文章编号:1006-9348(2022)09-0370-07

基于种群熵的变步长布谷鸟搜索算法

刘洪达[1,2],李德全*[1],王 栋[1]

(1. 中国科学院长春光学精密机械与物理研究所,吉林 长春 130033;
2. 中国科学院大学,北京 100049)

摘要:为了进一步提高布谷鸟搜索(Cuckoo Search,CS)算法的寻优性能,将种群熵的概念引入到莱维飞行机制中,基于寻优过程中布谷鸟种群的分布情况,来构建出可以实时变化的改进步长控制因子,提出了基于种群熵的变步长布谷鸟搜索(Variable Step-size Cuckoo Search based on Population Entropy,PE-VSCS)算法。利用8个标准测试函数分别对CS算法和PE-VSCS算法进行性能测试,从仿真结果可以得知,面对单峰函数问题或多峰函数问题时,无论函数维度是低是高,PE-VSCS算法的寻优性能均超过CS算法,从而证明了改进步长控制因子的有效性。

关键词:布谷鸟搜索算法;种群熵;步长控制因子

中图分类号:TP301.6 **文献标识码**:B

Variable Step-Size Cuckoo Search Based on Population Entropy

LIU Hong-da[1,2],LI De-quan*[1],WANG Dong[1]

(1. Changchun Institute of Optics,Fine Mechanics and Physics,Chinese Academy of Sciences,Changchun 130033,China;
2. University of Chinese Academy of Sciences,Beijing 100049)

ABSTRACT:In order to further improve the optimization performance of Cuckoo Search (CS) algorithm,the concept of population entropy is introduced into the Levy flight mechanism. Based on the distribution of the cuckoo population during the optimization,an improved step-size control factor that can change in real-time is constructed and an algorithm called Variable Step-size Cuckoo Search based on Population Entropy (PE-VSCS) is proposed. By using 8 standard test functions,the performance of the CS algorithm and the PE-VSCS algorithm are tested. From the simulation results,it can be known that when faced with a unimodal function problem or a multimodal function problem,whether the function dimension is low or high,the optimization performance of the PE-VSCS algorithm exceeds the performance of the CS algorithm,which proves the effectiveness of the improved step-size control factor.

KEYWORDS:Cuckoo search; Population entropy; Step-size control factor

1 引言

随着科学技术的发展,最优化问题成为了各个领域的重点研究对象。由于各行各业提出的最优化问题变得越来越抽象复杂,存在着非线性约束、计算量大等问题,经典的优化计算方法已经无法满足实际的需求。为获得复杂的优化问题合理有效的近似解,研究者们基于没有免费午餐(No Free Lunch,NFL)定理[1],针对不同的优化问题设计了大量的元启发式算法,例如模仿动物种群群集行为的粒子群优化算法[2](Particle Swarm Optimization,PSO)、模拟自然选择与生物进化过程的遗传算法[3](Genetic Algorithm,GA)、模仿蜜蜂行为的人工蜂群算法[4](Artificial Bee Colony Algorithm,ABC)等。在这些元启发式算法的基础上,诸多研究者对算法进行改进,提出了诸多改进策略,从而提高算法的效率。

许多优化算法由于其算法简单、易于实现等优点受人青睐,比如剑桥大学的Yang和Deb于2009年提出的布谷鸟搜索算法[5](Cuckoo Search,CS)。这是一种通过模拟布谷鸟寻窝产卵的行为所设计的优化算法,基于莱维飞行机制进行寻优,能够有效解决最优化问题,并成功应用于图像处理[6]、系统设计[7]、医学应用[8]、数据挖掘[9]等各个领域的实际问题中,也可与其它算法相结合以提高混合算法的性能,如BP神

基金项目:中国科学院"空间科学(二期)"战略性先导科技专项(XDA15020704)

收稿日期:2020-11-18 修回日期:2020-11-24

经网络[10]、PSO 算法[11]、极限学习机[12]等。该算法简单、参数少,易于实现、全局搜索能力强,采用的随机搜索策略使得算法更为高效。此外 CS 算法的通用性强,可以与其它算法相结合。但是 CS 算法也和诸多元启发式算法一样,存在着后期的搜索效率低、寻优精度不高等缺点。

为了克服这些缺点,研究者们纷纷对 CS 算法进行改进:Rodrigues[13]、冯登科[14]等学者将二进制这一概念引入了 CS 算法中,获得了更优秀的全局搜索能力与鲁棒性,但仍存在一些收敛速度较差的问题;Ouaarab[15]、王超[16]等学者提出了一类离散 CS 算法,但是面对一些条件时提高效果并不明显;马英辉[17]、Wang[18]等学者将混沌理论引入到 CS 算法中,提高了寻优速度与精度,但是算法复杂度增大,影响了寻优时间。

针对 CS 算法存在的问题,本文通过种群熵对布谷鸟步长控制因子进行改进:将寻优搜索空间等分为多个子空间,计算种群熵以确定种群的分布特性[19],基于分布特性控制步长控制因子实时变化,从而提高了算法的寻优效率。此外,这项改进保持了 CS 算法的通用性,可配合其它改进策略来进一步提高算法的性能。

2 布谷鸟搜索算法

布谷鸟算法(Cuckoo Search,CS)通过模拟布谷鸟巢寄生繁殖的原理,利用莱维飞行机制(Levy Flight)来有效地求解最优化问题。

部分布谷鸟不会自己孵卵育雏,而是偷偷地在其它鸟类的巢穴中产卵,并移除相同数量的寄主卵,从而保持巢穴内的卵总数不变,最终由该寄主代为孵化育雏。若寄生卵被发现,此寄生卵将会被寄主移除,导致布谷鸟的寄生繁殖失败。

莱维飞行是一种混合了短距离与长距离的随机游走机制。其步长服从于重尾分布,随机游走过程中有相对较高的几率出现大的步长。而对于搜索算法而言,前期的大步长便于增强种群多样性,提高前期的全局搜索能力,易于搜索到全局最优值所在邻域;后期的小步长提高了局部搜索能力,缩小了搜索范围,从而获得全局最优解。

CS 算法通过假设三个理想条件来模拟布谷鸟的巢寄生繁殖机制[5]:

1)布谷鸟一次仅产一个蛋,并随机选择宿主巢穴来孵化自己的蛋;

2)在随机选择的一组巢穴中,孵化条件最好的巢穴将会被保留到下一代;

3)可寄生的宿主巢穴数量 N 是固定的,一个宿主巢穴的主人能发现一个寄生鸟蛋的概率 $p_a \in [0,1]$。

在以上 3 个理想条件的基础上,*CS* 基于莱维飞行机制的位置更新公式如下

$$z^{k+1} = z^k + \alpha(z^k - z_{best}) \oplus Le'vy(\lambda) \tag{1}$$

其中莱维飞行可以利用 Mantegna 方法进行模拟

$$Le'vy(\lambda) = \frac{\mu}{|\nu|^{1/\beta}} \tag{2}$$

式(1)中,z^{k+1} 与 z^k 为第 k+1 代与第 k 代的宿主巢穴位置,z_{best} 为当前最优解,α 为步长控制因子,$\oplus$为点对点乘法,*Levy*(λ)为莱维飞行公式。

式(2)中,β=1.5,$\mu \sim N(0,\sigma_\mu^2)$,$\nu \sim N(0,1)$。正态分布的标准差 σ_μ 根据式(3)计算

$$\sigma_\mu = \left\{\frac{\Gamma(1+\beta) * \sin(\pi\beta/2)}{\Gamma[(1+\beta)/2] * \beta * 2^{(\beta-1)/2}}\right\}^{1/\beta} \tag{3}$$

宿主鸟发现寄生鸟蛋的概率称为发现概率 p_a,在 *CS* 算法中常取 p_a=0.25。*CS* 算法通过莱维飞行更新位置后,生成一个随机数 $r \in [0,1]$,当 $r>p_a$ 时,则对 z^k 进行随机改变,采用偏好游走的方式生成等量的新解以替换旧解,象征着宿主发现并处理掉寄生鸟蛋的行为,使得布谷鸟随机选择一个新的巢穴进行寄生繁殖。偏好游走公式如式(4)所示

$$z^{k+1} = z^k + \eta(z_a^k - z_b^k) \tag{4}$$

式中,压缩因子 $\eta \sim U[0,1]$,z_a^k 和 z_b^k 表示第 k 代的随机两个解。

在处理实际的优化问题时,巢穴的位置代表待辨识变量的解值,巢穴的适应度代表待辨识变量取不同解的时候对应的目标函数值。*CS* 算法的实现过程如下:

1) 参数初始化。设置最大迭代次数、种群规模、搜索空间维度、搜索上下界,生成鸟巢初始位置 $Z_0=(z_1,z_2,\cdots,z_N)$,并定义目标函数为 $f(z_i)$,i=1,2,…,N;

2) 计算每个巢穴的对应适应度,选出最优函数值作为当前最优适应度;

3) 基于莱维飞行机制,利用式(1)对巢穴的位置进行更新,获得一组新的巢穴位置并计算对应的适应度,此时式(1)中的步长控制因子 α=0.01。选择新巢穴的最优适应度与更新前的巢穴最优适应度进行对比,获得当前最优适应度与当前最优巢穴位置;

4) 随机产生一个随机数 r,令 r 与 p_a 进行比较。若 r 大于 p_a 则利用式(4)对巢穴位置进行随机更新,否则巢穴位置不发生变化;

5) 若达到最大迭代次数或满足指定搜索精度时,输出全局最优巢穴位置,即全局最优解,否则循环步骤 2-4。

3 改进布谷鸟搜索算法

由式(2)与式(3)可知,Levy(λ)的值取决于正态分布随机数 μ 和 v 这导致莱维飞行搜索步长和方向都是高度随机变化的。对于步长控制因子而言,较大的控制因子可以增强全局搜索能力,但是局部搜索能力较差,容易造成搜索至在真值附近摆动;较小的控制因子会导致算法的效率低下,需要更多的迭代次数。为改善该问题,本文提出了 PE-VSCS 算法(Variable Step-size Cuckoo Search based on Population Entropy),将种群的分布熵引入到 CS 算法中,对步长控制因子进行改进。

熵为某一系统体系混乱程度的度量,种群熵为算法寻优过程中种群分布混乱程度的度量。假设布谷鸟种群总数为 N、搜索维度为 dim,将寻优搜索空间分为 K 个搜索子空间,每个子空间含有个体数为 $n_i(i=1,2,\cdots,K)$,则第 j 维度($j=1,2,\cdots,dim$)的分布熵为

$$S_j = -\sum_{i=1}^{K}\frac{n_i}{N}\ln\frac{n_i}{N} \tag{5}$$

式(5)为布谷鸟种群在寻优搜索空间中的分布熵的表达式。为使分布熵更能体现出寻优解的分布信息,一般搜索子空间 K 取较大值。S 越大,则种群越分散,表明算法未收敛,需要较大的更新步长进行全局搜索;S 越小,则种群越集中,表明算法开始收敛,需要较小的更新步长进行局部搜索。

为满足上述关系,对改进步长控制因子进行构造,其表达式如下

$$\alpha_j = \theta * \frac{S_j - S_{\min}}{S_{\max} - S_{\min}} \tag{6}$$

式中,α_j 为第 j 维的步长控制因子,$S_{min}=0$ 为分布熵理论最小值,$S_{max}=lnK$ 为分布熵的理论最大值,θ 为调整参数。由于一般 $K>N$,由式(5)可知,分布熵实际最小值为 0,实际最大值为

$$S'_{\max} = -N*\left(\frac{1}{N}\ln\frac{1}{N}\right) - (K-N)*0 = \ln N \tag{7}$$

则步长控制因子的取值范围为 $\alpha\in\left[0,\theta*\frac{\ln N}{\ln K}\right]$。令步长控制因子在[0,1]区间内变化,则 $\theta=\frac{\ln K}{\ln N}$。

然而当全部种群聚集在同一区域内,会导致分布熵 $S=0$,使得算法无法再继续更新。针对上述情况引入了最小取值规则,即当 $S=0$ 的时候,令步长控制因子 $\alpha=\frac{1}{K}$。

PE-VSCS 算法的实现过程如下:

1) 参数初始化。设置最大迭代次数、种群规模、搜索空间维度、搜索上下界、搜索子空间数量,生成鸟巢初始位置 $Z_0=(z_1,z_2,\cdots,z_N)$,并定义目标函数为 $f(z_i)$,$i=1,2,\cdots,N$;

2) 计算每个巢穴的对应适应度,选出最优函数值作为当前最优适应度;

3) 基于莱维飞行机制,利用式(1)对巢穴的位置进行更新,获得一组新的巢穴位置并计算对应的适应度,此时步长控制因子 α 由式(6)决定。选择新巢穴的最优适应度与更新前的巢穴最优适应度进行对比,获得当前最优适应度与当前最优巢穴位置;

4) 随机产生一个随机数 r,令 r 与 p_a 进行比较。若 r 大于 p_a 则利用式(4)对巢穴位置进行随机更新,否则巢穴位置不发生变化;

5) 若达到最大迭代次数或满足指定搜索精度时,输出全局最优巢穴位置,即全局最优解,否则循环步骤 2-4。

4 仿真研究

为验证 PE-VSCS 算法的寻优性能,本文选用了如表 1 所示的不同特点的测试函数[20]。针对不同的测试函数,利用 PE-VSCS 算法与 CS 算法各进行 50 次仿真,并记录结果进行分析。其中 $f_1(x)$-$f_4(x)$ 为多维函数,$f_5(x)$-$f_8(x)$ 为二维函数,$f_1(x)$ $f_5(x)$ $f_7(x)$ $f_8(x)$ 为多峰函数,其余为单峰函数。

记录每一次的仿真结果,将 50 次仿真结果中的最优值、最差值、平均值、标准差作为算法性能的评价指标。最优值代表了算法寻优可达到的精度上限,平均值代表了算法寻优的平均精度,标准差代表了算法的稳定性。

测试结果如表 3 至表 10 所示。图 1 至图 8 为 PE-VSCS 与 CS 算法的最优收敛曲线图,图 1 至图 4 所示曲线图的维度取值为 10,图 5 至图 8 所示曲线图的维度取值为 2。

4.1 测试函数

表 1 测试函数

函数名	函数表达式	解空间	最优值
Ackley	$f_1(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	$[-32,32]$	0
Sphere	$f_2(x) = \sum_{i=1}^{n}x_i^2$	$[-100,100]$	0
Schwefel's P2.22	$f_3(x) = \sum_{i=1}^{n}\lvert x_i\rvert + \prod_{i=1}^{n}\lvert x_i\rvert$	$[-100,100]$	0
Sum Square	$f_4(x) = \sum_{i=1}^{n}ix_i^2$	$[-10,10]$	0
Drop-Wave	$f_5(x) = -\frac{1+\cos\left(12\sqrt{x_1^2+x_2^2}\right)}{0.5(x_1^2+x_2^2)+2}$	$[-5.12,5.12]$	-1
Easom	$f_6(x) = -\cos(x_1)\cos(x_2)\exp(-(x_1-\pi)^2-(x_2-\pi)^2)$	$[-100,100]$	-1
Shubert	$f_7(x) = \left(\sum_{i=1}^{5}i\cos((i+1)x_1+i)\right)\left(\sum_{i=1}^{5}i\cos((i+1)x_2+i)\right)$	$[-10,10]$	-186.7309
Schaffer	$f_8(x) = 0.5 + \frac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{[1+0.001(x_1^2+x_2^2)]^2}$	$[-10,10]$	0

4.2 测试环境与算法参数

仿真环境操作系统为 Windows 7，CPU 为 Intel Core i5-6400 2.7GHz，运行内存为 8G，编程环境为 MATLAB R2014b。

算法的最大迭代次数根据测试函数的搜索难度适当调整。其余的参数设置见表 2[5]。

表 2 算法参数设置

算法名称	参数设置
CS	$P_a=0.25, n=25, \alpha=0.01$
PE-VSCS	$P_a=0.25, n=25, K=1000$

4.3 算法性能比较

表 3 Ackley 函数 $f_1(x)$ 仿真结果

维度/迭代次数	评价指标	PE-VSCS	CS
10/1000	Best	3.553e-15	1.032e-6
	Worst	1.174e-11	1.371e-4
	Mean	2.471e-13	2.878e-5
	Std	1.642e-12	2.904e-5
50/5000	Best	6.022e-12	0.879
	Worst	4.768e-5	3.011
	Mean	1.068e-6	1.645
	Std	6.666e-6	0.428

表 4 Sphere 函数 $f_2(x)$ 仿真结果

维度/迭代次数	评价指标	PE-VSCS	CS
10/1000	Best	3.573e-38	3.389e-16
	Worst	5.101e-35	5.860e-14
	Mean	4.277e-36	7.101e-15
	Std	9.223e-36	1.043e-14
50/5000	Best	9.292e-27	2.433e-18
	Worst	2.637e-24	3.066e-16
	Mean	5.133e-25	8.446e-17
	Std	5.997e-25	8.169e-17

表 5 Schwefel's P2.22 函数 $f_3(x)$ 仿真结果

维度/迭代次数	评价指标	PE-VSCS	CS
10/1000	Best	3.104e-22	8.754e-8
	Worst	1.015e-20	8.042e-7
	Mean	2.893e-21	3.482e-7
	Std	2.047e-21	1.684e-7
50/5000	Best	8.757e-16	5.661e-9
	Worst	8.018e-5	1.000e+10
	Mean	3.730e-6	8.038e+8
	Std	1.327e-5	2.712e+9

表 6 Sum Square 函数 $f_4(x)$ 仿真结果

维度/迭代次数	评价指标	PE-VSCS	CS
10/1000	Best	6.564e-40	9.861e-18
	Worst	1.498e-36	6.711e-15
	Mean	1.011e-37	4.562e-16
	Std	2.330e-37	9.532e-16
50/5000	Best	2.208e-27	2.978e-19
	Worst	1.015e-24	8.572e-17
	Mean	2.047e-25	1.555e-17
	Std	2.738e-25	1.715e-17

表 7 Drop-Wave 函数 $f_5(x)$ 仿真结果

维度/迭代次数	评价指标	PE-VSCS	CS
2/200	Best	-1	-0.9999
	Worst	-1	-0.9362
	Mean	-1	-0.9963
	Std	0	0.0125

表 8 Easom 函数 $f_6(x)$ 仿真结果

维度/迭代次数	评价指标	PE-VSCS	CS
2/300	Best	-1	-1
	Worst	-1	-0.9974
	Mean	-1	-0.9999
	Std	0	4.204e-4

表 9 Shubert 函数 $f_7(x)$ 仿真结果

维度/迭代次数	评价指标	PE-VSCS	CS
2/1000	Best	-186.7309	-186.7309
	Worst	-186.7309	-186.7309
	Mean	-186.7309	-186.7309
	Std	6.239e-14	2.365e-10

表 10 Schaffer 函数 $f_8(x)$ 仿真结果

维度/迭代次数	评价指标	PE-VSCS	CS
2/1000	Best	0	1.140e-13
	Worst	3.220e-13	9.716e-3
	Mean	6.441e-15	2.291e-4
	Std	4.508e-14	0.0014

由上述测试结果可以看出，与 CS 算法进行对比，PE-VSCS 算法的性能大幅度提高。

面对低维度寻优问题时，PE-VSCS 算法的收敛速度更快。从表 7 与表 8 可以看出，在面对一些较简单的测试函数时，PE-VSCS 算法能在最大迭代次数较少的情况下，获得最优值并使得标准差为 0，表明面对该类问题时，PE-VSCS 算法能得到最优解且稳定性高。而 CS 算法在同样的最大迭代次数内，无法稳定寻出最优值甚至无法寻出最优值。从表 9

与表 10 可以看出，面对一些难度较高的测试函数时，PE-VSCS 算法的四个评价指标的数量级均小于 CS 算法的评价指标数量级，表明 PE-VSCS 算法的稳定性与寻优精度优于 CS 算法。

面对多维问题时，PE-VSCS 算法能达到一个很高的精度，优于 CS 算法。从表 4 与表 6 可以看出，PE-VSCS 算法可以使得评价函数的数量级达到-30 甚至-40，在一些实际问题中该数量级的误差可近似视为 0。而 CS 算法所达到的数量级与 PE-VSCS 算法相比，数量级相差了 10 甚至 20，表明了 PE-VSCS 算法的寻优精度要强于 CS 算法。

随着寻优维度的增加，PE-VSCS 算法的寻优能力有所下降，如表 3 至表 6 所示，当寻优维度为 50 时，需要的迭代次数相比于 10 维度而言变大，且四个评价指标相比于低维度的寻优精度而言变差。但是与 CS 算法相比，仍有显著的提高，PE-VSCS 算法与 CS 算法的四个指标相差至少 5 个数量级。

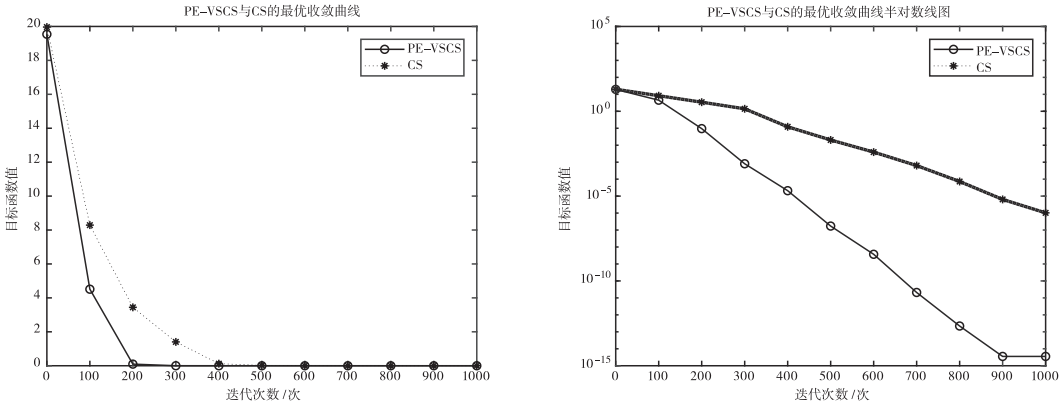


图 1　基于 $f_1(x)$ 的 PE-VSCS 与 CS 算法的最优收敛曲线图

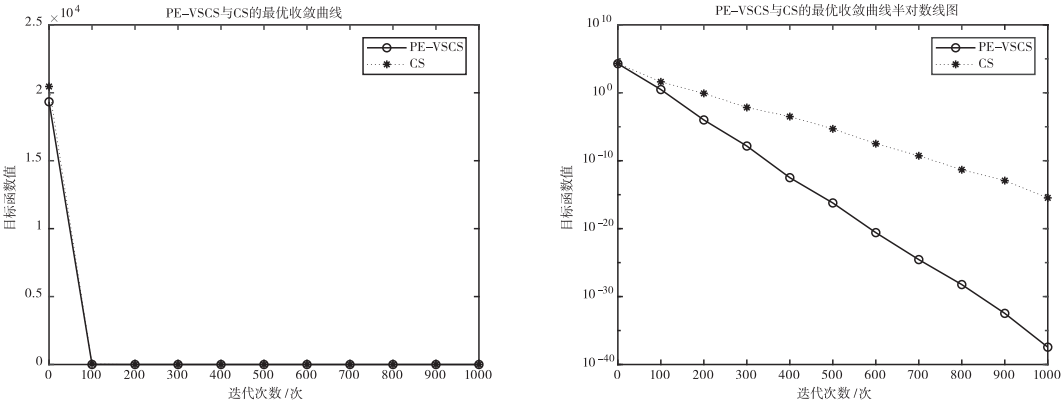


图 2　基于 $f_2(x)$ 的 PE-VSCS 与 CS 算法的最优收敛曲线图

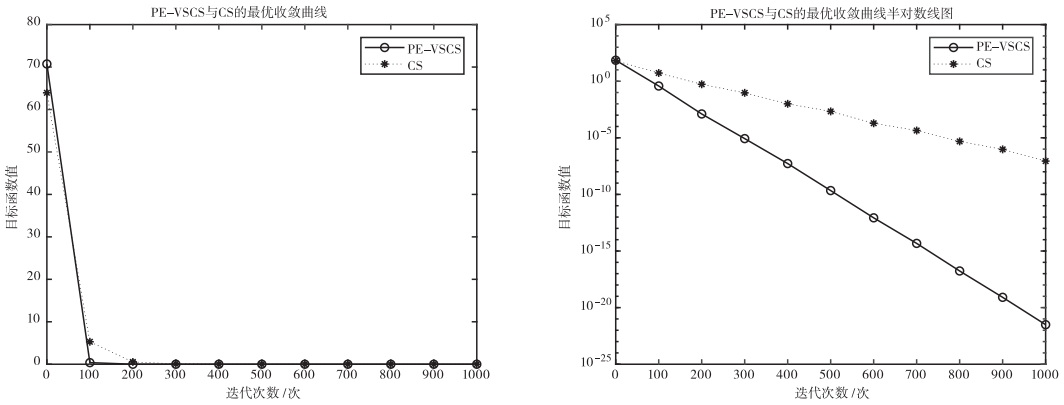
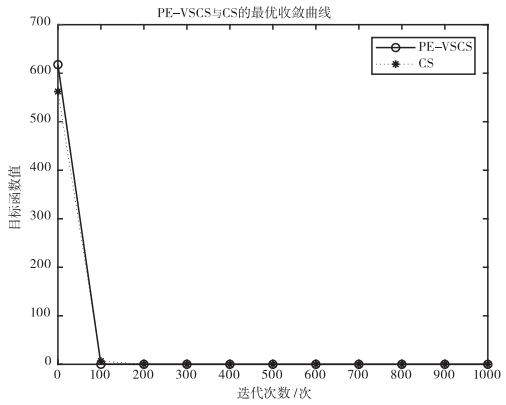


图 3　基于 $f_3(x)$ 的 PE-VSCS 与 CS 算法的最优收敛曲线图



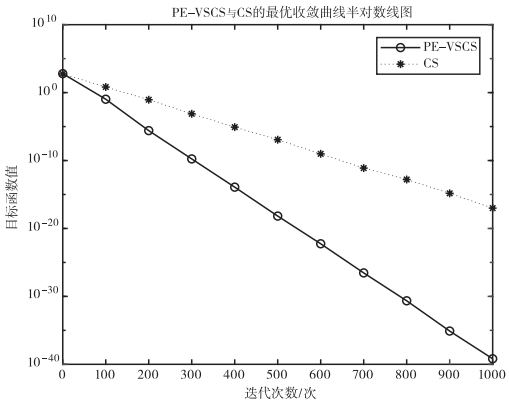


图 4　基于 $f_4(x)$ 的 PE-VSCS 与 CS 算法的最优收敛曲线图

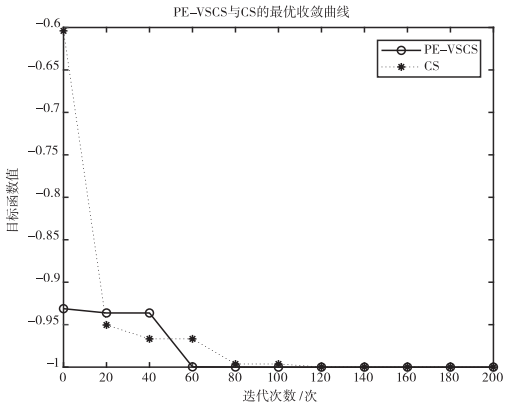


图 5　基于 $f_5(x)$ 的 PE-VSCS 与 CS 算法的最优收敛曲线图

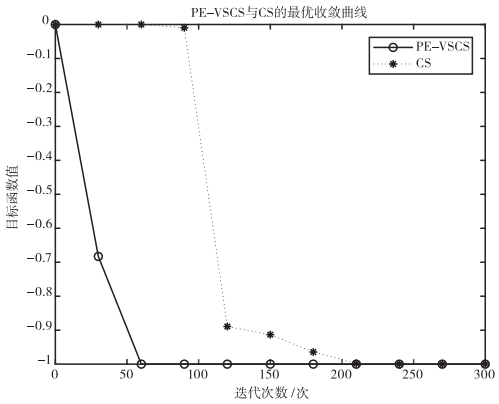


图 6　基于 $f_6(x)$ 的 PE-VSCS 与 CS 算法的最优收敛曲线图

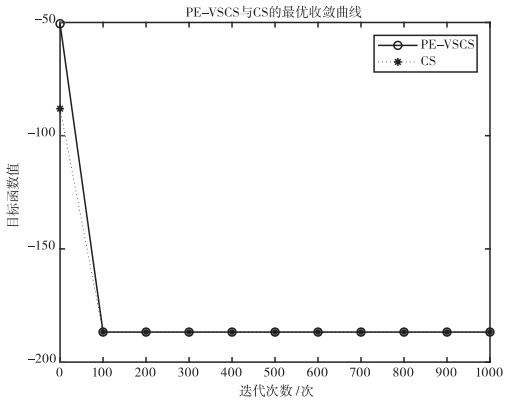


图 7　基于 $f_7(x)$ 的 PE-VSCS 与 CS 算法的最优收敛曲线图

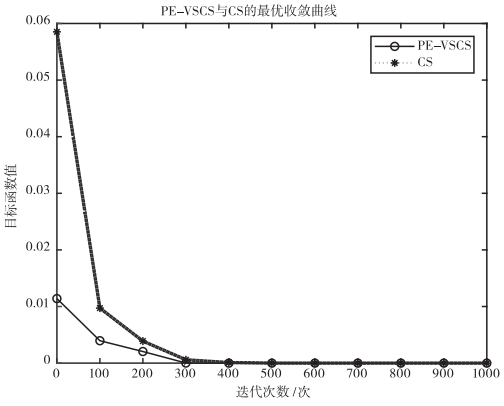


图 8　基于 $f_8(x)$ 的 PE-VSCS 与 CS 算法的最优收敛曲线图

4.4　算法收敛曲线分析

图 1 至图 4 为 10 维度测试函数的收敛曲线与对应的半对数线图，图 5 至图 8 为二维度测试函数的收敛曲线图。

多峰函数 $f_1(x)$、$f_5(x)$、$f_7(x)$、$f_8(x)$ 拥有多个局部极值点，可以测试算法跳出局部最优的能力。单峰函数 $f_2(x)$、$f_3(x)$、$f_4(x)$、$f_6(x)$ 可以测试算法的收敛速度。由曲线图可以看出 PE-VSCS 算法的收敛速度要优于 CS 算法。由半对数线图可以看出 PE-VSCS 在指定最大迭代次数内，评价函数的量级要优于 CS 算法所能达到的量级。

综上所述，通过八个测试函数的收敛曲线图进行对比分析，无论是低维度寻优还是高维度寻优，无论是单峰函数还是多峰函数，PE-VSCS 算法都具有优秀的寻优能力与收敛速度，证明了引入种群熵的步长控制因子的有效性

4.5 改进步长控制因子的分析

以 Easom 测试函数的测试结果的步长控制因子的变化为例,给出各维步长控制因子随迭代次数变化的曲线如图 9 所示。

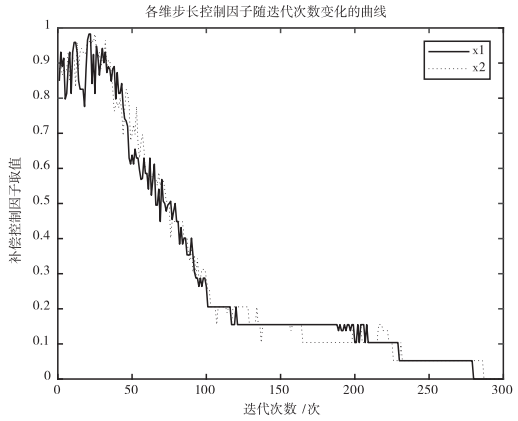


图 9 各维步长控制因子随迭代次数变化的曲线

由图 9 可以看出,各维步长控制因子随种群熵不断变化,整体呈现下降趋势。在迭代初期时,步长控制因子取较大值以提高前期的全局搜索能力;迭代后期时,步长控制因子处于一个较小值,提高了局部搜索能力。在迭代过程中,步长控制因子会偶尔出现变大的情况,以对应由于随机更新导致的种群分散情况。算法随迭代次数的增加逐渐由全局搜索转为局部搜索。

5 总结

熵可以度量某一系统体系混乱程度,CS 算法寻优模拟了布谷鸟种群从众多分散的巢穴中寻找出最优巢穴进行寄生繁殖的过程,种群从发散最终收敛于某一点,种群熵也随之变化从大变小。种群熵的变化代表了种群在寻优过程中的变化。本文通过将种群熵引入到莱维飞行机制中,构建了新的步长控制因子。通过 8 个测试函数的性能测试,可以表明 PE-VSCS 算法的寻优能力、求解精度、收敛速度等方面相比于 CS 算法均有提高。仿真结果表明,无论是单峰函数还是多峰函数,无论是低维函数还是多维函数,PE-VSCS 算法的寻优性能均超过 CS 算法,证明了基于种群熵的改进步长控制因子是有效的。且这项改进保持了 CS 算法的通用性,可配合其它改进策略来进一步提高算法的性能。本课题组将在后续工作中开展相关工作。

参考文献:

[1] Loris,Serafino,谢金星,等. 无导数优化:没有免费午餐定理的真实含义[J]. 数学译林,2015,(3):233-239.

[2] Kennedy J. Particle swarm optimization[J]. Proc. of 1995 IEEE Int. Conf. Neural Networks,(Perth,Australia),Nov. 27 - Dec. 2011,4(8):1942-1948.

[3] Goldberg D E. Genetic Algorithm in Search Optimization and Machine Learning[J]. Addison Wesley,1989,(7):2104-2116.

[4] Karaboga D. Artificial Bee Colony Algorithm[J]. Scholarpedia,2010,5(3):6915.

[5] Yang X S, Deb S. Cuckoo Search via Levy Flights [J]. mathematics,2010.

[6] 朱浩亮,李光平. 基于改进布谷鸟搜索算法的图像分割[J]. 计算机工程与设计,2018,39(5):1428-1432,1456.

[7] Xiaohua Zhu,Ning Wang. Cuckoo search algorithm with membrane communication mechanism for modeling overhead crane systems using RBF neural networks[J]. Applied Soft Computing,2017,56.

[8] 李爱菊,钮文良,王廷梅. 改进布鸟搜索算法最大熵值的医学图像分割[J]. 计算机仿真,2014,31(8):421-426.

[9] T Kumaresan,C Palanisamy. E-mail spam classification using S-cuckoo search and support vector machine[J]. Int. J. of Bio-Inspired Computation,2017,9(3).

[10] 侯启真,李泽,姬雨初,王阳. 基于 CS-BPNN 算法的飞机客舱 PMV 指标预测[J]. 计算机仿真,2020,37(3):52-55,99.

[11] 金明春,白云. 基于混合群智能算法的交通路径研究[J]. 计算机仿真,2020,37(9): 109-112,144.

[12] 赵坤,覃锡忠,贾振红. 采用改进的布谷鸟算法优化极限学习机[J]. 计算机仿真,2018,35(11):236-241.

[13] Rodrigues D,Pereira L A M,Almeida T N S,et al. BCS: A Binary Cuckoo Search algorithm for feature selection[C]. IEEE International Symposium on Circuits & Systems. IEEE,2013.

[14] 冯登科,阮奇,杜利敏. 二进制布谷鸟搜索算法[J]. 计算机应用,2013,33(6):1566-1570.

[15] Aziz Ouaarab, Belaïd Ahiod, Xin-She Yang. Discrete cuckoo search algorithm for the travelling salesman problem[J]. Neural Computing and Applications,2014,24(7-8).

[16] 王超,刘超,穆东,高扬. 基于离散布谷鸟算法求解带时间窗和同时取送货的车辆路径问题[J]. 计算机集成制造系统,2018,24(3): 570-582.

[17] 马英辉,吴一全. 利用混沌布谷鸟优化的二维 Renyi 灰度熵图像阈值选取[J]. 智能系统学报,2018,13(1):152-158.

[18] Wang G G, Deb S, Gandomi A H, et al. Chaotic cuckoo search [J]. Soft Computing,2016,20(9):3349-3362.

[19] 邵增珍,王洪国,刘弘. 具有启发式探测及自学习特征的降维对称微粒群算法[J]. 计算机科学,2010,37(5):219-222.

[20] 陈程,贺兴时,杨新社. 动态调整概率的双重布谷鸟搜索算法[J/OL]. 计算机科学与探索,2020-11-18:1-21

[作者简介]

刘洪达(1995.2-),男(汉族),黑龙江省望奎县人,硕士研究生,主要从事精密仪器方面的研究工作。

李德全(1991.11-),男(汉族),辽宁省海城市人,博士研究生,主要从事精密仪器方面的研究工作。

王 栋(1979.10-),男(汉族),山西省阳泉市人,博士,硕士研究生导师,副主任,研究员,主要从事精密仪器方面的研究工作。