Journal of Computers Vol. 37 No. 1, February 2026, pp. 129-142
https://doi.org/10.63367/199115992026023701008

Multi-solution Robust Optimization Over Time with Adaptive Population Control

Yiyong Wang[1], Lanlan Kang[2,1*], Wenliang Cao[3], Hongping Xiao[1], Peijia Zhu[1], and Juan Li[2]

[1] School of Information Engineering, Jiangxi University of Science and Technology,
Ganzhou 341000, China
6720230870@mail.jxust.edu.cn, 6720230785@mail.jxust.edu.cn,
6720230777@mail.jxust.edu.cn

[2] School of Information Engineering, Gannan University of Science and Technology,
Ganzhou 341000, China
victorykll@163.com, 9320220026@gnust.edu.cn

[3] School of Electronic Information, Dongguan Polytechnic,
Guangdong 523808, China
caowl22@163.com

Received 10 March 2025; Revised 9 April 2025; Accepted 10 April 2025

Abstract. Robust Optimization Over Time (ROOT) is used to solve dynamic optimization problems with the aim of finding solutions that can be accepted over a long time. Most of the researches in this field try to seek new robust solutions by predicting the future fitness values of candidate solutions. However, predicting future fitness value is error prone. Therefore, this paper propose a multi-solution robust optimization over time with adaptive population control (MROOT-AC). Firstly, a adaptive population control mechanism was proposed to optimize the population in the unexplored promising region according to the convergence state of the population, so as to improve the diversity of the population. Secondly, in order to prevent the inefficiency and resource waste caused by frequent changes of deployment scheme, a multi-solution archive management mechanism has been proposed. Compared with the existing ROOT algorithms, the experimental results on the Generalized Moving Peak Benchmark (GMPB) show that the proposed algorithm can significantly improve the performance of the robust solution.

Keywords: robust optimization over time, multi-solution, dynamic optimization, multi-population

1 Introduction

Many real-world optimization problems are dynamic and change over time, which is called dynamic optimization problems (DOPs) [1]. For example, in the scenario of image dehazing, the images captured under a hazy environment will significantly affect image processing tasks such as target recognition. However, the factors that influence the formation of the haze, such as atmospheric flow and changes in the distribution of suspended particles - all exhibit dynamic characteristics [2]. Most existing approaches solve DOPs by tracking the move global optimal solution (TMO) after each environment change. However, TMO is impractical for solving many real-world DOPs because frequently changing the deployed solution leads to high switching cost and the system intolerance. In taking-off scheduling problem, it is desirable to keep the current implemented schedule after an environmental change to avoid unfavorable disruptions in airport operations [3].

In order to solve the above problems, researchers have proposed various ROOT methods in recent years. ROOT methods can be divided into two categories: 1) Approaches that leverage the predicted fitness of candidate solutions to estimate their robustness; 2) Approaches that operate by utilizing the estimated robustness of the promising regions.

In [4], the authors first proposed a ROOT method to solve DOPs, where to find an "optimal" solution, so that the solution is feasible or acceptable in a period of time as much as possible. In [5], the authors proposed a hybrid prediction algorithm containing multiple prediction strategies. In [6], the authors compared the effects of dif-

* Corresponding Author

ferent prediction models. In [7], the authors approximate the robustness of the solution using a surrogate assisted model.

In [8], the authors solve dynamic optimization problems by tracking promising region and collecting information about problem space characteristics. In [9], the author proposed an algorithm that incorporates a robustness evaluation component and a dual-mode computing resource allocation component. In [10], the authors used the objective function value of the solution in the current environment and the floating value of the objective function in the adjacent environment to select a better robust solution. In [11], the authors propose a data stream driven multi-form evolutionary robust optimization algorithm. In [12], a new formulation of the performance measures are proposed for finding robust solutions over time.

In [13], the authors proposed a multi-objective ROOT algorithm considering switching cost in order to maximize the number of solutions and minimize the energy consumption of solutions during switching. In [14], the authors used the advantages of TMO and RPOOT under various environmental changes to propose a hybrid dynamic multi-objective evolutionary optimization algorithm. In [15], the authors introduced three methods to solve ROOT by sampling in the search space, and the experiment proved that the method of sampling in the search space was very good. In [16], the authors experimentally compared DE with three random sampling methods on dynamic robust optimization problems and showed that the DE algorithm is suitable for all but this type of problem. In [17], the authors proposed a new multi-objective robust optimization algorithm framework to solve the disassembly sequence planning problem in dynamic environment. In [18], the authors proposed a dynamic constrained robust optimization method in order to improve the effect of carbon fiber pre-oxidation, reduce energy consumption and switching cost. In [19], the authors introducing a dynamic balancing mechanism to adjusts the search direction. In [20], the authors proposed HS-DRPSO to improve the search performance dynamic robust optimization algorithm. In [21], the authors proposes a new method, called fluctuation and robustness based reinforcement learning in particle swarm optimization. In [22], A multi-objective evolutionary algorithm called constrained non-dominated sorting differential evolution based on decision variable classification is developed to search for robust order schedules. In [23], the authors introduce a novel robust evolutionary algorithm named the dual-stage robust evolutionary algorithm (DREA) aimed at discovering robust solutions.

As mentioned earlier, methods that use the predicted fitness of a candidate solution to predict its robustness suffer from extremely large prediction errors; Methods based on promising region robustness estimation can easily make the population fall into local optima. To this end, this paper proposes a new ROOT algorithm named MROOT-AC. The main technical achievements and contributions are as follows:

1) The proposed non-inertia particle swarm optimization algorithm with local search to accelerate the search speed of the population.
2) The proposed a adaptive population control strategy, which can make all the converging populations explore the promising region and avoid the waste of resources.
3) The proposed multi-solution archive management mechanism to avoid frequent switching of solutions by deploying primary solutions and secondary solutions.

2 Related Works

This section provide the concepts for ROOT, population control strategy, and environment change response strategy.

2.1 Robust Optimization Over Time Problem

Dynamic optimization problems (DOPs) [24] can be defined as

$$\max f^{(t)}(\vec{x}) = f(\vec{x}, \vec{\alpha}^{(t)}) . \tag{1}$$

Where $\vec{x}$ is a solution in the search space, $\vec{\alpha}^{(t)}$ denotes the vector of time-dependent control parameters, f is the objective function, $t \in [0, T]$ is the time index. Almost all existing work in the field of DOPs considers the problem where environmental changes occur only at discrete time steps. Assuming a total of T environments, Eq. (1)

this kind of problem can be dispersed into a function sequence composed of multiple static functions with the change of parameter $\vec{\alpha}^{(t)}$:

$$\left\{f\left(\vec{x},\vec{\alpha}^{(k)}\right)\right\}_{k=1}^{T}=\left\{f\left(\vec{x},\vec{\alpha}^{(1)}\right),f\left(\vec{x},\vec{\alpha}^{(2)}\right),\ldots,f\left(\vec{x},\vec{\alpha}^{(T)}\right)\right\}. \tag{2}$$

ROOT problems based on acceptability of the deployed solution in which the deployed solution is kept as long as it remains acceptable.

2.2 Population Control Strategy

Sub-populations may explore the same area, resulting in a waste of resources. And continuous evolution, the population may converge prematurely, so the convergence detection mechanism and rejection mechanism are introduced [25].

Exclusion mechanism can avoid aggregating multiple sub-populations on the same promising region, so as to improve the utilization efficiency of resources.

$$dis_{i,j}=\left\|gbest_i-gbest_j\right\|. \tag{3}$$

where $gbest_i$ and $gbest_j$ denotes the i^{th} and j^{th} sub-population global optimal positions, ||·|| represents the euclidean distance calculation method.

Convergence detection mechanism is used to determine the convergence status of population. The convergence status of pop_i is defined based on its spacial size λ_i which is the Euclidean distance of the farthest pair of individuals and is formally defined as:

$$\lambda_i=\max_{\vec{x}_j,\vec{x}_k\in \mathrm{pop}_i}\left\|\vec{x}_j-\vec{x}_k\right\|. \tag{4}$$

2.3 Environment Change Response Strategy

All existing ROOT methods are based on change reaction, and their performance depends on how quickly and effectively they react to changes in the environment. After environmental changes [9], one of the individuals is located on the best found position from the previous environment $\vec{g}_i^{(t-1)}$, other individuals are randomized around this position with the radius of the estimated shift severity by (5):

$$\vec{x}_{i,j}=\vec{g}_i^{*(t-1)}+(\tilde{s}_i*\vec{r}). \tag{5}$$

Where $\vec{x}_{i,j}$ is the position of the j^{th} individual of pop_i, $\vec{r}$ is a uniformly distributed random number vector in range [−1, 1], $\tilde{s}_i$ is the degree of variation in the coverage promising region of population i, which is calculated by (6):

$$\tilde{s}_i=\frac{1}{t-b_i-1}\times\sum_{k=b_i+1}^{t-1}\left\|gbest_i^{(k),end}-gbest_i^{(k-1),end}\right\|. \tag{6}$$

Where b_i is the time index of the environment that pop_i has converged.

3 Multi-solution Robust Optimization Over Time with Adaptive Population Control

This section proposes a multi-solution robust optimization over time algorithm with adaptive population control (MROOT-AC). First, the algorithm use the non-inertia particle swarm optimization with local search (LNPSO) algorithm as the optimizer. Second, proposed a adaptive population control mechanism (APCM), to ensure that all the convergent populations have explored the promising region and avoiding the waste of computing resources. Finally, due to the intolerance of the system and the high switching cost caused by the frequent switching of the deployed solutions, a multi-solution archiving mechanism (MSAM) was proposed. The main execution steps of MROOT-AC are described in Algorithm 1. And the flow chart of the proposed algorithm is shown in Fig. 1. First, the population is initialized. Then, the solution is sought through the evolutionary optimization algorithm. Finally, the solution is deployed.

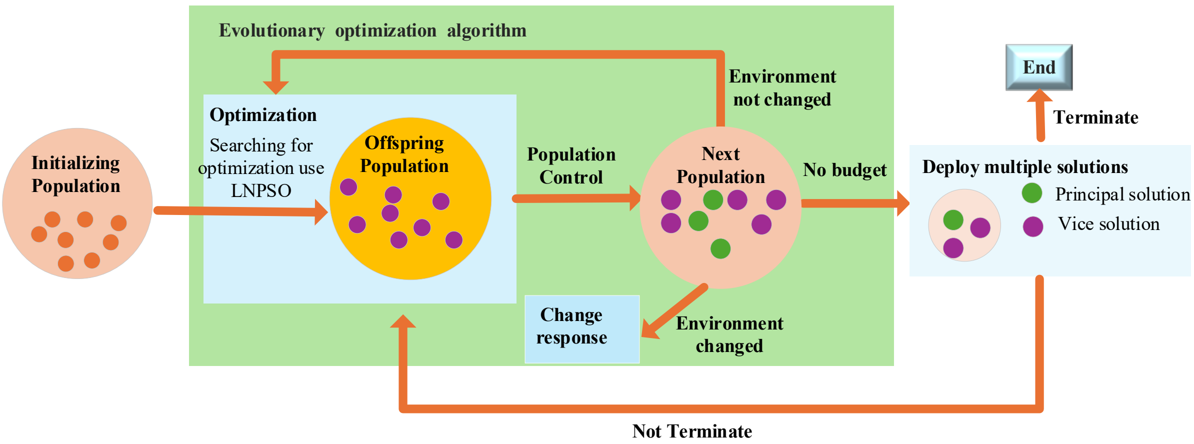


Fig. 1. Flow chart of the Multi-solution robust optimization over time algorithm with adaptive population control

```
Algorithm 1. The main steps of MROOT-AC
Initialize a population;
repeat
 for each population pop_i
    Execute LNPSO;
 end
 for any two populations
   calculate the dis_i,j by Eq.(3);
    if dis_i,j < φ_excl
       Remove the inferior sub-population;
    end
 end
 for each population pop_i
    Update λ_i by Eq.(4);
    Execute APCM;
 end
 if all the populations have converged
    Initialize a new sub-population;
 end
 if environment has changed
     Execute Change Response Strategy;
 end
 if a solution need to be deployed
    Execute MSAM;
 end
until stopping criterion is met;
```

3.1 Non-inertia Particle Swarm Optimization with Local Search

In [26], the authors proposed an Non-Inertia particle swarm optimization algorithm, the formula is given in (7). At the same time, the optimization of each dimension of the particle will have a positive impact on the global optimal solution [27]. Taking advantage of this, a new Non-Inertia particle swarm optimization with local search (LNPSO) is formed by combining the dimension exchange strategy with NPSO algorithm.

This strategy ensures a more thorough exploration of the search space and enhances the ability of the algorithm to converge to the global optimum. The LNPSO specific implementation process is shown in Algorithm 2.

$$\begin{aligned} v_{i,j}(t+1) &= c*(u(t+1)-u(t))+c_1*rand_1(pbest_{i,j}-x_{i,j}(t))+ \\ &\quad c_2*rand_2(gbest_j-x_{i,j}(t)) \\ x_{i,j}(t+1) &= x_{i,j}(t)+v_{i,j}(t+1) \end{aligned} \quad . \tag{7}$$

where $c \in (0, 1)$ is the difference coefficient, $u(t)$ and $u(t-1)$ is the mean of all particle positions of the population at time t and time $t-1$, $v_{i,j}(t)$ and $x_{i,j}(t)$ is the velocity and position of the i^{th} particle in the j^{th} dimension. c_1 and c_2 is the self-learning factor and the global learning factor, $pbest_{i,j}$ is the best position found by particle j of population i and $gbest_i$ is the best position found by the whole swarm i.

Algorithm 2. LNPSO

```
Update particle j according to (7);
If the particle's current position x_j is better than its historical optimal solution pbest_j;
 Update the historical optimal solution pbest_j:pbest_j= x_j;
 If the particle is inferior to the global optimum (gbest_i)
    for each dimension d of gbest_i
       temp_gbest=gbest_j;temp_gbest_d=particle_j,d;
        if Fit(gbest_i)<Fit(temp_gbest)
            gbest_i,d=temp_gbest_d;
         end
     end
 else
    Update the population global optimal solution gbest_i:gbest_i=x_j;
 end
end
```

3.2 Adaptive Population Control Mechanism

Although traditional methods based on the size of population space to judge the convergence situation have the advantages of computational convenience, they have significant defects: over-reliance on metric space measures can easily lead to the algorithm falling into the trap of local optimum, resulting in the convergence of the population does not explore the promising regions. Therefore, this section proposes a adaptive population control mechanism (APCM) to achieve accurate identification and adaptive adjustment of population states.

When the population convergence degree is in the preset threshold interval (r_{cover}, r_{conv}) and the population has not explored promising regions, it indicates that the population shows local aggregation characteristics. At this time, the rand strategy is implemented for part of the individual dimensions to break through the search stagnation by enhancing the local escape ability. When the population convergence degree is smaller than r_{cover} and the population has not explored the promising regions, it is determined that the population has entered a deep convergence state. At this time, randomly initialize the population to effectively balance the global exploration and local development capabilities. The specific implementation process is shown in Algorithm 3.

Algorithm 3. APCM

```
Calculate the diversity λ_i of the population according to (4)
If the population has not explored the promising region
 if r_cover < λ_i < r_conv
  For each dimension of each individual
   if  rand< pc
     Randomize this dimension
   end
  end
 else if λ_i < r_cover
    This population is initialized randomly;
end
end
```

3.3 Multi-solution Archive Management Mechanism

Aiming at the waste of computing resources and the increase of computer cost caused by frequent solution switching in dynamic optimization problems, this paper proposes a multi-solution archive management mechanism (MSAM). In this strategy, a solution set storage mechanism was constructed to respond to the environment. The principal solution was the optimal solution of the current environment, and the vice solution archives (VSA) was used to store the sub-optimal solution set satisfying the quality threshold constraint, this is shown in Fig. 2, the δ in the figure represents the quality threshold. When the environment changes, a three-stage evaluation is performed:

1. Archive maintenance phase: all solutions in VSA are evaluated in the current environment and the solutions that do not satisfy the acceptable threshold constraints are eliminated;
2. Principal solution evaluation phase: the environmental adaptability of the principal solution (PS) is verified, and if the principal solution fitness value is lower than the threshold, the principal solution is removed;
3. Solution selection phase: If the principal solution is empty, the optimal solution replacement is selected from the VSA.

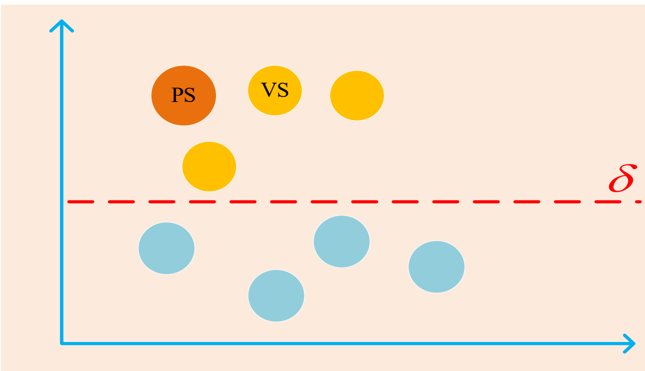


Fig. 2. Diagram of multi-solution archive management mechanism

4 Experimental Design and Result Analysis

This section introduces the benchmark generator, performance evaluation, experimental parameter settings, and provides a detailed analysis of the experimental results. The performance of the MROOT-AC algorithm is compared with four robust optimization algorithms ROOT-CRE, ROOT-LNPSO, ROOT-RB, ROOT-SHV use the GMPB test function. To ensure experimental fairness, all algorithms were run 31 times independently, and all experiments are conducted on matlabR2021a.

4.1 Benchmark Generator

Generalized Moving Peaks Benchmark (GMPB) [28] is a benchmark generator that fits the characteristics of real-world problems. To graphically represent the dynamic behavior of the GMPB generating problem instances, Fig. 3 shows the landscape in the 10th and 11th environments environment. The baseline function of GMPB is:

$$f^{(t)}(\vec{x}) = \max_{i\in\{1,\dots,m\}}\left\{h_i^{(t)} - \sqrt{\mathrm{T}((\vec{x}-\vec{c}_i^{(t)})^\top R_i^{(t)\top}, i)W_i^{(t)}\mathrm{T}(R_i^{(t)}(\vec{x}-\vec{c}_i^{(t)}), i)}\right\}. \quad (8)$$

where $T(y,i):\mathbb{R}^d \mapsto \mathbb{R}^d$ is calculated as:

$$T(y_l,i) = \begin{cases} \exp(\log(y_l) + \tau_i^{(t)}(\sin(\eta_{i,1}^{(t)}\log(y_l)) + \sin(\eta_{i,2}^{(t)}\log(y_l)))) & y_l > 0 \\ 0 & y_l = 0 \\ -\exp(\log(|y_l|) + \tau_i^{(t)}(\sin(\eta_{i,3}^{(t)}\log(|y_l|)) + \sin(\eta_{i,4}^{(t)}\log(|y_l|)))) & y_l < 0 \end{cases}. \quad (9)$$

Where $\vec{x}$ is a solution in d-dimension space, h_i^t is the height of peak i in the t^{th} environment, $\vec{c}_i^{(t)}$ is the position at the top of the i summit in the t^{th} environment, $\eta_{i,l\in\{1,2,3,4\}}^{(t)}$ with $\tau_i^{(t)}$ are irregularity parameters of the i^{th} peak in the t^{th} environment, they make the peaks become asymmetric and present multi-modal characteristics, $R_i^{(t)}$ is the rotation matrix of i^{th} peak in the t^{th} environment, each peak i is rotated using $R_i^{(t)}$, $W_i^{(t)}$ is a $d \times d$ diagonal matrix whose diagonal elements represent the width of the k^{th} peak in different dimensions in the t^{th} environment, it controls the ill-conditioned nature of the peak m is the number of peak, y_l is l^{th} dimension of y.

For each peak, the angle, height, irregularity parameters, width vector and the center of the i^{th} peak change from current environment to the next according to the following update rules:

$$\theta_i^{(t+1)} = \theta_i^{(t)} + \tilde{\theta}_i N(0,1). \quad (10)$$

$$h_i^{(t+1)} = h_i^{(t)} + \tilde{h}_i N(0,1). \quad (11)$$

$$\tau_i^{(t+1)} = \tau_i^{(t)} + \tilde{\tau}_i N(0,1). \quad (12)$$

$$\eta_{i,k}^{(t+1)} = \eta_{i,k}^{(t)} + \tilde{\eta}_i N(0,1),\ k \in \{1,2,3,4\}. \quad (13)$$

$$\omega_{i,l}^{(t+1)} = \omega_{i,l}^{(t)} + \tilde{\omega}_i N(0,1),\ l \in \{1,2,\cdots,d\}. \quad (14)$$

$$c_{i,l}^{(t+1)} = c_{i,l}^{(t)} + N(0, \frac{\tilde{s}_i}{\sqrt{d}}),\ l \in \{1,2,\cdots,d\}. \quad (15)$$

Where $\tilde{\theta}_i$, $\tilde{h}_i$, $\tilde{\omega}_i$, $\tilde{s}_i$, $\tilde{\eta}_i$ and $\tilde{\tau}_i$ are angle, height, width, shift and irregularity parameters in the i^{th} function, respectively.

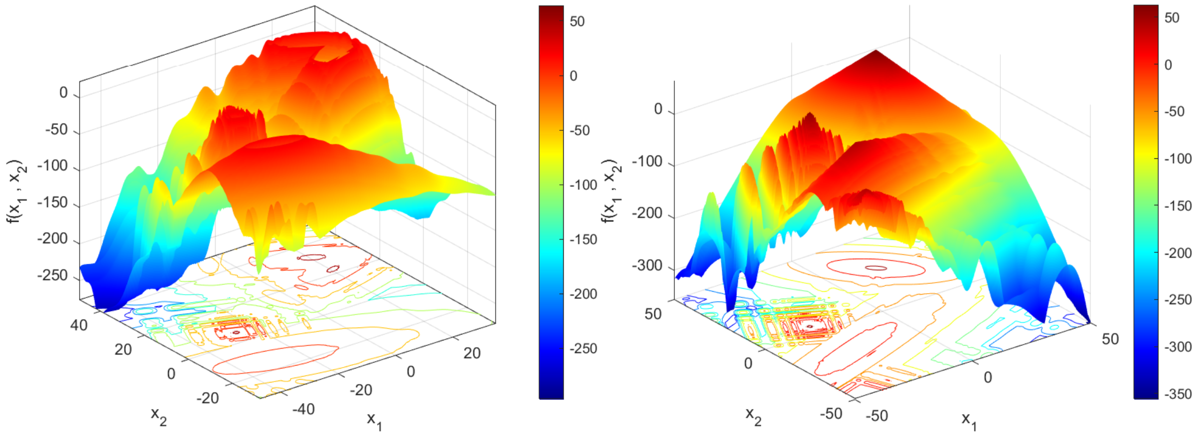


Fig. 3. GMPB fitness value surface in the 10th and 11th environments with five promising regions under 2D decision variables

4.2 Performance Evaluation

In this paper, Survival time (ST) [8] is selected as performance metrics due to it ability to quantify solution robustness.

ST is the maximum the number of successive environments starting from environment t during which the fitness value of the solution remains acceptable. The definition of ST is illustrated in the following equation:

$$ST(\vec{x}) = \begin{cases} 0, & \text{if } f^{(t)}(\vec{x}) < \mu \\ 1 + \max\left\{l \mid \max\left\{\forall i \in \{1,2,\ldots,l\} : f^{"(t+i)}(\vec{x}) \geq \mu\right\}, & otherwise \end{cases} . \quad \textbf{(16)}$$

Where μ is a user defined quality threshold to evaluate the acceptability of $\vec{x}$, $f^{"(t+i)}(\vec{x})$ is the fitness value of the solution $\vec{x}$ in the $t + i$ environment.

4.3 Parameter Settings

The parameter Settings of GMPB and MROOT-AC are shown in Table 1. This paper conducts experiment on problem instances with different number of promising regions (m), quality thresholds (μ), and dimensions (d).

Table 1. Parameter setting of the GMPB and MROOT-AC
(The default parameter values are highlighted when several values are used in experiments.)

Parameter	Symbol	Value	Parameter	Symbol	Value
Quality threshold	μ	**40**, 45, 50	Dimension	d	2, **5**, 10
Promising regions	m	10, **25**, 50, 100	Height	$h_i^{(0)}$	$\mathcal{U}\ [h_{\min}, h_{\max}]$
Shift severity	$\tilde{s}_i$	$\mathcal{U}$ [1, 15]/(sqrt(d))	Width	$w_i^{(0)}$	$\mathcal{U}\ [w_{\min}, w_{\max}]$
Height severity	$\tilde{h}_i$	$\mathcal{U}$ [1, 15]	Height range	$[h_{\min}, h_{\max}]$	[30, 70]
Width severity	$\tilde{\omega}_i$	$\mathcal{U}$ [0.1, 1.5]	Width range	$[w_{\min}, w_{\max}]$	[1, 12]
Center position	$c_i^{(0)}$	$\mathcal{U}\ [Lb, \text{Ub}]^d$	Rotation angle	$[\vartheta_{\min}, \vartheta_{\max}]$	$[-\pi, \pi]$
Change frequency	f	2500	Angle	$\vartheta_i^{(0)}$	U $[\vartheta_{\min}, \vartheta_{\max}]$

Sub-population size	p	5	Angle severity	$\tilde{\theta}$	$\pi/9$
Environment number	T	100	Parameter η	$\eta_i^{(0)}$	$\mathcal{U}\,[\eta_{\min}, \eta_{\max}]$
Rotation matrix	$R_I^{(0)}$	$GS(\mathcal{N}(0,1)^{d\times d})$	Parameter τ	$\tau_i^{(0)}$	$\mathcal{U}\,[\tau_{\min}, \tau_{\max}]$
Computational budget	δ	$f/2$	η range	$[\eta_{\min}, \eta_{\max}]$	[10, 25]
η severity	$\tilde{\eta}$	2	τ severity	$\tilde{\tau}$	0.05
τ range	$[\tau_{\min}, \tau_{\max}]$	[0, 0.4]	Search range	$[Lb, \mathrm{Ub}]^d$	$[-50, 50]^d$
Convergence threshold	r_{conv}	$0.5(Ub-Lb)/\sqrt[d]{p}$	Exclusion threshold	φ_{excl}	$0.5(Ub-Lb)/\sqrt[d]{p}$
Summit threshold	r_{conver}	5	Sleep threshold	$r_{\min}$	0.75
Difference coefficient	c	0.1	Learning factor	c_1 and c_2	2.05
Stop	$evals$	$f*T$	Random threshold	pc	0.5

4.4 Algorithms

In the experiments, this paper use the multi-population framework for all multi-population based methods [9] and the proposed LNPSO optimizer. The performance of the MROOT-AC algorithm is compared with four ROOT algorithm, ROOT-SHV, ROOT-CRE, ROOT-RB and ROOT-LNPSO. ROOT-SHV [8] uses the estimated shift and height severity values in the collected promising region information to select the next solution to deploy. ROOT-CRE [9] is selects the next solution by collecting the robustness of promising regions. ROOT-RB [29] is to select the optimal solution deployment among reliable promising regions. ROOT-LNPSO [26] is to deploy the global optimal solution found by LNPSO in the search space after environment change.

4.5 Experimental Results and Analysis

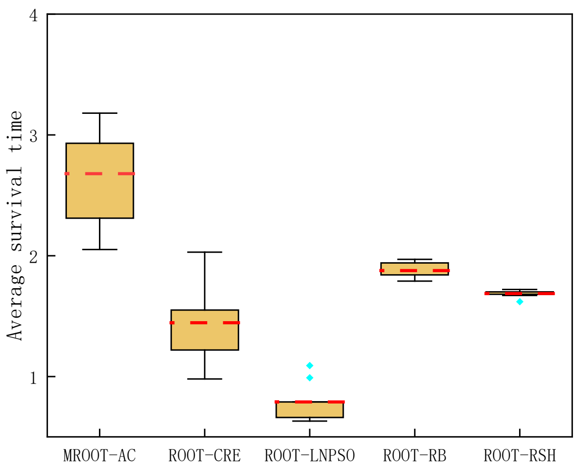
To comprehensively validate the competitiveness of the algorithms MROOT-RC proposed in this paper, their performance is compared with that of the benchmark algorithms ROOT-SHV, ROOT-CRE, ROOT-RB and ROOT-LNPSO in solving problem instances generated by GMPB with different dimensions d, numbers of promising regions m, and quality thresholds values μ. Table 2 present the results of each algorithm, with the algorithms exhibiting optimal performance highlighted by horizontal lines beneath the data.

The experimental results in Table 2 clearly show that the MROOT-AC algorithm significantly outperforms the four comparison algorithms in all test scenarios. Fig. 4 shows the box plots of the average survival time of the five algorithms under different promising regions, and it can be seen from the figure that the average survival time of MROOT-AC is due to other algorithms, which further intuitively verifies the superiority of the proposed algorithm. In addition, with the increase of the number of promising regions, the average survival time of all algorithms shows an increasing trend, this is due to the increase in the number of robust solution regions with better quality in landscapes with a large number of promising regions.

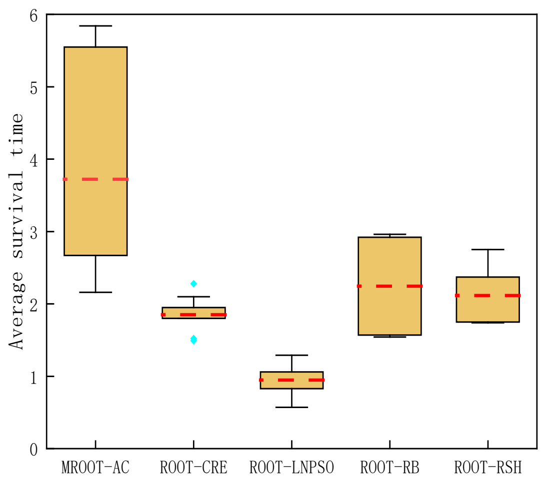
Fig. 5(a) shows the change trend of the average survival time of the five algorithms under different dimensions. It can be clearly seen from the figure that with the increase of the dimension of the problem space, the average survival time of the algorithms gradually decreases, and when the dimension is 10, the average survival time of each algorithm has little difference. This is because as the dimension increases, the problem instances generated by GMPB are more complex, and it is more difficult to search for solutions in the high-dimensional space. Fig. 5(b) further shows the average survival time variation of the five algorithms under different acceptable thresholds. The results show that the average survival time of the solution decreases significantly with the increase of the quality threshold, this is due to the fact that increasing μ, the region containing robust solutions shrinks, the maximum possible survival time decreases, and finding robust solutions becomes more challenging.

Table 2. Mean survival time (and standard error) of the five algorithms on GMPB with different dimension (d), promising region (m), quality threshold (μ) values (The highlighted entries are significantly better.)

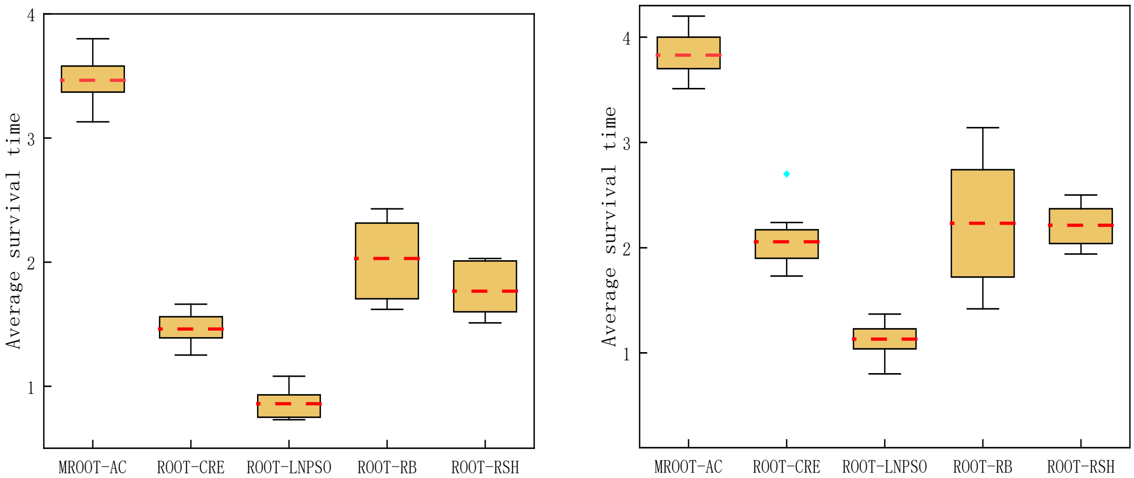
d	μ	m	Algorithms				
			MROOT-AC	ROOT-CRE	ROOT-LNPSO	ROOT-RB	ROOT-SHV
2	40	10	**4.52(0.49)**	4.22(0.49)	4.10(0.80)	3.75(0.56)	3.69(0.72)
		25	**6.12(0.99)**	4.59(0.05)	4.40(0.54)	4.83(0.68)	4.51(0.45)
		50	**7.98(1.02)**	4.33(0.33)	5.14(0.46)	5.39(0.55)	4.53(0.39)
		100	**8.20(0.83)**	5.37(0.32)	6.44(0.90)	6.06(0.15)	7.52(1.05)
	45	10	**4.11(0.80)**	3.51(0.34)	3.14(0.43)	3.03(0.35)	2.54(0.30)
		25	**4.39(0.58)**	3.61(0.57)	3.36(0.44)	3.35(0.56)	3.27(0.47)
		50	**5.16(0.66)**	3.92(0.76)	3.44(0.22)	3.19(0.19)	3.62(0.11)
		100	**6.30(0.80)**	4.70(0.66)	4.79(0.88)	4.38(1.03)	5.53(0.70)
	50	10	**2.78(0.41)**	2.36(0.25)	2.32(0.25)	2.01(0.15)	1.51(0.22)
		25	**3.12(0.24)**	2.55(0.19)	2.52(0.13)	2.24(0.25)	2.17(0.29)
		50	**3.69(0.22)**	2.80(0.26)	2.66(0.22)	2.31(0.08)	2.87(0.33)
		100	**4.60(0.49)**	3.30(0.17)	3.18(0.34)	3.99(0.16)	3.50(0.16)
5	40	10	**2.67(0.09)**	1.87(0.09)	1.68(0.03)	1.43(0.11)	0.79(0.19)
		25	**3.44(0.40)**	2.03(0.10)	1.76(0.18)	1.47(0.10)	0.85(0.15)
		50	**3.72(0.53)**	2.24(0.22)	2.11(0.03)	1.89(0.27)	0.95(0.34)
		100	**3.76(0.23)**	2.23(0.23)	2.21(0.08)	2.08(0.36)	1.14(0.14)
	45	10	**1.82(0.16)**	1.59(0.11)	1.45(0.02)	0.89(0.15)	0.75(0.23)
		25	**2.10(0.09)**	1.83(0.11)	1.70(0.04)	1.61(0.39)	0.82(0.17)
		50	**2.61(0.46)**	1.90(0.17)	1.87(0.02)	1.66(0.27)	0.92(0.13)
		100	**2.82(0.50)**	1.96(0.18)	1.92(0.10)	1.66(0.28)	1.05(0.02)
	50	10	**1.26(0.04)**	1.21(0.06)	1.22(0.10)	0.39(0.03)	0.34(0.03)
		25	**1.82(0.22)**	1.29(0.06)	1.68(0.12)	0.89(0.12)	0.39(0.12)
		50	**2.05(0.25)**	1.43(0.11)	1.73(0.14)	0.91(0.15)	0.31(0.09)
		100	**2.21(0.25)**	1.49(0.13)	1.77(0.12)	1.34(0.42)	1.00(0.02)
10	40	10	**1.49(0.07)**	1.46(0.39)	1.45(0.19)	0.05(0.02)	0.54(0.11)
		25	**1.54(0.06)**	1.51(0.14)	1.47(0.14)	0.13(0.04)	0.57(0.14)
		50	**1.71(0.13)**	1.70(0.23)	1.56(0.17)	0.16(0.09)	0.71(0.17)
		100	**1.88(0.20)**	1.74(0.46)	1.61(0.22)	0.26(0.02)	0.86(0.22)
	45	10	**1.39(0.27)**	1.14(0.08)	0.98(0.04)	0.02(0.04)	0.54(0.16)
		25	**1.41(0.49)**	1.40(0.16)	0.96(0.10)	0.05(0.01)	0.55(0.18)
		50	**1.75(0.40)**	1.27(0.17)	1.25(0.21)	0.06(0.04)	0.55(0.41)
		100	**1.75(0.40)**	1.56(0.13)	1.08(0.07)	0.09(0.03)	0.26(0.01)
	50	10	**0.69(0.03)**	0.63(0.16)	0.62(0.16)	0.02(0.01)	0.26(0.03)
		25	**0.81(0.14)**	0.67(0.11)	0.58(0.06)	0.04(0.01)	0.38(0.06)
		50	**0.78(0.11)**	0.60(0.06)	0.61(0.16)	0.04(0.01)	0.51(0.16)
		100	**0.67(0.04)**	0.64(0.11)	0.62(0.22)	0.05(0.00)	0.23(0.01)



(a) Robustness in GMPB with 10 promising regions



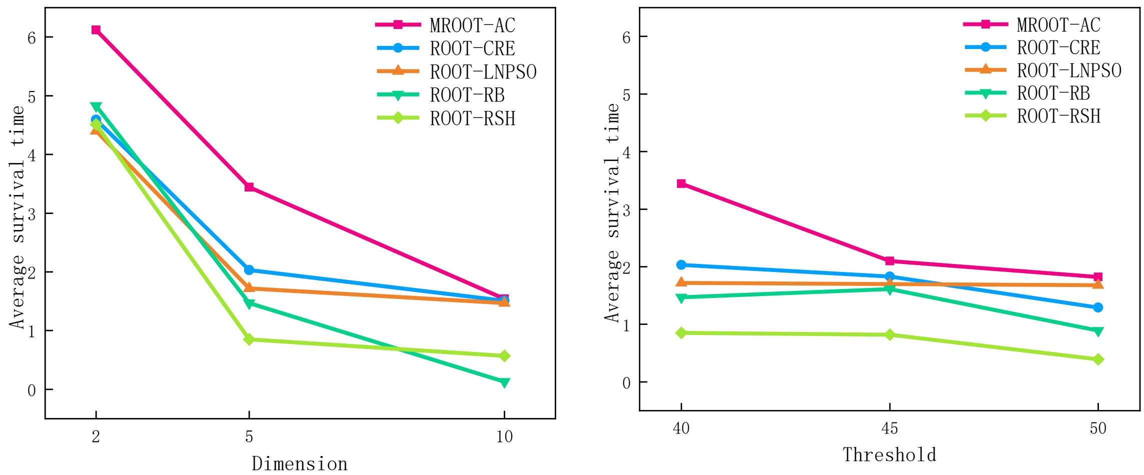
(b) Robustness in GMPB with 25 promising regions



(c) Robustness in GMPB with 50 promising regions

(d) Robustness in GMPB with 100 promising regions

Fig. 4. Average survival time by running the five algorithms on the GMPB test function in problem instances generated by GMPB with different numbers of promising regions and the default parameter settings from Table 2 for the rest of the parameters



(a) Robustness in GMPB with 25 promising regions and 40 threshold

(b) Robustness in GMPB with 25 promising regions and 5 dimension

Fig. 5. Trend of the average survival time of the algorithm as a function of the dimension and quality threshold

4.6 Ablation Experiment

MROOT-AC includes two main parts: an adaptive population control mechanism and a multi-solution archive management mechanism. To further illustrate the role of the multi-solution deployment in MROOT-AC, ablation experiments are conducted in this subsection.

MROOT-AC and its variant, ROOT-AC with only a single solution deployed is experimented on the GMPB problem with different promising regions, dimensions and quality thresholds, and the results are shown in Table 3. The average survival time of MROOT-AC is better than that of ROOT-AC in all cases. This is because MROOT-AC stores all the solutions that meet the quality threshold in the archive when deploying solution, and deploys them to increase the diversity of solutions. It increases the possibility that the solution deployed in the current environment can still satisfy the next environment.

Table 3. Mean survival time (and standard error) of MROOT-AC and its variant ROOT-AC on GMPB with different dimension (d), promising region (m), quality threshold (μ) values (The highlighted entries are significantly better.)

m	μ	d	Algorithms		m	μ	d	Algorithms	
			MROOT-AC	ROOT-AC				MROOT-AC	ROOT-AC
10	40	2	**4.52(0.49)**	3.63(1.46)	25	40	2	**6.12(0.99)**	3.80(0.91)
		5	**2.67(0.09)**	2.26(0.11)			5	**3.44(0.40)**	2.67(0.25)
		10	**1.49(0.07)**	1.17(0.23)			10	**1.54(0.06)**	1.39(0.21)
	45	2	**4.11(0.80)**	3.22(0.41)		45	2	**4.39(0.58)**	3.27(0.41)
		5	**1.82(0.16)**	1.63(0.09)			5	**2.10(0.09)**	1.81(0.21)
		10	**1.39(0.27)**	0.94(0.17)			10	**1.41(0.49)**	0.92(0.14)
	50	2	**2.78(0.41)**	2.16(0.20)		50	2	**3.12(0.24)**	2.41(0.18)
		5	**1.26(0.04)**	1.14(0.08)			5	**1.82(0.22)**	1.51(0.11)
		10	**0.69(0.03)**	0.60(0.15)			10	**0.81(0.14)**	0.67(0.09)
50	40	2	**7.98(1.02)**	3.90(0.36)	100	40	2	**8.20(0.83)**	7.33(0.31)
		5	**3.72(0.53)**	2.76(0.35)			5	**3.76(0.23)**	3.05(0.48)
		10	**1.71(0.13)**	1.44(0.14)			10	**1.88(0.20)**	1.54(0.34)
	45	2	**5.16(0.66)**	3.56(0.04)		45	2	**6.30(0.80)**	5.72(1.40)
		5	**2.61(0.46)**	2.26(0.45)			5	**2.82(0.50)**	2.53(0.22)
		10	**1.75(0.40)**	1.09(0.28)			10	**1.75(0.40)**	1.24(0.11)
	50	2	**3.69(0.22)**	2.58(0.08)		50	2	**4.60(0.49)**	3.58(0.10)
		5	**2.05(0.25)**	1.71(0.21)			5	**2.21(0.25)**	2.05(0.31)
		10	**0.78(0.11)**	0.68(0.07)			10	**0.97(0.04)**	0.86(0.08)

5 Conclusion

This paper proposes the Multi-solution Robust Optimization Over Time Algorithm with Adaptive Population Change (MROOT-AC). Firstly, the LNPSO is used as the optimizer to speed up the process of finding solutions. Secondly, in order to avoid the occurrence of premature population, APCM is introduced to increase the diversity of the population. Finally, the MSAM is used to deploy multiple solutions after the environment changes to avoid the high switching cost caused by the frequent switching of the deployed solutions.

MROOT-AC has significant advantages in solving dynamic optimization problems, but its performance is not ideal when solving high-dimensional problems. Improving the performance of MROOT-AC on high-dimensional problems will be future work.

Acknowledgement

The work of LanLan Kang was supported in part by the Natural Science Foundation of China under Grant No. 62166019, and the Natural Science Foundation of Jiangxi Province under Grant No. 20252BAC240195, and the Science and Technology Research Project in Department of Education of Jiangxi Province under Grant GJJ2403701. The work of Wenliang Cao was supported the Special for key fields of colleges and universities in Guangdong Province under Grant No. 2025ZDZX1084.

References

[1] K.J. Li, S. Elsayed, R. Sarker, D. Essam, Multiple landscape measure-based approach for dynamic optimization problems, Swarm and Evolutionary Computation 88(2024) 101578.
https://doi.org/10.1016/j.swevo.2024.101578

[2] J. Zhang, G. Zhao, W.J. Song, Z.Y. Xu, Black-Box Dehazing Network via Joint Loss Function, IEICE Transactions on Fundamentals E108-A(7)(2025) 964-972.
https://doi.org/10.1587/transfun.2024EAP1104

[3] K.K.H. Ng, C.K.M. Lee, S.Z. Zhang, K.L. Keung, The impact of heterogeneous arrival and departure rates of flights on runway configuration optimization, Transportation Letters 14(3)(2022) 215-226.

https://doi.org/10.1080/19427867.2020.1852496
[4] X. Yu, Y.C. Jin, K. Tang, X. Yao, Robust Optimization over Time — A New Perspective on Dynamic Optimization Problems, in: Proc. 2010 IEEE Congress on Evolutionary Computation, 2010. https://doi.org/10.1109/CEC.2010.5586024
[5] Y.N. Guo, H. Yang, M.R. Chen, J. Cheng, D.W. Gong, Ensemble prediction-based dynamic robust multi-objective optimization methods, Swarm and evolutionary computation 48(2019) 156-171. https://doi.org/10.1016/j.swevo.2019.03.015
[6] M. Fox, S.X. Yang, F. Caraffini, An Experimental Study of Prediction Methods in Robust Optimization over Time, in: Proc. 2020 IEEE Congress on Evolutionary Computation, 2020. https://doi.org/10.1109/CEC48606.2020.9185910
[7] P. Novoa-Hernandez, C.C. Cruz, D.A. Pelta, A Surrogate Assisted Approach for Fitness Computation in Robust Optimization over Time, in: Proc. 2024 Advances in Artificial Intelligence, 2024. https://doi.org/10.1007/978-3-031-62799-6_11
[8] D. Yazdani, T.T. Nguyen, J. Branke, Robust Optimization over Time by Learning Problem Space Characteristics, IEEE Transactions on Evolutionary Computation 23(1)(2019) 143-155. https://doi.org/10.1109/TEVC.2018.2843566
[9] D. Yazdani, D. Yazdani, J. Branke, M.N. Omidvar, A.H. Gandomi, X. Yao, Robust Optimization over Time by Estimating Robustness of Promising Regions, IEEE Transactions on Evolutionary Computation 27(3)(2023) 657-670. https://doi.org/10.1109/TEVC.2022.3180590
[10] E.C. Li, F.K. Zhao, Dynamic Robust Optimization Algorithm Based on Problem Feature Change Guidance, Computer Engineering and Applications 60(6)(2024) 130-146. https://doi.org/10.3778/j.issn.1002-8331.2306-0033
[11] Z.N. Liu, H.D. Wang, J.L. Ding, C. Yang, Y. Jin, Data Stream driven evolutionary algorithm for cost sensitive robust optimization over time, Swarm and Evolutionary Computation 94(2025) 101880. https://doi.org/10.1016/j.swevo.2025.101880
[12] Y.J. Huang, Y.C. Jin, Y.S. Ding, New performance indicators for robust optimization over time, in: Proc. 2015 IEEE Congress on Evolutionary Computation (CEC), 2015. https://doi.org/10.1109/CEC.2015.7257049
[13] Y.J. Huang, Y.C. Jin, K.G. Hao, Decision-making and multi-objectivization for cost sensitive robust optimization over time, Knowledge-Based Systems 199(2020) 105857. https://doi.org/10.1016/j.knosys.2020.105857
[14] M.R. Chen, Y.N. Guo, Y.C. Jin, S.X. Yang, D.W. Gong, Z.K. Yu, An Environment-Driven Hybrid Evolutionary Algorithm for Dynamic Multi-Objective Optimization Problems, Complex & Intelligent System 9(1)(2023) 659-675. https://doi.org/10.1007/s40747-022-00824-4
[15] L. Adam, X. Yao, A Simple Yet Effective Approach to Robust Optimization over Time, in: Proc. 2019 IEEE Symposium Series on Computational Intelligence (SSCI), 2019. https://doi.org/10.1109/SSCI44817.2019.9002697
[16] J-Y. Guzman-Gaspar, E. Mezura-Montes, S. Dominguez-Isidro, Differential Evolution in Robust Optimization over Time Using a Survival Time Approach, Mathematical and Computational Applications 25(4)(2020) 72. https://doi.org/10.3390/mca25040072
[17] X. Zhang, Y.L. Fang, Q. Liu, D. Yazdani, Multi-objective Robust Optimization over Time for Dynamic Disassembly Sequence Planning, International Journal of Precision Engineering and Manufacturing 25(1)(2024) 111-130. https://doi.org/10.1007/s12541-023-00900-w
[18] Y.L. Fang, Z.H. Zhao, L. Jin, Dynamic Constrained Robust Optimization over Time for Operational Indices of Preoxidation Process, in: Proc. 2024 18th International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), 2024. https://doi.org/10.1007/978-981-97-2272-3_16
[19] Z.N. Liu, H.D. Wang, Balancing objective and switch cost for robust optimization over time, Swarm and Evolutionary Computation 97(2025) 102022. https://doi.org/10.1016/j.swevo.2025.102022
[20] J. Zeng, X.Y. Yu, G.Y. Yang, H.T. Gui, Dynamic Robust Particle Swarm Optimization Algorithm Based on Hybrid Strategy, International Journal of Swarm Intelligence Research 14(1)(2023) 1-14. https://doi.org/10.4018/IJSIR.325006
[21] C.C. Lee, Y.R. Chen, C.K. Ting, Robust Optimization Over Time by Integrating Reinforcement Learning and Particle Swarm Optimization, in: Proc. 2025 IEEE Congress on Evolutionary Computation (CEC), 2025. https://doi.org/10.1109/CEC65147.2025.11043102
[22] W. Du, W.M. Zhong, Y. Tang, W.L. Du, Y.C. Jin, High-dimensional robust multi-objective optimization for order scheduling: a decision variable classification approach, IEEE Transactions on Industrial Informatics 15(1)(2019) 293-304. https://doi.org/10.1109/TII.2018.2836189
[23] W. Du, W.X. Fang, C. Liang, Y. Tang, Y.C. Jin, A Novel Dual-Stage Evolutionary Algorithm for Finding Robust Solutions, IEEE Transactions on Emerging Topics in Computational Intelligence 8(5)(2024) 3589-3602. https://doi.org/10.1109/TETCI.2024.3369710

[24] H.J. Li, G. Zhang, X.Y. Sun, J. Deng, Y. Li, Dynamic univariate sampling algorithm for dynamic optimization problems, Applied Soft Computing 180(2025) 113324.
https://doi.org/10.1016/j.asoc.2025.113324
[25] D. Yazdani, R. Cheng, D. Yazdani, J. Branke, Y.C. Jin, X. Yao, A Survey of Evolutionary Continuous Dynamic Optimization Over Two Decades—Part A, IEEE Transactions on Evolutionary Computation 25(4)(2021) 609-629.
https://doi.org/10.1109/TEVC.2021.3060014
[26] L.L. Kang, N.W. Liu, X.R. Lu, W.L. Cao, Y. Peng, A back-diffusion median integrated evolutionary optimization algorithm, Information Sciences 610(2022) 144-155.
https://doi.org/10.1016/j.ins.2022.07.168
[27] C.H. Li, S.X. Yang, A General Framework of Multipopulation Methods With Clustering in Undetectable Dynamic Environments, IEEE transactions on evolutionary computation 16(4)(2012) 556-577.
https://doi.org/10.1109/TEVC.2011.2169966
[28] D. Yazdani, M.N. Omidvar, R. Cheng, J. Branke, T.T. Nguyen, X. Yao, Benchmarking Continuous Dynamic Optimization: Survey and Generalized Test Suite, IEEE Transactions on Cybernetics 52(5)(2022) 3380-3393.
https://doi.org/10.1109/TCYB.2020.3011828
[29] D. Yazdani, T.T. Nguyen, J. Branke, J. Wang, A new multi-swarm particle swarm optimization for robust optimization over time, in: Proc. 2017 20th European Conference on the Applications of Evolutionary Computation (EvoApplications), 2017.
https://doi.org/10.1007/978-3-319-55792-2_7