控制与决策
Control and Decision

基于群智能求解带约束问题的时域鲁棒优化算法

黄元君， 金耀初， 郝矿荣

引用本文:
黄元君, 金耀初, 郝矿荣. 基于群智能求解带约束问题的时域鲁棒优化算法[J]. 控制与决策, 2020, 35(3): 740−748.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2018.0591

文章编号: 1001-0920(2020)03-0740-09 DOI: 10.13195/j.kzyjc.2018.0591

基于群智能求解带约束问题的时域鲁棒优化算法

黄元君[1,2], 金耀初[1,3†], 郝矿荣[1]

(1. 东华大学 信息科学与技术学院，上海 201620；2. 嘉兴职业技术学院 机电与汽车分学院，浙江 嘉兴 314036；3. 英国萨里大学 计算机科学系，吉尔福德 GU27XH)

摘 要: 针对现有的时域鲁棒优化算法无法解决带约束的优化问题,基于群智能优化方法,提出一种求解带约束优化问题的时域鲁棒优化算法. 首先,用约束条件构造罚函数,将带约束优化问题处理成为无约束优化问题;然后,采用一个分段函数作为粒子的适应度评价函数,通过竞争规则筛选粒子,设计带约束问题的时域鲁棒优化算法. 以优化碳纤维原丝的性能为背景,将算法在多组参数下进行测试和对比分析,结果表明了所提出算法的有效性. 进一步分析AR模型对算法性能的影响,指出预测模型的改进是提升算法性能的一个重要手段.

关键词: 鲁棒优化；动态优化；时域鲁棒优化；群智能算法；约束优化

中图分类号: TP18 **文献标志码:** A

Robust optimization over time for constrained optimization based on swarm intelligence

HUANG Yuan-jun[1,2], *JIN Yao-chu*[1,3†], *HAO Kuang-rong*[1]

(1. College of Information Science and Technology，Donghua University，Shanghai 201620，China；2. Mechanical and Automotive Branches，Jiaxing Vocational and Technology College，Jiaxing 314036，China；3. Department of Computer Science，University of Surrey，Guildford GU27XH，United Kingdom)

Abstract: As constrained optimization problem can not be solved by existing robust optimization over time , this paper proposes a new algorithm for robust optimization over time for solving constrained optimization problems based on swarm intelligence. Firstly, a penalty function with constraints is constructed and the constrained optimization problems are processed into the unconstrained optimization problems. Then, the algorithm of robust optimization over time for constrained is designed, which the piecewise function is used as the fitness evaluation function of the particle, and the particle is selected by the competition rule. Finally, The proposed algorithm is evaluated and compared under different parameter settings for optimization of the performance of a carbon fiber precursor to demonstrate its effectiveness. Furthermore, the influence of AR model on the performance of the algorithm is analyzed. It is pointed out that the improvement of the prediction model is an important way to improve the performance of the algorithm.

Keywords: robust optimization；dynamic optimization；robust optimization over time；swarm intelligence algorithm；constrained optimization

0 引 言

在优化问题中,外界因素往往存在一定的不确定性(或随时间的推移而发生变化)[1-2],如元器件老化[3]、工艺参数波动、供求关系改变等. 在进化算法领域里处理不确定性优化问题主要有以下两种方法[1]:一是用鲁棒优化方法(robust optimization, RO)[4],使其获得解的性能在不确定条件下维持在可接受的范围内;二是采用动态跟踪优化(tracking moving optimization, TMO)[1,5-7],即一旦条件发生变化时,对问题进行重新优化. 但这两种方法均有不足之处,前者是基于不确定量较小的情况下,而实际问题中会存在较大的不确定量,后者则要求算法能及时地找到新的最优解[7],并且能顺利地将新解切入到系统,而实际中算法往往达不到这样的灵敏度且无法保证每一

收稿日期: 2018-05-05；**修回日期:** 2018-10-10.

基金项目: 国家自然科学基金项目(61806051, 61503075, 61603090)；英国自然和工程科学研究会基金项目(EP/M017869/1)；上海市科学技术委员会国际合作项目(16510711100)；浙江省自然科学基金项目(LY18F030010)；中央高校基本科研业务费专项资金项目(2232015D3-32, 2232016D-32)；上海市扬帆计划项目(17YF1426100).

责任编委: 林崇.

†通讯作者. E-mail: yaochu.jin@surrey.ac.uk.

次解都能顺利地被切换[8].

为此,Jin 等将鲁棒优化和跟踪优化的思想相结合提出了时域鲁棒优化(robust optimization over time, ROOT)[8]. 其思想为:找一个尽可能长时间被使用的鲁棒解,即当外界条件发生变化后,若该解还能使得目标性能维持在一定的水平时,则仍然使用该解,直到后续某个新的动态环境下,该解无法满足性能要求时,再重新优化寻找新的鲁棒解. 其优点是,减少解的更换次数,避免因更换解带来的许多问题. 自ROOT提出后,其相关的研究得到了进一步发展[9-17]. 如文献[9]中给出了一个求解ROOT的基本框架. 文献[10]中指出了文献[8]中的鲁棒性定义存在不足,重新定义了两种鲁棒性,并给出了求解ROOT的新算法. 文献[11]针对现有鲁棒算法的评价指标存在不足,提出了3种新的评价指标. ROOT方法现面临的困难和挑战在文献[12]中做了分析. 文献[14-15]中将ROOT方法拓展到了动态多目标优化领域,提出了求解时域鲁棒pareto解的算法. 文献[16]基于测试问题的空间特性提出了4种策略,并根据策略选择鲁棒解. 为考虑时域鲁棒优化解的切换代价,文献[17]提出了一种考虑切换代价的ROOT/SC算法. 综上所述,对ROOT已有的研究都还集中在不带约束的动态优化问题中,然而,实际优化问题中往往会带有一些约束条件,例如设备、资金、能耗等限制.

另外,基于群体信息共享机制的群智能优化算法,例如粒子群算法(PSO)[18]、蚁群算法[19]、人工蜂群算法[20]、人工鱼群算法[21]、蝙蝠算法[22]、布谷鸟搜索算法[23]等,它们与传统的优化方法相比,求解不受目标函数的可微性及连续性等条件的限制,且具有搜索效率高、容易实现、无需求偏导等优点,在处理优化问题上显示出巨大的优势[7, 24]. 如文献[5]和文献[7]分别采用了单种群和多种群PSO方法来跟踪动态最优解获得了良好的效果. 文献[25]采用多种群协同微粒群算法来优化流数据. 文献[26]提出了多群体的单变量边缘分布算法求解动态优化问题. 文献[24]对群智能体优化处理动态优化问题做了详细的综述. 此外它们在求解带约束的优化问题上也取得了丰富的成果[27-31]. 如文献[28]将PSO与遗传算法相结合,提出PSO-GA启发式算法求解约束优化问题. 文献[29]将区间分析方法与粒子群优化相结合求解约束优化问题. 文献[31]用粒子群算法求解复杂约束的鲁棒均值投资组合问题等.

鉴于现有的ROOT算法都是解决无约束优化问题,未见有关对求解带约束优化问题的ROOT算法的研究,本文基于群智能优化策略,提出一种求解带约束优化问题的ROOT新算法.

1　相关知识

1.1　问题描述

本文针对目标函数中参数随时间变化且带约束条件的一类动态优化问题进行研究,其数学描述为

$$\max\ f(\boldsymbol{x}, \boldsymbol{\alpha}(t)). \tag{1}$$

$$\text{s.t.}\ \begin{cases} g_i(\boldsymbol{x}) \leqslant 0,\ i = 1, 2, \ldots, q; \\ g_j(\boldsymbol{x}) = 0,\ j = q+1, q+2, \ldots, m; \\ h_k \leqslant \boldsymbol{x}_k \leqslant \mu_k,\ k = 1, 2, \ldots, n. \end{cases}$$

其中: f 为目标函数; $g_i(\boldsymbol{x})(i = 1, 2, \ldots, m)$ 为约束函数; m 为约束函数的总个数; $\boldsymbol{x}$ 为决策向量, $\boldsymbol{x}_k$ 为第 k 维的决策分量,其中 $k = 1, 2, \ldots, n$, n 为决策变量总的维度; $S = \{\boldsymbol{x} \in R^n | h_k \leqslant \boldsymbol{x}_k \leqslant \mu_k, k = 1, 2, \ldots, n\}$ 为搜索空间; $D = \{\boldsymbol{x} \in S | g_i(\boldsymbol{x}) \leqslant 0, i = 1, 2, \ldots, q, \bigcap g_j(\boldsymbol{x}) = 0, j = q+1, q+2, \ldots, m\}$ 为问题的可行域; $\boldsymbol{x} \in D$ 为问题的可行解; $\boldsymbol{\alpha}(t)$ 为时变参数, t 表示连续或离散的时间变量. 与本领域内以往的研究工作一样[7-13],假定 $\boldsymbol{\alpha}(t)$ 变化都是发生在某一个时刻,且在一段时间(或称为一个环境)内保持不变. 因此,动态目标函数可以描述成由一系列静态函数构成的函数序列[9-13],即

$$\langle f(\boldsymbol{x}, \boldsymbol{\alpha}_1), f(\boldsymbol{x}, \boldsymbol{\alpha}_2), \ldots, f(\boldsymbol{x}, \boldsymbol{\alpha}_i), \ldots, f(\boldsymbol{x}, \boldsymbol{\alpha}_L) \rangle. \tag{2}$$

其中: $\boldsymbol{\alpha}_i(i = 1, 2, \ldots, L)$ 表示在第 i 个环境里保持不变的参数, L 为总变化次数(或称总环境个数).

解决式(1)这类动态优化问题的传统方法是采用跟踪优化:即一旦参数 $\boldsymbol{\alpha}(t)$(或称环境)变化后,重新找 $f(\boldsymbol{x}, \boldsymbol{\alpha}_i)$ 的最优解 s^*. 但是,这种方法要求算法能及时地感知到每一个环境的变化,且能迅速地找到新解进行切换,这使其在实际应用中遇到了制约[8].

1.2　时域鲁棒优化(ROOT)理论

为克服跟踪优化的制约,文献[8]中以全新的角度提出了ROOT方法,使得每个解尽可能地在多个动态连续环境中使用,以减少解的切换次数. 根据ROOT思想,文献[10]将解的使用时间(或称生存时间)定义为解的鲁棒性(或称鲁棒值),即

$$R(\boldsymbol{x}, t, \delta) = \begin{cases} 0,\ f(\boldsymbol{x}, \boldsymbol{\alpha}_i) < \delta; \\ \max\{l | t \leqslant i \leqslant t + l : f(\boldsymbol{x}, \boldsymbol{\alpha}_i) \geqslant \delta\},\ \text{otherwise}. \end{cases} \tag{3}$$

其中: $R(\boldsymbol{x},t,\delta)$表示解$\boldsymbol{x}$在时域上的鲁棒性,即从$t$时刻开始,满足目标函数$f(\boldsymbol{x},\boldsymbol{\alpha}_i)\geqslant\delta$下该$\boldsymbol{x}$能维持的时间长度(或环境个数),$\delta$为(用户给定)鲁棒阈值,$l$为时间长度(或环境个数). 于是,在时域鲁棒优化中优化的目标函数变为

$$\max R(\boldsymbol{x},t,\delta) \text{ or } \min -R(\boldsymbol{x},t,\delta), \tag{4}$$

即找鲁棒性最大的解$\boldsymbol{x}$代替最优解$s^{\star}$.

当ROOT找到鲁棒解$\boldsymbol{x}$,若其对应的鲁棒性$R(\boldsymbol{x})=l=L$,则表明该解能适用于整个时期内所有的动态环境,这类似于一般鲁棒性定义[4]下的鲁棒解;若其鲁棒性$R(\boldsymbol{x})=l=0$,则表明该解不能维持当前的动态环境变化,需要时时更换该解,这类似于跟踪优化;若解的鲁棒性$0<R(\boldsymbol{x})=l<L$,则表示介于两者之间. 因此,时域鲁棒优化是联系跟踪优化和鲁棒优化这两者间的一种新型优化方法.

1.3 ROOT算法框架

图1给出了ROOT算法框架,它包括求解器(solver)、估计器(estimator)和数据库(database). 常用的进化算法都可作为求解器,如粒子群算法、遗传算法、差分进化算法等. 数据库用来存放历史数据、当前数据和预测数据. 这些数据用于构建预测器(predictor),只要是合适的预测技术都可以采用,如线性回归模型(autoregressive models, AR)、支持向量机

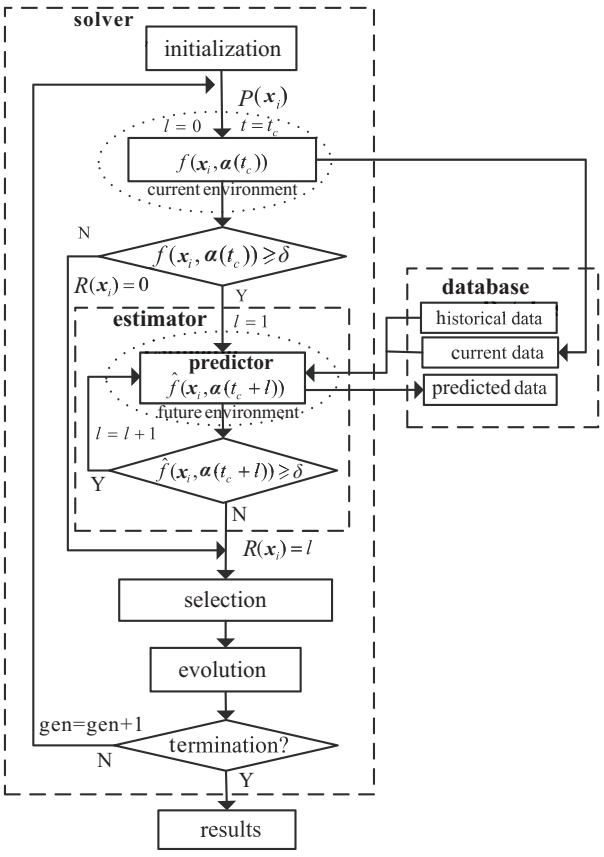

图1 ROOT算法框架

等. 整个算法的工作原理:算法初始化种群$P(\boldsymbol{x}_i)$后,以解的鲁棒性作为适应度评价函数评估每个解. 由式(3)的鲁棒性定义,若当前环境下(图1中,$t=t_c,l=0$表示$f(\boldsymbol{x}_i,\boldsymbol{\alpha}(t_c))$在当前时刻环境),不满足$f(\boldsymbol{x}_i,\boldsymbol{\alpha}(t_c))\geqslant\delta$,则$\boldsymbol{x}_i$的鲁棒性为$R(\boldsymbol{x}_i)=0$;若满足$f(\boldsymbol{x}_i,\boldsymbol{\alpha}(t_c))\geqslant\delta$,则$\boldsymbol{x}_i$通过estimator估计输出该解的鲁棒性$R(\boldsymbol{x}_i)=l$. 注意:在estimator里时间变量$l=l+1$表示转入下一时刻(或称下一个环境). 因实际问题中,未来时刻的参数$\boldsymbol{\alpha}(t)$是动态的,使得未来的目标函数$f(\boldsymbol{x}_i,\boldsymbol{\alpha}(t_c+l))$很难获得,因此$\boldsymbol{x}_i$在未来时刻的目标函数值通过predictor预测获得$\hat{f}(\boldsymbol{x}_i,\boldsymbol{\alpha}(t_c+l))$来代替真实的$f(\boldsymbol{x}_i,\boldsymbol{\alpha}(t_c+l))$,从而根据式(3)估计出该解的鲁棒性为$R(\boldsymbol{x}_i)=l$.然后,算法进入选择(selection)、进化(evolution),直到进化代数终止(termination),返回最优鲁棒解. 更具体的描述可见文献[10-12].

2 带约束问题的ROOT算法

现有ROOT算法研究都是集中在求解无约束条件的优化问题,因此,无法用它来求解如问题(1)这类带约束条件优化问题的时域鲁棒解.

2.1 约束处理

为求解式(1)这类带约束优化问题的时域鲁棒解,本文先将式(1)中约束条件构造成一个罚函数

$$V(\boldsymbol{x})=\sum_{i=1}^{q}\max\{g_i(\boldsymbol{x}),0\}+\sum_{j=q+1}^{m}|g_j(\boldsymbol{x})|. \tag{5}$$

其中:$\max\{g_i(\boldsymbol{x}),0\}(i=1,2,\ldots,q)$表示$\boldsymbol{x}$在第$i$个不等式约束的违反程度,$|g_j(\boldsymbol{x})|(j=q+1,q+2,\ldots,m)$表示$\boldsymbol{x}$在第$j$个不等式约束的违反程度,罚函数$V(\boldsymbol{x})$表示总的违反程度. 显然,若点$\boldsymbol{x}\in D$,则有$V(\boldsymbol{x})=0$. 求解这类有约束优化问题的常用方法[27]是引入参数λ,合并罚函数与目标函数得到如下公式:

$$Q(\boldsymbol{x})=-R(\boldsymbol{x},t,\delta)+\lambda V(\boldsymbol{x}), \tag{6}$$

从而将有约束问题转化为无约束问题,再按无约束的方法来求解. 注意:本文按最小化的标准形式推导,所以式(6)中取$-R(\boldsymbol{x},t,\delta)$. 但是,罚因子的选取是一个较难的问题,罚因子太小,难以产生可行解;罚因子太大,则会出现过早的收敛[29].

为避免以上问题,本文结合群智能算法的评价、选择、进化机制,将优化目标函数$-R(\boldsymbol{x},t,\delta)$和罚函数$V(x)$分离,采用一个分段函数

$$\text{Fitness}(\boldsymbol{x})=\begin{cases}-R(\boldsymbol{x},t,\delta),\ V(\boldsymbol{x})=0;\\ V(\boldsymbol{x}),\ V(\boldsymbol{x})\neq 0\end{cases} \tag{7}$$

作为粒子的适应度评价函数,通过竞争规则筛选粒子,设计求解带约束问题的时域鲁棒优化算法. 式(7)表示:若罚函数$V(\boldsymbol{x})=0$,则该粒子落在可行区域,以$-R(\boldsymbol{x},t,\delta)$为适应度;若罚函数$V(\boldsymbol{x})\neq 0$,则该粒子落在非可行区域,以$V(\boldsymbol{x})$为适应度.

2.2　算法设计

PSO算法[18]是模仿鸟群觅食行为的一种典型的群智能优化算法. 该算法通过粒子在解空间追随最优粒子进行迭代搜索寻找最优解,具有简单、容易实现、调整参数少等优点. 因此,本文采用PSO为优化器,结合以上的约束处理方法和后文对选择算子的定义,提出一种求解带约束优化问题的ROOT算法,流程见图2,其基本步骤如下:

1) 初始化种群$P(\boldsymbol{x})$.

2) 根据式(5)计算每个粒子的$V(\boldsymbol{x}_i)$值.

3) 根据式(7)确定每个粒子的fitness$(\boldsymbol{x}_i)$.

4) 根据fitness$(\boldsymbol{x}_i)$,通过后文给出的算法2选出更优粒子.

5) 更新局部、全局最优粒子和粒子位置.

6) 迭代是否终止? 是,终止算法,输出结果;否,gen = gen + 1,跳到第2)步.

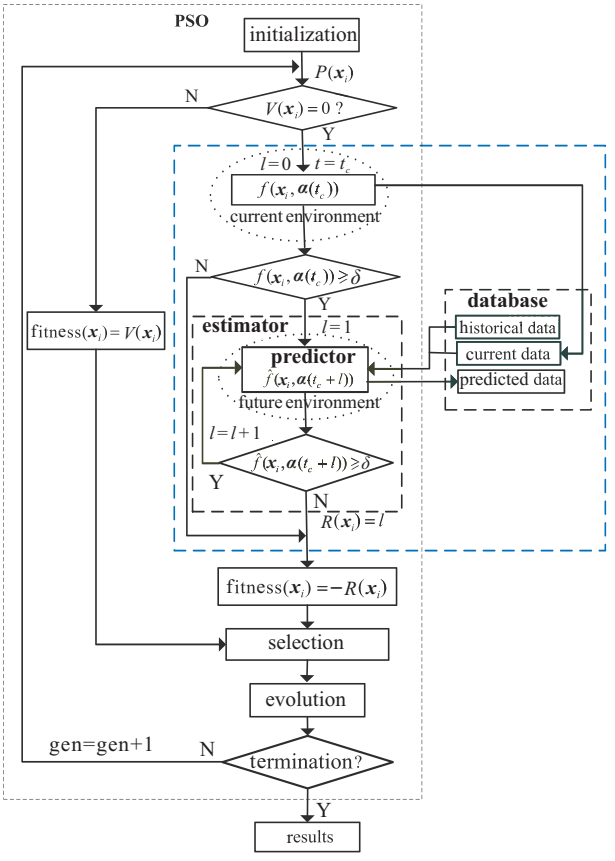

图 2　求解带约束优化问题的ROOT算法流程

算法1　求解带约束优化问题的ROOT主算法.

input: δ和$\boldsymbol{\alpha}(t)$// 调入参数;

return: $\boldsymbol{x}_{\text{gbest}}$ // 返回最优鲁棒解.

1) np, w, c_1, c_2, gen // 设置算法参数

2) for $i=1$ to np

3)　particle$(\boldsymbol{x}_i)\leftarrow$初始化粒子//初始化每个粒子

4)　$\boldsymbol{x}_{i,\text{pbest}}=\boldsymbol{x}_i$ // 初始化局部最优粒子

5)　$\boldsymbol{x}_{\text{gbest}}\leftarrow$选择算子$(\boldsymbol{x}_{i,\text{pbest}},\boldsymbol{x}_{\text{gbest}})$//见算法2

6) end for i

7) indx = 0

8) while (indx $\leqslant$ gen) // 终止条件

9)　for $i=1$ to np

10)　$\boldsymbol{v}_i^j=w\boldsymbol{v}_i^{j-1}+c_1\gamma_1(\boldsymbol{x}_{i,\text{pbest}}^{j-1}-\boldsymbol{x}_i^{j-1})+c_2\gamma_2(\boldsymbol{x}_{\text{gbest}}^{j-1}-\boldsymbol{x}_i^{j-1})$

11)　$\boldsymbol{x}_i^j=\boldsymbol{x}_i^{j-1}+\boldsymbol{v}_i^j$

12)　$\boldsymbol{x}_{i,\text{pbest}}^j\leftarrow$选择算子$\boldsymbol{x}_i,\boldsymbol{x}_{i,\text{pbest}}$// 更新局部最优

13)　$\boldsymbol{x}_{\text{gbest}}\leftarrow$选择算子$\boldsymbol{x}_{i,\text{pbest}},\boldsymbol{x}_{\text{gbest}}$//更新全局最优

14)　end for i

15)　indx = indx + 1

16) end while

为描述算法的机理,算法1给出了主算法的伪代码. 其第1行为种群大小np,粒子的惯性权重w,加速因子c_1和c_2,进化代数gen等参数的设置. 第3行表示对每个粒子进行初始化,第5行中选择算子具体见算法2(表示对比两粒子,返回更优者). 第8～第16行表示迭代更新粒子,满足终止条件后,返回最优解. 第10和第11行为更新每个粒子的速度和位置,第12和第13行为通过选择算子更新局部最优、全局最优粒子.

算法2　选择算子.

input: $\boldsymbol{x}_1,\boldsymbol{x}_2$ // 调入竞争的粒子;

return: $\boldsymbol{x}_{\text{better}}$ // 返回胜出粒子.

1) if $V(\boldsymbol{x}_1)=0, V(\boldsymbol{x}_2)=0$ and $-R(\boldsymbol{x}_1,t,\delta)\leqslant -R(\boldsymbol{x}_2,t,\delta)$

2)　$\boldsymbol{x}_{\text{better}}=\boldsymbol{x}_1$

3) end if

4) if $V(\boldsymbol{x}_1)=0, V(\boldsymbol{x}_2)>0$

5)　$\boldsymbol{x}_{\text{better}}=\boldsymbol{x}_1$

6) end if

7) if $V(\boldsymbol{x}_1)>0, V(\boldsymbol{x}_2)>0$ and $V(\boldsymbol{x}_1)\leqslant V(\boldsymbol{x}_2)$

8)　$\boldsymbol{x}_{\text{better}}=\boldsymbol{x}_1$

9) end if

对于选择算子中的任意两个粒子$\boldsymbol{x}_1,\boldsymbol{x}_2\in S$,若

有以下情形之一成立,则$\boldsymbol{x}_1$优于$\boldsymbol{x}_2$:

1) 若$\boldsymbol{x}_1,\boldsymbol{x}_2\in D$且$-R(\boldsymbol{x}_1,t,\delta)\leqslant -R(\boldsymbol{x}_2,t,\delta)$;

2) 若$\boldsymbol{x}_1\in D,\boldsymbol{x}_1\in S\setminus D$;

3) 若$\boldsymbol{x}_1,\boldsymbol{x}_2\in S\setminus D$且$V(\boldsymbol{x}_1)\leqslant V(\boldsymbol{x}_2)$.

算法2给出了选择算子的伪代码. 其中,里面涉及到对鲁棒性$-R(\boldsymbol{x}_i,t,\delta)$的计算见算法3.

算法3 计算解的鲁棒性.

input: $\boldsymbol{x},t,\delta$// 调入参数;

return: $R(\boldsymbol{x},t,\delta)$// 返回结果.

1) $l=0,t_c=t$// 初始化时间指引l,t_c为当前时刻

2) if $f(\boldsymbol{x},\boldsymbol{\alpha}(t_c+l))\geqslant\delta$

3) $l\leftarrow l+1$

4) repeat

5) updata DB// DB为数据库

6) $\hat{f}(\boldsymbol{x},\boldsymbol{\alpha}(t_c+l))\leftarrow \text{predictor(DB)}$

7) $l\leftarrow l+1$// 转移到下一时刻

8) until $\hat{f}(\boldsymbol{x},\boldsymbol{\alpha}(t_c+l))<\delta$

9) end if

10) $R(\boldsymbol{x},t,\delta)=l$ // 返回解$\boldsymbol{x}$的鲁棒性

算法3给出了计算每个解的鲁棒性$R(\boldsymbol{x},t,\delta)$的伪代码,其中$t_c=t$表示当前时刻,$l=0$表示初始化时间指引. 假如$\boldsymbol{x}_i$满足条件$f(\boldsymbol{x},\boldsymbol{\alpha}(t_c+l))\geqslant\delta$,则$l\leftarrow l+1$表示进入下一时刻(新的环境). 算法中第5~第8行表示不断地更新$l\leftarrow l+1$,直到$\hat{f}(\boldsymbol{x},\boldsymbol{\alpha}(t_c+l))<\delta$为止. 最终返回每个解$\boldsymbol{x}$的鲁棒性$R(\boldsymbol{x},t,\delta)=l$. 因未来时刻的$f(\boldsymbol{x},\boldsymbol{\alpha}(t_c+l))$是未知的,需用历史数据构建预测器(predictor)预测出其函数值[9-10](第5和第6行).

3 实验测试

3.1 工程实例

为验证算法的有效性,本节以碳纤维纺丝过程中的牵伸比与原丝性能之间的优化为例. 其中,干喷湿纺法制备聚丙烯腈基碳纤维原丝工艺流程如图3所示,根据文献[32]可得到如$\boldsymbol{\alpha}(t)=0.364$的6级总牵伸比与原丝性能之间的数学模型

$$f(\boldsymbol{x},\boldsymbol{\alpha}(t))=\boldsymbol{\alpha}(t)\boldsymbol{x}^{0.99}\mathrm{e}^{-\frac{5.49}{\boldsymbol{x}}};\tag{8}$$

$$\text{s.t. } 0.37\boldsymbol{x}^{0.99}\mathrm{e}^{-\frac{14.28}{\boldsymbol{x}}}\geqslant 10,\tag{9}$$

$$10\leqslant \boldsymbol{x}=\prod_{i=1}^{6}x_i\leqslant 18,\tag{10}$$

$$1\leqslant x_1\leqslant 4.5,\tag{11}$$

$$1\leqslant x_2\leqslant 2,\tag{12}$$

$$1\leqslant x_3\leqslant 2,\tag{13}$$

$$2\leqslant x_4\leqslant 6,\tag{14}$$

$$1\leqslant x_5\leqslant 1.3,\tag{15}$$

$$1\leqslant x_6\leqslant 2.5.\tag{16}$$

其中$f(\boldsymbol{x},\boldsymbol{\alpha}(t))$为原丝的性能,$\boldsymbol{x}=\prod_{i=1}^{6}x_i$为总牵伸比,$x_i(i=1,2,\ldots,6)$为各级的牵伸比. 式(9)表示断裂伸长率与总牵伸比之间关系,且断裂伸长率要求大于等于10;式(10)表示总牵伸比应控制在10~18为宜;式(11)~(16)表示各级牵伸比都有一定范围的限制. 实际生产过程中,原丝的性能还受到牵伸的温度、牵伸的速度、牵伸的段数等因素的影响[32],这些因素有所波动,都会影响模型参数的变化,因此用时变参数$\boldsymbol{\alpha}(t)$来描述以上总牵伸比与原丝性能之间的关系更为合理. 显然,这是一个带约束的动态优化问题,因此,可采用本文所提出的时域鲁棒优化方法来处理该问题,即保证原丝性能$f(\boldsymbol{x},\boldsymbol{\alpha}(t))\geqslant\delta$($\delta$为用户给定)的要求下,找到在动态环境下尽可能长时间使用的一个拉伸比方案. 所以该优化问题的目标函数转化为$\max R(\boldsymbol{x},t,\delta)$.

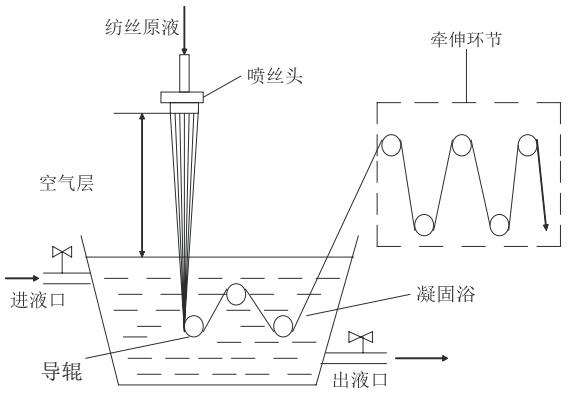

图3 干喷湿纺工艺流程图

3.2 实验设计

1) 动态环境. 实验假定式(8)中的参数在时间$t\in[0,120)$内按

$$\boldsymbol{\alpha}(t)=\boldsymbol{\alpha}(t_0)+\beta\times N(\mu,\sigma)\tag{17}$$

的形式发生了120次的动态变化,即在每一个t时刻发生一次变动,对应得到$L=120$个动态环境. 其中:β表示变化幅度,$N(\mu,\sigma)$表示产生服从均值为μ、标准差为σ的正态分布的随机数,实验中取$\boldsymbol{\alpha}(t_0)=0.364,\beta=\boldsymbol{\alpha}(t_0)10\,\%,\mu=0$. 注:实验侧重点是验证算法对动态问题能否有效求解,暂且不考虑这种动态设置是否符合实际工艺过程. 根据文献[10]以$t=16$(前15个时刻作为历史时刻,产生的数据用于

构造预测模型)到$t = 100$依次为$f(\boldsymbol{x}, \boldsymbol{\alpha}_i)$的当前起始时刻(即在每个时刻都重启算法获得当前环境下的鲁棒解),用于测试算法在不同环境下的求解效果.

2) 性能指标.采用文献[10]定义的如下指标来评价算法性能:

$$P = \bar{R} = \frac{1}{N}\sum_{i=1}^{N} R_i, \tag{18}$$

其中R_i表示$f(\boldsymbol{x}, \boldsymbol{\alpha}_i)$在第$i$个环境里解的鲁棒性. 并对获得解是否在可行域内进行检验,标记如下:

$$C = \begin{cases} 1, & \text{可行解;} \\ 0, & \text{非可行解.} \end{cases} \tag{19}$$

3) 参数设置. PSO算法的种群大小$\mathrm{np} = 50$,粒子的惯性权重$w = 0.298$,加速因子$c_1 = c_2 = 0.496$,进化代数$\mathrm{gen} = 20$. 采用一个$\psi = 5$阶的AR模型为预测器,其数学形式为$Y_t = \varepsilon + \sum_{i=1}^{\psi} \boldsymbol{\eta}_i Y_{t-i}$,$\varepsilon$为白噪声,$Y_t$为$t$时刻的时间序列数据. 每次采用距离当前最近的前15个时刻的数据作为训练模型的样本数据. 用最小二乘法估计AR模型的参数$\boldsymbol{\eta}(\boldsymbol{\eta} = (\boldsymbol{\eta}_1, \boldsymbol{\eta}_2, \ldots, \boldsymbol{\eta}_\psi))$[10].

4) 对比实验组.为更好地分析不同参数对算法的影响及算法的优化效果,需要排除AR预测模型偏差带来的干扰[11,17]. 增加一组以真实函数代替AR模型来计算鲁棒值的对比实验.

3.3　结果分析

算法在不同的鲁棒阈值$\delta = 4.4, \delta = 4.5, \delta = 4.6$和标准差(扰动程度)$\sigma = 0.5, \sigma = 1, \sigma = 10$的9种情况下分别独立运行30次,各参数下算法所获得的总体性能指标$\bar{R}_{\text{真实}}$、$\bar{R}_{\text{预测}}$及两者误差$\bar{e} = \|\bar{R}_{\text{真实}} - \bar{R}_{\text{预测}}\|$的平均值和标准差分别列在表1中(通过置信度水平0.05的Wilcoxon秩和检验,对比相同δ不同σ情况下的指标值,其中最好的标黑;同理,对比相同σ不同δ情况下的指标值,其中最好的标*). 对应每个动态测试时刻获得解的可行性标记C(式(19))值、解的鲁棒性$\bar{R}_{\text{真实}}$、$\bar{R}_{\text{预测}}$及两者误差$\bar{e} = \|\bar{R}_{\text{真实}} - \bar{R}_{\text{预测}}\|$值分别显示在图4～图6中.

表1　各参数下的指标$\bar{R}_{\text{真实}}, \bar{R}_{\text{预测}}, \bar{e}_{\text{误差}}$结果

δ	指标	σ		
		0.5	1	10
4.4	$\bar{R}_{\text{真实}}$	***4.06(0.05)**	*1.49(0.05)	*0.70(0.01)
	$\bar{R}_{\text{预测}}$	***2.52(0.20)**	*0.72(0.07)	*0.42(0.06)
	$\bar{e}_{\text{误差}}$	1.53(0.20)	0.76(0.08)	**0.28(0.06)**
4.5	$\bar{R}_{\text{真实}}$	**1.75(0.04)**	1.08(0.01)	0.64(0.01)
	$\bar{R}_{\text{预测}}$	**0.87(0.09)**	0.46(0.08)	0.39(0.04)
	$\bar{e}_{\text{误差}}$	0.87(0.09)	0.62(0.08)	**0.26(0.04)**
4.6	$\bar{R}_{\text{真实}}$	**1.06(0.02)**	0.91(0.02)	0.59(0.00)
	$\bar{R}_{\text{预测}}$	**0.41(0.07)**	0.31(0.05)	0.37(0.04)
	$\bar{e}_{\text{误差}}$	*0.65(0.08)	*0.59(0.06)	***0.21(0.04)**

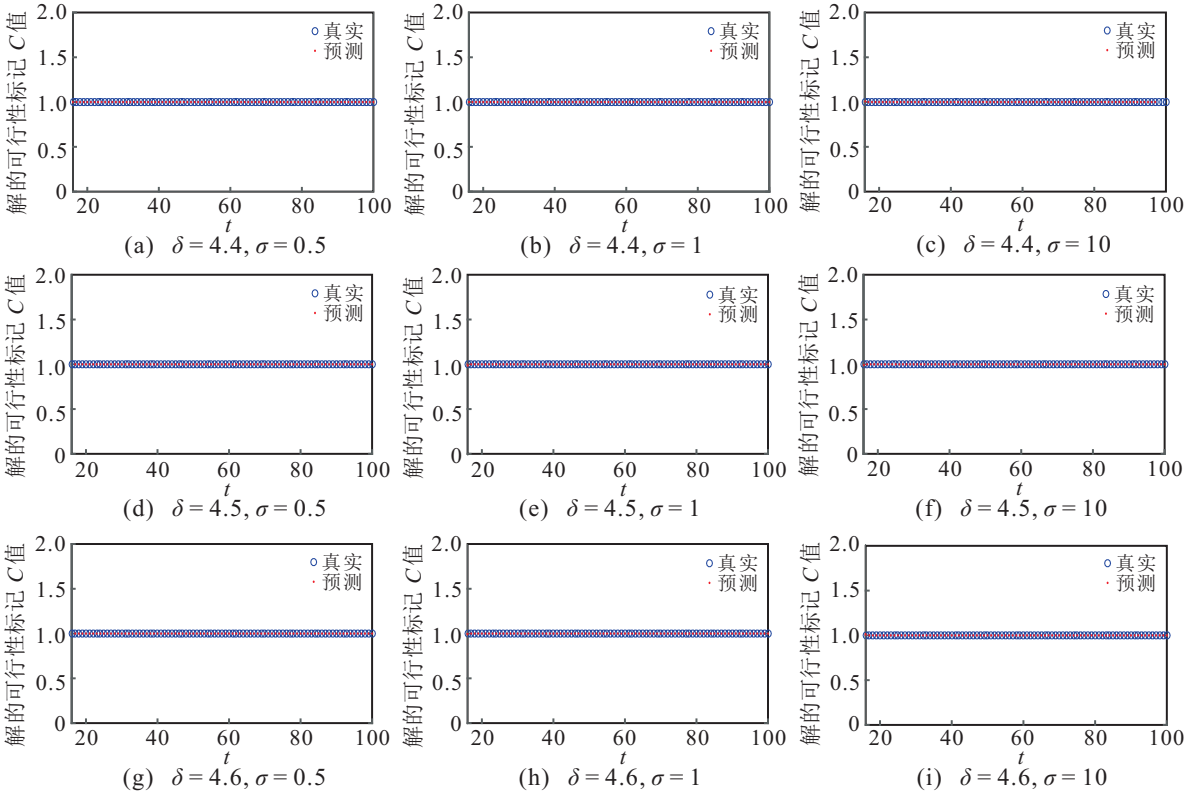

图4　不同参数下,各测试时刻(环境)下获得解的可行性标记C值

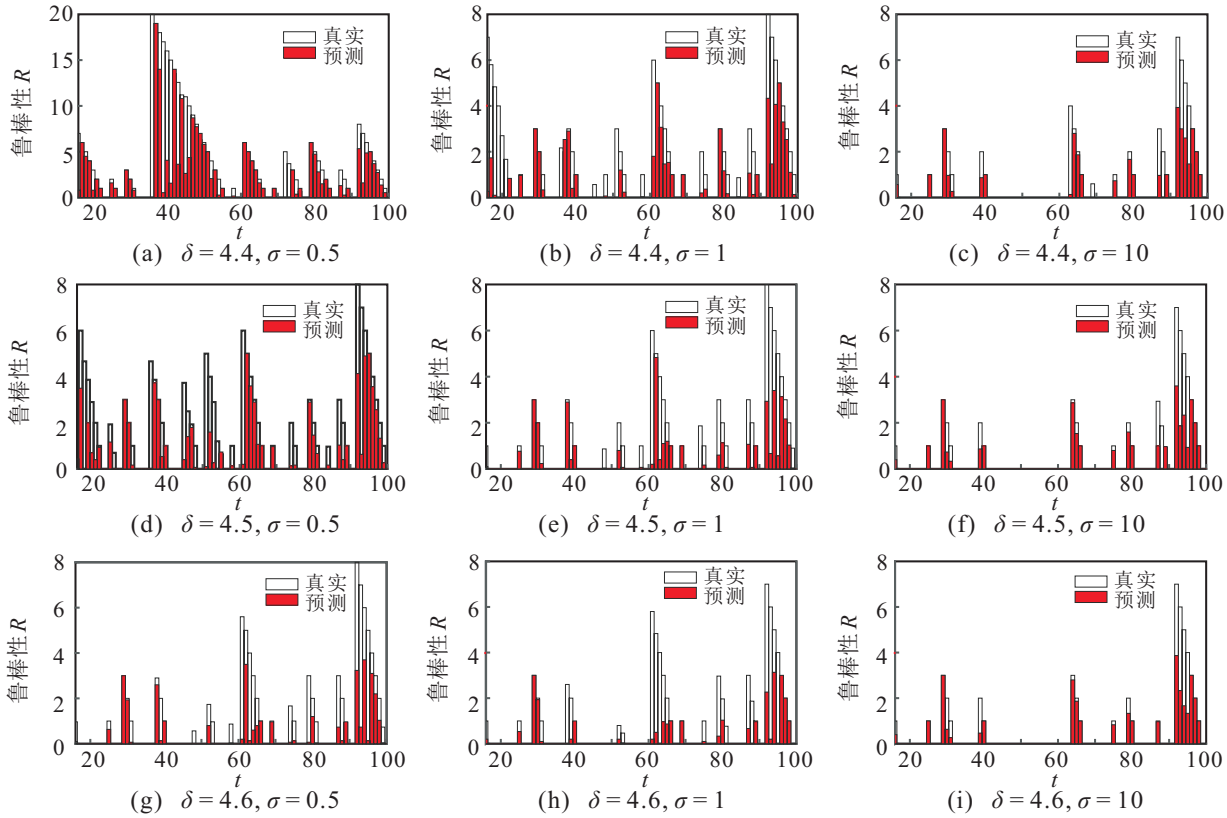

图 5 不同参数下,各测试时刻(环境)下获得解的鲁棒性

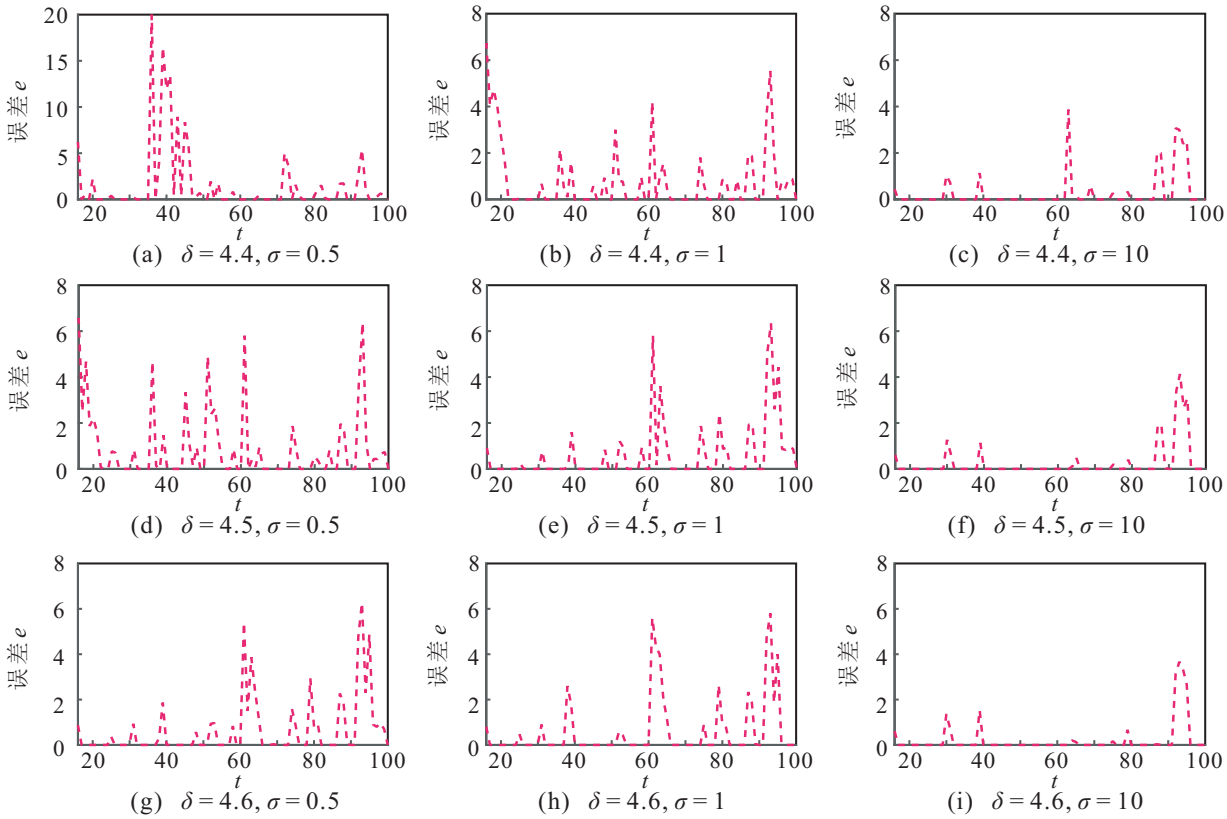

图 6 不同参数下,各测试时刻(环境)下两组实验的鲁棒性误差曲线

3.3.1 解的可行性

由图4可知:各参数下,统计两组算法在每个测试时刻(环境)进行30次独立运行所获得解的可行性标记C值都为1,表明算法每次找到的解都落在可行域内,即获得的解都是满足约束条件的可行解.

3.3.2 参数影响

由表1的性能指标及图5每一个时刻(环境)下获得解的鲁棒性可知:同一δ值下,参数σ越小,鲁棒性

指标$\bar{R}_{真实}$、$\bar{R}_{预测}$越好;同一σ值下,参数δ越小,鲁棒性指标$\bar{R}_{真实}$、$\bar{R}_{预测}$越好;这些结果与实际情况相符合. 因为σ越大表示动态变化越剧烈,δ越大表示鲁棒阈值的要求越高,这必然导致解的鲁棒性有下降的趋势. 这表明所提出的算法是有效的(排除预测偏差).

3.3.3 预测效果

由表1中的误差指标$\bar{e}$和图6各测试时刻(环境)下两组的鲁棒性误差e曲线可知:随着参数σ和δ的增大,其平均误差$\bar{e}$越小,表明预测越接近真实. 其原因是随着这些参数的变大,意味着问题本身的动态特性加剧,鲁棒特性减弱,因此算法在对鲁棒值预测时,被预测器预测的时刻会更短,显然预测的准确度会提高,估计出解的鲁棒性也越接近真实值. 这反映了预测器的效果对算法性能的影响较大.

4 结　论

时域鲁棒优化是具有跟踪优化和鲁棒优化特性的一种新型优化方法. 本文为求解带约束问题的时域鲁棒优化,首先,通过约束条件构造罚函数,将有约束问题处理成为无约束问题;其次,将目标函数与罚函数分别作为两个独立的适应度评价函数,这样避免了引入新的罚因子λ及其对λ值选择的困难,进而,凭借群智能算法的竞争选择机制来优化搜索时域鲁棒解;最后,以优化碳纤维原丝性能为背景,将所提出算法在不同组参数下进行测试,结果验证了所提出算法的有效性; 进一步对比分析了AR模型的预测效果及其对算法性能的影响,由此可知:在后续的研究中,对不同的动态环境,如何找到有效的预测方法,减少预测误差,将是提升算法性能的一个重要手段.

参考文献(References)

[1] Yang S, Jin Y, Ong Y S. Evolutionary computation in dynamic and uncertain environments[M]. Berlin, Heidelberg: Springer-Verlag, 2007: 1-28.

[2] 陈美蓉. 动态鲁棒进化优化方法研究[D]. 徐州: 中国矿业大学信息与控制工程学院, 2017.
(Chen M R. Research on dynamic robust evolutionary optimization method[D]. Xuzhou: School of Information and Control Engineering, China University of Mining and Technology, 2017.)

[3] 郭一楠, 张芹英, 巩敦卫, 等. 一类时变时延网络控制系统的鲁棒容错控制[J]. 控制与决策, 2008, 23(6): 689-696.
(Guo Y N, Zhang Q Y, Gong D W, et al. Robust fault-tolerant control of networked control systems with time-varying delays[J]. Control and Decision, 2008, 23(6): 689-696.)

[4] Lim D, Ong Y S, Jin Y, et al. Inverse multi-objective robust evolutionary optimization[J]. Genetic Programming and Evolvable Machines, 2006, 7(4): 383-404.

[5] Eberhart R C, Shi Y. Tracking and optimizing dynamic systems with particle swarms[C]. Proceeding of the IEEE Congress on Evolutionary Computation. Seoul: IEEE, 2001: 94-100.

[6] Nguyen T T, Yang S, Branke J. Evolutionary dynamic optimization: A survey of the state of the art[J]. Swarm and Evolutionary Computation, 2012, 6: 1−24.

[7] Parrott D, Li X. Locating and tracking multiple dynamic optima by a particle swarm model using speciation[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(4): 440-458.

[8] Yu X, Jin Y, Tang K, et al. Robust optimization over time a new perspective on dynamic optimization problems[C]. WCCI 2010 IEEE World Congress on Computaltion Intelligence. Barcelona: IEEE, 2010: 3998-4003.

[9] Jin Y, Tang K, Yu X, et al. A framework for finding robust optimal solutions over time[J]. Memetic Computing, 2012, 5(1): 3-18.

[10] Fu H, Sendhoff B, Tang K, et al. Finding robust solutions to dynamic optimization problems[C]. The 16th European Conference on Applications of Evolutionary Computation. Vienna: Springer, 2013: 616-625.

[11] Huang Y, Jin Y, Ding Y. New performance indicators for robust optimization over time[C]. 2015 IEEE Congress on Evolutionary Computation. Sendai: IEEE, 2015: 1380-1387.

[12] Fu H, Sendhoff B, Tang K, et al. Robust optimization over time: Problem difficulties and benchmark problems[J]. IEEE Transactions on Evolutionary Computation, 2015, 19(5): 731-745.

[13] Guo Y, Chen M, Fu H. Find robust solutions over time by two-layer multi-objective optimization method[C]. 2014 IEEE Congress on Evolutionary Computation. Beijing: IEEE, 2014: 1528-1535.

[14] Chen M, Guo Y, Liu H. The evolutionary algorithm to find robust Pareto-optimal solutions over time[J]. Mathematical Problems in Engineering, 2015, 6: 1-18.

[15] 陈美蓉, 郭一楠, 巩敦卫, 等. 一类新型动态多目标鲁棒进化优化方法[J]. 自动化学报, 2017, 43(11): 2014-2032.
(Chen M R, Guo Y N, Gong D W, et al. A novel dynamic multi-objective robust evolutionary optimization method[J]. Acta Automatica Sinica, 2017, 43(11): 2014-2032.)

[16] Yazdani D, Nguyen T T, Branke J. Robust optimization over time by learning problem space characteristics[J].

IEEE Transactions on Evolutionary Computation, 2019, 23(1): 143-155.

[17] Huang Y, Ding Y, Jin Y. A multi-objective approach to robust solutions over time considering switching cost[J]. Information Sciences, 2017, 394: 183-197.

[18] Kennedy J, Eberhart R. Particle swarm optimization[C]. Proceeding of IEEE International Congress on Neural Networks. Perth: IEEE, 1995: 1942-1948.

[19] Yang Q, Chen W, Yu Z. Adaptive multimodal continuous ant colony optimization[J]. IEEE Transactions on Evolutionary Computation, 2017, 21(2): 191-205.

[20] 宁爱平, 张雪英. 人工蜂群算法的收敛性分析[J]. 控制与决策, 2013, 28(9): 1554-1558.
(Ning A P, Zhang X Y. Convergence analysis of artificial bee colony algorithm[J]. Control and Decision, 2013, 28(9): 1554-1558.)

[21] Zhang Z, Wang K, Zhu L, et al. A pareto improved artificial sh swarm algorithm for solving a multi-objective fuzzy disassembly line balancing problem[J]. Expert Systems with Applications, 2017, 86: 1339-1351.

[22] AzmiAl-Betar M, Awadallah M, Faris H, et al. Bat-inspired algorithms with natural selection mechanisms for global optimization[J]. Neurocomputing, 2018, 273: 448-465.

[23] Abdelbasset M, Hessin A, Abdelfatah L. A comprehensive study of cuckoo-inspired algorithms[J]. Neural Computing and Applications, 2018, 29(2): 345-361.

[24] Mavrovouniotis M, Li C, Yang S. A survey of swarm intelligence for dynamic optimization: Algorithms and applications[J]. Swarm and Evolutionary Computation, 2017, 33: 1-17.

[25] 张勇, 夏长红, 巩敦卫, 等. 基于多种群协同微粒群优化的流数据聚类算法[J]. 控制与决策, 2016, 31(10): 1879-1883.
(Zhang Y, Xia C H, Gong D W, et al. Streaming data clustering using cooperative particle swarm optimization[J]. Control and Decision, 2016, 31(10): 1879-1883.)

[26] 武燕, 王宇平, 刘小雄, 等. 求解动态优化问题的多群体UMDA[J]. 控制与决策, 2008, 23(12): 1401-1412.
(Wu Y, Wang Y P, Liu X X, et al. Multi-population univariate marginal distribution algorithm for dynamic optimization problems[J]. Control and Decision, 2008, 23(12): 1401-1412.)

[27] 卜晨阳. 演化约束优化及演化动态优化求解算法研h究[D]. 安徽: 中国科学技术大学计算机科学与技术学院, 2017.
(Bu C Y. Research on evolutionary constrained optimization and evolutionary dynamic optimization algorithms[D]. Anhui: School of Computer Science and Technology, University of Science and Technology of China, 2017.)

[28] Ggarg H. A hybrid PSO-GA algorithm for constrained optimization problems[J]. Applied Mathematics and Computation, 2017, 274: 292-305.

[29] Coelho T M, Machado A M C, Soares G L. An interval space reducing method for constrained problems with particle swarm optimization[J]. Applied Soft Computing, 2017, 59: 405-417.

[30] 刘衍民. 解决约束优化问题的一种改进PSO研究[J]. 系统仿真学报, 2011, 23(10): 2130-2133.
(Liu Y M. Research of improved PSO for solving constrained optimization problems[J]. Journal of System Simulation, 2011, 23(10): 2130-2133.)

[31] 李军, 周建力. 考虑复杂约束的鲁棒均值-CVaR投资组合模型及粒子群算法[J]. 控制与决策, 2016, 31(12): 2219-2224.
(Li J, Zhou J L. Robust mean-CVaR portfolio selection model with complicated realistic constraints and its improved particle swarm optimization algorithm[J]. Control and Decision 2016, 31(12): 2219-2224.)

[32] 陈佳佳. 碳纤维纺丝过程的协同模型与智能优化研究[D]. 上海: 东华大学信息科学与技术学院, 2013.
(Chen J J. Study on the cooperative modelling and intelligent optimization of carbon fiber spinning process[D]. Shanghai: College of Information Science and Technology, Donghua University, 2013.)

作者简介

黄元君(1983−), 男, 博士生, 从事智能算法、动态优化、鲁棒优化的研究, E-mail: huangyuanjungege@126.com;

金耀初(1966−), 男, 教授, 博士生导师, 从事进化优化、机器学习、进化发育系统等研究, E-mail: yaochu.jin@surrey.ac.uk;

郝矿荣(1964−), 女, 教授, 博士生导师, 从事智能优化、类脑智能、机器视觉等研究, E-mail: krhao@dhu.edu.cn.

(责任编辑：孙艺红)