|\geqslant 1$时,有助于下一代个体$X_{t+1}$背离$X_t^*$运动,加强算法的探索.

2　SCA寻优性能分析

种群多样性影响着群体智能算法的寻优特性. 种群多样性高利于全局探索,但延缓收敛;种群多样性低利于局部搜索,但易陷入停滞. 通过调控种群多样性平衡全局和局部搜索一直是研究热点. 下面将分析SCA种群多样性在群体进化过程中的变化规律. 采用种群重心定义群体多样性的方法.

定义1 [11]　第t代种群多样性定义为

$$I(X_{t+1})=\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\Big[X_{t+1}(i,j)-\frac{1}{N}\Big(\sum_{i=1}^{N}X_{t+1}(i,j)\Big)\Big]^2,\tag{3}$$

其中r_1、r_2、r_3均为随机数,第$t+1$时刻的个体X_{t+1}是随机的,因此第$t+1$时刻种群多样性$I(X_{t+1})$也是随机的. 期望效用$E(I(X_{t+1}))$用于刻画$I(X_{t+1})$的统计特征,有

$$E(I(X_{t+1}))=\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}E\Big[X_{t+1}(i,j)-\frac{1}{N}\Big(\sum_{i=1}^{N}X_{t+1}(i,j)\Big)\Big]^2.\tag{4}$$

定理1　SCA在$t+1$时刻种群多样性期望为

$$\frac{(N-1)^2}{MN^3}\sum_{i=1}^{N}\sum_{j=1}^{M}\Big(D(X_t(i,j))+\frac{1}{2}a^2(t)\times E[|r_2X_t^*(i,j)-X_t(i,j)|]^2\Big)+\frac{1}{NM}I(E(X_t(i,j))).$$

证明　由式(4)展开,得到

$$E[I(X_{t+1})]=\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\Big\{E[X_{t+1}(i,j)-EX_{t+1}(i,j)]^2-2E\Big([X_{t+1}(i,j)-$$

$$EX_{t+1}(i,j)]\Big[\frac{1}{N}\sum_{i=1}^{N}X_{t+1}(i,j)-$$
$$EX_{t+1}(i,j)\Big]\Big)+E\Big[\frac{1}{N}\sum_{i=1}^{N}X_{t+1}(i,j)-$$
$$EX_{t+1}(i,j)\Big]^2\Big\}. \tag{5}$$

对式(5)中的第2项展开计算,得到

$$E\Big([X_{t+1}(i,j)-EX_{t+1}(i,j)]\times$$
$$\Big[\frac{1}{N}\sum_{i=1}^{N}X_{t+1}(i,j)-EX_{t+1}(i,j)\Big]\Big)=$$
$$\frac{1}{N}E\Big([X_{t+1}(i,j)-EX_{t+1}(i,j)]\times$$
$$\Big[\sum_{k=1}^{N}X_{t+1}(k,j)-N\cdot EX_{t+1}(i,j)\Big]\Big)=$$
$$\frac{1}{N}\Big(\sum_{k=1,k\neq i}^{N}E([X_{t+1}(i,j)-EX_{t+1}(i,j)]\times$$
$$[X_{t+1}(k,j)-EX_{t+1}(i,j)])+$$
$$E[X_{t+1}(i,j)-EX_{t+1}(i,j)]^2\Big)=$$
$$(1/N)(E[X_{t+1}(i,j)-EX_{t+1}(i,j)]^2)=$$
$$(1/N)D(X_{t+1}(i,j)), \tag{6}$$

其中$D(X_{t+1}(i,j))$为第i个个体X_{t+1}第j维的方差.对式(5)中的第3项展开计算,得到

$$E\Big[\frac{1}{N}\sum_{k=1}^{N}X_{t+1}(k,j)-EX_{t+1}(i,j)\Big]^2=$$
$$D\Big(\frac{1}{N}\sum_{k=1}^{N}X_{t+1}(k,j)-EX_{t+1}(i,j)\Big)+$$
$$\Big(E\Big[\frac{1}{N}\sum_{k=1}^{N}X_{t+1}(k,j)-E(X_{t+1}(i,j))\Big]\Big)^2=$$
$$\frac{1}{N^2}\Big(\sum_{k=1}^{N}D(X_{t+1}(k,j))\Big)+$$
$$\Big(E\Big[X_{t+1}(i,j)-\frac{1}{N}\sum_{k=1}^{N}E(X_{t+1}(k,j))\Big]\Big)^2. \tag{7}$$

为了研究第$t+1$时刻个体X_{t+1}方差与更新方程中参数的关系,将r_1、r_3的关系记作

$$b(r_1,r_3)=\begin{cases}\sin(r_1),\ r_3>0.5;\\ \cos(r_1),\ r_3<0.5.\end{cases} \tag{8}$$

由式(1)和(8),得到

$$D(X_{t+1}(i,j))=$$
$$E((X_{t+1}(i,j))^2-(E(X_{t+1}(i,j)))^2=$$
$$E[X_t(i,j)+a(t)b(r_1,r_3)|r_2X_t^*(i,j)-X_t(i,j)|]^2-$$
$$(E(X_{t+1}(i,j)))^2. \tag{9}$$

由于

$$E(\sin(r_1))=\int_0^{2\pi}\sin r\cdot\frac{1}{2\pi}\mathrm{d}r=$$
$$\frac{1}{2\pi}(-\cos r)\,|_0^{2\pi}=0,$$

类似地,$E(\cos(r_1))=0$,得到

$$E(b(r_1,r_3))=E(\sin r_1)\cdot P\{r_3>0.5\}+$$
$$E(\cos r_1)P\{r_3<0.5\}=0, \tag{10}$$
$$D(b(r_1,r_3))=E(b^2(r_1,r_3))=$$
$$E(\sin^2 r_1)\cdot P\{r_3>0.5\}+$$
$$E(\cos^2 r_1)P\{r_3<0.5\}=0.5. \tag{11}$$

由式(10),得到

$$EX_{t+1}(i,j)=$$
$$E[X_t(i,j)+a(t)b(r_1,r_3)|r_2X_t^*(i,j)-X_t(i,j)|]=$$
$$EX_t(i,j)+a(t)Eb(r_1,r_3)E|r_2X_t^*(i,j)-X_t(i,j)|=$$
$$EX_t(i,j). \tag{12}$$

由式(9)和(12),得到

$$D(X_{t+1}(i,j))=$$
$$E[X_t(i,j)+a(t)b(r_1,r_3)|r_2X_t^*(i,j)-X_t(i,j)|]^2-$$
$$(E(X_{t+1}(i,j)))^2=$$
$$E[X_t(i,j)]^2+2a(t)Eb(r_1,r_3)E[X_t(i,j)\times$$
$$|r_2X_t^*(i,j)-X_t(i,j)|]+a^2(t)Eb^2(r_1,r_3)\times$$
$$E[|r_2X_t^*(i,j)-X_t(i,j)|]^2-(E(X_{t+1}(i,j)))^2=$$
$$D(X_t(i,j))+\frac{1}{2}a^2(t)E[|r_2X_t^*(i,j)-X_t(i,j)|]^2. \tag{13}$$

将式(6)、(7)和(13)代入(5),得到

$$E[I(X_{t+1})]=$$
$$\frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\Big\{D(X_{t+1}(i,j))-$$
$$\frac{2}{N}D(X_{t+1}(i,j))+\frac{1}{N^2}D(X_{t+1}(i,j))+$$
$$\Big(E(X_{t+1}(i,j))-\frac{1}{N}\sum_{i=k}^{N}E(X_{t+1}(k,j))\Big)^2\Big\}=$$
$$\frac{(N-1)^2}{MN^3}\sum_{i=1}^{N}\sum_{j=1}^{M}D(X_{t+1}(k,j))+$$
$$\frac{1}{NM}I(E(X_{t+1}(i,j)))=$$
$$\frac{(N-1)^2}{MN^3}\sum_{i=1}^{N}\sum_{j=1}^{M}\{D(X_t(i,j)+$$
$$\frac{1}{2}a^2(t)E[|r_2X_t^*(i,j)-X_t(i,j)|]^2\}+$$

$$\frac{1}{NM}I(E(X_t(i,j))), \tag{14}$$

其中$I(EX_{t(i,j)})$为第t时刻个体X_t期望的多样性. □

定理1给出了第$t+1$代SCA群体多样性与算法参数的关系. 若第t代种群给定,种群规模、解的维度和种群个体的取值确定,则X_t的方差$D(X_t(i,j))$和$I(E(X_t(i,j)))$也确定. 可知,式(14)中求和的第2项为$a(t)^2$与$E|r_2X_t^*(i,j)-X_t(i,j)|$的乘积,由于种群取值已确定,随机数$r_2$和控制因子$a(t)$影响第$t+1$代群体多样性. 种群多样性与控制因子成指数关系,当$a(t)>1$时,$a^2(t)$对$a(t)$值为指数放大,利于维持群体多样性作用;当$a(t)<1$时,$a^2(t)$对$a(t)$值为指数递减,致使群体多样性快速下降.

在标准SCA算法中,由式(2)可知控制因子$a(t)$随着迭代次数从2下降到0. 当$t=T/2$时,$a(t)=1$;当$t>T/2$时,$a(t)<1$,使得群体加速收敛. 若陷入局部最优,则群体易出现停滞情况.

3　多尺度正余弦算法(MSCA)

通过对标准SCA种群多样性的分析可知,当控制因子$a(t)<1$时,将使得群体加速收敛. 标准SCA采用的线性递减策略,将使得一半的进化时间都用于局部搜索,加速群体收敛. 该策略针对单峰函数的搜索是快速有效的,但是针对复杂多模问题,会出现进化后期探索能力不足的问题. 为解决该问题,本文提出多尺度控制因子策略,使得在进化过程中群体可以实现全局或局部的交替搜索. 在标准SCA算法中,由于随机因子r_2的影响使得算法的局部搜索能力偏弱,MSCA采用双种群策略,主种群采用多尺度控制因子学习策略,辅助群采用局部学习算法.

3.1　多尺度控制因子策略

假设总迭代次数为T,将整个进化过程划分为k段,每个过程迭代次数分别设置为$T_1,T_2,\ldots,T_k$,并满足$T_1+T_2+\ldots+T_k=T$.在每段进化过程,群体可采用不同的控制因子,多尺度控制因子定义如下:

$$a(t)=f(\lambda_i,\beta_i,T_i),\ i\in k. \tag{15}$$

其中:λ_i和β_i分别为控制因子的起始值和终止值;T_i为在第i段的迭代次数;$f(\cdot)$为定义的函数,为线性函数或非线性函数. 本文设计了两阶段多尺度控制因子,即

$$a(t)=\begin{cases}\lambda_1\left(1-\dfrac{t}{T_1}\right)+\beta_1,\ t<T_1;\\ \lambda_2\left(1-\dfrac{t-T_1}{T-T_1}\right),\ t\geqslant T_1.\end{cases} \tag{16}$$

当$t<T_1$时,控制因子$a(t)$从$\lambda_1+\beta_1$递减至β_1;当$t\geqslant T_1$时,控制因子$a(t)$从λ_2递减至0,其中$\lambda_2>\beta_1$. MSCA通过多尺度的控制因子可以实施每个阶段探索和利用,从而在整个进化过程中实现全局或局部的交替搜索,使得群体避免停滞.

3.2　群体学习机制

MSCA利用多尺度控制因子参数值的设置可以实现算法的局部搜索,但是群体的学习榜样(X_t^*)受r_2的影响,使得群体无法及时跟踪X_t^*,从而降低了收敛速度. 为了保持较高的收敛速度,MSCA采用双种群策略,即主群和协助群. 主群采用多尺度控制因子学习策略;协助群专注对已有最优解的利用学习. 协助群局部学习算法的思想来源于PSO算法. 在PSO算法中粒子同时学习群体最优粒子gbest和个体历史最优粒子pbest,最终收敛于gbest与pbest的算术平均点. 为了加快收敛速度,本文设计的局部学习算法直接学习种群gbest与个体pbest的算术平均点. 更新方程如下:

$$X_{t+1}(i)=\begin{cases}X_t(i)+b(t)\cdot\xi(X_t(i)-0.5(G_t+P_t(i))),\\ \quad f(G_t)>f(X_t^*);\\ X_t(i)+b(t)\cdot\xi(X_t(i)-0.5(X_t^*+P_t(i))),\\ \quad f(G_t)\leqslant f(X_t^*).\end{cases} \tag{17}$$

其中:ξ为服从$[0,1]$分布的随机数;$b(t)$为学习步长,$b(t)=2(1-t/T)+2$,T为总迭代次数;X_t^*为主群群体最优个体;G_t为协助群群体最优个体;P_t为个体历史最优粒子.

主群和协助群独立进化,当主群种群的最优解适应度值大于协助群的最优解适应度值时,协助群不再学习自身的最优解而是学习主群的最优解.

3.3　MSCA算法流程

MSCA算法具体步骤如下.

step 1: 种群初始化. 种群$S=S_m\bigcup S_s$,其中S_m为主群,S_s为协助群. 设S_m种群规模为N_1,S_s种群规模为N_2,$N_2<N_1$,按给定搜索空间的范围随机生成初始种群S的个体.

step 2: 计算S_m和S_s个体的适应度$f(X_i)$,并分别将S_m和S_s适应度最大的个体赋值给S_m群体最优个体X^*和S_s群体最优个体G.

step 3: 对于$X_i\in S_m(i=1,2,\ldots,N_1)$,按照式(1)和(16)产生个体$X_i$的新位置$f(X_i^{\text{new}})$. 若适应度$f(X_i^{\text{new}})>f(X_i)$,则$X_i\leftarrow X_i^{\text{new}}$.

step 4: 若$f(X^*)>f(G)$,则$G\leftarrow X^*$;否则转至step 5.

step 5: 对于$X_i\in S_s(i=1,2,\ldots,N_2)$,按照式(17)产生个体$X_i$的新位置$f(X_i^{\text{new}})$. 若适应度$f(X_i^{\text{new}})$

$> f(X_i)$,则$X_i \leftarrow X_i^{\text{new}}$.

step 6: 判断是否达到最大迭代次数T,若满足条件则停止迭代,将G作为结果输出;否则转至step 3.

4 数值实验与结果分析

4.1 测试函数和参数设置

为了验证所提出算法的优化性能,选取文献[5-10]采用的23个测试函数,包括单调测试函数($f_1 \sim f_7$)、多模测试函数($f_8 \sim f_{13}$)和固定维数多模测试函数($f_{14} \sim f_{23}$).

将所提出算法(MSCA)与标准SCA和改进SCA进行实验比较,包括基于反向学习的混合SCA(HSCA)[7]、基于精英混沌学习的SCA(COSCA)[8]、基于记忆导向的SCA(MESCA)[9]、时变加速系数PSO算法(TVACPSO)[12]、灰狼算法(GWO)[13]、人工蜂群算法(ABC)[14]和动态教与学算法(DTLBO)[11]. 各算法参数设置如表1所示.

表 1 参数设置

算法	参数设置
SCA	$a(t): 2 \to 0$
COSCA	$a(t): 1 \to 0, \eta = 1, \text{pr} = 0.1$
HSCA	$a(t): 2 \to 0$
MESCA	$a(t): 2 \to 0$,设置不同引导个体Δ_t
TVACPSO	$b_1: 2.5 \to 0.5, b_2: 0.5 \to 2.5, \omega: 0.9 \to 0.4$
GWO	$a: 2 \to 0$
QABC	bestlimit $= 45$
DTLBO	$J_r = 0.3, \omega = 10, \text{tf} \in (0, 2)$
MSCA	$t > 0.5T, a(t): 2.5 \to 0.5;$ $t \leqslant 0.5T, a(t): 1.5 \to 0$

仿真实验计算机配置为Intel(R) Core(TM) i5-10210U CPU @2.30 GHz 1.60 GHz, 16 GB内存. 在对比实验中,所有算法均使用相同的种群规模$N = 30$,其中MSCA主群规模为20,协助群规模为10. 算法的最大迭代次数$T = 500$,独立运行25次,维数为30. 每种算法运行后记录其平均适应度值和标准差,并用Wilcoxon秩和检验(显著性水平$\alpha = 0.05$)评价MSCA与其他算法之间的性能差异.

4.2 优化性能分析

9种算法运行25次的平均适应度和标准差如表2所示,每个函数获取的最优结果用加粗字体显示. 由表2可见,MSCA已找到5个函数的理论最优值,分别为f_9、f_{10}、f_{11}、f_{16}和f_{18},其中f_9、f_{10}和f_{11}为复杂多模函数,存在多个局部极值,MSCA可以找到理论最优值,表明该算法具有良好的处理多模函数的能力. f_5为Rosenbrock函数,虽然是单调函数,但只有MSCA的解可以到达10^{-1}数量级. f_8为Schwefe's函数,其复杂性是由于在远离理论最优值的地方,存在一个深度很深的局部值,致使很多优化算法陷入局部极小值,MSCA找到的解可以到达10^{-1}数量级,这表明MSCA具有较好的全局搜索能力. 对于10个固定维数的函数问题,9种算法均可获得满意解,MSCA在7个函数上获得了最优解,表明MSCA也具有较好的局部搜索能力.

在$\alpha = 0.05$的显著性水平下,MSCA与其他对比算法的Wilcoxon秩、检测概率值p和检测结果D如表3所示. 符号"+""="和"−"分别表示MSCA优于、相当和劣于某个算法. "−/=/+"显示了算法比较的最终统计结果,DTBLO算法的统计结果为"2/5/16",表示与MSCA相比,有2个函数的优化结果优于MSCA,5个函数的优化结果相当于MSCA,16个函数的优化结果劣于MSCA. 依据Wilcoxon秩和检测的结果,可获得9个算法平均排名,结果如表2中"rank"所示. 可见,MSCA的综合优化性能最好,DTBLO次之,COSCA第3.

为了直观地显示各算法的进化过程,6个测试函数的9种算法收敛曲线如图1∼图6所示. 由图1∼图6可见,SCA、HSCA和MESCA算法在进化中期后逐渐停滞. 虽然HSCA融入新的操作算子,MESCA采用了多榜样学习的方式,但是由于进化中期以后控制因子小于1加速了群体的聚集,使其陷入停滞. COSCA算法中增加了混沌搜索和方向搜索增加进化后期的多样性. MSCA能够在进化后期保持搜索能力,表明多尺度控制因子策略是有效的. GWO、DTBLO、PSO和ABC虽然在多数函数上进化后期没有停滞,但是收敛速度稍缓慢. MSCA较好地平衡了算法的收敛速度和收敛精度.

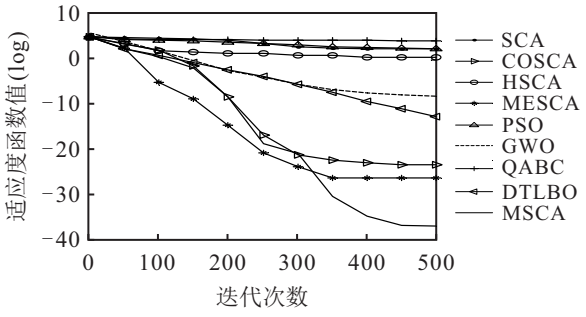

图 1 f_3收敛曲线

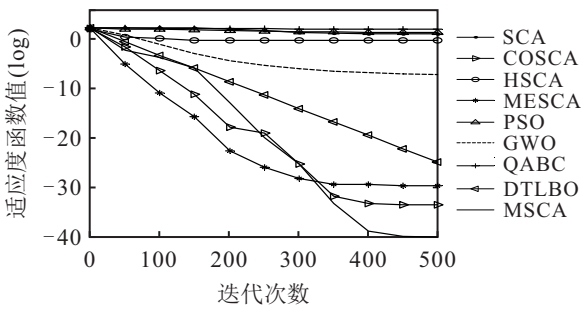

图 2 f_4收敛曲线

表 2　9种算法优化结果

函数	SCA	COSCA	HSCA	MESCA	PSO	GWO	ABC	DTBLO	MSCA	f_{min}
f_1	2.00e-04 (3.88e-03)	**2.43e-78** (3.22e-80)	2.53e-06 (3.19e-06)	2.87e-73 (4.73e-73)	5.49e-06 (1.24e-06)	1.51e-31 (1.80e-31)	7.16e-05 (6.25e-06)	5.80e-76 (4.64e-76)	1.04e-75 (2.23e-74)	0
f_2	7.22e-1 (1.19e-1)	**1.69e-37** **(2.88e-37)**	1.17e-05 (1.82e-6)	1.61e-36 (2.23e-35)	1.02e-01 (3.45e-02)	8.62e-19 (4.95e-19)	7.84e-03 (3.09e-03)	9.23e-33 (8.88e-39)	4.45e-35 (2.15e-36)	0
f_3	3.45e+02 (3.72e+02)	4.27e-16 (4.88e-17)	1.70e-01 (3.12e-01)	6.27e-27 (7.30e-27)	6.95e-01 (2.34e-01)	9.31e-09 (1.49e-08)	6.24e+01 (4.18e+01)	2.37e-13 (2.55e-13)	**5.66e-35** **(3.12e-35)**	0
f_4	5.36e+00 (6.38 e+00)	8.54e-35 (1.75e-35)	6.55e-1 (1.46e-1)	5.61e-28 (1.25e-28)	6.95e-01 (2.34e-01)	4.18e-08 (4.43e-08)	4.40e+01 (2.34e+00)	1.21e-25 (1.75e-25)	**3.25e-37** **(2.02e-37)**	0
f_5	3.81e+01 (1.82e+01)	2.13e+01 (2.67e+00)	2.84e+01 (4.26e-01)	2.81e+01 (5.20e-01)	5.32e+01 (3.15e-01)	2.71e+01 (5.20e-01)	9.46e+01 (3.61e+01)	2.68e+01 (5.80e-01)	**1.63e-01** **(2.50e-02)**	0
f_6	4.02e+00 (2.42e-01)	2.98e+00 (1.12e-3)	6.03e-01 (1.21e-01)	1.82e+00 (3.02e-01)	1.88e-02 (3.52e-02)	6.28e-01 (4.02e-01)	**1.52e-03** **(5.21e-05)**	2.28e-02 (3.09e-02)	4.11e-03 (1.21e-03)	0
f_7	3.65e-01 (3.61e-03)	3.22e-04 (1.62e-04)	5.55e-03 (2.32e-03)	2.59e-03 (1.41e-03)	6.42e-02 (5.23e-03)	3.30e-01 (2.73e-01)	3.68e-01 (3.54e-02)	1.82e-03 (4.26e-04)	**2.49e-04** **(1.62e-04)**	0
f_8	5.62e+03 (8.49e+02)	5.98e+01 (7.53e+00)	3.21e+03 (7.15e+02)	5.14e+03 (1.00e+03)	2.24e+01 (1.23e+01)	5.14e+03 (1.00e+03)	1.88e+02 (9.49e+01)	9.24e+02 (5.90e+02)	**7.16e-02** **(2.63e-02)**	0
f_9	8.11e-01 (9.03e-01)	**0.00e+00** **(0.00e+00)**	8.71e+00 (5.13e+00)	**0.00e+00** **(0.00e+00)**	3.31e+01 (1.21e+01)	**0.00e+00** **(0.00e+00)**	6.76e+00 (2.15e+00)	1.46e+01 (9.15e+00)	**0.00e+00** **(0.00e+00)**	0
f_{10}	2.77e-01 (3.39e-01)	**0.00e+00** **(0.00e+00)**	1.22e-04 (3.18e-05)	9.43e-05 (3.18e-05)	2.12e+00 (5.78e-02)	1.06e-13 (3.18e-14)	1.13e-01 (8.62e-02)	4.25e-15 (1.58e-15)	**0.00e+00** **(0.00e+00)**	0
f_{11}	4.25e-01 (1.04e-01)	**0.00e+00** **(0.00e+00)**	2.45e-03 (3.09e-04)	7.56e-08 (1.49e-08)	9.96e-02 (3.45e-03)	3.56e-03 (6.49e-03)	4.21e-02 (3.38e-02)	**0.00e+00** **(0.00e+00)**	**0.00e+00** **(0.00e+00)**	0
f_{12}	8.17e-01 (5.34e-01)	3.45e-01 (5.79e-02)	6.43e-02 (8.14e-03)	5.01e-02 (2.08e-02)	1.84e+01 (1.17e+01)	8.01e-02 (2.08e-2)	**8.98e-05** **(1.43e-04)**	3.24e+00 (6.54e-1)	6.32e-04 (1.38e-4)	0
f_{13}	2.58e+00 (2.43e-01)	2.32e+00 (3.13e-01)	1.40e+00 (8.24e-01)	2.73e+00 (3.55e-01)	8.32e+01 (4.74e+01)	8.38e-01 (3.08e-01)	**1.22e-04** **(5.01e-05)**	1.14e-01 (5.84e-02)	6.78e-03 (8.25e-04)	0
f_{14}	1.511 (3.84e-01)	3.552 (2.18e-02)	**0.998** (1.82e-01)	**0.998** (1.02e-03)	**0.998** (7.01e-14)	1.11e+01 (3.04e+00)	**0.998** (6.15e-09)	**0.998** **(0.00e+00)**	**0.998** 5.68e-10	1
f_{15}	1.12e-03 (2.54e-03)	7.879e-04 (6.34e-04)	5.01e-04 (1.00e-04)	5.67e-04 (1.52e-04)	5.06e-04 (3.51e-04)	4.85e-04 (4.50e-04)	5.99e-04 (1.58e-04)	7.15e-04 (5.26e-04)	**4.04e-04** **(4.617e-05)**	3e-04
f_{16}	−1.030 1 (4.46e-05)	−1.031 1 (1.17e-05)	**−1.031 6** **(0.00e+00)**	−1.031 1 (3.58e-06)	**−1.031 6** **(0.00e+00)**	**−1.031 6** (3.58e-06)	**−1.031 6** **(0.00e+00)**	**−1.031 6** **(0.00e+00)**	**−1.031 6** (6.027e-09)	−1.031 6
f_{17}	0.399 (1.43e-02)	**0.398** **(0.00e+00)**	0.397 **(0.00e+00)**	**0.398** (1.14e-03)	0.397 **(0.00e+00)**	0.397 (2.95e-00)	0.397 (9.41e-11)	0.397 **(0.00e+00)**	**0.398** (1.99e-02)	0.398
f_{18}	**3.000** (1.56e-04)	**3.000** (7.94e-09)	**3.000** (1.00e-04)	**3.000** (3.51e-05)	2.999 (6.58e-15)	**3.000** (7.05e-06)	2.999 (1.69e-15)	2.999 **(0.00e+00)**	**3.000** (4.757e-09)	3
f_{19}	−3.855 0 (2.26e-03)	−3.858 9 (1.32e-04)	**−3.862 7** (2.01e-04)	−3.859 9 (1.04e-03)	**−3.862 7** **(2.26e-16)**	−3.858 1 (2.81e-03)	−3.842 7 (4.26e-16)	−3.862 0 (2.26e-16)	−3.845 8 (3.10e-02)	−3.86
f_{20}	−2.873 4 (1.16e-01)	−3.162 (3.416e-02)	−3.318 0 (5.01e-03)	−3.169 (6.26e-02)	−3.270 (5.92e-2)	**−3.320** (7.03e-6)	**−3.320** **(6.25e-16)**	**−3.320** (4.25e-11)	−3.181 (0.140 7)	−3.32
f_{21}	−2.236 (1.71)	−9.583 4 (9.216e-04)	−10.012 4 (1.99e-01)	−9.923 4 (2.25e-01)	−10.151 2 (2.34e-05)	−10.152 0 (2.351e-04)	**−10.153 1** (2.96e-05)	**−10.153 1** (3.55e-05)	**−10.153 1** **(1.96e-05)**	−10.153 2
f_{22}	−3.398 2 (2.09e+00)	−10.320 8 (3.12e-06)	−10.270 6 (2.12e-01)	−10.339 1 (6.21e-02)	−10.270 6 **(2.51e-06)**	−10.402 0 (7.29e-04)	−10.402 4 (1.36e-05)	**−10.402 6** (1.25e-05)	**−10.402 6** (9.50e-05)	−10.402 8
f_{23}	−3.344 5 (1.320)	−10.536 0 (2.42e-05)	−10.366 0 (1.13e-01)	−10.502 1 (3.52e-02)	−10.536 0 (8.28e-07)	−10.482 1 (3.12e-04)	−10.536 0 **(1.57e-10)**	−9.196 2 (2.99e+00)	**−10.536 2** (1.50e-05)	−10.536 2
rank	7.91(9)	3.96(3)	4.83(6)	4.26(4)	5.52(8)	4.39(5)	4.83(6)	3.70(2)	1.78(1)	

表 3 MSCA与对比算法的检验值P和检验结果D

函数	SCA		COSCA		HSCA		MESCA		TVACPSO		GWO		ABC		DBLO	
	P	D	P	D	P	D	P	D	P	D	P	D	P	D	P	D
f_1	1.41e-9	+	1.72e-4	−	1.41e-9	+	1.72e-4	+	1.41e-9	+	1.41e-9	+	1.41e-9	+	0.309	=
f_2	5.46e-6	+	1.72e-4	−	1.41e-9	+	0.622	=	1.41e-9	+	1.41e-9	+	1.41e-9	+	2.41e-6	+
f_3	1.41e-9	+	1.41e-9	+	1.41e-9	+	1.72e-4	+	1.41e-9	+	1.41e-9	+	1.41e-9	+	1.41e-9	+
f_4	1.41e-9	+	1.41e-9	+	1.41e-9	+	1.72e-4	+	1.41e-9	+	1.41e-9	+	1.41e-9	+	.41e-9	+
f_5	1.41e-9	+	1.41e-9	+	1.41e-9	+	1.41e-9	+	1.41e-9	+	1.41e-9	+	1.41e-9	+	1.41e-9	+
f_6	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	1.82e-4	+	5.46e-6	+	1.72e-4	−	5.46e-6	+
f_7	5.46e-6	+	1.82e-4	+	5.46e-6	+	3.99e-5	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	1.27e-5	+
f_8	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+
f_9	2.19e-4	+	NAN	=	4.23e-6	+	NAN	=	4.23e-6	+	NAN	=	4.23e-6	+	4.23e-6	+
f_{10}	4.23e-6	+	NAN	=	4.23e-6	+	4.23e-6	+	4.23e-6	+	2.38e-6	+	4.23e-6	+	4.23e-6	+
f_{11}	4.23e-6	+	NAN	=	4.23e-6	+	4.23e-6	+	4.23e-6	+	4.23e-6	+	4.23e-6	+	NAN	=
f_{12}	4.23e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	1.72e-4	−	5.46e-6	+
f_{13}	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	5.46e-6	+	1.72e-4	−	5.46e-6	+
f_{14}	2.49e-4	+	2.49e-4	+	5.21e-6	+	1.08e-5	+	0.568	=	5.35e-5	+	1.56e-8	+	3.35e-8	−
f_{15}	2.49e-6	+	2.49e-6	+	2.49e-4	+	2.49e-4	+	2.49e-4	+	2.49e-4	+	2.49e-4	+	2.49e-6	+
f_{16}	2.49e-6	+	2.49e-6	+	6.15e-5	−	2.49e-6	+	6.15e-5	−	6.15e-5	−	6.15e-5	−	6.15e-5	−
f_{17}	5.46e-6	+	5.15e-4	−	2.89e-9	+	6.15e-5	−	9.7e-11	+	1.41e-9	+	9.7e-11	+	9.7e-11	+
f_{18}	1.33e-2	+	0.521	=	1.33e-2	+	2.99e-3	+	9.9e-10	+	0.521	=	1.22e-9	+	3.9e-10	+
f_{19}	1.59e-4	−	1.59e-4	−	3.99e-4	−	1.59e-4	−	1.59e-4	−	1.59e-4	−	1.31e-2	+	1.59e-4	−
f_{20}	1.95e-4	+	2.19e-9	+	3.32e-4	−	2.19e-9	+	2.10e-8	−	1.14e-9	−	1.14e-9	−	1.14e-9	−
f_{21}	1.95e-4	+	1.41e-9	+	3.99e-9	+	1.41e-9	+	3.5e-10	+	0.521	=	0.521	=	0.521	=
f_{22}	1.95e-4	+	1.59e-9	+	8.61e-9	+	2.56e-8	+	8.61e-9	+	9.7e-10	+	3.1e-10	+	0.245	=
f_{23}	1.7e-10	+	1.7e-10	+	1.59e-9	+	7.7e-10	+	1.71e-10	+	8.51e-9	+	1.7e-10	+	3.99e-4	+
$-/=/+$	1/0/22		3/4/16		2/1/20		1/2/20		2/2/19		2/3/18		4/2/17		2/5/16	

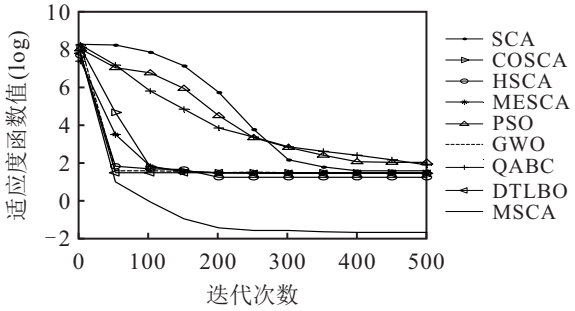

图 3 f_5收敛曲线

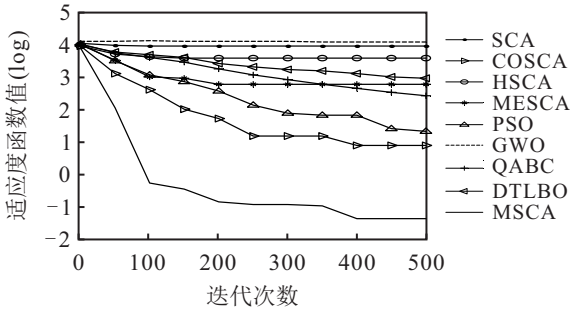

图 4 f_8收敛曲线

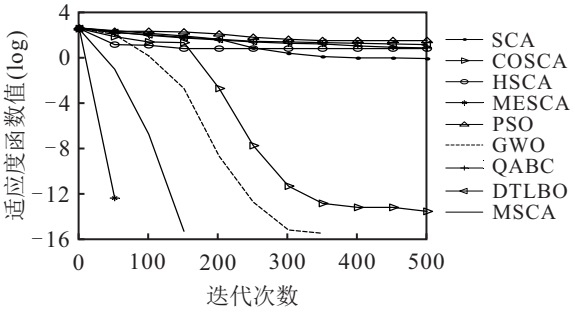

图 5 f_9收敛曲线

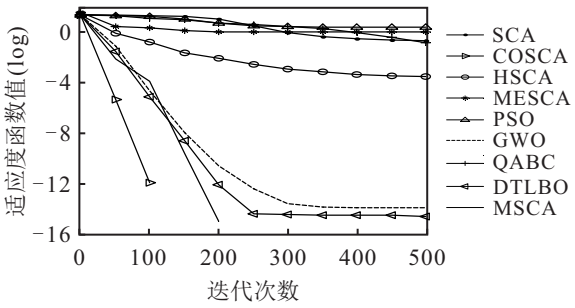

图 6 f_{10}收敛曲线

4.3 算法的时间复杂度分析

设种群规模为N,待搜索空间的维数为M,适应度计算复杂度为$O(f_s)$. 标准的SCA在运行过程中群体计算适应度的复杂度为$O(Nf_s)$,个体更新的计算复杂度为$O(NM)$,适应度比较运算的复杂度为$O(N)$,故SCA计算复杂度为

$$O(\text{SCA}) = O(NM + Nf_s + N). \tag{18}$$

与标准的SCA相比,MESCA在运行过程中增加了个体选择的比较运算,其复杂度为$O(N)$,故计算复杂度为

$$O(\text{MESCA}) = O(NM + Nf_s + 2N). \tag{19}$$

HSCA和COSCA算法中含有快速排序操作,对于N个个体快速排序的计算复杂为$O(N\log N)$,最差的情况为$O(N^2)$,将快速排序的计算复杂度记作O_s,$O(\text{HSCA})$和$O(\text{COSCA})$[8]分别为

$$O(\text{HSCA}) = O(NM + Nf_s + N + O_s), \tag{20}$$

$$O(\text{COSCA})=O(1.1NM+Nf_s+N+O_s+O_g),\quad (21)$$

其中O_g为COSCA精英混沌策略计算复杂度. MSCA主群在运行过程中计算控制因子需判断当前的迭代次数,其计算复杂度为$O(1)$. 评价个体适应度的计算复杂度为$O((N_1+N_2)f_s)$. 主群的个体更新计算复杂度为$O(N_1M)$(N_1为主群规模),协助群的个体更新计算复杂度为$O(N_2M)$(N_2为协助群规模),由于$N=N_1+N_2$,个体更新计算复杂度为$O(NM)$. 与标准的SCA相比,MSCA的协助群增加了个体最优值更新的计算复杂度$O(N_2)$,故MSCA计算复杂度为

$$O(\text{MSCA})=O(NM+Nf_s+N+N_2).\quad (22)$$

算法运行时,CPU的总耗时可从一定程度上反映算法的时间复杂度,表4为9种算法优化每个函数运行25次的总耗时. 由表4可见:9种算法中人工蜂群算法(ABC)的总耗时86.266 s为最短;SCA、MESCA、MSCA的总耗时分别为129.165 s、129.363 s、129.227 s;COSCA总耗时170.698 s为最长.

表4　9种算法运行25次总耗时　　**单位**: s

函数	SCA	COSCA	HSCA	MESCA	PSO	GWO	ABC	DTBLO	MSCA
f_1	5.056	6.879	5.842	5.180	4.771	3.083	2.258	8.449	5.012
f_2	5.485	7.521	7.218	5.762	5.274	3.905	3.052	8.160	5.504
f_3	8.930	12.873	12.431	8.627	8.762	7.001	5.954	16.984	8.990
f_4	4.947	6.629	6.152	4.471	4.785	4.167	3.912	10.661	4.805
f_5	6.138	8.998	8.843	6.884	5.675	4.862	4.939	12.674	6.110
f_6	5.730	7.855	7.931	5.160	5.323	4.148	3.247	9.936	5.662
f_7	6.691	8.738	8.728	6.699	5.884	4.347	3.462	10.437	6.689
f_8	5.001	6.839	5.683	5.180	4.155	3.390	3.462	9.990	5.160
f_9	5.425	7.131	6.187	5.447	4.734	3.403	3.181	12.156	5.502
f_{10}	6.845	7.367	7.921	6.282	5.652	4.104	3.540	11.070	6.976
f_{11}	5.530	7.148	6.397	5.568	5.457	4.750	3.714	10.906	5.618
f_{12}	6.236	8.192	7.690	6.374	5.959	4.176	3.967	11.291	6.323
f_{13}	6.533	8.241	7.585	6.556	5.294	6.692	5.752	10.729	6.518
f_{14}	6.556	8.842	7.693	6.561	5.798	5.285	3.842	11.987	6.560
f_{15}	4.003	6.473	6.196	4.019	3.851	3.282	2.168	7.401	4.005
f_{16}	5.091	6.843	6.630	5.096	4.899	3.029	2.310	8.980	5.104
f_{17}	4.077	5.832	5.447	4.081	3.691	2.836	3.148	8.974	4.002
f_{18}	4.267	6.688	6.196	4.273	3.982	2.960	3.264	7.682	4.226
f_{19}	4.789	5.439	5.375	4.796	4.728	2.192	2.180	8.295	4.740
f_{20}	4.381	5.352	5.864	4.488	4.372	3.085	4.092	8.649	4.346
f_{21}	5.942	7.349	7.831	6.045	5.262	3.306	4.596	8.522	5.945
f_{22}	5.889	6.293	6.730	5.990	5.543	3.118	4.960	10.085	5.802
f_{23}	5.623	7.176	6.965	5.824	5.628	4.056	5.266	10.682	5.628
总时	129.165	170.698	163.535	129.363	119.479	91.177	86.266	234.700	129.227

表5　参数的设置对算法的影响

参数设置	f_3 (p=7.15e-3)		f_5 (p=1.56e-4)		f_6 (p=6.83e-3)		f_8 (p=7.21e-4)		f_{12} (p=1.25e-4)		f_{13} (p=7.15e-3)	
	mean(std)	R	mean(std)	R	mean(std)	R	mean(std)	R	mean(std)	R	mean(std)	R
$\lambda_1=2.5,\beta_1=1$	7.15e-30	5	6.71e+0	7	6.73e-2	7	1.02e+1	5	9.25e-4	4	8.95e-2	6
$\lambda_1=2$	(1.42e-29)		(7.12e+0)		(6.54e-2)		(1.11e+1)		(1.54e-3)		(9.63e-2)	
$\lambda_1=2.5,\beta_1=0.5$	4.16e-32	3	4.72e-1	2	7.62e-3	3	1.93e-1	4	6.89e-4	2	9.78e-3	4
$\lambda_1=2$	(1.21e-33)		(7.02e-1)		(8.54e-3)		(1.65e-1)		(7.34e-4)		(8.54e-3)	
$\lambda_1=2,\beta_1=0.5$	7.82e-31	4	8.87e-1	4	9.18e-3	4	8.62e-2	3	1.50e-3	5	8.63e-3	2
$\lambda_1=2$	(1.42e-31)		(6.11e-1)		(7.86e-3)		(7.43e-2)		(1.01e-3)		(8.31e-3)	
$\lambda_1=2.5,\beta_1=1$	1.71e-25	7	6.58e+0	6	5.11e-2	5	4.17e+1	7	1.21e-3	6	5.56e-1	7
$\lambda_1=1.5$	(3.32e-24)		(3.23e+0)		(5.42e-2)		(2.46e+1)		(9.25e-4)		(4.70e-1)	
$\lambda_1=2,\beta_1=1$	4.56e-23	8	8.57e+0	8	5.46e-2	6	8.63e+1	8	2.32e-3	7	7.43e-1	8
$\lambda_1=1.5$	(6.34e-23)		(6.91e+0)		(4.34e-2)		(7.86e+1)		(1.26e-3)		(6.32e-1)	
$\lambda_1=2.5,\beta_1=0.5$	5.66e-35	1	1.63e-1	1	4.11e-3	1	7.16e-2	1	6.32e-4	1	6.78e-3	1
$\lambda_1=1.5$	(3.12e-35)		(2.50e-2)		(1.21e-3)		(2.63e-2)		(1.38e-4)		(8.25e-4)	
$\lambda1=2,\beta_1=0.5$	8.92e-35	2	5.52e-1	3	7.31e-3	2	8.03e-2	2	7.28e-4	3	9.11e-3	3
$\lambda_1=1.5$	(9.21e-35)		(3.24e-1)		(6.28e-3)		(7.32e+1)		(5.12e-4)		(8.16e-3)	

MSCA的计算开销与SCA和MESCA相仿,低于COSCA、HSCA和DTLBO,高于PSO、GWO和ABC.

4.4 参数设置讨论

MSCA的控制参数$a(t)$起着协调算法的探索和开发的作用,当$a(t)$取较大值时,利于群体的探索;取较小值时,加强群体的开发. MSCA采用两阶段多尺度控制因子,式(15)中含有λ_1、β_1和λ_2,本节对参数取不同的值时算法的性能进行分析. λ_1取值为2和2.5;β_1取值为0.5和1. λ_2取值为2和1.5,形成8个不同的参数组合. MSCA算法在不同的参数组合下获得函数f_3、f_5、f_6、f_8、f_{12}和f_{13}的优化结果,如表5所示,表中mean(std)为均值(方差),R为排序(rank),p为在$\alpha=0.05$的显著性水平下Friedman检验的概率值. 由表5可见,测试函数对应的Friedman检测值P均小于0.05,表明5种pr取值对应的算法结果之间存在显著差异. 对于8个不同的参组组合,其中$\lambda_1=2.5$、$\beta_1=0.5$、$\lambda_1=1.5$获得的平均排名最高.

5 结 论

本文首先给出了标准SCA进化过程中群体多样性期望,并指出控制因子的设置是导致在进化后期多样性缺失的重要因素之一;然后提出了多尺度SCA(MSCA),新算法通过自适应的多尺度控制因子调节群体多样性从而实现多层次的搜索,同时设计了协助种群实施局部搜索兼顾加快收敛速度和提高解的质量. 将所提出算法与改进的正余弦算法和新型群智能算法进行对比,实验统计结果表明,所提出算法较好地平衡了进化过程中的探索和开采,提高了全局优化的能力.

参考文献(References)

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

[2] Abualigah L, Diabat A. Advances in sine cosine algorithm: A comprehensive survey[J]. Artificial Intelligence Review, 2021, 54(4): 2567-2608.

[3] Nenavath H, Kumar Jatoth D R, Das D S. A synergy of the sine-cosine algorithm and particle swarm optimizer for improved global optimization and object tracking[J]. Swarm and Evolutionary Computation, 2018, 43: 1-30.

[4] Belazzoug M, Touahria M, Nouioua F, et al. An improved sine cosine algorithm to select features for text categorization[J]. Journal of King Saud University-Computer and Information Sciences, 2020, 32(4): 454-464.

[5] Abd Elaziz M, Oliva D, Xiong S W. An improved opposition-based sine cosine algorithm for global optimization[J]. Expert Systems with Applications, 2017, 90: 484-500.

[6] Chegini S N, Bagheri A, Najafi F. PSOSCALF: A new hybrid PSO based on sine cosine algorithm and levy flight for solving optimization problems[J]. Applied Soft Computing, 2018, 73: 697-726.

[7] Gupta S, Deep K. A hybrid self-adaptive sine cosine algorithm with opposition based learning[J]. Expert Systems With Applications, 2019, 119: 210-230.

[8] 郭文艳, 王远, 戴芳, 等. 基于精英混沌搜索策略的交替正余弦算法[J]. 控制与决策, 2019, 34(8): 1654-1662.
(Guo W Y, Wang Y, Dai F, et al. Alternating sine cosine algorithm based on elite chaotic search strategy[J]. Control and Decision, 2019, 34(8): 1654-1662.)

[9] Gupta S, Deep K, Engelbrecht A P. A memory guided sine cosine algorithm for global optimization[J]. Engineering Applications of Artificial Intelligence, 2020, 93: 103718.

[10] Chen H L, Wang M J, Zhao X H. A multi-strategy enhanced sine cosine algorithm for global optimization and constrained practical engineering problems[J]. Applied Mathematics and Computation, 2020, 369: 124872.

[11] Xu Y L, Yang Z L, Li X P, et al. Dynamic opposite learning enhanced teaching-learning-based optimization[J]. Knowledge-Based Systems, 2020, 188: 104966.

[12] Ratnaweera A, Halgamuge S K, Watson H C. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 240-255.

[13] Mirjalili S, Mirjalili S M, Lewis A. Grey wolf optimizer[J]. Advances in Engineering Software, 2014, 69: 46-61.

[14] Aslan S, Badem H, Karaboga D. Improved quick artificial bee colony algorithm for global optimization[J]. Soft Computing, 2019, 23(24): 13161-13182.

[15] Shi Y H, Eberhart R C. Population diversity of particle swarms[C]. IEEE Congress on Evolutionary Computation. Piscataway: IEEE, 2008: 1063-1067.

作者简介

申元霞(1979–), 女, 副教授, 博士, 从事智能算法、智能信息处理等研究, E-mail: yuanxiashen@163.com;

张学锋(1977–), 男, 教授, 博士, 从事虚拟现实技术与人工智能等研究, E-mail: zxf_06@ahut.edu.cn;

方馨(1997–), 女, 硕士生, 从事智能算法的研究, E-mail: fangxin2345@163.com;

汪小燕(1974–), 女, 副教授, 硕士, 从事粗糙集、粒计算等研究, E-mail: wxyzjx@ahut.edu.cn.

(责任编辑: 魏 冰)