基于拟反向学习的自适应 QPSO 算法及其在工程中的应用

何光

(重庆工商大学 数学与统计学院;统计智能计算与监测重庆市重点实验室,重庆 400067)

摘 要:为改善量子粒子群优化(quantum-behaved particle swarm optimization algorithm,QPSO)算法在求解复杂的多模问题时表现出的收敛精度差和易于陷入局部最优的问题,提出了一种基于拟反向学习的自适应 QPSO 算法.首先,借鉴拟反向学习的思路,对粒子初始位置进行优化调整,增加算法搜索效率,加快收敛速度;其次,在粒子运动幅度的设置中考虑了种群进化程度和粒子聚集程度,构造了具有自适应特点的收缩-扩张因子,用于增强算法的局部挖掘和全局搜索能力;然后,将混沌映射的方法引入到越界粒子的处理上,有助于算法逃离局部最优.接着,基于 14 个测试函数将改进算法与 8 种智能优化算法进行对比分析.最后借助 2 个具体的工程设计问题进一步检验改进算法在实际应用中的效果.实验结果表明改进算法无论在基准测试中还是在工程应用上,其搜索能力更强,整体性能表现更为均衡.

关键词:量子粒子群优化算法;拟反向学习;收缩-扩张因子;混沌映射;工程应用

中图分类号:TP301.6 **文献标志码**:A **文章编号**:1000-2367(2025)05-0081-09

作为人工智能方法的一种重要类型,群智能算法源于对以蚂蚁、鸟类、蜜蜂等为代表的生物群体社会行为的分析.在迭代过程中,基于群体的算法可以不断使用个体和群体之间的交互信息,促使个体在搜索空间中探索,从而完成对问题最优解的寻找.与传统的优化方法相比,群智能算法不需要目标函数的导数信息,只使用原始数学运算符进行计算,操作原理简单,具有自组织、无中心控制、高鲁棒性、灵活且低消耗等特点,面对大规模的复杂问题时仍能给出最优解,目前成为世界各地众多研究者关注的焦点.

作为一种重要的群智能优化算法,量子粒子群优化(QPSO)算法[1]是基于粒子群优化算法的改良版本.QPSO 算法去除了粒子群算法中粒子方向的属性,使得粒子的更新与其先前的运动无关,从而大大增加了个体位置的随机性.与粒子群优化算法相比,QPSO 算法需要设置更少的参数,同时在收敛速度和最优解质量方面表现更好.在过去的 10 年里,QPSO 算法逐渐受到工业界和学术界的关注,主要体现在以下两个方面.其一,如何提高算法的全局搜索能力,改善算法在迭代后期种群的多样性缺失等问题.HE 等[2]设计了一种改进的 QPSO 算法,该算法利用了列维飞行中长距离搜索的特点,扩大了粒子的活动空间,增强了种群的探索能力,同时还应用了非线性收缩扩张因子,提高了算法的收敛精度.WANG 等[3]提出了一种混合自适应 QPSO 算法,该算法采用高斯混沌映射和列维飞行策略来增强局部挖掘和全局探索之间的平衡能力,保持种群多样性.其二,研究 QPSO 算法在各个领域的应用.近年来,QPSO 算法受到不同领域研究人员的青睐,成功应用于医学图像分割[4]、投资组合优化[5]、软件缺陷预测[6]和其他实际问题[7-8]中.

收稿日期:2024-04-23;**修回日期**:2024-09-26.

基金项目:重庆市科委项目(cstc2016jcyjA0564);重庆市教委项目(KJQN202100815).

作者简介(通信作者):何光(1981—),男,重庆万州人,重庆工商大学教授,博士,主要研究方向为智能优化算法及应用,E-mail:heguang6896@163.com.

引用本文:何光.基于拟反向学习的自适应 QPSO 算法及其在工程中的应用[J].河南师范大学学报(自然科学版),2025,53(5):81-89.(He Guang.Adaptive QPSO algorithm based on quasi-opposite learning and its applications in engineering[J].Journal of Henan Normal University(Natural Science Edition),2025,53(5):81-89.DOI:10.16366/j.cnki.1000-2367.2024.04.23.0004.)

虽然现有大多数改进的 QPSO 算法,在处理单模问题以及部分简单的多模问题上表现突出,但是面对一些复杂的多模问题时,仍然存在收敛效果差和算法早熟的现象.一是在算法的初始化环节,未能充分利用粒子分布位置的信息,由于粒子的初始位置对随后进行的算法迭代具有重要的引导作用,设置好粒子更新的出发点,将有助于推动算法的整体搜索,提升其收敛的速度,那么在改进种群的初始化方面还有需要深入分析的地方.另外,在算法的迭代过程中,单个粒子信息与种群信息的运用效率不够,反映出算法在处理复杂问题时局部挖掘和全局搜索的平衡能力不够,寻优性能较差,从而在个体和种群之间的信息交互和经验吸取方面还存在优化的空间.

于是,为改善 QPSO 算法在面对复杂的多模问题时表现出的收敛精度差和易于陷入局部最优的状况,将作出如下改进.首先,借鉴拟反向学习的思路,对粒子初始位置进行优化调整,增加算法搜索效率,加快收敛速度;其次,在粒子运动幅度的设置中考虑了种群进化程度和粒子聚集程度,设计了具有自适应特点的收缩-扩张因子,用于增强算法的局部挖掘和全局搜索能力;然后,将混沌映射的方法引入到越界粒子的处理上,有助于算法逃离局部最优.接着,基于 14 个测试函数将改进算法与 8 种智能优化算法进行对比分析,最后借助 2 个具体的工程设计问题进一步展示改进算法在实际应用中的性能优势.

1 标准的 QPSO 算法

在标准的 QPSO 系统中,设种群规模为 N,搜索空间维数为 D,搜索空间第 j 维的上界和下界分别为 U_j 和 L_j,最大迭代次数为 $T_{\max}$.第 i 个粒子在搜索过程中会趋近于局部吸引点 $P_i=(P_{i,1},P_{i,2},\cdots,P_{i,D})$,形如:

$$P_{i,j}(t+1)=\varphi p_{i,j}(t)+(1-\varphi)p_{g,j}(t), \tag{1}$$

其中 $i=1,2,\cdots,N$,$j=1,2,\cdots,D$,φ 表示 0 到 1 之间均匀分布的随机数,$p_{i,j}(t)$ 表示第 i 个粒子在第 t 次迭代时历史最优位置 $p_i(t)$ 的第 j 维,$p_{g,j}(t)$ 表示第 t 次迭代时全局最优位置 $p_g(t)$ 的第 j 维.在 QPSO 算法中定义最优平均位置 $m(t)$ 如下:$m(t)=(\frac{1}{N}\sum_{i=1}^{N}p_{i,1}(t),\frac{1}{N}\sum_{i=1}^{N}p_{i,2}(t),\cdots,\frac{1}{N}\sum_{i=1}^{N}p_{i,D}(t))$.

在设计 QPSO 算法的粒子运行轨迹时,引入了量子态粒子行为,运用蒙特卡罗方法得到粒子位置更新公式如下:

$$x_{i,j}(t+1)=P_{i,j}(t)\pm\beta\cdot|m_j(t)-x_{i,j}(t)|\cdot\ln(1/u), \tag{2}$$

其中,β 称为收缩-扩张因子,u 表示 0 到 1 之间均匀分布的随机数,$m_i(t)$ 表示 $m(t)$ 的第 j 维.当 u 大于0.5 时,式(2)中取“+”,否则取“-”.

2 基于拟反向学习的自适应 QPSO 算法

2.1 粒子初始位置的改进

QPSO 算法的搜索效率实际上与初始种群中每个个体离最优解的距离有关.当某个粒子最初在最优解附近生成时,种群中其他粒子的运动趋势将保持与该粒子一致,进而加快算法搜索的效率.为提高算法迭代的速度和效率,TIZHOOSH[9] 提出了一种基于反向学习的技术:对种群中的每个粒子寻找其反向粒子,挑选粒子与反向粒子中表现更佳的作为初始个体,从而增加种群快速接近最优解的可能性.在此基础上,RAHNAMAYAN 等[10] 进一步提出了一种称为拟反向学习的方法,该方法已被证明在确定全局最优解方面更快、更有效.

将应用拟反向学习的策略来初始化种群,有助于提升粒子在迭代开始前快速接近最优解的机会,进而增加算法的搜索效率和收敛速度.基于拟反向学习的初始化流程见算法Ⅰ.

算法Ⅰ 基于拟反向学习的初始化

1:随机生成初始种群 $P_0=\{x_{ij}\}$,$i=1,2,\cdots,N$,$j=1,2,\cdots,D$,定义反向种群和拟反向种群分别为 $RP_0=\{x_{ij}^r\}$ 和 $QRP_0=\{x_{ij}^{qr}\}$

2:for $i=1:N$

3: for $j=1:D$

4: $x_{ij}^r=L_j+U_j-x_{ij}$,$M_{ij}=(L_j+U_j)/2$

5: if $x_{ij}<M_{ij}$

6: $x_{ij}^{qr}=M_{ij}+\text{random}\cdot(x_{ij}^r-M_{ij})$

7:　　else
8:　　$x_{ij}^{qr} = x_{ij}^{r} + \text{random} \cdot (M_{ij} - x_{ij}^{r})$
9:　　end if
10:　end for
11:end for
12:选择 $P_0 \cup QRP_0$ 中适应值最好的 N 个粒子构成初始种群

2.2　收缩-扩张因子的调整

在 QPSO 算法的迭代过程中,收缩-扩张因子 β 用于调整粒子在搜索空间中的运动幅度,影响算法收敛的速度.标准 QPSO 算法中,随迭代次数增加,β 呈线性变化,然而对迭代过程中粒子信息的运用不够充分,特别是在迭代后期面临种群多样性缺失时,不能及时做出调整和应对.希望随着算法迭代的推进,能够充分利用好个体历史最优和种群全局最优的相关信息,用于增强算法搜索的有效性.

一方面,引入种群的进化程度(记为 $D(e)$),用于刻画整个种群的迭代速度,根据当前全局最优值与过去的全局最优值之间的比较来体现,公式如下:

$$D(e) = \frac{f(p_g(t))}{f(p_g(t-1))}, \tag{3}$$

其中 $f(p_g(t))$ 表示第 t 次迭代的全局最优值.另一方面,考虑粒子聚合程度(记为 $D(a)$),用于体现种群中所有粒子的聚集情况,具体公式如下:

$$D(a) = \frac{N \cdot f(p_g(t))}{\sum_{i=1}^{N} f(p_i(t))}, \tag{4}$$

其中 $f(p_i(t))$ 表示第 t 次迭代第 i 个粒子的历史最优值.

于是,借助 D_e 和 D_a 重新设计收缩-扩张因子 β,增强粒子运动的自我适应能力和灵活性,从而提升算法的局部挖掘和全局搜索性能,具有这种自适应的 β 形式如下:

$$\beta = a - b \cdot D_e + c \cdot D_a, \tag{5}$$

其中 a 表示 β 的初始值,b 和 c 分别为种群进化程度和粒子聚集程度的调节系数.根据式(5)可见,随着迭代的增加,D_e 逐渐趋于 1,β 的取值会相应减小,表明在算法的后期粒子的运动范围会减少,在小范围内进行搜索;同时,D_a 会随着迭代逐渐增加,导致 β 的取值增大,拓展了算法的全局搜索范围.那么,b 和 c 这两个调节系数的设置对粒子的搜索效果影响明显,具体操作思路将在性能测试中进行阐述.

2.3　越界粒子的处理

混沌是非线性确定性系统内在随机性的外在复杂表现形式.它是一种形状类似于随机的非随机运动,具有遍历性、随机性和规律性的特点.近年来,许多学者将混沌优化原理与现代优化算法相结合,将混沌映射产生的序列映射到搜索空间中,或者用混沌映射生成的序列代替算法中的随机参数.高斯混沌映射是一种重要的映射方法,被用于生成混沌序列,具体如下:

$$y_j = \begin{cases} 1, \text{若 } y_j^0 \in [0, 0.5), \\ \text{mod}(1/y_j^0, 1), \text{若 } y_j^0 \in [0.5, 1], \end{cases} \tag{6}$$

其中 y_j^0 为 0 到 1 之间均匀分布的随机数.

将由式(6)生成的 y_j 用于更新越界粒子的位置,一方面可以保证更新后的粒子能够在可行的搜索空间中运动,另一方面能够赋予粒子具有一定的随机扰动能力,有助于算法摆脱早熟的困境,从而维持种群的多样性.具体更新公式如下:

$$x_{ij}(t) = \begin{cases} U_j - y_i(U_j - L_j), \text{若 } x_{ij}(t) > U_j, \\ L_j + y_j(U_j - L_j), \text{若 } x_{ij}(t) < L_j, \end{cases} \tag{7}$$

其中 U_j 和 L_j 分别表示搜索空间第 j 维的上界和下界.

改进后的算法主要运用了拟反向学习策略和自适应调整思路,命名为 QAQPSO,其算法流程见算法Ⅱ.

算法Ⅱ　QAQPSO 算法

1:设定种群规模 N,搜索空间维数 D,最大迭代次数为 $T_{\max}$,运用算法Ⅰ生成每个粒子的初始位置
2:初始化 $p_i(t)$ 和 $P_g(t)$,令 $k=0$
3:While $k < T_{\max}$

4: 分别运用式(3)和(4)计算 D_e 和 D_a
5: for $i=1:N$
6: for $j=1:D$
7: 由式(1)计算得出局部吸引点 $P_i(t)$,由式(5)计算得 β
8: if $u>0.5$
9: $x_{i,j}(t+1)=P_{i,j}(t)+\beta\cdot|m_j(t)-x_{i,j}(t)|\cdot\ln(1/u)$
10: else
11: $x_{i,j}(t+1)=P_{i,j}(t)-\beta\cdot|m_j(t)-x_{i,j}(t)|\cdot\ln(1/u)$
12: end if
13: 运用式(6)和(7)对越界的粒子进行重新调整
14: end for
15: 通过比较原则,更新 $P_i(t)$ 和 $P_g(t)$,$k=k+1$,
16: end for
17:end while
18:输出结果

2.4 算法的时间复杂度

QAQPSO 算法的时间复杂度取决于种群的初始化、适应值计算以及个体更新等环节.设种群规模为 N,搜索空间维数为 D.在初始化阶段,设产生粒子位置分量的时间为 t_1,单个粒子适应值的计算时间为 $f(D)$;在粒子的更新过程中,设粒子位置分量的更新时间为 t_2,其适应值的计算时间为 $f(D)$,个体最优和种群最优的更新时间为 t_3.从而,进行一次迭代时 QAQPSO 算法的时间复杂度为:$O[2N\cdot(D\cdot t_1+f(D))]+O[N\cdot(D\cdot t_2+f(D)+t_3)]=O[N\cdot(D+f(D))]$.相比 QPSO 算法,QAQPSO 算法主要在种群初始化时增加了反向种群的生成和函数评价的时间,而对收缩-扩张因子进行自适应调整时并没有增加算法的时间复杂度.然而,从时间复杂度上来看,QAQPSO 算法与标准 QPSO 算法是一样的.

3 性能测试

3.1 测试准备

将 QAQPSO 算法与 AVOA[11]、HHO[12]、GWO[13]、BOA[14]、SSA[15]、QPSO[1]、LQPSO[2]和 HAQPSO[3]算法进行对比分析.所有 14 个测试函数来源于 CEC2017,具体信息见表 1,其中 f_1-f_2 为单模函数,用于测试算法的局部挖掘能力;f_3-f_4 为多模函数,用于测试算法的全局探索能力;f_5-f_{14} 为复合函数,用于测试算法的均衡能力.关于 QAQPSO 的参数设置,a 的取值与标准 QPSO 中收缩-扩张因子的初始值一样,选为 0.9;参数 b 和 c 分别考虑了 0.1、0.3、0.5 和 0.7,然后根据面中心复合设计的思想[16],在测试函数上对以上参数进行了性能对比分析,最终确定 $b=0.3$,$c=0.1$.另外,各对比算法的参数设定来源于相应的参考文献.

表 1 测试函数

Tab. 1 Benchmark functions

函数	名称	取值范围	最优值	函数	名称	取值范围	最优值
f_1	Shifted and Rotated Bent Cigar	[−100,100]	100	f_8	Hybrid Function 6($N=5$)	[−100,100]	1 800
f_2	Shifted and Rotated Zakharov	[−100,100]	300	f_9	Hybrid Function 6($N=5$)	[−100,100]	1 900
f_3	Shifted and Rotated Rosenbrock	[−100,100]	400	f_{10}	Hybrid Function 6($N=6$)	[−100,100]	2 000
f_4	Shifted and Rotated Expanded Scaffer's F6	[−100,100]	600	f_{11}	Composition Function 2($N=3$)	[−100,100]	2 200
f_5	Hybrid Function 1($N=3$)	[−100,100]	1 100	f_{12}	Composition Function 4($N=4$)	[−100,100]	2 400
f_6	Hybrid Function 2($N=3$)	[−100,100]	1 200	f_{13}	Composition Function 6($N=5$)	[−100,100]	2 600
f_7	Hybrid Function 4($N=4$)	[−100,100]	1 400	f_{14}	Composition Function 8($N=6$)	[−100,100]	2 800

3.2 改进算法的消融实验

在进入不同算法的性能对比之前,对前面提出的 3 种改进策略进行了消融实验,以确定每种策略对原始算法性能改进的重要性.这里用 S1 表示初始种群的改进策略,S2 表示收缩扩张因子的优化方案,S3 表示越界个体的调整措施.在实验中,种群规模为 50,搜索空间为 30 维,最大迭代次数为 1 000.每种组合在各个测试函数上独立运行 30 次,并记录结果的均值和标准差.通过实验,发现单独使用某一种改进策略,以及同时使用 S1 和 S3 对算法性能的改善效果并不明显,然而融合 S2 之后算法有明显的改进,现将实验结果展示如下,见表 2.

表 2　消融实验结果

Tab. 2　Results of ablation experiment

函数	指标	S2	S1+S2	S2+S3	S1+S2+S3	函数	指标	S2	S1+S2	S2+S3	S1+S2+S3
f_1	均值	3.35e+03	3.65e+03	1.68e+03	1.30e+03	f_8	均值	2.19e+05	4.11e+05	2.55e+05	1.99e+05
	标准差	3.44e+03	4.33e+03	2.18e+03	2.13e+03		标准差	1.30e+05	5.64e+05	3.57e+05	2.37e+05
f_2	均值	3.11e+04	2.67e+04	1.69e+04	1.51e+04	f_9	均值	9.52e+03	1.34e+04	6.90e+03	6.62e+03
	标准差	1.14e+04	1.36e+04	6.77e+03	4.92e+03		标准差	1.07e+04	1.32e+04	6.85e+03	4.96e+03
f_3	均值	4.92e+02	4.97e+02	4.89e+02	4.76e+02	f_{10}	均值	2.32e+03	2.32e+03	2.29e+03	2.27e+03
	标准差	2.33e+01	2.38e+01	2.91e+01	3.80e+01		标准差	2.10e+02	1.38e+02	1.69e+02	1.59e+02
f_4	均值	6.00e+02	6.00e+02	6.00e+02	6.00e+02	f_{11}	均值	6.41e+03	5.52e+03	3.77e+03	2.85e+03
	标准差	1.20e-02	2.44e-02	6.86e-02	7.65e-03		标准差	2.95e+03	3.10e+03	2.55e+03	1.70e+03
f_5	均值	1.15e+03	1.17e+03	1.14e+03	1.14e+03	f_{12}	均值	2.89e+03	2.88e+03	2.87e+03	2.87e+03
	标准差	3.29e+01	4.11e+01	3.11e+01	2.39e+01		标准差	3.15e+01	2.16e+01	1.42e+01	2.13e+01
f_6	均值	4.02e+05	2.202+05	1.90e+05	1.71e+05	f_{13}	均值	4.42e+03	4.30e+03	3.87e+03	3.82e+03
	标准差	3.97e+05	1.36e+05	1.42e+05	1.07e+05		标准差	3.85e+02	1.82e+02	7.54e+02	7.16e+02
f_7	均值	5.14e+04	2.94e+04	1.92e+04	1.93e+04	f_{14}	均值	3.23e+03	3.23e+03	3.21e+03	3.21e+03
	标准差	4.79e+04	2.81e+04	1.51e+04	2.28e+04		标准差	2.67e+01	2.19e+01	3.36e+01	2.05e+01

根据表 2 的结果,首先对比单独使用 S2 策略与融合 S1 和 S2 两种策略,发现融合 S1 和 S2 两种策略取得的优化结果在函数 f_2、f_6、f_7、f_{10}、f_{11}、f_{12}、f_{13}、f_{14}上更好,而在其余函数上表现不如单独使用 S2 策略.这表明初始种群的改进策略是有效果的,在部分测试函数上呈现出一定的优势.随后,比较单独使用 S2 策略与融合 S3 和 S2 两种策略的结果,发现除了函数 f_4 和 f_8 之外,在其余函数上融合 S3 和 S2 两种策略带来的性能提升效果更好.这意味着对越界个体的处理方式能够明显改善算法的寻优性能.另外,从融合两种策略的结果来看,S1+S2 的组合整体上不如 S3+S2 组合带来的性能提升效果.然而,在将 3 种策略综合运用之后,相比其余组合,S1+S2+S3 的组合在 12 个测试函数上取得更好的表现,这意味着 3 种策略的融合运用能有效改进算法性能,与改进策略设计的初衷契合.

3.3　不同算法的性能对比

在测试实验中,所有算法的种群规模均为 50,搜索空间分别取 30、50 和 100 维,最大迭代次数为 1 000.各算法在每个测试函数上独立运行 30 次,并记录结果的均值和标准差.在 30 维和 50 维的对比中,改进算法均在超过 70%的函数上表现最好,展现出比其他对比算法更好的搜索性能.至于在更高维的情况,这里展示了 100 维时的实验结果,见附录表 S1.

通过附录表 S1 的结果,QAQPSO 算法在 11 个测试函数上取得了最好的均值结果,同时在 f_1、f_3、f_5-f_6、f_8-f_9 以及 f_{12}-f_{14}这些函数上也获得了更好的稳定性结果.在单模函数和多模函数方面,改进算法表现出不错的寻优性能.在复合函数上,除了在 f_{10}和 f_{11}两个函数上寻优效果较差外,在其余 8 个函数上都有着不错的表现.综合来看,QAQPSO 算法在不同维度的测试中都展现出更有效的搜索性能,整体表现优于其他比较算法.进一步,为了对比 9 种算法的收敛效率,展示了 100 维下各个算法在测试函数 f_1、f_6、f_7、f_{13}上的迭代情况,见图 1.由图 1 所示,改进算法 QAQPSO 在处理单模函数和复杂的多模函数上,均有着不错的表现,无论收敛速度还是寻优质量都优于其他 8 种比较算法.

3.4　显著性检验

进一步,验证改进算法的显著性.首先将 QAQPSO 与其他各个算法之间进行 Wilcoxon 检验,其中"+"、"−"和"="分别表示相比给定的算法,QAQPSO 算法的性能明显更好、明显更差或无明显差异,并记录在 14 个测试函数下的对比结果.随后,通过 Friedman 检验,计算 9 种算法的秩均值,获得各个算法在不同函数中得分排名的平均值,进而确定各算法性能的最终名次.具体的非参数检验结果见表 3 至表 5.

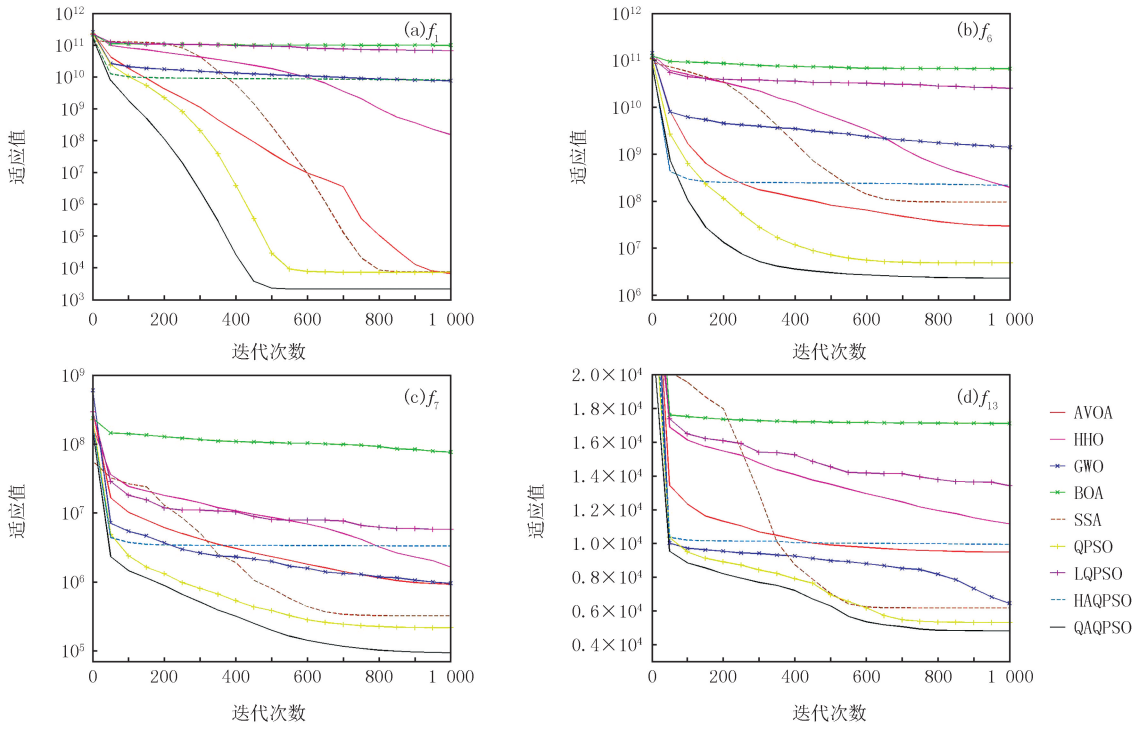

图1 在100维情况下各算法的收敛曲线

Fig. 1 Convergence curves of different algorithms with 100-dimensional

表 3 在 30 维情况下的非参数检验结果

Tab. 3 Non-parameter test results with 30-dimensional

算法	Wilcoxon			Friedman	
	+	−	=	得分	排名
AVOA	14	0	0	5.00	5
HHO	14	0	0	6.14	7
GWO	14	0	0	5.64	6
BOA	14	0	0	8.64	9
SSA	12	0	2	4.42	4
QPSO	9	0	5	3.21	3
LQPSO	14	0	0	7.71	8
HAQPSO	6	0	8	3.00	2
QAQPSO	−	−	−	1.21	1

表 4 在 50 维情况下的非参数检验结果

Tab. 4 Non-parameter test results with 50-dimensional

算法	Wilcoxon			Friedman	
	+	−	=	得分	排名
AVOA	11	1	2	4.07	4
HHO	12	1	1	5.36	7
GWO	11	1	2	4.79	5
BOA	14	0	0	8.93	9
SSA	10	1	3	3.93	3
QPSO	8	0	6	3.07	2
LQPSO	14	0	0	8.00	8
HAQPSO	12	1	1	5.21	6
QAQPSO	−	−	−	1.64	1

表 5 在 100 维情况下的非参数检验结果

Tab. 5 Non-parameter test results with 100-dimensional

算法	Wilcoxon			Friedman	
	+	−	=	得分	排名
AVOA	10	3	1	3.71	4
HHO	11	3	0	5.00	5
GWO	11	3	0	5.07	6
BOA	14	0	0	8.71	9
SSA	12	2	0	3.57	3
QPSO	9	0	5	3.50	2
LQPSO	14	0	0	7.93	8
HAQPSO	12	2	0	5.57	7
QAQPSO	−	−	−	1.93	1

通过表 3 至表 5 中 Wilcoxon 检验的结果可见,改进算法在大多数测试函数中展现出更佳的寻优能力;同时,Friedman 检验结果可见,QAQPSO 算法在 9 种算法中的综合排名最高,表明其搜索能力更强,整体性能表现更为均衡.

4　改进算法在工程问题中的应用

将 QAQPSO 算法用于寻找 2 个工程设计问题的数值解.在具体应用问题的求解中,各种比较算法的参数设置均来源于相关文献,QAQPSO 算法中的各个参数与上一节性能测试中设置一样.所有算法的规模均为 50,每种算法独立运行 30 次后,记录下相应的均值、标准差、最好值和对应的最好解.

问题 1　压力容器设计问题.该问题的目的是使压力容器的生产成本最小化,包含 4 个变量,即壳体的厚度 T_s(记为 x_1),半球形部分的厚度 T_h(记为 x_2),内半径 R(记为 x_3) 和圆柱零件的长度 L(记为 x_4),以及 4 个约束条件.其数学模型表示如下:

$$\begin{aligned}&\min\quad f(x)=0.622\ 4x_1x_3x_4+1.778\ 1x_2x_3^2+3.166\ 1x_1^2x_4+19.84x_1^2x_3,\\&\text{s.t.}\quad g_1(x)=0.019\ 3x_2-x_1\le 0;g_2(x)=0.009\ 54x_3-x_2\le 0;\\&\qquad g_3(x)=-\pi x_3^2x_4-\frac{4}{3}\pi x_3^3+1\ 296\ 000\le 0;g_4(x)=x_4-240\le 0.\end{aligned}$$

其中,$0\le x_1,x_2\le 100$,$10\le x_3,x_4\le 200$.

将 QAQPSO 算法与其他 8 种智能算法:EA2[17]、EA3[18]、QPSO[1]、ESs[19]、CPSO[20]、GWO[13]、WOA[21]和 HHO[12]进行对比,获得的最优值和最优解具体见表 6.根据表 6 中展示的结果,可见在最优值的获取上 GA2、GA3、ESs、CPSO、GWO 和 WOA 这 6 种智能算法在数值结果上比较接近,而 HHO、QPSO 和 QAQPSO 算法在压力容器设计问题上的求解结果更好,其中改进算法体现出更佳的寻优能力.

表 6　不同算法在压力容器设计问题上的对比结果

Tab. 6　Comparison results of different algorithms for the pressure vessel design problem

算法	GA2	GA3	QPSO	ESs	CPSO	GWO	WOA	HHO	QAQPSO
均值	6 177.235 6	6 074.334 1	6 012.487 6	6 350.123 2	6 147.133 2	6 078.234 2	6 122.453 4	6 081.227 6	5 982.876 5
标准差	1.31e+02	1.59e+02	9.29e+01	4.26e+02	8.61e+01	7.92e+01	1.22e+02	6.13e+01	5.97e+01
最好值	6 059.946 3	6 059.946 3	5 996.134 0	6 059.745 6	6 061.077 7	6 051.563 9	6 059.741 0	6 000.462 59	5 891.867 98
T_s	0.812 500	0.812 500	0.838 161	0.812 500	0.812 500	0.812 5	0.812 500	0.817 583 83	0.781 968
T_h	0.437 500	0.437 500	0.414 33	0.437 500	0.437 500	0.437 5	0.437 500	0.407 292 7	0.386 53
R	42.097 398	42.097 4	43.427 857	42.098 087	42.091 266	42.089 181	42.098 269 9	42.091 745 76	40.516 472
L	176.654 050	176.654 0	160.832 633	176.640 518	176.746 500	176.758 731	176.638 998	176.719 635 2	197.277 7

问题 2　拉伸/压缩弹簧设计问题.该问题的目的是最小化弹簧重量,受到导线直径 d(记为 x_1),平均线圈直径 D(记为 x_2) 和有效线圈数 P(记为 x_3) 的限制. 其数学模型由以下公式表示.

$$\begin{aligned}&\min\quad f(x)=(x_3+2)x_2x_1^2,\\&\text{s.t.}\quad g_1(1)=1-\frac{x_2^3x_3}{71\ 785x_1^4}\le 0;g_2(x)=\frac{4x_2^2-x_1x_2}{12\ 566(x_2x_1^3-x_1^4)}+\frac{1}{5\ 108x_1^2}-1\le 0;\\&\qquad g_3(x)=1-\frac{140.45x_1}{x_2^2x_3}\le 0;g_4(x)=\frac{x_1+x_2}{1.5}-1\le 0.\end{aligned}$$

其中,$0.05\le x_1\le 2$,$0.25\le x_2\le 1.30$,$2\le x_3\le 15$.

将 QAQPSO 算法与 EA2[17]、EA3[18]、QPSO[1]、ESs[19]、CPSO[20]、GWO[13]、WOA[21]和 HHO[12]算法的比较结果记录在表 7 中.由表 7 可知,对于这个非线性的低维优化问题,各算法在寻优效果方面比较接近,其中 GA2 和 QPSO 算法的表现稍差,而其他 7 种算法的精度需要在小数点后第 5 位才能体现出来.在求解弹簧设计问题的最佳值上,HHO 表现最好,GWO 和 QAQPSO 两个算法在搜索能力上与之相近.同时,改进算法在均值和稳定性方面表现更好.

表 7 不同算法在拉伸/压缩弹簧设计问题上的对比结果

Tab. 7 Comparison results of different algorithms for the tension/compression spring design problem

算法	GA2	GA3	QPSO	ESs	CPSO	GWO	WOA	HHO	QAQPSO
均值	0.012 734	0.012 721	0.012 867	0.013 227	0.013 429	0.013 667	0.013 324	0.012 745	0.012 713
标准差	4.67e-05	5.88e-05	6.33e-05	8.67e-04	7.34e-04	9.67e-04	6.75e-04	4.03e-05	3.85e-05
最好值	0.012 704	0.012 681	0.012 717 6	0.012 698	0.012 674	0.012 666	0.012 676 3	0.012 665 443	0.012 665 67
d	0.051 480	0.051 989	0.052 071 8	0.051 643	0.051 728	0.051 69	0.051 207	0.051 796 393	0.051 771
D	0.351 661	0.363 965	0.365 663	0.355 360	0.357 644	0.356 737	0.345 215	0.359 305 355	0.358 69
P	11.632 201	10.890 522	10.826 8	11.397 926	11.244 543	11.288 85	12.004 032	11.138 859	11.174 549 5

5 结束语

为了改善 QPSO 算法在面对复杂的多模问题时表现出的收敛精度差和易于陷入局部最优的状况,设计出基于拟反向学习的自适应 QPSO 算法(QAQPSO).在改进算法设计中,运用拟反向学习的思想对种群进行初始化,提升算法的搜索效率,加快收敛速度;借助自适应的收缩-扩张因子,用于改善算法的全局搜索和局部挖掘能力,增强算法性能的均衡性;通过高斯混沌映射调整越界粒子的落点范围,有助于算法多样性的保持.在性能测试中,QAQPSO 算法无论在收敛精度还是寻优结果的鲁棒性方面,比其他 8 种智能优化算法表现更好.进而,在 2 个具体的工程设计问题上,QAQPSO 算法同样展现出优良的全局搜索能力,在求解复杂多模问题时取得了更具竞争优势的结果.

然而,在改善算法的收敛精度和整体性能方面,如何进一步挖掘种群与个体之间的有用信息,选择新的思路来提升粒子的迭代质量都是值得深入研究的方向.另外,将算法继续运用到更多的实际问题中,也存在很多思考的空间.

附录见电子版(DOI:10.16366/j.cnki.1000-2367.2024.04.23.0004).

参 考 文 献

[1] SUN J,FENG B,XU W B.Particle swarm optimization with particles having quantum behavior[C]//Proceedings of the 2004 Congress on Evolutionary Computation.Portland:IEEE,2004:325-331.

[2] HE G,LU X L.An improved QPSO algorithm and its application in fuzzy portfolio model with constraints[J].Soft Computing,2021,25(12):7695-7706.

[3] WANG H Q,CHENG X W,CHEN G C.A hybrid adaptive quantum behaved particle swarm optimization algorithm based multilevel thresholding for image segmentation[C]//2021 IEEE International Conference on Information Communication and Software Engineering (ICICSE).Chengdu:[s.n.],2021:97-102.

[4] LI Y Y,BAI X Y,JIAO L C,et al.Partitioned-cooperative quantum-behaved particle swarm optimization based on multilevel thresholding applied to medical image segmentation[J].Applied Soft Computing,2017,56:345-356.

[5] LU X L,HE G.QPSO algorithm based on Lévy flight and its application in fuzzy portfolio[J].Applied Soft Computing,2021,99:106894.

[6] ANJU A J,JUDITH J E.Retraction Note:Adaptive recurrent neural network for software defect prediction with the aid of quantum theory particle swarm optimization[J].Multimedia Tools and Applications,2023,82(11):16257-16278.

[7] 郭强,时胜国,何辉辉.基于量子行为粒子群算法的舱室噪声监测点优化布置[J].哈尔滨工程大学学报,2024,45(8):1488-1496.
GUO Q,SHI S G,HE H H.Optimal sensor placement at cabin-inner-sound monitoring points based on the quantum-behaved particle swarm optimization algorithm[J].Journal of Harbin Engineering University,2024,45(8):1488-1496.

[8] 陈卓凡,周坤,秦菲菲,等.基于改进量子粒子群优化算法的机器人逆运动学求解[J].中国机械工程,2024,35(2):293-304.
CHEN Z F,ZHOU K,QIN F F,et al.Inverse kinematics solution of robots based on IQPSO algorithm[J].China Mechanical Engineering,2024,35(2):293-304.

[9] TIZHOOSH H R.Opposition-based learning:a new scheme for machine intelligence[C]//International Conference on Computational Intelligence for Modelling,Control and Automation and International Conference on Intelligent Agents,Web Technologies and Internet Commerce (CIMCA-IAWTIC'06).Vienna:IEEE,2005:695-701.

[10] RAHNAMAYAN S,TIZHOOSH H R,SALAMA M M A.Quasi-oppositional differential evolution[C]//2007 IEEE Congress on Evolu-

tionary Computation. Singapore: IEEE, 2007: 2229-2236.
[11] ABDOLLAHZADEH B, GHAREHCHOPOGH F S, MIRJALILI S. African vultures optimization algorithm: a new nature-inspired metaheuristic algorithm for global optimization problems[J]. Computers & Industrial Engineering, 2021, 158: 107408.
[12] HEIDARI A A, MIRJALILI S, FARIS H, et al. Harris Hawks optimization: algorithm and applications[J]. Future Generation Computer Systems, 2019, 97: 849-872.
[13] MIRJALILI S, MIRJALILI S M, LEWIS A. Grey wolf optimizer[J]. Advances in Engineering Software, 2014, 69: 46-61.
[14] ARORA S, SINGH S. Butterfly optimization algorithm: a novel approach for global optimization[J]. Soft Computing, 2019, 23(3): 715-734.
[15] 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.
[16] BALACHANDRAN M, DEVANATHAN S, MURALEEKRISHNAN R, et al. Optimizing properties of nanoclay-nitrile rubber (NBR) composites using Face Centred Central Composite Design[J]. Materials & Design, 2012, 35: 854-862.
[17] DEB K. Optimal design of a welded beam via genetic algorithms[J]. AIAA Journal, 1991, 29(11): 2013-2015.
[18] COELLO COELLO C A, MEZURA MONTES E. Constraint-handling in genetic algorithms through the use of dominance-based tournament selection[J]. Advanced Engineering Informatics, 2002, 16(3): 193-203.
[19] MEZURA-MONTES E, COELLO C A C. A simple multimembered evolution strategy to solve constrained optimization problems[J]. IEEE Transactions on Evolutionary Computation, 2005, 9(1): 1-17.
[20] HE Q, WANG L. An effective co-evolutionary particle swarm optimization for constrained engineering design problems[J]. Engineering Applications of Artificial Intelligence, 2007, 20(1): 89-99.
[21] MIRJALILI S, LEWIS A. The whale optimization algorithm[J]. Advances in Engineering Software, 2016, 95: 51-67.

Adaptive QPSO algorithm based on quasi-opposite learning and its applications in engineering

He Guang

(School of Mathematics and Statistics; Chongqing Key Laboratory of Statistics Intelligent Computing and Monitoring, Chongqing Technology and Business University, Chongqing 400067, China)

Abstract: To address the issues of poor convergence accuracy and susceptibility to local optima exhibited by the quantum-behaved particle swarm optimization(QPSO) algorithm in solving complex multi-modal problems, an adaptive QPSO algorithm based on quasi-opposite learning is proposed. Firstly, drawing on the idea of quasi-opposite learning, the initial positions of particles are optimized and adjusted to increase algorithm search efficiency and accelerate convergence speed. Secondly, in the setting of particle movement range, population evolution degree and particle aggregation degree are taken into account, and a contraction-expansion factor with adaptive characteristics is constructed to enhance the local exploitation and global exploration capabilities of the algorithm. And the method of chaotic mapping is introduced to handle out-of-range particles, which helps the algorithm escape local optima. Then the improved algorithm is compared with eight existing intelligent optimization algorithms on 14 benchmark test functions. Additionally, the application effectiveness of the improved algorithm is examined through two real-world engineering design problems. The experimental results demonstrate that the improved algorithm exhibits stronger search capability and more balanced overall performance.

Keywords: Quantum-behaved particle swarm optimization algorithm; quasi-opposite learning; contraction-expansion factor; chaotic map; engineering application

[责任编校　陈留院　杨浦]

附　录

表 S1　在 100 维情况下测试算法的对比结果

Tab. S1　Comparison results of test algorithms with 100-dimensional

函数	指标	AVOA	HHO	GWO	BOA	SSA	QPSO	LQPSO	HAQPSO	QAQPSO
f_1	均值	9.75e+07	2.57e+09	6.54e+10	2.52e+11	1.25e+06	3.20e+07	2.18e+11	4.26e+10	3.38e+05
	标准差	3.43e+08	6.17e+08	1.32e+10	1.05e+10	1.55e+06	2.91e+07	1.02e+10	8.82e+09	4.05e+05
f_2	均值	3.05e+05	2.82e+05	3.47e+05	4.00e+05	4.61e+05	5.59e+05	4.88e+05	4.97e+05	3.72e+05
	标准差	3.39e+04	1.43e+04	3.29e+04	7.16e+04	9.80e+04	7.34e+04	4.31e+04	8.41e+04	4.17e+04
f_3	均值	9.96e+02	1.74e+03	4.07e+03	1.06e+05	9.35e+02	7.94e+02	5.02e+04	9.04e+03	7.80e+02
	标准差	1.01e+02	1.84e+02	1.26e+03	9.36e+03	8.44e+01	6.19e+01	5.32e+03	3.38e+03	5.32e+01
f_4	均值	6.62e+02	6.85e+02	6.38e+02	7.11e+02	6.70e+02	6.06e+02	7.04e+02	6.57e+02	6.04e+02
	标准差	3.10e+00	3.92e+00	4.87e+00	3.54e+00	4.81e+00	1.19e+00	3.49e+00	5.92e+00	1.35e+00
f_5	均值	2.65e+04	4.41e+04	6.46e+04	2.86e+05	2.30e+04	2.06e+04	1.55e+05	6.01e+04	1.38e+04
	标准差	8.95e+03	1.67e+04	1.88e+04	8.44e+04	7.77e+03	8.64e+03	2.05e+04	2.00e+04	4.80e+03
f_6	均值	2.22e+08	9.90e+08	7.58e+09	1.75e+11	6.34e+08	5.17e+07	1.00e+11	8.14e+09	1.88e+07
	标准差	1.11e+08	3.10e+08	3.92e+09	1.71e+10	3.70e+08	2.89e+07	9.10e+09	3.40e+09	7.57e+06
f_7	均值	2.88e+06	4.13e+06	6.05e+06	7.33e+07	3.34e+06	2.77e+06	5.30e+07	1.91e+07	1.20e+06
	标准差	1.14e+06	1.58e+06	3.14e+06	4.11e+07	1.80e+06	1.73e+06	1.71e+07	1.00e+07	5.28e+06
f_8	均值	3.11e+06	4.77e+06	6.94e+06	1.51e+08	4.33e+06	6.08e+06	9.12e+07	1.27e+07	2.99e+06
	标准差	1.33e+06	1.74e+06	4.64e+06	7.09e+07	2.53e+06	2.97e+06	2.67e+07	6.69e+06	1.24e+06
f_9	均值	6.38e+05	1.28e+07	1.30e+08	1.97e+10	1.54e+07	7.91e+03	6.96e+09	4.75e+06	3.43e+03
	标准差	3.44e+05	6.34e+06	1.49e+08	5.23e+09	9.81e+06	6.16e+03	1.36e+09	7.06e+06	1.52e+03
f_{10}	均值	6.08e+03	6.04e+03	5.21e+05	7.86e+03	5.31e+03	6.91e+03	7.61e+03	5.10e+03	6.79e+03
	标准差	4.76e+02	6.06e+02	9.33e+02	2.96e+02	5.82e+02	7.91e+02	2.84e+02	5.56e+02	7.17e+02
f_{11}	均值	2.02e+04	2.55e+04	2.21e+04	3.53e+04	1.92e+04	3.32e+04	3.46e+04	1.95e+04	3.30e+04
	标准差	1.68e+03	1.49e+03	6.08e+03	7.07e+02	1.93e+03	1.59e+03	5.97e+02	4.18e+03	1.04e+03
f_{12}	均值	5.06e+03	7.49e+03	4.17e+03	1.11e+04	4.11e+03	3.63e+03	7.82e+03	5.29e+03	3.60e+03
	标准差	2.72e+02	4.45e+02	1.09e+02	1.29e+03	1.54e+02	6.72e+01	2.63e+02	2.62e+02	6.27e+01
f_{13}	均值	2.38e+04	2.75e+04	1.46e+04	5.62e+04	1.43e+04	9.51e+03	4.01e+04	2.60e+04	9.35e+03
	标准差	2.53e+03	3.56e+03	1.03e+03	2.07e+03	3.42e+03	1.03e+03	2.57e+03	2.70e+03	8.04e+02
f_{14}	均值	3.80e+03	4.78e+03	8.22e+03	3.59e+04	3.72e+03	3.65e+03	2.47e+04	1.48e+04	3.57e+03
	标准差	1.02e+02	3.70e+02	1.40e+03	1.93e+03	1.30e+02	1.11e+02	1.67e+03	2.16e+03	4.14e+01