DOI:10.13196/j.cims.2021.07.004

改进鲸鱼算法求解工程设计优化问题

刘景森[1,2],马义想[1],李 煜[3+]

(1.河南大学 软件学院,河南 开封 475004;
2.河南大学 河南省智能数据处理工程研究中心,河南 开封 475004;
3.河南大学 管理科学与工程研究所,河南 开封 475004)

摘 要:为了更好求解工程设计优化问题,提升鲸鱼算法的寻优性能和应用能力,提出一种基于分段式随机惯性权重和最优反馈机制的鲸鱼优化算法。在随机游走觅食策略中引入基于当前全局最优解的反馈机制,加快算法收敛速度,增强求解稳定性;在收缩包围策略和螺旋气泡网捕食策略中引入分段式随机惯性权重,提高算法的寻优精度和跳出局部极值的能力;对越界处理进行修正和改进,消除了进化成果可能丢失的隐患。通过理论分析证明了该改进算法与基本鲸鱼算法的时间复杂度相同。6 种代表性对比算法在 12 个复杂基准测试函数和 3 个工程优化设计问题上的实验结果表明,该改进算法的寻优性能、求解稳定性、对不同问题的适用性和有效性均明显优于其他 5 种对比算法。

关键词:工程设计;优化;鲸鱼优化算法;反馈机制;越界处理;时间复杂度

中图分类号:TP301.6 **文献标识码**:A

Improved whale algorithm for solving engineering design optimization problems

LIU Jingsen[1,2], *MA Yixiang*[1], *LI Yu*[3+]

(1. College of Software, Henan University, Kaifeng 475004, China;
2. Henan Provincial Intelligent Data Processing Engineering Research Center, Henan University, Kaifeng 475004, China;
3. Institute of Management Science and Engineering, Henan University, Kaifeng 475004, China)

Abstract: To better solve the engineering design optimization problems and improve the optimization performance and application ability of the whale optimization algorithm, the whale optimization algorithm based on piecewise random inertia weight and optimal feedback mechanism was proposed. For the random walk foraging strategy, a feedback mechanism based on the current global optimal solution was introduced to speed up the algorithm's convergence speed and enhance the stability of the solution. The piecewise random inertia weight was introduced into the shrinkage encirclement strategy and the spiral bubble net predation strategy, which improved the optimization accuracy and enhanced the ability of algorithm to jump out of the local extremum. The Cross-border processing was modified and improved to eliminate the potential loss of evolution results. Theoretical analysis proved that the improved algorithm had the same time complexity as the basic whale optimization algorithm. The experimental results of 6 representative comparison algorithms on 12 complex benchmark test functions and 3 engineering optimization design problems showed that the proposed algorithm had significantly better optimization performance, solution stability, applicability and effectiveness to different problems by comparing with 5 other comparison algorithms.

Keywords: engineering design; optimization; whale optimization algorithm; feedback mechanism; cross-border handling; time complexity

收稿日期:2019-09-06;**修订日期**:2020-02-24。Received 06 Sep. 2019; accepted 24 Feb. 2020.

基金项目:国家自然科学基金资助项目(71601071);河南省重点研发与推广专项资助项目(182102310886);河南大学研究生教育创新与质量提升资助项目(SYL18060145,SYL19050104)。**Foundation items**: Project supported by the National Natural Science Foundation, China (No. 71601071), the Key Research & Development and Promotion Program in Henan Province, China (No. 182102310886), and the Graduate Education Innovation and Quality Improvement Program of Henan University, China (No. SYL18060145, SYL19050104).

0　引言

优化是一个非常活跃的研究领域,它在指定约束条件下探索给定问题的最优解,主要目的是解决一些复杂的大规模工程问题。目前优化领域最常用的有效方法之一是基于仿生学的生物元启发式优化算法,如受蚂蚁寻找食物过程中释放信息素发现路径行为启发提出的蚁群算法(Ant Colony Optimization, ACO)[1-3];受飞鸟集群捕食行为启发提出的粒子群算法(Particle Swarm Optimization, PSO)[4-6];受布谷鸟寄生育雏行为启发提出的布谷鸟搜索算法(Cuckoo Search Algorithm, CSA)[7-8];受蝙蝠利用声呐探测猎物、避开障碍物启发提出的蝙蝠算法(Bat Algorithm, BA)[9-11];受花朵授粉方式不同启发提出的花朵授粉算法(Flower Pollination Algorithm, FPA)[12-13];受樽海鞘在海洋中游弋和觅食行为启发提出的樽海鞘群算法(Salp Swarm Algorithm, SSA)[14-15];模拟蝴蝶觅食行为及散发香味特性提出的蝴蝶优化算法(Butterfly Optimization Algorithm, BOA)[16-17]等。这些机制各有特点、性能各具优势的算法,在不断地提出、改善、比较和优胜劣汰中,为解决大规模复杂实际工程优化问题提供了良好的解决思路与设计方案,也吸引了大量国内外学者的关注与研究。

函数极值优化问题是测试算法寻优性能的主要方法,而工程设计约束优化问题则是启发式优化算法的一个重要应用领域和研究热点。对工程设计问题的优化方法大致分为确定性抽象方法和元启发式方法两大类。确定性抽象方法对初始值敏感,是解决简单优化问题的一种有效方法,却通常无法解决复杂工程设计问题。而元启发式方法在求解复杂工程设计问题时有着独特优势,它使用目标函数与概率而不是确定性规则,有更多的机会找到更好的工程优化问题解决方案,如:ZHANG 等[18]提出一种基于神经网络算法和教与学优化算法的有效混合方法;TSAI[19]提出一种改进的差分进化算法;DU 等[20]提出了基于递减步长和逻辑混沌映射的果蝇优化算法;ZHANG 等[21]提出一种基于布谷鸟搜索算法和差分进化算法的新型混合优化算法;DE MELO 等[22]提出一种多视图差分进化算法;KUMAR 等[23]提出一种改进的灰狼优化算法。这些改进算法均被应用于求解工程优化设计问题,并取得了较好效果,但其最优解质量和适用性都有进一步提升的空间。解决工程设计约束优化问题还需要探索寻优精度更高、普适性更强的算法。

鲸鱼优化算法(Whale Optimization Algorithm, WOA)是澳大利亚学者 MIRJALILI 等[24]受座头鲸特殊捕食行为启发提出的一种新型启发式搜索算法。该算法机制优越,寻优精度高,收敛速度快,且几乎不涉及参数设置,因此近年成为进化计算领域重要的研究和改进算法之一。如褚鼎立等[25]提出一种结合自适应权重和模拟退火的鲸鱼优化算法,提高了算法的寻优精度与收敛速度;CHEN 等[26]提出了融入莱维飞行和混沌局部搜索的改进鲸鱼优化算法,并将其应用到工程设计问题;吴泽忠等[27]提出一种基于改进螺旋更新位置模型的鲸鱼优化算法,提高了算法的普遍适用性和稳定性;OLIVA 等[28]提出一种基于混沌搜索的鲸鱼优化算法,提高了太阳能电池参数估计的准确性;肖子雅等[29]提出一种精英反向学习的黄金正弦鲸鱼优化算法,并通过仿真实验证明算法可以有效解决大规模问题,也可以应用于实际工程优化问题;WANG 等[30]提出一种基于鲸鱼优化算法的新型混合系统,提高了风速预测的精度与准确性;黄清宝等[31]提出一种基于余弦控制因子和多项式变异的鲸鱼优化算法,提高了函数极值优化的寻优精度和稳定性;JIANG 等[32]提出一种修改关键参数以实现分类搜索的鲸鱼优化算法,通过在 CEC 2014 函数的测试和 3 个约束工程问题的评估,证明了改进算法能够有效解决复杂、大规模、有约束的优化问题;ABDEL-BASSET 等[33]针对置换流车间调度问题提出一种将鲸鱼优化算法与基于贪婪局部搜索策略结合的算法,提高了算法的性能。

虽然这些改进在相应领域提升了算法的寻优和应用能力,但鲸鱼优化算法的寻优稳定性、跳出局部极值的能力和对于工程设计约束优化应用的求解效果与适用性仍需得到更大程度的改进和提高。为了更好求解工程设计优化问题,提升鲸鱼优化算法的应用能力和寻优性能,本文提出一种基于分段式随机惯性权重和最优反馈机制的鲸鱼优化算法(Feedback Whale Optimization Algorithm, FWOA)。在随机游走觅食策略中引入基于当前全局最优解的反馈机制,在维系种群多样性的同时避免随机选择个体的盲目性,加快算法收敛速度,增强求解稳定性;在收缩包围策略和螺旋气泡网捕食策略中引入分段式随机惯性权重,调节算法全局勘测能力和局部开

采能力的平衡，不仅加快了算法的收敛速度，也降低了后期因惯性权重单调递减直接陷入局部极值无法跳出的风险，有利于算法摆脱局部极值，找到全局最优值；对越界处理进行修正和改进，消除了已有进化成果可能丢失的隐患，解决了因较多个体越界导致越界处理后大量趋同，损害多样性的问题。本文通过理论分析证明了改进算法 FWOA 和基本鲸鱼算法 WOA 的时间复杂度相同，算法改进但并没有降低执行效率。通过对比 6 种算法在 12 个复杂基准测试函数上进行不同维度的函数极值优化实验，测试结果表明，该改进算法的寻优性能和求解稳定性明显优于其他 5 种代表性算法。而对焊接梁、拉伸弹簧和压力容器 3 个具有挑战性的工程设计约束优化问题的求解，其结果也充分显示了 FWOA 算法在处理不同类型工程优化设计问题上优越的求解效果和良好适用性。

1 基本鲸鱼优化算法

鲸鱼优化算法(WOA)将鲸鱼捕食猎物的行为分为随机游走觅食、收缩包围猎物和螺旋气泡网捕食 3 种不同的策略，其具体步骤如下：

步骤 1 初始化 WOA 算法各项参数：种群规模 N，空间维度 dim，迭代次数 Max_iter，鲸鱼初始种群 $x_i(i=1,2,\cdots,n)$和最优鲸鱼的位置与适应度初值。

步骤 2 进入迭代，先对整个种群集中进行边界条件处理，然后对种群中的每个个体分别进行进化处理，计算系数向量 $\boldsymbol{A}$ 和 $\boldsymbol{C}$，并产生均匀分布决策随机数 p。

$$\boldsymbol{A}=2\times\boldsymbol{a}\times\boldsymbol{r}-\boldsymbol{a}; \tag{1}$$

$$\boldsymbol{C}=2\times\boldsymbol{r}。 \tag{2}$$

其中：a 在整个迭代周期内随着迭代次数的增加从 2 下降到 0；$\boldsymbol{r}$ 为[0,1]上的随机向量。

步骤 3 当决策系数 $p<0.5$ 且 $|\boldsymbol{A}|\geqslant 1$ 时，当前鲸鱼在全局空间内进行随机游走觅食，勘测范围较广，此策略的数学模型为：

$$\boldsymbol{D}=|\boldsymbol{C}\times\boldsymbol{X}_{rand}-\boldsymbol{X}(t)|; \tag{3}$$

$$\boldsymbol{X}(t+1)=\boldsymbol{X}_{rand}-\boldsymbol{A}\times\boldsymbol{D}。 \tag{4}$$

其中：t 为当前迭代次数；$\boldsymbol{X}_{rand}$ 为当前种群中随机选择个体的位置向量；$\boldsymbol{X}(t)$、$\boldsymbol{X}(t+1)$分别为当前鲸鱼个体在第 t、$t+1$ 代的位置。

步骤 4 当决策系数 $p<0.5$ 且 $|\boldsymbol{A}|<1$ 时，说明当前座头鲸找到了猎物，正在进行收缩包围，此时将当前全局最优鲸鱼位置当作目标猎物位置(或最靠近目标猎物的位置)，鲸鱼以当前全局最优鲸鱼的位置为基础解进行位置更新，这一过程的数学模型如下：

$$\boldsymbol{D}=|\boldsymbol{C}\times\boldsymbol{X}_{best}-\boldsymbol{X}(t)|; \tag{5}$$

$$\boldsymbol{X}(t+1)=\boldsymbol{X}_{best}-\boldsymbol{A}\times\boldsymbol{D}。 \tag{6}$$

其中 $\boldsymbol{X}_{best}$ 为当前全局最优鲸鱼个体的位置。

步骤 5 当决策系数 $p\geqslant 0.5$ 时，表示鲸鱼正在进行螺旋气泡捕食。首先计算当前鲸鱼与当前最优位置鲸鱼之间的距离，然后在两者的位置之间建立一个螺旋方程，即鲸鱼以螺旋气泡网捕食猎物。模拟座头鲸螺旋捕食行为的数学模型如下：

$$\boldsymbol{D}'=|\boldsymbol{X}_{best}-\boldsymbol{X}(t)|; \tag{7}$$

$$\boldsymbol{X}(t+1)=\boldsymbol{D}'\times e^{bl}\times\cos(2\times\pi\times l)+\boldsymbol{X}_{best}。 \tag{8}$$

其中：$\boldsymbol{D}'$为当前位置鲸鱼到当前全局最优位置鲸鱼的距离；b 为用于定义对数螺旋形状的常数；l 为[−1,1]之间的均匀分布随机数。

步骤 6 判断是否符合结束条件，若符合则记录当前最优解及其目标值，否则转步骤 2 进行下一轮迭代。

2 基于分段式随机惯性权重和最优反馈机制的鲸鱼优化算法(FWOA)

2.1 基于最优解反馈机制的随机游走觅食

在鲸鱼优化算法的随机游走觅食策略中，随机选取种群中一头鲸鱼的位置作为基础解进行位置更新，这种完全随机性选择虽然可以维系种群的活跃性，但也增加了算法的不确定性，导致算法的稳定性较差，减缓了收敛速度。其实在随机游走觅食过程中，可以通过反馈机制，使得鲸鱼并不仅进行无目的游走，还与当前最优位置的鲸鱼进行信息交流，这种方式不仅可以保持种群的多样性，还可避免随机选择个体的盲目性，本文引入的反馈数学模型如下：

$$\boldsymbol{X}_{new\_rand}=\boldsymbol{X}_{rand}+rand\times(\boldsymbol{X}_{rand}-\boldsymbol{X}_{best})。 \tag{9}$$

其中：$\boldsymbol{X}_{rand}$ 为随机选择的个体；$\boldsymbol{X}_{best}$ 为最优鲸鱼位置。

由式(9)可以看出，新鲸鱼的位置是由随机鲸鱼位置和最优鲸鱼位置共同决定的，在保证种群活跃性的同时加快了收敛速度。引入反馈机制后的鲸鱼随机游走公式为：

$$\boldsymbol{X}(t+1)=\boldsymbol{X}_{new\_rand}-\boldsymbol{A}\times\boldsymbol{D}。 \tag{10}$$

2.2　分段式非线性递减惯性权重

动态惯性权重是平衡和调节算法全局勘测与局部开采能力的重要机制,较大的惯性权重意味着有较大的搜索步长,有利于算法进行全局搜索,增强跳出局部极值的能力,找到全局最优解;较小的惯性权重则是有利于算法进行局部开采,提高算法的收敛精度,加快收敛速度。而在鲸鱼优化算法的整个迭代期间,随着进化代数的增加,惯性权重的取值并没有发生改变,即采用较大的固定权重值,这虽然保证了算法的全局勘探能力,却不利于迭代后期鲸鱼向全局最优解附近进行精细搜索。

基于以上分析,本文将鲸鱼优化算法的收缩包围策略和螺旋捕食策略都分为前期、中期、后期 3 个进化阶段。在进化前期算法的惯性权重不变,即仍采用较大固定值,使算法在全局空间中进行充分搜索;而在中后期则引入非线性递减的动态惯性权重,使得鲸鱼向全局最优解附近靠近,加快算法收敛速度,引入的权重公式为:

$$\begin{cases}\omega = 1 & t < \dfrac{1}{3} \times Max\_iter \\ \omega = 1 - e^{rand \times \left(\frac{t}{Max\_iter}-1\right)} & \text{其他}\end{cases}。\tag{11}$$

式(11)中,惯性权重 ω 保持从 1 到 0 整体递减趋势的同时,也呈现出一定的随机性,这种随机性降低了算法如在中前期未能搜索到理论最优值附近,而在后期因惯性权重的单调递减直接陷入局部极值无法跳出的风险,有利于算法摆脱局部最优。加入惯性权重后,收缩包围策略与螺旋捕食策略的公式如下:

$$\boldsymbol{X}(t+1) = \omega \times \boldsymbol{X}_{best} - \boldsymbol{A} \times \boldsymbol{D};\tag{12}$$

$$\boldsymbol{X}(t+1) = \omega \times \boldsymbol{X}_{best} + \boldsymbol{D} \times e^{bl} \times \cos(2 \times \pi \times l)。\tag{13}$$

2.3　修正且改进的边界处理

鲸鱼优化算法中的边界条件处理,是在每一轮迭代的开始处对上轮迭代形成的整个种群中所有个体集中进行各维度越界检查与处理,这种处理方式存在问题。因为在一轮迭代之中,对种群中每一个鲸鱼个体进行位置更新时,就应当考虑该个体的新位置是否越界,而不是在整个种群都已更新完毕后的下一轮迭代开始处,再集中进行边界条件处理。由于鲸鱼在游走觅食策略的位置更新时,就已经用到了本代种群中随机选择的鲸鱼,如果采取现有的越界处理方式,会导致选择的随机个体可能是位置越界的鲸鱼,从而造成连锁反应引起更多个体越界,而大量的越界和下轮迭代开始处的集中式越界处理会使种群中大量个体重新随机化,中断和破坏了已有的进化成果,降低了算法的收敛速度和求解精度,这也是 WOA 求解不太稳定的一个重要原因。因此,本文采取将边界条件处理放在每个个体位置更新后立即进行,并对边界处理方法进行了改进。在 WOA 中,如果某一维或若干维的位置超出设定边界范围时,都是将其设置为边界的最大值或最小值,这样虽然可以使鲸鱼的位置限定在合法区域内,但却降低了种群的活跃性,如果越界情况较多,处理后的结果将过于趋同,不利于算法找到最优解。本文算法在进行边界条件处理时,会选择在靠近相应边界的区域内随机产生一个数值代替越界值,如果在某个维度上低于该维度的边界下限,就产生一个靠近此下限的值代替其值,如果在某维度上大于此维度的边界上限,则产生一个接近该上限的值代替其值,代码如下:

```
for i=1:dim
  for j=1:N
    if(u(i,j)<Xmin)
    u(i,j)=unifrnd(0,0.05)*(Xmax-Xmin)+Xmin
    end if
    if(u(i,j)>Xmax)
    u(i,j)=unifrnd(0.95,1)*(Xmax-Xmin)+Xmin
    end if
  end
end
```

由上述代码可以看出,改进公式不仅能增强种群的活跃性,还能提高算法的稳定性,符合改进规律,使边界处理更加合理和有效。

2.4　FWOA 算法流程

FWOA 算法描述如下:

```
初始化鲸鱼种群 xi(i=1,2,…,n)
设置最优鲸鱼位置与适应度的初值
while(t≤Max_iter)
  由公式(11)计算非线性惯性权重 ω
  for i=1:n
    由公式(3)和(4)更新系数向量 A 和 C 的值
    if p<0.5
      if|A|≥1
        由反馈公式(9)更新随机鲸鱼位置
        以随机鲸鱼位置为基础解,依据公式(10)进行随
        机游走觅食
      else
```

```
            由式(12)以当前最优解为基础进行收缩包围猎物
        end if
      else
        由式(13)以当前最优解为基础进行螺旋捕食
      end if
      进行边界条件处理
      计算新的适应度值,确定当前最优值和最优值位置
    end for
  end while
  输出结果
```

3 时间复杂度分析

时间复杂度体现了算法的运行效率,是评判算法性能优劣的重要因素。在基本鲸鱼优化算法中,假设种群规模为 N,个体位置的维度为 n,设置最优鲸鱼的初始位置与适应度初值的时间为 t_1,初始化鲸鱼个体位置中每一维的时间为 t_2,则初始化阶段的时间复杂度为:

$$T_1 = O(t_1 + N(n \cdot t_2)) = O(n)。$$

进入迭代后,总迭代次数为 Max_iter。假设对种群中每只鲸鱼每一维进行边界条件处理的时间为 t_3,计算目标函数适应度值的时间为 $f(n)$,与当前最优适应度值比较替换的时间为 t_4,系数向量 $\boldsymbol{A}$ 和 $\boldsymbol{C}$ 的计算时间为 t_5,则该阶段的时间复杂度

$$T_2 = O(N(n \cdot t_3 + f(n) + t_4 + t_5)) = O(n + f(n))。$$

假设种群中有 m_1 头鲸鱼进行随机游走觅食,m_2 头鲸鱼进行收缩包围猎物,m_3 头鲸鱼进行螺旋路径游走攻击猎物($N = m_1 + m_2 + m_3$,$0 \leqslant m_1, m_2, m_3 \leqslant N$),鲸鱼个体在执行这 3 种不同策略时,每维位置更新的时间分别为 t_6、t_7、t_8,则该阶段的时间复杂度

$$T_3 = O(N(m_1(n \cdot t_6) + m_2(n \cdot t_7) + m_3(n \cdot t_8))) = O(n)。$$

综上所述,WOA 总的时间复杂度

$$T = T_1 + Max\_iter(T_2 + T_3) = O(n + f(n))。$$

在 FWOA 算法中,种群规模为 N,个体维度为 n,均与基本鲸鱼优化算法一致。两者初始化过程一样,这一阶段的时间复杂度相同,均为 $T_1 = O(t_1 + N(n \times t_2)) = O(n)$。

进入迭代后,总迭代次数仍为 Max_iter。假设:惯性权重 w 的计算时间为 ξ_1,系数向量 $\boldsymbol{A}$ 和 $\boldsymbol{C}$ 的计算时间仍为 t_5,则该阶段的时间复杂度

$$T''_1 = \frac{2}{3} Max\_iter(O(N(\xi_1))) + Max\_iter(O(N(t_5))) = O(\xi_1 + t_5)。$$

同样假设种群中有 m'_1 头鲸鱼进行随机游走觅食,m'_2 头鲸鱼进行收缩包围猎物,m'_3 头鲸鱼进行螺旋路径游走攻击猎物($N = m'_1 + m'_2 + m'_3$,$0 \leqslant m'_1, m'_2, m'_3 \leqslant N$)。在进行随机游走觅食时,引入反馈机制的计算时间为 ξ_2,执行该策略鲸鱼个体每一维位置的更新时间为 ξ_3,而加入惯性权重后执行收缩包围猎物策略与螺旋气泡网攻击策略的鲸鱼个体每一维位置的更新时间分别为 ξ_4 和 ξ_5,则该阶段的时间复杂度

$$T''_2 = Max\_iter(O(m'_1(\xi_2 + n \cdot \xi_3) + m'_2(n \cdot \xi_4) + m'_3(n \cdot \xi_5)))) = O(n)。$$

假设鲸鱼个体每一维进行越界处理的时间为 ξ_6,计算目标函数适应度值的时间 $f(n)$、与当前最优适应度值比较替换的时间 t_4 都和基本 WOA 算法一致,则该阶段的时间复杂度

$$T''_3 = Max\_iter(O(N(n \cdot \xi_6 + f(n) + t_4))) = O(n + f(n))。$$

由上可得,改进算法 FWOA 的总时间复杂度

$$T'' = T_1 + T''_2 + T''_3 + T''_4 = O(n + f(n))。$$

基于上述分析可知,本文改进算法 FWOA 和基本鲸鱼优化算法 WOA 的时间复杂度相同,并没有降低算法的执行效率。

4 仿真实验

为了验证本文算法 FWOA 的整体性能,本文进行了两部分实验。4.1 节将算法在 12 个复杂无约束基准测试函数上进行低、中、高 3 种不同维度的函数极值优化测试,用以验证算法的寻优性能与收敛能力。4.2 节研究了 3 个具有挑战性的工程设计约束优化问题,用以检验算法在不同类型工程优化设计问题上的求解能力和应用潜力。两部分实验都将本文算法 FWOA 与基本鲸鱼优化算法(WOA)、A-CWOA(A-C parametric whale optimization algorithm)[34]、AWOA(adaptive whale optimization algorithm)[35]、蝙蝠算法(BA)、蝴蝶优化算法(BOA)共 6 种算法进行了对比。基于算法性能比较的公平性原则,6 种对比算法采用相同的软、硬件平台,运行环境为 Windows10 操作系统、编程语言为 MATLAB R2016a。仿真实验中,6 种算法的运行次数、种群规模、空间维度和最大迭代次数都保持一致,即 $N = 30$,$dim = 10/50/100$,$N\_iter = 1\,000$,每种算法分别独立运行 30 次。

在算法参数设置方面,4 个鲸鱼类算法中用于定义对数螺旋形状的常数 b 的值均设为 1;蝴蝶优

化算法中幂指数 a 的值为 0.1,感知形态 c 的初值为 $c=0.01$,转换概率 $p=0.8$;蝙蝠算法的声波响度 $A_0=0.25$,脉冲发动速率 $r_0=0.5$。6 个算法的参数设置采用的都是其各自基本算法的原有标准值,本文未另做挑选和更改,这进一步增强了实验结果与对比分析的客观性、公正性和可信性。

4.1　函数极值优化仿真实验

4.1.1　寻优精度分析

为了测试本文算法 FWOA 的求解性能,将其与上述 5 种对比算法在 12 个具有不同特征的基准测试函数上进行函数优化对比测试。具体测试函数如表 1 所示。

表 1　测试函数

函数名	函数公式	取值范围	最优值
Sphere	$F_1(x)=\sum_{i=1}^{D}x_i^2$	$[-100,100]$	0
Schwefel's problem 2.22	$F_2(x)=\sum_{i=1}^{d}\lvert x_i\rvert+\prod_{i=1}^{d}\lvert x_i\rvert$	$[-10,10]$	0
Expansion of F10	$F_3(x)=f_{10}(x_1,x_2)+\cdots+f_{10}(x_{i-1},x_i)+f_{10}(x_d,x_1)$ where $f_{10}(x,y)=(x^2+y^2)^{0.25}[\sin^2(50(x^2+y^2)^{0.1})+1]$	$[-100,100]$	0
Zakharov	$F_4(x)=\sum_{i=1}^{D}x_i^2+(\sum_{i=1}^{D}0.5x_i)^2+(\sum_{i=1}^{D}0.5x_i)^4$	$[-5,10]$	0
Expanded Schaffer's F6	$F_5(x)=f_s(x_1,x_2)+f_s(x_2,x_3)+\cdots+f_s(x_d,x_1)$ where $f_s(x,y)=0.5+\frac{\sin^2(\sqrt{x^2+y^2})-0.5}{(1+0.001(x^2+y^2))^2}$	$[-10,10]$	0
Schwefel's problem 2.21	$F_6(x)=\max\{\lvert x_i\rvert,1\leqslant i\leqslant d\}$	$[-100,100]$	0
Rotated hyper-ellipsoid	$F_7(x)=\sum_{i=1}^{d}\sum_{j=1}^{i}x_j^2$	$[-65,65]$	0
Powell	$F_8(x)=\sum_{i=1}^{d/4}[(x_{4i-3}+10x_{4i-2})^2+5(x_{4i-1}-x_{4i})^2+(x_{4i-2}-2x_{4i-1})^4+10(x_{4i-3}+x_{4i})^4]$	$[-10,10]$	0
Salomon	$F_9(x)=1-\cos(2\pi\sum_{i=1}^{d}x_i)+0.1\sum_{i=1}^{d}x_i^2$	$[-100,100]$	0
Schaffer F7	$F_{10}(x)=\frac{1}{d-1}\sum_{i=1}^{d-1}[(x_i^2+x_{i+1}^2)^{0.25}+(x_i^2+x_{i+1}^2)^{0.25}\sin^2(50(x_i^2+x_{i+1}^2)^{0.1})]$	$[-100,100]$	0
Levy	$F_{11}(x)=\sin^2(\pi w_1)+\sum_{i=1}^{D-1}(w_i-1)^2[1+10\sin^2(\pi w_i+1)]+(w_D-1)^2[1+\sin^2(2\pi w_D)]$ $w_i=1+\frac{x_i-1}{4},\forall i=1,\cdots,D$	$[-5.12,5.12]$	0
ACKley's	$F_{12}(x)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right)+20+e$	$[-32,32]$	0

表 1 所列的 12 个基准测试函数中:$F_1(x)$、$F_2(x)$、$F_4(x)$、$F_6(x)$、$F_7(x)$是单峰函数,全局搜索相对比较简单,大多数算法的优化结果都能达到理论最优值附近,主要用来测试算法的求解精度与收敛能力。$F_3(x)$、$F_5(x)$、$F_8(x)-F_{12}(x)$是多峰函数,这类函数大多具有多个极值点,用来检测算法能否摆脱局部极值找到全局最优解,其中:函数 $F_5(x)$的全局最优值周围有无数个极小值点,且全局最优值

与这些极小值点之间存在极大值点，因此该函数具有强烈的震荡特性，很难找到全局最优值，非常考验算法跳出局部极值的能力；而函数 $F_{10}(x)$ 是 $F_5(x)$ 的一个升级版函数，不但具有 $F_5(x)$ 的所有特点，而且函数图像是层层叠加的，求解难度上升了很多倍；函数 $F_{11}(x)$ 是通过增加正弦函数频繁产生局部极小值，且极小值的位置分布不是对称的；函数 $F_{12}(x)$ 则是通过指数函数进行调节，含有大量的障碍物，且极值点个数随着空间维度的增加呈梯度增加，更能考验算法求解高维复杂函数的能力。这些各具特点的不同函数都有一定的求解难度，很适于测试算法的性能。

4.1.2 寻优精度分析

表 2 统计了 6 种对比算法在空间维度为 10 维、50 维和 100 维的情况下，各自独立运行 30 次的最差解、最优解和平均值，并将各函数相应的最佳结果进行了加粗标注。

表 2 6 种算法在固定迭代次数下的寻优性能比较

函数	算法	$dim=10$			$dim=50$			$dim=100$		
		最差解	最优解	平均值	最差解	最优解	平均值	最差解	最优解	平均值
F_1	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	0	0	0	0	0	0	0	0	0
	A-CWOA	9.8E−262	2.1E−305	3.4E−263	1.7E−251	2.7E−278	5.8E−253	4.0E−248	4.0E−292	1.4E−249
	WOA	7.0E−157	6.6E−178	5.7E−158	8.5E−147	5.3E−166	2.9E−148	3.8E−145	2.2E−165	1.3E−146
	BA	1.46E+04	1019.7	4.82E+03	6.38E+04	14 430	3.49E+04	1.12E+05	49 120	7.07E+04
	BOA	1.77E−14	9.70E−15	1.43E−14	2.06E−14	1.64E−14	1.86E−14	2.03E−14	1.72E−14	1.88E−14
F_2	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	9.9E−293	5.0E−307	3.6E−294	1.2E−293	5.9E−308	4.6E−295	2.3E−292	3.0E−302	2.5E−293
	A-CWOA	2.3E−140	8.6E−157	1.0E−141	5.9E−132	1.2E−152	2.0E−133	9.5E−135	2.8E−147	4.8E−136
	WOA	4.8E−102	2.6E−118	1.6E−103	4.4E−102	2.1E−112	1.7E−103	4.2E−102	3.5E−112	3.5E−103
	BA	5.84E+01	0.335 51	1.86E+01	5.42E+17	8.03E+01	1.81E+16	6.69E+35	1.10E+02	2.23E+34
	BOA	1.14E−11	8.87E−12	1.03E−11	3.71E+24	8.53E−20	1.67E+23	2.09E+51	5.64E+42	7.80E+49
F_3	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	0	0	0	0	0	0	0	0	0
	A-CWOA	2.53E−65	8.28E−78	8.45E−67	4.02E−66	2.37E−77	1.37E−67	9.16E−66	3.45E−75	5.16E−67
	WOA	2.01E−55	1.18E−65	8.37E−57	1.03E−58	1.19E−66	6.1E−60	1.99E−60	9.89E−67	1.72E−61
	BA	8.21E+01	3.87E+01	6.12E+01	4.35E+02	2.69E+02	3.45E+02	8.77E+02	5.97E+02	7.25E+02
	BOA	9.30E−01	1.60E−08	1.35E−01	3.61E−09	4.46E−10	1.37E−09	4.95E−09	7.66E−10	1.91E−09
F_4	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	0	0	0	0	0	0	7.1E−185	0	2.8E−186
	A-CWOA	1.3E−200	2.1E−231	4.7E−202	9.0E−159	8.5E−182	3.7E−160	9.9E−101	1.9E−125	3.3E−102
	WOA	1.81E+00	7.49E−18	1.76E−01	1.22E+03	629.07	8.73E+02	2.11E+03	1137.4	1.66E+03
	BA	4.19E+02	9.08E−07	6.56E+01	1.22E+04	1.07E+02	3.58E+03	6.08E+04	4.04E+03	1.62E+04
	BOA	1.74E−14	1.34E−14	1.59E−14	1.93E−14	1.42E−14	1.70E−14	1.94E−14	1.49E−14	1.70E−14
F_5	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	0	0	0	0	0	0	0	0	0
	A-CWOA	0	0	0	0	0	0	0	0	0
	WOA	1.06E+00	0	5.57E−01	1.13E+01	0	6.33E+00	2.48E+01	0	8.83E+00
	BA	1.75E+00	2.07E−01	7.97E−01	1.12E+01	2.85E+00	5.45E+00	2.80E+01	9.88E+00	1.46E+01
	BOA	1.41E+00	5.86E−01	1.13E+00	1.68E+01	1.34E+01	1.56E+01	3.89E+01	3.28E+01	3.63E+01

续表 2

F_6	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	2.5E−277	1.2E−289	8.8E−279	1.5E−271	1.7E−288	5.1E−271	2.7E−273	4.8E−284	1.2E−274
	A-CWOA	1.5E−111	1.3E−128	8.75E−01	9.33E+01	5.0555	7.13E+01	9.41E+01	7.407 9	6.35E+01
	WOA	1.36E+01	0	4.56E−01	2.04E+01	0	3.103 24	2.86E+01	0	1.048 22
	BA	5.92E+01	1.92E+01	3.64E+01	7.21E+01	3.72E+01	5.71E+01	7.26E+01	4.94E+01	6.17E+01
	BOA	1.30E−11	7.93E−12	1.13E−11	1.40E−11	1.05E−11	1.22E−11	1.37E−11	1.06E−11	1.24E−11
F_7	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	0	0	0	0	0	0	0	0	0
	A-CWOA	3.0E−253	1.0E−296	9.9E−255	2.3E−245	3.8E−279	7.5E−247	3.4E−244	4.8E−282	1.9E−245
	WOA	2.0E−154	5.9E−185	6.8E−156	6.2E−146	1.6E−172	2.3E−147	1.7E−140	2.3E−166	5.5E−142
	BA	9.10E+04	5.50E+03	2.98E+04	1.94E+07	2.04E+06	6.31E+06	8.02E+07	2.62E+07	4.75E+07
	BOA	1.86E−14	1.13E−14	1.59E−14	2.17E−14	1.65E−14	1.97E−14	2.19E−14	1.81E−14	1.97E−14
F_8	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	0	0	0	0	0	0	0	0	0
	A-CWOA	1.27E−20	3.0E−292	4.3E−208	2.2E−239	6.0E−289	7.2E−241	4.8E−252	7.7E−285	1.6E−253
	WOA	2.99E−06	6.47E−63	3.70E−07	1.6E−144	5.1E−167	5.4E−146	3.4E−145	2.0E−166	1.7E−146
	BA	7.15E+03	1.07E−03	9.31E+02	3.27E+05	1.69E+04	1.08E+05	9.04E+05	1.28E+05	3.52E+05
	BOA	1.85E−14	9.11E−15	1.38E−14	2.00E−14	1.65E−14	1.83E−14	2.08E−14	1.61E−14	1.86E−14
F_9	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	0	0	0	0	0	0	0	0	0
	A-CWOA	2.9E−255	3.4E−290	9.6E−257	2.8E−246	9.8E−288	9.3E−248	9.0E−241	2.1E−276	3.0E−242
	WOA	1.56E−02	5.86E−09	5.19E−03	5.89E−03	1.1E−109	2.05E−03	4.10E−03	2.25E−13	1.21E−03
	BA	8.03E+02	1.90E+02	4.44E+02	6.78E+03	2.12E+03	3.73E+03	1.61E+04	4.07E+03	8.03E+03
	BOA	3.34E−19	1.18E−20	9.04E−20	2.48E−19	7.89E−21	6.27E−20	2.92E−19	3.04E−21	6.20E−20
F_{10}	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	0	0	0	0	0	0	0	0	0
	A-CWOA	1.50E−70	1.00E−79	7.03E−72	3.79E−68	1.61E−76	1.93E−69	1.44E−68	5.29E−77	8.52E−70
	WOA	7.02E−52	4.37E−66	2.34E−53	3.16E−61	6.89E−70	3.0E−62	2.41E−60	5.46E−68	1.09E−61
	BA	7.50E+00	4.27E+00	5.87E+00	9.49E+00	5.31E+00	7.15E+00	9.70E+00	6.43E+00	7.33E+00
	BOA	2.31E−01	8.25E−08	8.52E−02	9.79E−08	1.59E−09	2.02E−08	7.27E−09	1.49E−09	4.09E−09
F_{11}	FWOA	**2.07E−03**	**5.86E−07**	**4.89E−04**	**1.24E−01**	**3.63E−06**	**2.97E−02**	**5.85E−01**	**1.82E−04**	**1.25E−01**
	AWOA	9.34E−02	6.55E−05	7.25E−03	4.33E−01	5.60E−04	1.08E−01	8.65E−01	5.93E−02	3.36E−01
	A-CWOA	2.28E−03	1.99E−05	5.62E−04	2.56E−01	1.21E−02	8.90E−02	1.30E+00	3.06E−02	3.32E−01
	WOA	2.72E−01	1.99E−05	5.48E−02	9.64E−01	3.53E−02	2.98E−01	2.23E+00	2.43E−01	7.66E−01
	BA	6.29E+00	8.95E−02	2.14E+00	2.42E+01	3.59E+00	8.66E+00	6.42E+01	6.37E+00	2.43E+01
	BOA	4.34E−01	1.08E−01	2.95E−01	4.77E+00	3.53E+00	4.20E+00	9.49E+00	8.53E+00	9.18E+00
F_{12}	FWOA	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	AWOA	0	0	0	0	0	0	0	0	0
	A-CWOA	3.55E−15	0	3.55E−16	3.55E−15	0	5.92E−16	3.55E−15	0	8.29E−16
	WOA	7.11E−15	0	3.08E−15	7.11E−15	0	3.08E−15	7.11E−15	0	3.67E−15
	BA	2.00E+01	1.10E+01	1.54E+01	1.79E+01	1.44E+01	1.65E+01	2.00E+01	1.47E+01	1.67E+01
	BOA	3.37E−12	2.84E−14	8.08E−13	1.38E−11	1.06E−11	1.23E−11	1.25E−11	1.08E−11	1.16E−11

观察表 2 的数据可知，在空间维度从 10 维增加到 100 维的过程中，FWOA、AWOA、A-CWOA、WOA 和 BOA 5 种算法的寻优性能并没有出现太大程度的降低，这说明 WOA 和 BOA 有着较好的维度变化适应性，适于求解高维函数。而 BA 不但在寻优结果上不如这 5 种算法，而且维度变化适应性不佳，这也反映出最近几年一些新兴算法在寻优机制和能力上的进步。

观察 6 种算法在单峰函数上的寻优结果，可以看出各算法的整体开发能力。对于函数 $F_1(x)$和 $F_7(x)$，FWOA 和 AWOA 算法在不同维度下 30 次运行所找到的最差解、最优解和平均值都是理论最优值，其寻优性能远远超过 A-CWOA、WOA、BOA 和 BA 4 种算法。而对于函数 $F_2(x)$和 $F_6(x)$，虽然 AWOA 算法找到的最优解非常接近于理论最优值，但仍稍逊于最差解、最优解和平均值都是理论最优值的 FWOA 算法。对于余下的单峰函数 $F_4(x)$，这是 12 个函数中唯一使鲸鱼类算法随着维度增加寻优性能降低的函数，不仅 WOA 和 A-CWOA 表现不佳，就连寻优性能一向良好的 AWOA 表现也不尽人意，由表 2 中 $F_4(x)$的数据可知，AWOA 算法在 10 维和 50 维下都找到了理论最优解，但仔细观察其在 100 维下的平均值和最差解可知，AWOA 在这个函数上的表现并不稳定，而本文改进算法不但找到了全局理论最优，而且测试结果非常稳定，说明 FWOA 在单峰函数上的表现更好，高维适应性更强，算法的整体开发能力更优。而 BOA 虽然也受维度变化影响不大，但其整体的开发能力却略逊于鲸鱼类算法；仔细观察 BA 在这 5 个单峰函数上的表现可以发现，该算法在搜索区间较小的函数 $F_2(x)$和 $F_4(x)$上的寻优结果要远远优越于搜索区间较大的其余 3 个函数，这说明 BA 的局部搜索能力较好，而全局寻优能力相对较弱，算法进行大范围全面搜索时跳出局部最优的能力不足。

与单峰函数不同，多峰函数包含许多局部最优解，这些局部最优解的数量随着空间维度的增加呈指数增长，非常适于测试算法跳出局部极值的能力和对于维度变化的适应性。观察 6 种算法在多峰函数上的寻优结果，可以看到除了函数 $F_{11}(x)$，FWOA 和 AWOA 在不同维度下 30 次运行找到的最差解、最优解和平均值都是理论最优值，说明这两种算法具有很好的探测能力且跳出局部极值的能力较强。而 A-CWOA、WOA、BOA 和 BA 4 种算法在除 $F_5(x)$和 $F_{12}(x)$以外的多峰函数上找到的最差解、最优解和平均值都不是全局理论最优。与在单峰函数上表现类似，BA 仍然在较大区间内搜索效果不太好，很容易陷入局部极值难以跳出。对于小搜索区间的多峰函数 $F_5(x)$，4 个鲸鱼类算法找到的最优解都是全局理论最优，而 BA 和 BOA 求得的最优解也很接近于此，但仔细观察在较大区间内搜索的多峰函数 $F_3(x)$、$F_9(x)$、$F_{10}(x)$、$F_{12}(x)$，BA 找到的最优解不仅远远落后于 WOA，与 BOA 相比，寻优结果也差了很多。

对于函数 $F_{11}(x)$，6 种算法虽都没找到全局理论最优值 0，但 FWOA 在不同空间维度下找到的最差解、最优解和平均值都优于 AWOA、A-CWOA、WOA、BOA 和 BA 5 种算法，依然显示出更好的求解能力和维度适应性。

以上寻优精度结果表明，无论是单峰函数还是多峰函数，FWOA 在各个维度下每次运行的结果几乎都是全局理论最优，表现出优越的求解性能。这些测试结果充分说明了 FWOA 较好地解决 WOA 在函数极值优化上寻优精度不高、容易陷入局部极值、寻优性能不稳定的问题。

4.1.3 收敛曲线分析

为了更直观地比较 6 种算法的寻优性能，各算法在迭代次数相同、维度为 100 情况下求解多峰函数 $F_3(x)$、$F_5(x)$、$F_8(x)-F_{12}(x)$的收敛曲线如图 1～图 7 所示。多峰函数能体现算法跳出局部极值的能力，更能说明算法的寻优能力，单峰函数的收敛曲线比较简单且结果与多峰函数类似，不再赘述。

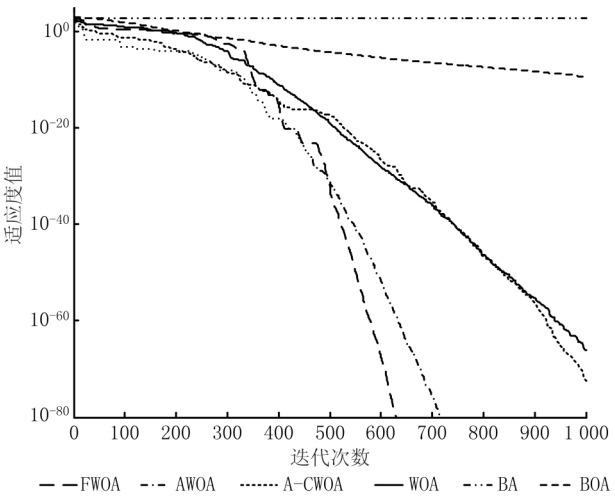

图1 F_3的收敛曲线

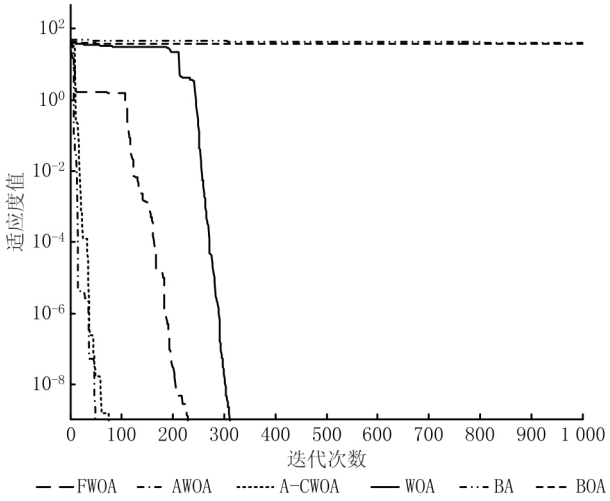

图2　F_5的收敛曲线

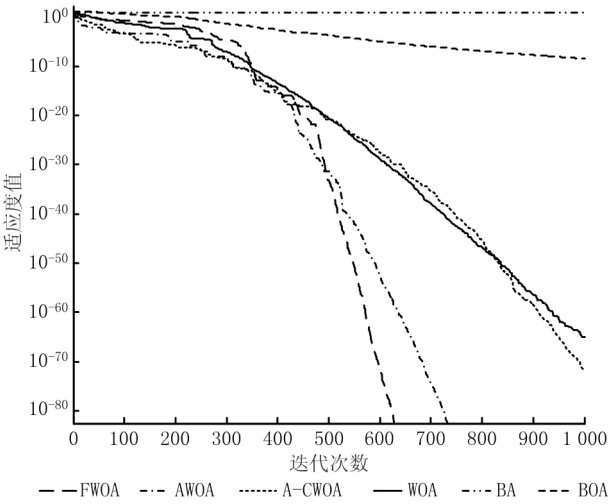

图5　F_{10}的收敛曲线

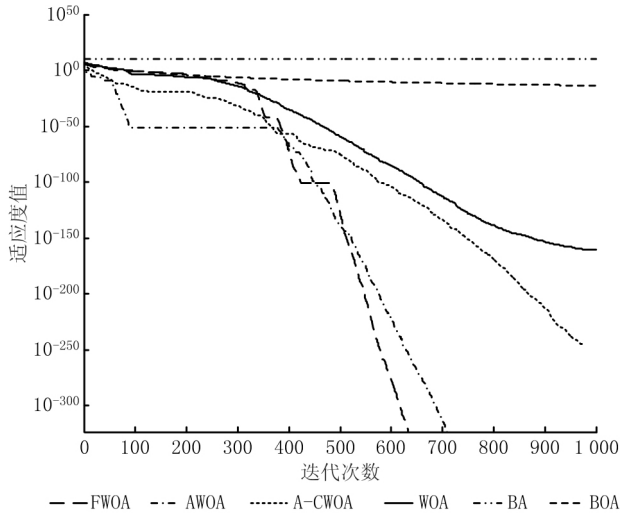

图3　F_8的收敛曲线

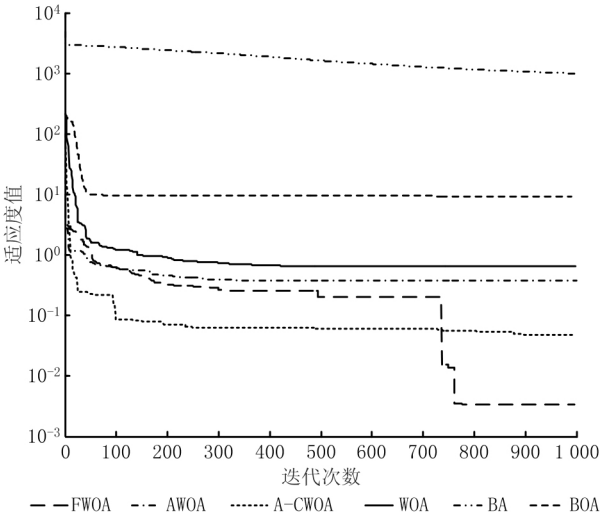

图6　F_{11}的收敛曲线

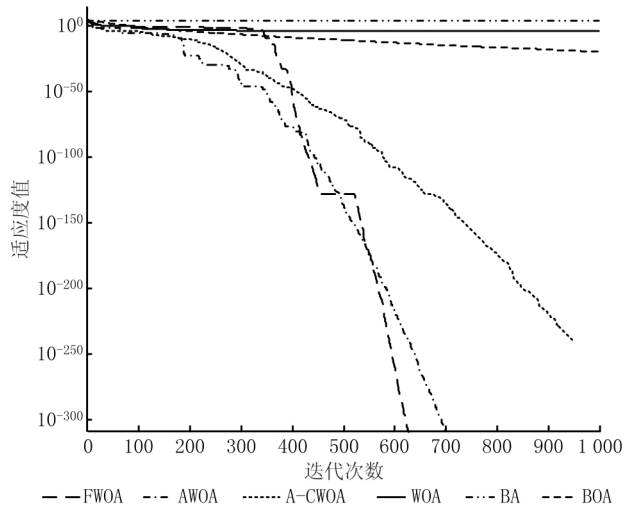

图4　F_9的收敛曲线

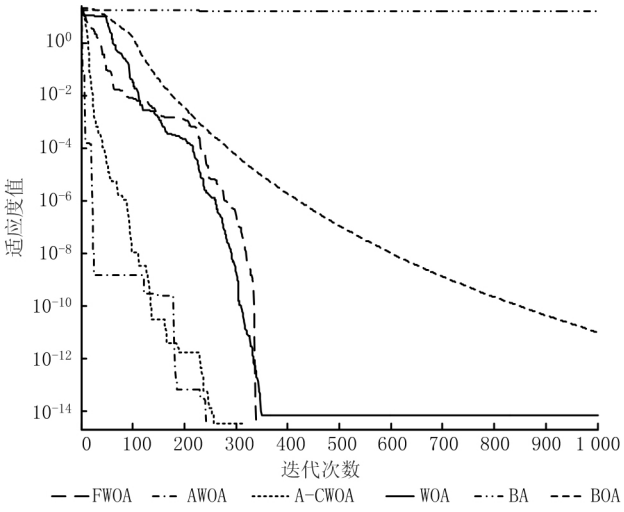

图7　F_{12}的收敛曲线

从以上 7 个多峰函数图上可以看出算法的收敛速度和摆脱局部极值、寻找全局最优值的能力。FWOA 在除了图 2 和图 7 以外的 5 个收敛曲线图上，收敛速度都是 6 种算法中最快的；虽在 $F_5(x)$和 $F_{12}(x)$上的收敛速度落后于 AWOA 和 A-CWOA，但同样也找到了全局理论最优值，且观察表 2 中函数 $F_{12}(x)$数据可以发现，A-CWOA 虽可以找到最优解，但并不稳定，而 FWOA 和 AWOA 算法找到的最优解、最差解和平均值都是全局理论最优，说明这两种算法不但跳出局部极值的能力较强，而且寻优过程中表现比较稳定。在图 1 和图 3～图 6 中，A-CWOA、WOA、BOA 三种算法的收敛曲线虽仍呈现下降趋势，但收敛速度比较缓慢，无法在 1 000 代内收敛到全局理论最优。BA 更是从一开始便陷入了局部极值，收敛曲线几乎呈现水平状态，下降趋势极不明显，完全无法在 1 000 代内收敛到全局最优值附近。

由 4.1.2 节的寻优精度和图 1～图 7 分析可知，在空间维度分别是 10 维、50 维、100 维的情况下，FWOA 表现出非常优越的求解性能，在测试的 12 个基准函数各个维度上几乎都找到了理论最优值，且 FWOA 的收敛速度大部分明显比 AWOA、A-CWOA、WOA、BA 和 BOA 算法更快。这是因为 FWOA 算法在随机游走觅食策略中引入了基于当前全局最优解的反馈机制，不仅加快了算法的收敛速度，还增强了求解的稳定性；而在收缩包围策略和螺旋捕食策略中引入分段式随机惯性权重，可以在加快算法收敛速度的同时，降低算法在中前期如果未能搜索到理论最优值附近便会在后期因惯性权重的单调递减直接陷入局部极值无法跳出的风险，有利于算法摆脱局部最优，找到全局最优值。而对越界处理的修正和改进，不但消除了已有进化成果可能丢失的隐患，也解决了若种群中较多个体越界而在越界处理后过于趋同损失多样性的问题，有利于算法找到最优解，提高收敛速度与寻优精度。上述实验充分说明 FWOA 具有出色的寻优性能，其寻优精度、收敛速度、跳出局部极值的能力、维度变化适应性和求解稳定性均优于 AWOA、A-CWOA、WOA、BA 和 BOA 5 种对比算法。

4.2 工程设计约束优化问题的求解与分析

为了验证本文算法 FWOA 求解工程约束优化问题的有效性，将 FWOA 和上述 5 种对比算法应用于焊接梁、拉伸弹簧和压力容器 3 个著名的标准工程优化设计问题。这 3 个工程问题具有各自不同的约束，本文采用通用的死刑惩罚函数机制处理这些约束条件，使算法在优化过程中自动丢弃不可行解。在求解这些工程优化问题时，每个算法均独立运行 100 次取其最佳寻优设计结果，参数设置也与 4.1 节相同。

4.2.1 求解焊接梁优化设计问题

焊接梁设计是一个著名的复杂约束工程设计问题，其目的是在剪应力、弯曲应力、屈曲载荷、端部挠度和侧面约束下找到最低制造成本。该问题有 7 个约束条件和 4 个设计变量，设计变量分别是焊缝厚度 $h(0.1\leqslant h\leqslant 2)$、焊缝长度 $l(0.1\leqslant l\leqslant 10)$、梁宽度 $t(0.1\leqslant t\leqslant 10)$、梁厚度 $b(0.1\leqslant b\leqslant 2)$，具体数学模型如下：

(1)设计变量：

$$x=[x_1x_2x_3x_4]=[hltb]。$$

(2)目标函数：

$$f(x)=1.10471x_1^2x_2+0.04811x_3x_4(14.0+x_2)。$$

(3)约束函数：

$$g_1(x)=\tau(x)-136\ 000\leqslant 0;$$

$$g_2(x)=\sigma(x)-30\ 000\leqslant 0;$$

$$g_3(x)=x_1-x_4\leqslant 0;$$

$$g_4(x)=0.104\ 71x_1^2+0.048\ 11x_3x_4(14.0+x_2)-5.0\leqslant 0;$$

$$g_5(x)=0.125-x_1\leqslant 0;$$

$$g_6(x)=\delta(x)-0.25\leqslant 0;$$

$$g_7(x)=6\ 000-P_c(x)\leqslant 0;$$

$$\tau(x)=\sqrt{(\tau')^2+2\tau'\tau''\frac{x_2}{2R}+(\tau'')^2};$$

$$\tau'=\frac{6\ 000}{\sqrt{2}x_1x_2},\tau''=\frac{MR}{J};$$

$$M=6\ 000\left(14+\frac{x_2}{2}\right),R=\sqrt{\frac{x_2^2}{4}+\left(\frac{x_1+x_2}{2}\right)^2};$$

$$J=2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{12}+\left(\frac{x_1+x_3}{2}\right)^2\right]\right\};$$

$$\sigma(x)=\frac{784\ 000}{x_4x_3^2},\delta(x)=\frac{4\times 600\times 14^3}{30\times 10^6x_3^3x_4};$$

$$P_c(x)=\frac{2.006\ 5\sqrt{x_3^2x_4^6}}{14^2}\left(1-\frac{50x_3}{28}\right)。$$

表 3 统计了 6 种算法在求解焊接梁设计问题时得到的最佳解决方案数据，由表 3 可以看出，本文算法 FWOA 在 $h=1.820\ 0$、$l=2.688\ 28$、$t=9.035\ 30$、$b=0.205\ 80$ 处得到的最小代价 $f(x)=$

1.591 25 是找到的焊接梁设计问题的最低成本。这是因为本文算法引入的反馈机制和惯性权重提高了算法的局部搜索能力,基本鲸鱼算法是一个更擅于全局搜索的算法,特别是在大范围空间搜索时,算法比较稳定,这一点从 4.1 节的求解函数极值实验部分就可以看出。而焊接梁的设计变量取值范围比较小并受到多个约束条件的限制,需要算法有着较强的局部搜索能力,而本文改进则很好地满足了这一要求。

表 3　6 种算法求解焊接梁设计问题的最佳解决方案

算法	X_1	X_2	X_3	X_4	最低成本
FWOA	0.182 00	2.688 28	9.035 30	0.205 80	1.591 25
WOA	0.164 91	2.869 99	9.080 15	0.205 51	1.600 77
A-CWOA	0.174 21	2.759 69	9.031 54	0.205 96	1.592 38
AWOA	0.172 90	2.863 43	9.052 21	0.205 65	1.604 88
BOA	0.187 97	2.700 08	8.579 88	0.234 35	1.720 84
BA	0.204 26	2.503 95	9.228 77	0.204 79	1.616 08

4.2.2　求解拉伸弹簧优化设计问题

拉伸弹簧设计是一个最小化约束问题,其目的是设计一种重量最轻且满足挠度、剪切应力、波动频率、外径等 4 个约束条件和 3 个设计变量的拉伸弹簧。3 个设计变量分别为线径 $d(0.05\leqslant d\leqslant 2.00)$、平均线圈直径 $D(0.25\leqslant D\leqslant 1.30)$、活性线圈数量 $N(2.00\leqslant N\leqslant 15.00)$,该设计问题的数学模型如下:

(1)设计变量:

$$x=[x_1x_2x_3]=[dDN]。$$

(2)目标函数:

$$f(x)=(x_3+2)x_2x_1^2。$$

(3)约束函数:

$$g_1(x)=1-\frac{x_2^3x_3}{71\ 785x_1^4}\leqslant 0;$$

$$g_2(x)=\frac{4x_2^2-x_1x_2}{12\ 566(x_2x_1^3-x_1^4)}+\frac{1}{5\ 108x_1^2}-1\leqslant 0;$$

$$g_3(x)=1-\frac{140.45x_1}{x_2^2x_3}\leqslant 0;$$

$$g_4(x)=\frac{x_1+x_2}{1.5}-1\leqslant 0。$$

表 4 所示为 FWOA 算法和其他 5 种算法求解拉伸弹簧设计问题的最小重量及对应设计变量的最优解。从该表数据可以看出,除了 BOA 和 A-CWOA 算法的求解结果相对较差外,其他几种算法求得的最终设计结果相差无几,且细微观察可以看出,FWOA 求解出的弹簧重量是 6 种算法中最小的,说明本文算法对于求解这类工程设计问题具有良好的性能。

表 4　6 种算法求解拉伸弹簧设计问题的最佳解决方案

算法	X_1	X_2	X_3	最小重量
FWOA	0.051 65	0.355 79	11.343 75	0.012 665 697
WOA	0.051 43	0.350 58	11.657 84	0.012 666 459
A-CWOA	0.052 90	0.386 27	9.759 16	0.012 710 938
AWOA	0.051 21	0.345 30	11.995 55	0.012 675 625
BOA	0.050 00	0.312 26	14.828 80	0.013 137 462

4.2.3　求解压力容器优化设计问题

压力容器设计问题的目标是使总成本最小化,包括材料成本、成型成本和焊接成本。该设计问题有 4 个约束条件和 4 个设计变量,设计变量分别是壳体厚度 $T_s(0\leqslant T_s\leqslant 99)$、封头厚度 $T_h(0\leqslant T_h\leqslant 99)$、壳体半径 $R(10\leqslant R\leqslant 200)$ 以及圆柱形截面长度 $L_s(10\leqslant L_s\leqslant 200)$,其具体数学模型如下:

(1)设计变量:

$$x=[x_1x_2x_3x_4]=[T_sT_hRL_s]。$$

(2)目标函数:

$$f(x)=0.622\ 4x_1x_3x_4+1.7781x_2x_3^2。$$

(3)约束函数:

$$g_1(x)=-x_1+0.019\ 3x_3\leqslant 0;$$

$$g_2(x)=-x_2+0.009\ 54x_3\leqslant 0;$$

$$g_3(x)=-\pi x_3^2x_4-\frac{4}{3}\pi x_3^3+1\ 296\ 000\leqslant 0;$$

$$g_4(x)=x_4-240\leqslant 0。$$

表 5 统计了 6 种算法在求解压力容器问题时,得到的最佳解决方案的数据。可以看出,本文算法 FWOA 在 $T_s=0.778\ 84$、$T_h=0.384\ 77$、$R=40.325\ 89$、$L_s=199.928\ 78$ 处得到的最小代价 $f(x)=5\ 886.970\ 27$ 不但是这几种算法求解压力容器设计问题的最低成本,而且其设计结果的成本远低于其他 5 种算法,充分说明 FWOA 在求解此类工程设计约束优化问题时具有优越的性能。

表 5　6 种算法求解压力容器设计问题的最佳解决方案

算法	X_1	X_2	X_3	X_4	最低成本
FWOA	0.778 42	0.384 77	40.325 89	199.928 78	5 886.970 27
WOA	0.838 97	0.422 72	43.245 55	162.922 38	6 051.712 79
A-CWOA	0.783 22	0.388 51	40.339 80	200.000 00	5 936.483 59
AWOA	0.845 91	0.484 49	43.787 57	156.772 55	6 242.774 51
BOA	1.278 77	0.628 33	61.910 03	25.253 46	7 665.852 47
BA	1.154 46	0.570 66	59.816 09	40.689 85	7 132.774 87

通过对以上 3 种不同复杂程度的著名工程约束优化问题的测试，充分说明了本文改进算法 FWOA 在处理不同类型工程优化设计问题上具有较好的应用潜力，不但能够提供很好的解决方案，而且求解结果也相当稳定。

5 结束语

为了更好求解工程设计约束优化问题，提升鲸鱼算法的寻优性能、求解稳定性和应用能力，本文提出一种基于分段式随机惯性权重和最优反馈机制的改进鲸鱼算法。在随机游走觅食策略中引入基于当前全局最优解的反馈机制，在维系种群多样性的同时避免了随机选择个体的盲目性，不仅加快了收敛速度，还改善了算法的稳定性。在收缩包围策略和螺旋气泡网捕食策略中引入分段式随机惯性权重，调节算法全局勘测能力和局部开采能力的平衡，不仅加快了算法的收敛速度，还降低了后期因惯性权重单调递减直接陷入局部极值无法跳出的风险，有利于算法摆脱局部极值，找到全局最优值。而对于越界处理的修正和改进，则消除了已有进化成果可能丢失的隐患，解决了当种群中较多个体越界而在越界处理后过于趋同造成多样性丧失的问题，提高了算法的寻优精度与收敛速度。通过理论分析证明了本文算法 FWOA 和基本鲸鱼算法(WOA)的时间复杂度相同，算法改进并没有降低执行效率。通过对 6 种算法在 12 个复杂基准测试函数上进行的不同维度函数极值优化仿真测试，结果表明，FWOA 的寻优性能和求解稳定性都明显优于其他 5 种对比算法。而对焊接梁、拉伸弹簧和压力容器 3 个具有挑战性的工程设计约束优化问题的求解结果，也充分证明了 FWOA 在处理不同类型工程优化设计问题上具有很好的适用性和优越的求解效果。下一步将继续改进鲸鱼优化算法的寻优机制，不断提高求解性能，并进一步提升算法的应用能力和适用性，将其应用于更多领域问题的求解中。

参考文献：

[1] COLORNI A, DORIGO M, MANIEZZO V. Distributed optimization by ant colonies[C]//Proceedings of European Conference on Artificial Life. Amsterdam, the Netherlands: Elsevier, 1991: 134-142.

[2] SILVA B N, HAN K J. Mutation operator integrated ant colony optimization based domestic appliance scheduling for lucrative demand side management[J]. Future Generation Computer Systems, 2019, 100: 557-568.

[3] BAI Jianglong, CHEN Hanning, HU Yabao, et al. Ant colony algorithm based on negative feedback and its application on robot path planning[J]. Computer Integrated Manufacturing Systems, 2019, 25(7): 1767-1774(in Chinese). [白建龙，陈瀚宁，胡亚宝，等. 基于负反馈机制的蚁群算法及其在机器人路径规划中的应用[J]. 计算机集成制造系统，2019，25(7)：1767-1774.]

[4] KENNEDY J, EBERHART R C. Particle swarm optimization [C]//Proceedings of the IEEE International Conference on Neural Networks. Washington, D. C., USA: IEEE, 1995: 1942-1948.

[5] ADHIKARI M, SRIRAMA S N. Multi-objective accelerated particle swarm optimization with a container-based scheduling for Internet-of-Things in cloud environment[J]. Journal of Network and Computer Applications, 2019, 137: 35-61.

[6] WU Yongming, DAI Longzhou, LI Shaobo, et al. Mixed assembly line evolution balancing based on improved particle swarm algorithm[J]. Computer Integrated Manufacturing Systems, 2017, 23(4): 781-790(in Chinese). [吴永明，戴隆州，李少波，等. 基于改进粒子群优化算法的混流装配线演进平衡[J]. 计算机集成制造系统，2017，23(4)：781-790.]

[7] YANG X S, DEB S. Cukoo search via levy flight[C]//Proceedings of World Congress on Nature & Biologically Inspired Computing. Washington, D. C., USA: IEEE, 2009: 210-214.

[8] CHEN Xu, YU Kunjie. Hybridizing cuckoo search algorithm with biogeography-based optimization for estimating photovoltaic model parameters[J]. Solar Energy, 2019, 180: 192-206.

[9] YANG X S. Bat algorithm for multi-objective optimization[J]. International Journal of Bio-Inspired Computation, 2012, 3(5): 267-274.

[10] LIU Jingsen, JI Hongyuan, LI Yu. Robot path planning based on improved bat algorithm and cubic spline interpolation[J/OL]. Acta Automatica Sinica, 2019: 1-10[2019-03-25]. https://doi.org/10.16383/j.aas.c180855(in Chinese). [刘景森，吉宏远，李　煜. 基于改进蝙蝠算法和三次样条插值的机器人路径规划[J/OL]. 自动化学报，2019：1-10[2019-03-25]. https://doi.org/10.16383/j.aas.c180855.]

[11] LI Yu, LI Xiaoting, LIU Jingsen, et al. An improved bat algorithm based on levy flights and adjustment factors[J]. Symmetry, 2019, 11(7), 925.

[12] YANG X S. Firefly algorithms for multimodal optimization [C]//Proceedings of the International Symposium on Stochastic Algorithms. Berlin, Germany: Springer-Verlag, 2009: 169-178.

[13] LIU Jingsen, LIU Li, LI Yu. A differential evolution flower pollination algorithm with dynamic switch probability[J]. Chinese Journal of Electronics, 2019, 28(4): 737-747.

[14] MIRJALILI S, GANDOMI A H, MIRJALILI S Z, et al. Salp swarm algorithm: A bio-inspired optimizer for engineering design problems[J]. Advances in Engineering Software, 2017, 114: 163-191.

[15] XING Zhikai, JIA Heming, SONG Wenlong. Levy flight trajectory-based salp swarm algorithm for multilevel threshol-

ding image segmentation[J]. Acta Automatica Sinica,2021,47(2):363-377(in Chinese).[邢致恺,贾鹤鸣,宋文龙. 基于莱维飞行樽海鞘群优化算法的多阈值图像分割[J]. 自动化学报,2021,47(2):363-377.]

[16] ARORA S, SINGH S. Butterfly optimization algorithm: a novel approach for global optimization[J]. Soft Computing, 2019,23(3):715-734.

[17] ARORA S, SINGH S. An improved butterfly optimization algorithm with chaos[J]. Journal of Intelligent & Fuzzy Systems,2017,32(1):1079-1088.

[18] ZHANG Yiying, JIN Zhigang, CHEN Ye. Hybrid teaching-learning-based optimization and neural network algorithm for engineering design optimization problems[J]. Knowledge-Based Systems,2019,187:104836.

[19] TSAI J T. Improved differential evolution algorithm for nonlinear programming and engineering design problems[J]. Neurocomputing,2015,148:628-640.

[20] DU Tingsong, KE Xianting, LIAO Jiagen, et al. DSLC-FOA: An improved fruit fly optimization algorithm application to structural engineering design optimization problems[J]. Applied Mathematical Modelling,2018,55:314-339.

[21] ZHANG Zichen, DING Shifei, JIA Weikuan. A hybrid optimization algorithm based on cuckoo search and differential evolution for solving constrained engineering problems[J]. Engineering Applications of Artificial Intelligence, 2019, 85: 254-268.

[22] DE MELO V V, CAROSIO G L C. Investigating multi-view differential evolution for solving constrained engineering design problems[J]. Expert Systems With Applications, 2013, 40(9):3370-3377.

[23] KUMAR V, KUMAR D. An astrophysics-inspired grey wolf algorithm for numerical optimization and its application to engineering design problems[J]. Advances in Engineering Software,2017,112:231-254.

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

[25] CHU Dingli, CHEN Hong, WANG Xuguang. Whale optimization algorithm based on adaptive weight and simulated Annealing[J]. Chinese Journal of Electronics,2019,47(5):992-999(in Chinese).[褚鼎立,陈　红,王旭光. 基于自适应权重和模拟退火的鲸鱼优化算法[J]. 电子学报,2019,47(5):992-999.]

[26] CHEN Huiling, XU Yueting, WANG Mingjing, et al. A balanced whale optimization algorithm for constrained engineering design problems[J]. Applied Mathematical Modelling,2019,71:45-59.

[27] WU Zezhong, SONG Fei. Whale optimization algorithm based on improved spiral update position model[J]. Systems Engineering—Theory & Practice, 2019, 39(11): 2928-2944(in Chinese).[吴泽忠,宋　菲. 基于改进螺旋更新位置模型的鲸鱼优化算法[J]. 系统工程理论与实践, 2019, 39(11): 2928-2944.]

[28] OLIVA D, AZIZ M A E, HASSANIEN A E. Parameter estimation of photovoltaic cells using an improved chaotic whale optimization algorithm[J]. Applied Energy, 2017, 200: 141-154.

[29] XIAO Ziya, LIU Sheng. Study on elite opposition-based golden-sine whale optimization algorithm and its application of project optimization[J]. Chinese Journal of Electronics, 2019,47(10):2177-2186(in Chinese).[肖子雅,刘　升. 精英反向黄金正弦鲸鱼算法及其工程优化研究[J]. 电子学报, 2019,47(10):2177-2186.]

[30] WANG Jianzhou, DU Pei, NIU Tong, et al. A novel hybrid system based on a new proposed algorithm—Multi-objective whale optimization algorithm for wind speed forecasting[J]. Applied Energy,2017,208:344-360.

[31] HUANG Qingbao, LI Junxing, SONG Chunning, et al. Whale optimization algorithm based on cosine control factor and polynomial mutation[J]. Control and Decision,2020,35(3):559-568(in Chinese).[黄清宝,李俊兴,宋春宁,等. 基于余弦控制因子和多项式变异的鲸鱼优化算法[J]. 控制与决策,2020,35(3):559-568.]

[32] JIANG Ruiye, YANG Ming, WANG Songyan, et al. An improved whale optimization algorithm with armed force program and strategic adjustment[J]. Applied Mathematical Modelling,2020,81:603-623.

[33] ABDEL-BASSET M, MANOGARAN G, EL-SHAHAT D, MIRJALILI S. A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem[J]. Future Generation Computer Systems, 2018,85:129-145.

[34] ELHOSSEINI M A, HAIKAL A Y, BADAWY M, et al. Biped robot stability based on an A—C parametric whale optimization algorithm[J]. Journal of Computational Science, 2019,31:17-32.

[35] SUN Wei, ZHANG Chongchong. Analysis and forecasting of the carbon price using multi-resolution singular value decomposition and extreme learning machine optimized by adaptive whale optimization algorithm[J]. Applied Energy,2018,231: 1354-1371.

作者简介:

刘景森(1968－),男,河南开封人,教授,博士,研究方向:智能算法、优化控制和网络信息安全;E-mail:ljs@henu.edu.cn;

马义想(1994－),女,河南驻马店人,硕士研究生,研究方向:智能算法;E-mail:myxhenu@163.com;

＋**李　煜**(1969－),女,河南开封人,教授,博士,研究方向:智能算法、电子商务,通讯作者,E-mail:leey@henu.edu.cn。