Neural Computing and Applications (2024) 36:6721–6740
https://doi.org/10.1007/s00521-024-09424-4

ORIGINAL ARTICLE

SRIME: a strengthened RIME with Latin hypercube sampling and embedded distance-based selection for engineering optimization problems

Rui Zhong[1] · Jun Yu[2] · Chao Zhang[3] · Masaharu Munetomo[4]

Received: 10 July 2023 / Accepted: 15 January 2024 / Published online: 12 February 2024

Abstract

This paper proposes a strengthened RIME algorithm to tackle continuous optimization problems. RIME is a newly proposed physical-based evolutionary algorithm (EA) inspired by the soft and hard rime growth process of rime-ice, which has a powerful exploitation ability. But in complex optimization problems, RIME will easily trap into local optima and the optimization will become stagnation. Noticing this issue, we introduce three techniques to the original RIME: (1) Latin hypercube sampling replaces the random generator as the initialization strategy, (2) modified hard rime search strategy, and (3) embedded distance-based selection mechanism. We evaluate our proposed SRIME in 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions and eight real-world engineering optimization problems with nine state-of-the-art EAs. Experimental and statistical results show that the introduction of three techniques can significantly accelerate the optimization of the RIME algorithm, and SRIME is a competitive optimization technique in real-world applications. Ablation experiments are also provided to analyze our proposed three techniques independently, and the embedded distance-based selection contributes most to the improvement of SRIME. The source code of SRIME can be found in https://github.com/RuiZhong961230/SRIME.

Keywords Strengthened RIME algorithm (SRIME) · Latin hypercube sampling · Modified hard rime exploitation · Embedded distance-based selection · Engineering optimization

✉ Rui Zhong
rui.zhong.u5@elms.hokudai.ac.jp

Jun Yu
yujun@ie.niigata-u.ac.jp

Chao Zhang
zhang@u-fukui.ac.jp

Masaharu Munetomo
munetomo@iic.hokudai.ac.jp

1 Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Japan

2 Institute of Science and Technology, Niigata University, Niigata, Japan

3 Department of Engineering, University of Fukui, Fukui, Japan

4 Information Initiative Center, Hokkaido University, Sapporo, Japan

1 Introduction

As the evolutionary computation (EC) community develops rapidly, hundreds and thousands of new meta-heuristic algorithms (MAs) have been proposed to deal with various optimization problems such as constrained problems [1], multi-objective problems [2], high-dimensional or large-scale problems [3, 4], combinatorial problems [5], real-world problems [6]. Thanks to the superior characteristics of MAs in solving complex optimization problems, and such as gradient-free, simple structure, and non-convex handle-able [7], MAs have become one of the most popular optimization techniques. Here, we list some representative MAs and classify them into two categories: the evolutionary algorithm (EA) and the swarm intelligence (SI), which is depicted in Fig. 1.

The first category is the EA, which is inspired by Darwin's theory of evolution and simulates the rule of competition and survival in nature. Genetic algorithm (GA) [8],

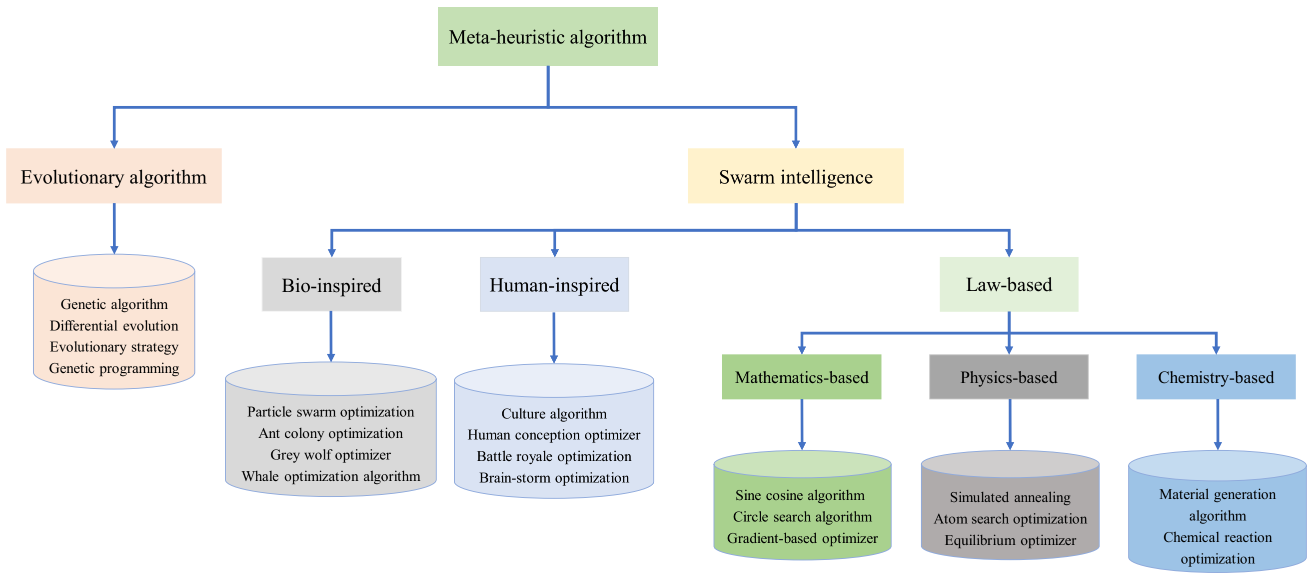


Fig. 1 Representative MAs and their classification

as one of the most classic evolutionary algorithms (EAs), imitates the breeding behaviors of organisms in real-world environments by crossover and mutation operators, and the selection operator reflects the process of natural selection where the fittest individuals are inclined to survival. Besides, the EA family also contains other approaches such as differential evolution (DE) [9], genetic programming (GP) [10], and evolutionary strategy (ES) [11], which have a similar foundation to GA.

Another category is swarm intelligence, which can be divided into three classes: bio-inspired, human-inspired, and law-based. We can further separate the law-based category into mathematics-based, physics-based, and chemistry-based. Among them, bio-inspired algorithms often simulate the social behaviors of organisms to realize optimization. For example, particle swarm optimization (PSO) [12] simulates the social behaviors of a bird flock or fish school, ant colony optimization (ACO) [13] is inspired by the behaviors of ants when seeking a path between their colony and a source of food, gray wolf optimizer (GWO) [14] mimics the leadership hierarchy and hunting mechanism of gray wolves in nature, and whale optimization algorithm, and whale optimization algorithm (WOA) [15] simulates the bubble-net hunting behavior of humpback whales. Human-inspired algorithms are inspired by human activities and typically contain the culture algorithm (CA) [16], the human conception optimizer (HCO) [17], the battle royale optimization (BRO) [18], and the brain-storm optimization (BSO) [19]. Another component is mathematics/physics/chemistry-law-based MAs, which are designed by the specific theory or phenomenon in the corresponding discipline and are typified by the sine cosine algorithm (SCA) [20], the simulated annealing (SA) [21], and the chemical reaction optimization (CRO) [22].

With the explosive development of EAs, some scholars also expressed their concerns to the EC community. In the meantime, the No Free Lunch Theorem (NFLT) [23] logically proved that there is no EA best suited for solving all optimization problems. In other words, a particular EA may outperform on a set of problems while showing poor performance on a different set of problems. Obviously, the existence of the NFLT makes this field of study highly meaningful.

RIME algorithm [24], as a novel physics-based optimization technique, simulates the motion of soft rime and hard rime particles in the rime ice and formulates these phenomena with unique mathematical models to realize the optimization, and we will introduce the RIME in Sect. 2 in detail. Meanwhile, the simplicity, efficiency, scalability, and applicability of RIME have attracted widespread attention from scholars. As one of the latest EAs, we also notice some shortages of RIME. For example, the unbalanced exploitation and exploration abilities limit the performance of RIME in complex optimization problems, and the enhanced greedy selection mechanism survives high-quality solutions while the diversity of the population is lost. These unsettled issues challenge the scalability of RIME in multi-modal problems.

The objective of this paper is to introduce some efficient strategies to improve the performance of RIME in complex optimization problems and propose a strengthened RIME (SRIME). SRIME inherits the main skeleton of RIME and modifies the strategy in three aspects: (1). initialization, (2) hard rime exploitation, and (3) selection mechanism. In numerical experiments, nine state-of-the-art EAs are employed as competitor algorithms while CEC2020 benchmark functions and eight engineering problems are adopted as standard evaluation environments. Besides, we implement ablation experiments to investigate the performance of each strategy that we proposed separately. More specifically, the main contribution of this paper is as follows:

(1) We propose an SRIME to improve the overall performance of the original RIME. In the swarm initialization stage, we replace the random seed generator with the Latin hypercube sampling, and the benefit of this method can generate more uniform solutions in the high-dimensional search space. Given the shortages that we mentioned before, we add uncertainty and randomness to the exploitation phase of RIME and modify the hard rime search strategy.

(2) The enhanced greedy selection mechanism in the original RIME can significantly accelerate the convergence speed of optimization, especially in unimodal problems, but on the contrary, this selection strategy is too greedy to make the algorithm trap into local optima easily in complex optimization problems, and thus, we introduce an embedded distance-based selection mechanism to RIME so that even the deteriorating solution has an opportunity to be accepted, which can enhance the diversity of the population and endow the capacity to RIME for escaping local optima.

(3) We evaluate our proposed SRIME on the standard CEC2020 benchmark functions and eight popular engineering optimization problems with nine powerful optimization techniques, the experimental and statistical analysis proves the efficiency and applicability of SRIME sufficiently.

(4) Comprehensive ablation experiments are provided in our experimental study to investigate the performance of our proposed three improvements independently.

The remainder of this paper is organized as follows: Sect. 2 introduced the RIME. Section 3 provides a detailed introduction to our proposal: SRIME. Section 4 focuses on numerical experiments and statistical analysis. Section 5 discusses the performance of SRIME and further lists some open topics for future research. Finally, Sect. 6 concludes this paper.

2 RIME algorithm

As same as most population-based EAs, population or swarm initialization is the first step of RIME. Similarly, RIME randomly generates the initial population matrix with uniform distribution. Equation (1) describes the structure of rime population R and the generation of each rime particle x_{ij}.

$$R = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ x_{31} & x_{32} & \cdots & x_{3m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{bmatrix} \quad (1)$$
$$x_{ij} = r_1 \cdot (UB_j - LB_j) + LB_j$$

LB_j and UB_j are the lower and the upper bound of the jth dimension and r_1 is a random number in (0, 1). x_{ij} represents the rime particle of the jth dimension of the ith individual, where $i \in [1, n]$ and $j \in [1, m]$.

Figure 2[1] shows two different forms of rime ice: soft rime ice and hard rime ice, which is the inspiration of the RIME.

In the breeze environment, soft rime ice grows along the surface of branches with high randomness, which is described as an exploration operator in RIME that enables the algorithm to cover the search space as much as possible in the early stage of optimization, and Eq. (2) formulates the growth of the soft rime ice.

$$R_{ij}^{t+1} = R_{best,j} + r_2 \cdot \cos(\theta) \cdot \beta \cdot (h \cdot (UB_j - LB_j) + LB_j), \; if \; r_3 < E \quad (2)$$

$R_{best,j}$ is the jth rime particle of the best rime individual. r_2 is a random number in (− 1, 1) to control the direction of particle movement cooperating with the $\cos(\theta)$, β is the self-adaptive environmental factor, h is a random number in (0, 1) represents the degree of adhesion, r_3 is also a random number in (0, 1), and E $(0 < E \leq 1)$ is the coefficient to affect the condensation probability of an agent and increases as the optimization promotes. Equation (3) shows the computation of parameter θ, β, and E.

$$\theta = \pi \cdot \frac{t}{10 \cdot t_{max}}$$
$$\beta = 1 - \left[\frac{w \cdot t}{t_{max}} \right] / w \quad (3)$$
$$E = \sqrt{t / t_{max}}$$

[1] Pictures are downloaded from https://pixabay.com/ as copyright-free images. (a) https://pixabay.com/photos/barbed-wire-frost-frozen-cold-ice-1938842/. (c) https://pixabay.com/photos/thuja-ice-winter-cold-frozen-6015613/.

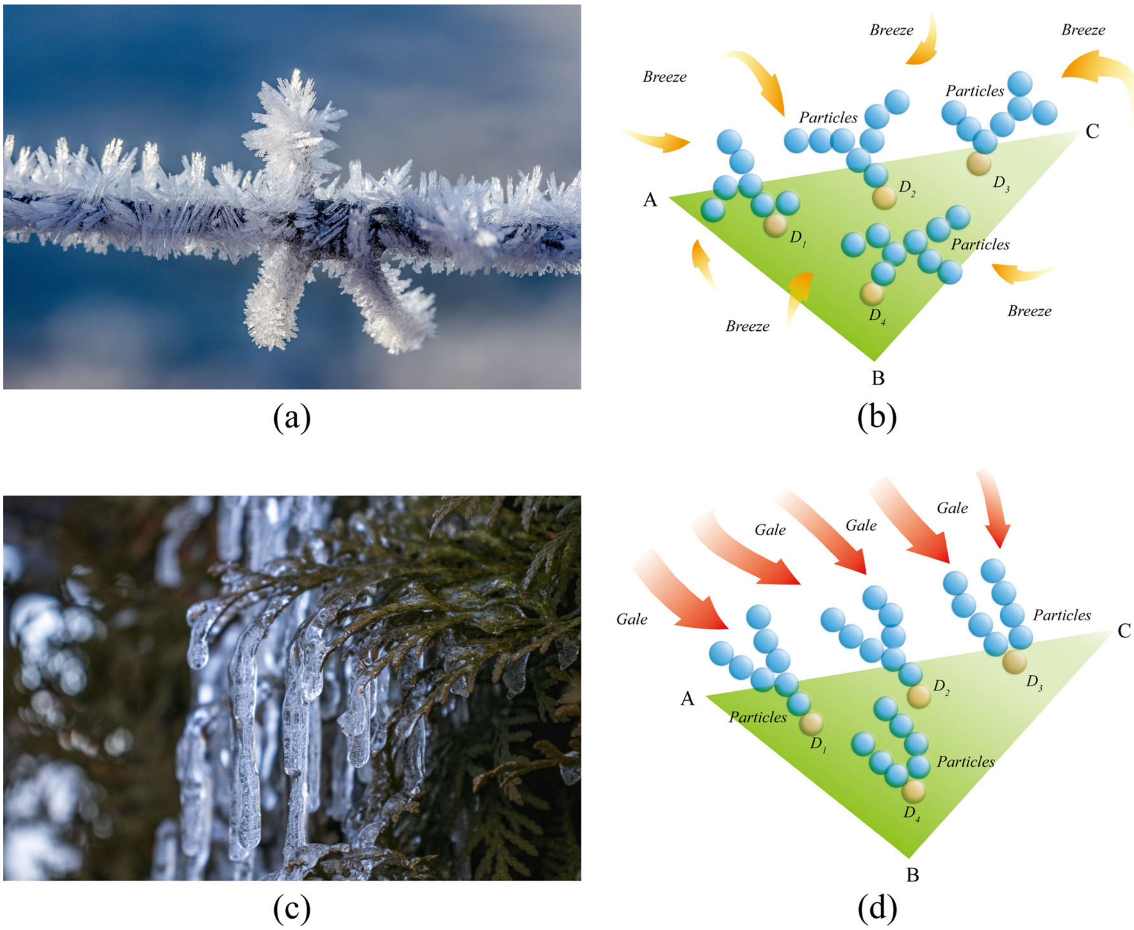


Fig. 2 The real-world rime ice and corresponding mathematical models [24]. **a** soft rime ice. **b** mathematical model of soft rime ice. **c** hard rime ice. **d** mathematical model of hard rime ice

where t and t_{max} are the current and the maximum number of the iteration time, respectively, w is a constant which is suggested to 5, and $[\cdot]$ denotes the rounding operator.

In the gale environment, the mechanism of the hard rime ice growth is simpler and more regular than the growth of soft rime ice, which is expressed by Eq. (4).

$$R_{ij}^{t+1} = R_{best,j},\ if\ r_4 < f^{nor}(R_i) \tag{4}$$

where r_4 is a random value in $(-1, 1)$ and $f^{nor}(R_i)$ is the normalized fitness of R_i. If the condition of Eq. (4) is satisfied, the ith offspring individual replaces the value of jth rime particle by the best rime individual R_{best}.

Additionally, RIME adopts an enhanced greedy selection mechanism, which compares the parent individual with the corresponding offspring individual one by one, if the offspring individual has better fitness, it will directly replace the parent individual and participate in the subsequent optimization process.

3 Our proposal: SRIME

This section introduces our proposed SRIME in detail. We first provide the overview of SRIME, then, the involved techniques are introduced subsequently.

3.1 The overview of SRIME

Figure 3 shows the flowchart of SRIME. First, SRIME initializes the population with the Latin hypercube sampling technique, and if the optimization is not terminated, SRIME updates rime individuals with the soft rime and modified hard rime search strategy in order, then, the embedded distance-based selection mechanism selects the offspring to the subsequent generation of optimization. These processes are repeated until computational resources are exhausted. Next, we will introduce the Latin hypercube sampling, the modified hard rime search strategy, and the embedded distance-based selection mechanism.

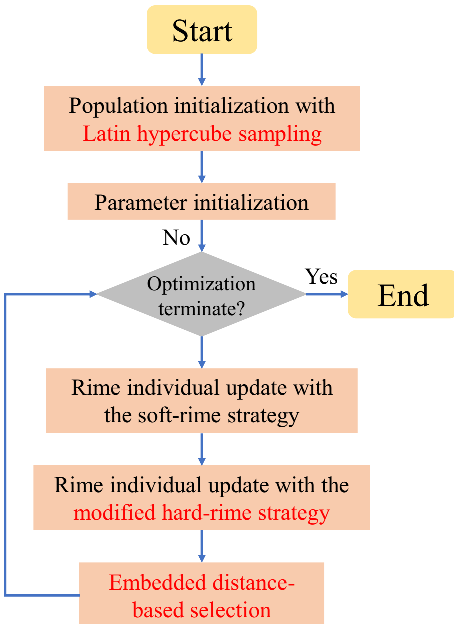


Fig. 3 The flowchart of SRIME

3.2 Latin hypercube sampling

The initial population is critical to the optimization process since a high-quality population can help the algorithm capture the characteristics of the fitness landscape while a poor population may lead the direction of optimization into the bad search area. The conventional random sampling strategy can generate an approximately uniform population in the low-dimensional search space, however, in the high-dimensional search space, this simple scheme cannot ensure that each interval is represented in the sample or that the sample points are well-spaced, and clusters or gaps may exist in the sample space, which will affect the distribution of solutions in the fitness landscape.

To overcome the shortage of simple random initialization, Latin hypercube sampling (LHS) was proposed by Michael Stein [25], which is a statistical method for constructing a near-random sample of parameter values from a multi-dimensional distribution. Here, we first provide a visual demonstration of Latin hypercube sampling with four samples and two-dimensional search space in Fig. 4.

Latin hypercube sampling first divides the search space into equal intervals, and the interval for every dimension is paired randomly to form the sampling areas. Then, each sample is generated randomly within the sampling areas in order. Mathematically, the generated samples can be denoted by Eq. (5) [26]

$$x_i = \frac{1}{n} r + \frac{i-1}{n} \tag{5}$$

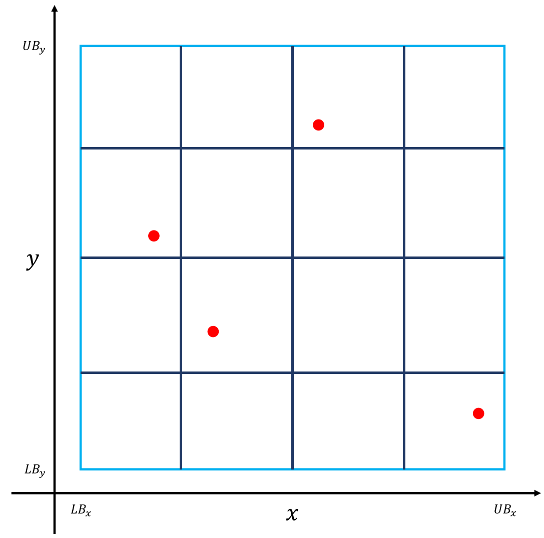


Fig. 4 A visual demonstration of Latin hypercube sampling in two-dimensional search space, and the sample size is equal to four

where r is a uniform random number in (0, 1) and x_i is the ith sample for the ith interval. Compared with random initialization, this technique can be easily extended to the high-dimensional search space while ensuring that each interval is represented by a sample so that the sample points are more uniformly distributed at the cheap cost of the computational budget.

3.3 Modified hard rime search strategy

The essence of the hard rime search strategy in the original RIME can be traced back to the crossover operator of DE [9]. Here, we list the hard rime search strategy of RIME and the conventional crossover operator of DE with a similar format for easy comparison.

$$R_{ij}^{t+1} = \begin{cases} R_{best,j}, & if\ r_4 < f^{nor}(R_i) \\ R_{ij}^{t+1}, & otherwise \end{cases} \tag{6a}$$

$$R_{ij}^{t+1} = \begin{cases} R_{ij}^{t+1}, & if\ r_4 < Cr\ or\ j = jrand \\ R_{ij}^{t}, & otherwise \end{cases} \tag{6b}$$

Eq. (6a) is the hard rime search strategy of RIME and Eq. (6b) is the crossover operator of DE. Cr is the crossover rate. Two different components of these two strategies are the crossover objects and the crossover probability. RIME blends the current best solution R_{best} and soft rime generated solution R_i^{t+1}, while DE mixes the offspring individual R_i^{t+1} generated by mutation operator with the parent solution R_i^t. The hard rime search strategy in RIME leverages the current best individual to do the crossover repeatedly, which significantly accelerates the convergence

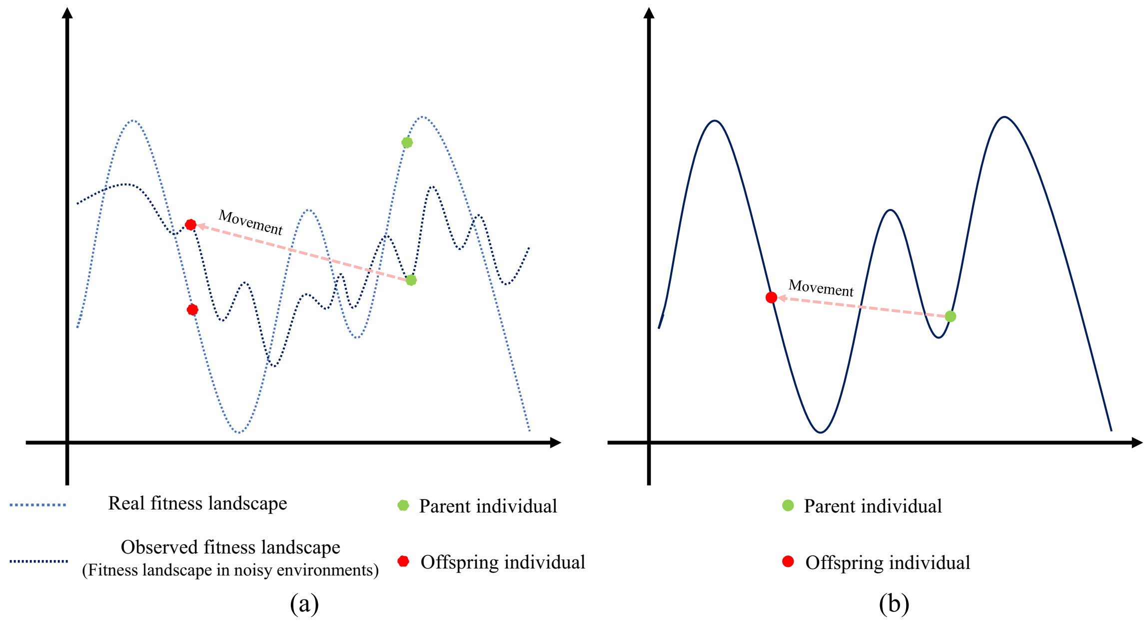


Fig. 5 A visual demonstration of the distance-based selection. **a** In noisy environments. **b** In the multi-modal fitness landscape

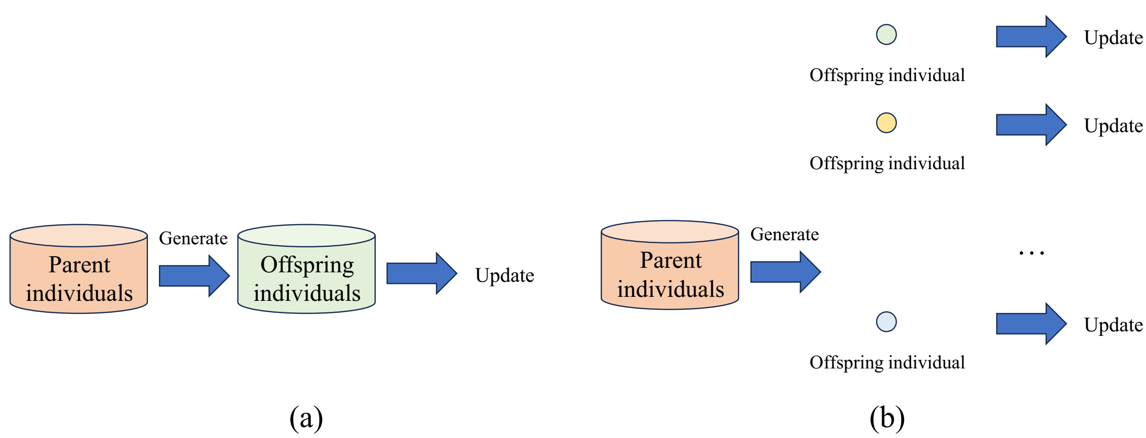


Fig. 6 The difference between RIME and SRIME in the offspring individuals generation and the update process. **a** In the RIME, these two processes are separated strictly. **b** In the SRIME, the update is embedded after each offspring individual is generated

of optimization. However, especially in complex multi-modal problems, the local optima have a strong attraction to the direction of optimization due to this operator, which is known as the basin of attraction [27]. Therefore, it is necessary to increase some randomness to the hard rime search strategy to prevent stagnation when trapping into local optima. Here, we propose a modified hard rime search strategy in Eq. (7)

$$R_{ij}^{t+1} = \begin{cases} R_{best,j} + f^{nor}(R_i) \cdot (R_{r_1,j} - R_{r_2,j}), & \text{if } r_4 < f^{nor}(R_i) \\ R_{ij}^{t+1}, & \text{otherwise} \end{cases} \tag{7}$$

where $R_{r_1} - R_{r_2}$ are two mutually different rime individuals, and we simply use the normalized fitness of R_i as the scaling factor, and this additional term is hoped to avoid the optimization premature.

3.4 Embedded distance-based selection

Distance-based selection is inspired by the simulated annealing (SA) [21] and first proposed in [28] to deal with the optimization in noisy environments. Differently from the SA, where the acceptance probability for the deteriorating solution decreases as the optimization promotes, the distance-based selection scheme adaptively accepts or rejects the offspring individual through two factors: fitness value and distance. Given the uncertainty in noisy environments, the domination between two solutions may change as the intensity of noise changes dynamically.

Thus, the distance-based selection allows the degenerated solution to still have a probability of being accepted and surviving. Equation (8) describes the distance-based selection mechanism with the formulation.

$$R_i^{t+1} = \begin{cases} R_i^{t+1}, & if\ f(R_i^{t+1}) < f(R_i^t) \\ R_i^{t+1}, & else\ if\ r_5 \leq e^{-\frac{\Delta f}{Dist}} \\ R_i^t, & otherwise \end{cases} \tag{8}$$

where r_5 is a random value in (0,1), $\Delta f = |f(R_i^{t+1}) - f(R_i^t)|$, and $Dist$ denotes the Manhattan distance between R_i^{t+1} and R_i^t, which can be calculated by Eq. (9).

$$Dist(R_i^{t+1}, R_i^t) = \sum_{j=1}^{D} |R_{ij}^{t+1} - R_{ij}^t| \tag{9}$$

Three cases are contained in the distance-based selection. Notice that this explanation is under the background of the minimization problem in noisy environments.

Case 1: If the offspring individual has better fitness in noisy environments, then it will replace the parent and survive.

Case 2: In this case, the fitness of the offspring individual is worse than the parent, but the condition of $r_5 \leq e^{-\frac{\Delta f}{Dist}}$ is satisfied, and the degrade offspring individual R_i^{t+1} is still accepted in this situation.

Case 3: If the degeneration of the offspring individual is unaffordable, then, the offspring individual will be rejected.

Here, we emphasize *Case 2* in the distance-based selection. Figure 5 shows a demonstration of the distance-based selection in noisy environments and the multi-modal functions.

The original intention of distance-based selection is to endow the anti-noise ability to the algorithm. As shown in Fig. 5a, we can only observe the fitness of solutions in noisy environments, and the movement from the parent individual to the offspring individual in Fig. 5a will be rejected by the conventional greedy selection, however, the offspring individual actually performs better than the parent individual in the real fitness landscape, and as the noise changes, the domination between these two solutions will also change. Thus, the introduction of distance-based selection to noisy environments is a great success, both in theory and experimental support.

Considering the greed level of the selection mechanism in the original RIME is too high to limit the performance of RIME in multi-modal problems, we introduce the distance-based selection to RIME. Figure 5b shows a demonstration in the multi-modal fitness landscape. Due to the existence of basin attraction [29], the green parent individual is difficult to escape local optimum, and the deteriorating movement in Fig. 5b will be rejected by enhanced greedy selection in the original RIME. However, to escape the local optimum and its attraction, this deteriorating movement is worthy to be accepted. Similarly, the distance-based selection also offers an opportunity for this acceptance. Besides, we focus on the threshold $e^{-\frac{\Delta f}{Dist}}$ in *Case 2* that we fix the Δf between two solutions, and if the Manhattan distance between these two solutions is farther, the value of this threshold $e^{-\frac{\Delta f}{Dist}}$ and the acceptance rate of deteriorating movement also become larger, which can help the direction of optimization escape the local optimum.

The above illustration explains the benefit of distance-based selection in complex optimization problems adequately, and we embed it into our proposed SRIME. Figure 6 shows the difference between RIME and our proposed SRIME in the offspring individual generation and the update process.

In the original RIME, the offspring individual generation and the update are divided independently, which can be analyzed from the source code provided in [24]. In our proposed SRIME, the update process is activated immediately after a new offspring individual is generated as shown in Fig. 6b. Given both the soft rime and the hard rime search strategies in the original RIME use the current optimum R_{best} to generate offspring individuals, our proposed embedded update strategy can update and utilize the current optimum R_{best} promptly.

In summary, the pseudocode of SRIME is shown in Algorithm 1.

Algorithm 1 SRIME

Require: Dimension: D, Population size: PS, Maximum iteration: T,
Ensure: Global optimum: GO

Initialize the rime population $R = \{R_1^0, R_2^0, ..., R_{PS}^0\}$ with Latin hypercube sampling
Get the current best solution GO from R
$t = 0$
while $t < T$ **do**
  Calculate coefficient E by Eq. (3)
  for $i = 0$ *to* PS **do**
    for $j = 0$ *to* D **do**
      Update the i^{th} rime individual R_i^t by Eq. (2) # soft rime
      Update the i^{th} rime individual R_i^t by Eq. (7) # modified hard rime
    end for
    Ensure the individual R_i^t is within search space and evaluate it
    if R_i^{t+1} has better fitness than R_i^t **then** # embedded selection
      Survive R_i^{t+1} to the next generation
      if R_i^{t+1} has better fitness than GO **then**
        Update GO with R_i^{t+1} # global optimum update
      end if
    else if $r_5 \leq e^{-\frac{\Delta f}{Dist}}$ is satisfied **then**
      Survive R_i^{t+1} to the next generation # distance-based selection
    end if
  end for
  $t = t + 1$
end while
return GO

Table 1 Summary of the CEC2020 benchmark functions: Uni. = Unimodal function, Multi. = Multimodal function, Hybrid. = Hybrid function, Comp. = Composition function

Fun	Description	Feature	Optimum
f_1	Shifted and Rotated Bent Cigar function	Uni	100
f_2	Shifted and Rotated Schwefel's function	Multi	1100
f_3	Shifted and Rotated Lunacek bi-Rastrigin function		700
f_4	Expanded Rosenbrock's plus Griewangk's function		1900
f_5	Hybrid function 1 ($N = 3$)	Hybrid	1700
f_6	Hybrid function 2 ($N = 4$)		1600
f_7	Hybrid function 3 ($N = 5$)		2100
f_8	Composition function 1 ($N = 3$)	Comp	2200
f_9	Composition function 2 ($N = 4$)		2400
f_{10}	Composition function 3 ($N = 5$)		2500

Table 2 Summary of eight engineering optimization problems

Name	Abbr	Dim	# of constraints
Welded beam problem	WBP	4	7
Speed reducer problem	SRD	7	11
Cantilever beam problem	CBD	5	1
I beam problem	IBD	4	2
Tubular column problem	TCD	2	6
Piston lever problem	PLD	4	4
Corrugated bulkhead problem	CBHD	4	6
Reinforced concrete beam problem	RCB	3	2

3.5 Computational complexity analysis

SRIME is a population-based stochastic optimization technique, thus, the skeleton of SRIME is like most EAs and can be roughly divided into two stages: initialization and iteration search (i.e., soft rime search and modified hard rime search). Given the dimension of the problem is D, the population size is N and the maximum iteration time is T. First, the computational complexity of population initialization is $O(N \cdot D)$, then, SRIME enters the soft rime search, modified hard rime search, and embedded distance-based selection asynchronously (i.e., Algorithm 1 line 8, 9, and from line 12 to 19). For a single individual, the computational complexity of soft rime search and modified hard rime search is $O(D)$ and $O(D)$. In embedded distance-based selection, as we analyzed in Eq. (8) three cases are involved: In case 1 when the offspring individual has better fitness, it will be accepted, and the computational complexity is $O(1)$. In case 2 when the offspring individual has worse fitness but $r_5 \leq e^{-\frac{\Delta f}{Dist}}$ is satisfied, the offspring individual will be also accepted. In this case, the Manhattan distance between the offspring individual and the parent individual is calculated, and the computational complexity

Table 3 The parameters of all compared optimization methods

EAs	Parameters	Value
PSO (1995)	Inertia factor w	1
	Acceleration coefficients c_1 and c_2	2.05
	Max. and min. speed	2, − 2
DE (1996)	Mutation scheme	DE/cur-to-rand/1
	Scaling factor F	0.8
	Crossover rate Cr	0.9
CMA-ES (2001)	σ	1.3
GWO (2014)	Explicit hyperparameter-free	
AEO (2020)	Explicit hyperparameter-free	
ZOA (2022)	Explicit hyperparameter-free	
WSO (2022)	parameter ρ_r	0.5
EVO (2023)	Explicit hyperparameter-free	
RIME (2023)	parameter w	5
SRIME	parameter w	5

Table 4 Experimental and statistical results on 10-D CEC2020 benchmark functions

Func.		PSO	DE	CMA-ES	GWO	AEO	ZOA	WSO	EVO	RIME	SRIME
f_1	Mean	3.69e+08 +	7.64e+08 +	2.48e+07 +	4.28e+06 +	1.19e+08 +	2.29e+08 +	2.81e+09 +	4.15e+08 +	3.86e+05 +	**6.93e+04**
	Std	5.84e+08	1.61e+08	5.32e+06	1.17e+07	1.41e+08	2.96e+08	1.46e+09	4.04e+08	4.57e+05	1.17e+05
f_2	Mean	9.95e+09 +	7.50e+10 +	2.08e+09 +	2.16e+08 +	9.04e+09 +	2.04e+10 +	2.14e+11 +	2.67e+10 +	2.72e+07 +	**6.25e+06**
	Std	1.25e+10	2.17e+10	6.73e+08	6.52e+08	8.35e+09	1.73e+10	8.71e+10	2.38e+10	2.44e+07	1.01e+07
f_3	Mean	5.51e+09 +	3.49e+10 +	1.00e+09 +	1.70e+08 +	3.59e+09 +	6.47e+09 +	6.72e+10 +	1.15e+10 +	1.18e+07 +	**3.13e+06**
	Std	1.21e+10	9.08e+09	3.14e+08	1.40e+08	4.42e+09	5.07e+09	3.49e+10	1.18e+10	1.23e+07	3.53e+06
f_4	Mean	1.91e+03 +	1.93e+03 +	1.91e+03 +	1.90e+03 +	1.91e+03 +	1.91e+03 +	2.50e+03 +	1.93e+03 +	1.90e+03 +	**1.90e+03**
	Std	1.32e+01	1.66e+01	8.08e−01	6.04e−01	6.97e+00	4.54e+00	9.95e+02	1.26e+02	9.00e−01	6.29e−01
f_5	Mean	1.13e+05 +	1.23e+04 ≈	**2.07e+03** −	2.53e+04 +	1.68e+04 ≈	6.54e+04 +	5.34e+05 +	4.48e+04 +	1.19e+04 ≈	1.22e+04
	Std	1.44e+05	3.46e+03	8.93e+01	1.23e+04	7.92e+03	9.13e+04	4.98e+05	7.45e+04	7.46e+03	8.46e+03
f_6	Mean	8.42e+03 +	2.88e+03 +	**2.00e+03** ≈	3.22e+03 +	2.38e+03 +	5.68e+03 +	1.67e+04 +	6.32e+03 +	3.27e+03 ≈	2.25e+03
	Std	7.56e+03	3.93e+02	1.92e+02	1.44e+03	5.16e+02	3.73e+03	7.96e+03	4.84e+03	2.52e+03	1.42e+03
f_7	Mean	2.87e+05 +	5.72e+04 +	**6.47e+03** −	2.35e+04 ≈	1.28e+04 ≈	1.78e+04 ≈	5.70e+05 +	8.59e+04 +	1.83e+04 ≈	2.13e+04
	Std	3.38e+05	2.27e+04	1.60e+03	1.48e+04	7.75e+03	8.90e+03	4.41e+05	2.32e+05	1.04e+04	1.61e+04
f_8	Mean	2.32e+03 +	2.32e+03 +	2.31e+03 +	2.30e+03 ≈	2.31e+03 +	2.31e+03 +	2.33e+03 +	2.31e+03 +	2.30e+03 ≈	**2.30e+03**
	Std	3.64e+00	1.48e+00	9.83e−01	3.62e+00	3.39e+00	2.40e+00	8.55e+00	2.92e+00	2.15e+00	1.43e+00
f_9	Mean	3.46e+03 +	3.82e+03 +	2.87e+03 +	2.70e+03 +	3.04e+03 +	3.17e+03 +	4.26e+03 +	3.34e+03 +	2.63e+03 +	**2.59e+03**
	Std	5.48e+02	1.50e+02	3.86e+01	5.77e+01	1.99e+02	3.34e+02	3.97e+02	4.22e+02	4.07e+01	3.38e+01
f_{10}	Mean	3.09e+03 +	3.09e+03 +	3.00e+03 +	3.00e+03 +	3.07e+03 +	3.06e+03 +	3.15e+03 +	3.09e+03 +	2.99e+03 ≈	**2.99e+03**
	Std	5.26e+01	3.37e+01	6.26e+00	1.26e+01	5.37e+01	3.80e+01	4.59e+01	3.44e+01	2.15e+01	1.99e+01
+/ ≈ /− summary:		10/0/0	9/1/0	7/1/2	8/2/0	8/2/0	9/1/0	10/0/0	10/0/0	5/5/0	
Ave. rank		7.7	7.7	3.1	3.9	4.9	10	5.8	7.6	2.6	**1.7**

f_1: Unimodal function; $f_2 - f_4$: Multimodal functions; $f_5 - f_7$: Hybrid functions; $f_8 - f_{10}$: Composition functions; mean and std: the mean and the standard deviation of 30 trial runs

is $O(D)$. In case 3, the above conditions are not satisfied, thus, the parent individual will survive to the next generation, and the computational complexity is $O(1)$. Besides, for a single RIME individual, only one selection operator will be activated, and the computational complexity of embedded distance-based selection is $\max\{O(1), O(D), O(1)\} = O(D)$.

In summary, the computational complexity of SRIME is

$$\begin{aligned} &O(N \cdot D + T \cdot (N \cdot (D + D + D))) \\ &= O(N \cdot D + 3 \cdot T \cdot N \cdot D)) \end{aligned} \tag{10}$$

4 Numerical experiments

This section introduces the numerical experiments in detail. Section 4.1 introduces the experiment settings: experimental environments, benchmark functions, and compared methods with their parameters. Section 4.2 shows the experimental and statistical results.

Table 5 Experimental and statistical results on 30-D CEC2020 benchmark functions

Func.		PSO	DE	CMA-ES	GWO	AEO	ZOA	WSO	EVO	RIME	SRIME
f_1	Mean	4.35e+09	3.87e+10	3.20e+09	5.10e+08	7.20e+09	1.75e+10	3.17e+10	1.01e+10	2.56e+07	**1.70e+06**
		+	+	+	+	+	+	+	+	+	
	Std	1.85e+09	5.06e+09	5.67e+08	4.17e+08	2.16e+09	3.55e+09	8.66e+09	3.78e+09	1.06e+07	1.55e+06
f_2	Mean	6.41e+11	3.79e+12	3.08e+11	6.21e+10	8.91e+11	2.00e+12	3.96e+12	9.99e+11	2.88e+09	**1.84e+08**
		+	+	+	+	+	+	+	+	+	
	Std	5.70e+11	3.83e+11	4.39e+10	7.24e+10	3.63e+11	4.40e+11	9.07e+11	3.69e+11	1.16e+09	2.12e+08
f_3	Mean	1.54e+11	1.24e+12	1.04e+11	1.52e+10	2.15e+11	6.14e+11	1.17e+12	3.54e+11	8.73e+08	**9.07e+07**
		+	+	+	+	+	+	+	+	+	
	Std	8.98e+10	1.56e+11	1.85e+10	1.27e+10	6.58e+10	1.33e+11	2.32e+11	1.06e+11	4.15e+08	1.24e+08
f_4	Mean	6.79e+03	9.84e+04	2.09e+03	1.92e+03	8.85e+03	2.43e+04	1.72e+05	9.15e+03	1.92e+03	**1.91e+03**
		+	+	+	+	+	+	+	+	+	
	Std	6.93e+03	3.84e+04	9.09e+01	5.56e+00	1.00e+04	1.76e+04	1.07e+05	9.48e+03	3.54e+00	2.57e+00
f_5	Mean	2.05e+07	1.51e+07	**5.51e+05**	7.50e+05	5.06e+06	8.03e+06	4.89e+07	2.26e+06	9.33e+05	6.37e+05
		+	+	≈	≈	+	+	+	+	≈	
	Std	1.45e+07	5.77e+06	1.41e+05	4.98e+05	5.57e+06	8.67e+06	2.63e+07	2.63e+06	6.25e+05	3.56e+05
f_6	Mean	3.84e+06	2.02e+05	**6.05e+03**	3.64e+04	9.62e+04	1.76e+05	1.13e+07	1.98e+06	4.87e+04	3.82e+04
		+	+	−	≈	+	+	+	+	≈	
	Std	9.37e+06	4.71e+05	1.51e+03	1.93e+04	1.16e+05	3.52e+05	1.46e+07	3.92e+06	1.91e+04	2.25e+04
f_7	Mean	6.94e+07	3.56e+07	**5.40e+05**	1.27e+06	1.36e+07	3.14e+07	2.29e+08	7.62e+06	1.40e+06	1.07e+06
		+	+	−	≈	+	+	+	+	≈	
	Std	7.15e+07	1.06e+07	1.58e+05	7.95e+05	1.17e+07	2.51e+07	1.31e+08	7.33e+06	6.66e+05	7.06e+05
f_8	Mean	2.79e+03	2.65e+03	2.45e+03	2.38e+03	2.91e+03	2.71e+03	3.20e+03	2.47e+03	2.40e+03	**2.38e+03**
		+	+	+	≈	+	+	+	+	+	
	Std	2.04e+02	3.37e+01	7.75e+00	9.81e+00	2.30e+02	8.70e+01	3.03e+02	2.77e+01	1.39e+01	8.46e+00
f_9	Mean	8.61e+03	1.30e+04	6.27e+03	3.56e+03	8.46e+03	1.64e+04	2.21e+04	1.17e+04	2.97e+03	**2.70e+03**
		+	+	+	+	+	+	+	+	+	
	Std	1.87e+03	6.30e+02	2.81e+02	8.88e+02	9.54e+02	2.20e+03	3.63e+03	1.88e+03	8.47e+01	7.07e+01
f_{10}	Mean	3.68e+03	5.11e+03	3.11e+03	2.99e+03	3.53e+03	3.89e+03	4.80e+03	3.63e+03	2.95e+03	**2.93e+03**
		+	+	+	+	+	+	+	+	+	
	Std	4.86e+02	4.66e+02	3.89e+01	4.86e+01	1.73e+02	2.56e+02	4.02e+02	1.82e+02	2.85e+01	1.51e+01
+/≈/− summary:		10/0/0	10/0/0	7/1/2	6/4/0	10/0/0	10/0/0	10/0/0	10/0/0	7/3/0	
Ave. rank		6.8	8.5	3.1	2.6	6.0	9.7	7.6	6.4	2.8	**1.5**

4.1 Experiment settings

4.1.1 Experimental environments and implementation

All optimization methods are programmed by Python 3.11 and implemented in Lenovo Legion R9000P, which is equipped with Windows 11, AMD Ryzen 7 5800 H with Radeon Graphics 3.20 GHz, and 16GB RAM. All compared algorithms are provided by the MEALPY library [30], CEC2020 benchmark functions by OpFuNu library [31], and eight engineering optimization problems are provided by ENOPPY library [32].

4.1.2 Benchmark functions

We evaluate our proposed SRIME on CEC2020 benchmark functions [33] and eight engineering optimization problems as same as in [34]. The summary of CEC2020 benchmark functions and eight engineering optimization problems are listed in Tables 1 and 2, respectively. More detailed formulation and visualization of engineering optimization problems can be found in [34].

4.1.3 Compared methods and parameters

We compare our proposed SRIME to two categories of EAs with a total of nine EAs: (1). Classic EAs including particle swarm optimization (PSO) [12], differential

Table 6 Experimental and statistical results on 50-D CEC2020 benchmark functions

Func.		PSO	DE	CMA-ES	GWO	AEO	ZOA	WSO	EVO	RIME	SRIME
f_1	Mean	1.91e+10 +	1.07e+11 +	1.11e+10 +	2.70e+09 +	2.24e+10 +	5.15e+10 +	7.83e+10 +	3.95e+10 +	2.22e+08 +	**1.28e+07**
	Std	3.76e+09	1.04e+10	3.09e+09	1.85e+09	4.87e+09	5.21e+09	9.89e+09	8.75e+09	6.34e+07	8.72e+06
f_2	Mean	2.32e+12 +	1.15e+13 +	1.34e+12 +	2.38e+11 +	2.48e+12 +	5.62e+12 +	9.18e+12 +	4.21e+12 +	2.63e+10 +	**1.10e+09**
	Std	5.45e+11	1.39e+12	4.36e+11	1.43e+11	5.73e+11	9.42e+11	1.25e+12	9.14e+11	8.67e+09	5.87e+08
f_3	Mean	6.79e+11 +	4.13e+12 +	4.05e+11 +	9.48e+10 +	8.26e+11 +	1.94e+12 +	2.98e+12 +	1.22e+12 +	8.56e+09 +	**3.69e+08**
	Std	1.98e+11	4.60e+11	8.35e+10	5.35e+10	1.99e+11	2.11e+11	5.35e+11	2.33e+11	2.52e+09	2.32e+08
f_4	Mean	8.92e+04 +	1.40e+06 +	1.01e+04 +	1.96e+03 +	5.59e+04 +	2.94e+05 +	1.01e+06 +	1.07e+05 +	1.95e+03 +	**1.93e+03**
	Std	1.31e+05	6.32e+05	6.69e+03	3.48e+01	5.80e+04	1.24e+05	4.79e+05	1.11e+05	1.10e+01	8.32e+00
f_5	Mean	6.63e+07 +	6.23e+07 +	**2.39e+06** −	3.91e+06 −	2.18e+07 +	4.55e+07 +	1.33e+08 +	2.18e+07 +	6.89e+06 +	4.93e+06
	Std	4.06e+07	2.18e+07	4.70e+05	3.78e+06	1.13e+07	2.84e+07	5.29e+07	1.64e+07	2.84e+06	2.44e+06
f_6	Mean	1.03e+07 +	9.22e+07 +	**3.51e+04** −	3.93e+05 +	7.34e+06 +	2.75e+08 +	7.37e+08 +	8.72e+07 +	4.49e+05 +	7.91e+04
	Std	1.46e+07	3.22e+07	6.68e+03	4.38e+05	8.34e+06	3.49e+08	5.46e+08	6.85e+07	4.49e+05	5.97e+04
f_7	Mean	9.15e+08 +	7.53e+08 +	1.27e+07 +	6.93e+06 ≈	1.46e+08 +	6.09e+08 +	2.55e+09 +	1.81e+08 +	8.38e+06 ≈	**6.14e+06**
	Std	7.65e+08	1.81e+08	3.96e+06	4.58e+06	1.10e+08	4.20e+08	1.56e+09	1.95e+08	4.46e+06	2.69e+06
f_8	Mean	4.14e+03 +	3.19e+03 +	2.66e+03 +	**2.45e+03** −	6.30e+03 +	4.71e+03 +	6.81e+03 +	2.84e+03 +	2.52e+03 −	2.63e+03
	Std	9.35e+02	1.06e+02	3.91e+01	1.90e+01	2.33e+03	7.31e+02	1.52e+03	9.98e+01	2.33e+01	8.87e+02
f_9	Mean	2.31e+04 +	2.74e+04 +	1.12e+04 +	6.06e+03 +	1.85e+04 +	4.52e+04 +	5.56e+04 +	4.01e+04 +	3.85e+03 +	**2.96e+03**
	Std	3.37e+03	1.91e+03	1.36e+03	1.51e+03	5.36e+03	5.31e+03	5.57e+03	8.14e+03	2.26e+02	2.41e+02
f_{10}	Mean	8.44e+03 +	1.23e+04 +	4.58e+03 +	3.85e+03 +	7.03e+03 +	1.09e+04 +	1.71e+04 +	8.80e+03 +	3.71e+03 +	**3.41e+03**
	Std	1.88e+03	1.78e+03	2.49e+02	2.59e+02	7.70e+02	1.38e+03	3.30e+03	1.00e+03	1.85e+02	1.00e+02
+/ ≈ /− summary:		10/0/0	10/0/0	8/0/2	7/1/2	10/0/0	10/0/0	10/0/0	10/0/0	8/1/1	
Ave. rank		6.4	8.6	3.4	2.6	5.8	9.6	8.0	6.6	2.5	**1.5**

evolution (DE) [9], covariance matrix adaptation evolutionary strategy (CMA-ES) [35], and gray wolf optimizer (GWO) [14]. (2). The latest EAs including artificial ecosystem-based optimization (AEO) [36], zebra optimization algorithm (ZOA) [37], war strategy optimization (WSO) [38], energy valley optimizer (EVO) [39], and the original RIME [24]. The population size of all algorithms is set to 100, the maximum FEs for CEC2020 functions is 1000 · D (D=dimension size), and the maximum FEs for engineering problems is 20,000. To alleviate the randomness in the optimization, each EA is executed with 30 trial runs. The detailed parameter settings of the competitor algorithms are listed in Table 3. All parameters are consistent with the suggested settings in corresponding papers.

Given the engineering optimization problem contains constraints and original EAs including SRIME cannot deal with constrained optimization problems, thus, we equip all EAs with the static penalty function [40], which is defined by Eq. (11)

$$F(R_i) = f(R_i) + w \cdot \sum_{i=1}^{m} (\max(0, g_i(R_i))) \tag{11}$$

$F(\cdot)$ is the fitness function, while $f(\cdot)$ and $g_i(\cdot)$ are the objective function and constraint function, respectively. w is a constant set to $10e7$ by default.

Besides, the ablation experiment on CEC2020 benchmark functions is also provided to evaluate the contribution of each proposed component. We re-emphasize our proposed three adjustments in RIME: (1). Latin hypercube

Table 7 Experimental and statistical results on 100-D CEC2020 benchmark functions

Func.		PSO	DE	CMA-ES	GWO	AEO	ZOA	WSO	EVO	RIME	SRIME
f_1	Mean	1.01e+11	3.33e+11	8.40e+10	1.50e+10	6.96e+10	1.75e+11	2.22e+11	1.48e+11	3.19e+09	**1.48e+08**
		+	+	+	+	+	+	+	+	+	
	Std	1.33e+10	2.41e+10	1.42e+10	5.21e+09	1.35e+10	9.53e+09	1.66e+10	1.28e+10	6.62e+08	7.70e+07
f_2	Mean	1.10e+13	3.08e+13	8.88e+12	1.42e+12	6.93e+12	1.95e+13	2.49e+13	1.40e+13	2.96e+11	**1.32e+10**
		+	+	+	+	+	+	+	+	+	
	Std	1.38e+12	2.17e+12	1.66e+12	4.22e+11	1.36e+12	1.23e+12	2.10e+12	1.89e+12	6.43e+10	7.51e+09
f_3	Mean	3.60e+12	1.17e+13	3.01e+12	6.19e+11	2.36e+12	6.53e+12	8.79e+12	5.05e+12	1.08e+11	**5.66e+09**
		+	+	+	+	+	+	+	+	+	
	Std	4.66e+11	8.57e+11	6.10e+11	1.68e+11	4.02e+11	4.80e+11	5.79e+11	6.98e+11	2.26e+10	3.17e+09
f_4	Mean	2.51e+05	2.10e+07	7.88e+04	3.40e+03	1.41e+05	1.13e+06	3.30e+06	6.52e+05	2.15e+03	**2.08e+03**
		+	+	+	+	+	+	+	+	+	
	Std	1.10e+05	6.64e+06	3.02e+04	1.13e+03	6.70e+04	4.05e+05	9.07e+05	2.63e+05	8.49e+01	4.84e+01
f_5	Mean	5.82e+08	7.70e+08	2.89e+07	**2.78e+07**	1.76e+08	3.72e+08	6.73e+08	1.83e+08	6.28e+07	3.37e+07
		+	+	≈	≈	+	+	+	+	+	
	Std	3.39e+08	1.04e+08	6.80e+06	1.18e+07	4.42e+07	1.07e+08	2.03e+08	5.64e+07	2.08e+07	9.38e+06
f_6	Mean	5.94e+08	3.43e+09	**3.24e+05**	2.45e+06	8.17e+07	5.82e+09	1.30e+10	2.21e+09	4.35e+05	3.42e+05
		+	+	≈	+	+	+	+	+	≈	
	Std	7.68e+08	1.24e+09	1.15e+05	2.37e+06	8.80e+07	2.34e+09	6.34e+09	1.69e+09	2.83e+05	2.66e+05
f_7	Mean	4.42e+09	5.21e+09	3.55e+07	4.64e+07	8.66e+08	4.98e+09	1.32e+10	1.90e+09	5.42e+07	2.32e+07
		+	+	+	+	+	+	+	+	+	
	Std	2.85e+09	1.48e+09	1.10e+07	3.28e+07	3.07e+08	1.64e+09	5.35e+09	6.33e+08	2.02e+07	**7.00e+06**
f_8	Mean	8.74e+03	4.51e+03	3.54e+03	3.39e+03	1.86e+04	1.43e+04	1.97e+04	4.19e+03	**2.70e+03**	3.26e+03
		+	+	+	≈	+	+	+	+	−	
	Std	1.73e+03	3.77e+02	2.52e+02	3.06e+03	5.17e+03	3.23e+03	3.54e+03	6.11e+02	5.94e+01	2.69e+03
f_9	Mean	1.02e+05	1.16e+05	6.75e+04	2.73e+04	9.41e+04	1.45e+05	1.68e+05	1.41e+05	1.34e+04	**4.65e+03**
		+	+	+	+	+	+	+	+	+	
	Std	1.02e+04	1.27e+04	8.73e+03	6.67e+03	1.98e+04	6.47e+03	6.61e+03	9.10e+03	1.17e+03	6.25e+02
f_{10}	Mean	1.24e+04	4.19e+04	8.62e+03	4.49e+03	9.26e+03	1.87e+04	2.86e+04	1.46e+04	4.14e+03	**3.55e+03**
		+	+	+	+	+	+	+	+	+	
	Std	1.60e+03	7.87e+03	1.11e+03	2.42e+02	1.05e+03	1.80e+03	3.36e+03	2.02e+03	1.65e+02	6.87e+01
+/≈/− summary:		10/0/0	10/0/0	8/2/0	8/2/0	10/0/0	10/0/0	10/0/0	10/0/0	8/1/1	
Ave. rank:		6.4	9.0	3.6	2.9	5.1	9.4	8.1	6.7	2.4	**1.4**

sampling for population initialization. (2). Modified hard rime search strategy. (3). Embedded distance-based selection mechanism, which combined with the original RIME independently.

4.2 Experimental and statistical results

This section shows the experimental and statistical results of optimization, all EAs in each function are independently implemented 30 times. To determine the statistical significance between every pair of competitor algorithms, we first apply the Kruskal–Wallis. If the statistical significance exists, then, the Mann–Whitney U test is employed to calculate the p-value between every pair of algorithms. Finally, we use the Holm multiple comparison test [41] to correct the p-value obtained by the Mann–Whitney U test and identify the statistical significance level. Symbols +, ≈, and − are applied to represent that our proposed SRIME is significantly better, with no significance, and significantly worse with the compared method. Additionally, the Friedman test is also conducted to calculate the averaging rank of every algorithm, and the best value is in bold.

4.2.1 Optimization performance on CEC2020 suite

Tables 4, 5, 6 and 7 summarize the experimental and statistical results of optimization on 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions among ten EAs. Due to the limitation of space, convergence curves of representative functions (the unimodal function f_1, the

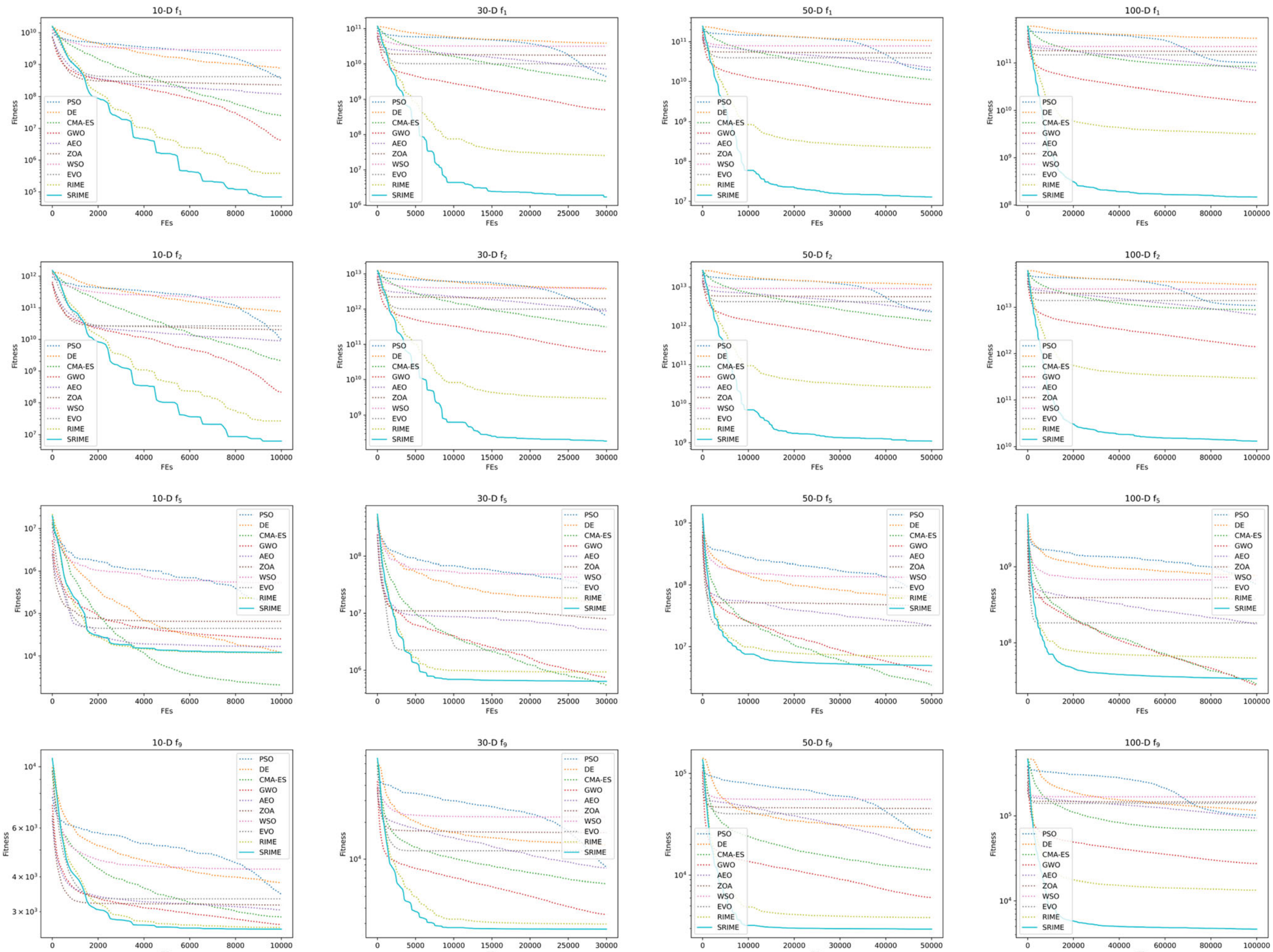


Fig. 7 The convergence curve of 10-D, 30-D, 50-D, and 100-D f_1, f_2, f_5, and f_9 in CEC2020 benchmark functions

multimodal function f_2, the hybrid function $f5$, and the composition function) f_9 are provided in Fig. 7.

4.2.2 Optimization performance on engineering problems

Table 8 summarizes the experimental and statistical results on eight engineering optimization problems among ten EAs, and Fig. 8 visualizes the convergence curves.

4.2.3 Ablation experiment on CEC2020 benchmark functions

Tables 9, 10, 11 and 12 summarize ablation experiments on 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions. Here, we first define the abbreviation of compared algorithms. RIME$_{lhs}$: The original RIME combines with Latin hypercube sampling, RIME$_{mhs}$: The original RIME combines with the modified hard rime search strategy, and RIME$_{edbs}$: The original RIME combines with the embedded distance-based selection mechanism.

5 Discussion

In this section, we analyze the efficiency of our proposed SRIME based on the performance in the CEC2020 suite and eight engineering optimization problems, and some open topics are provided to further improve the SRIME in future works.

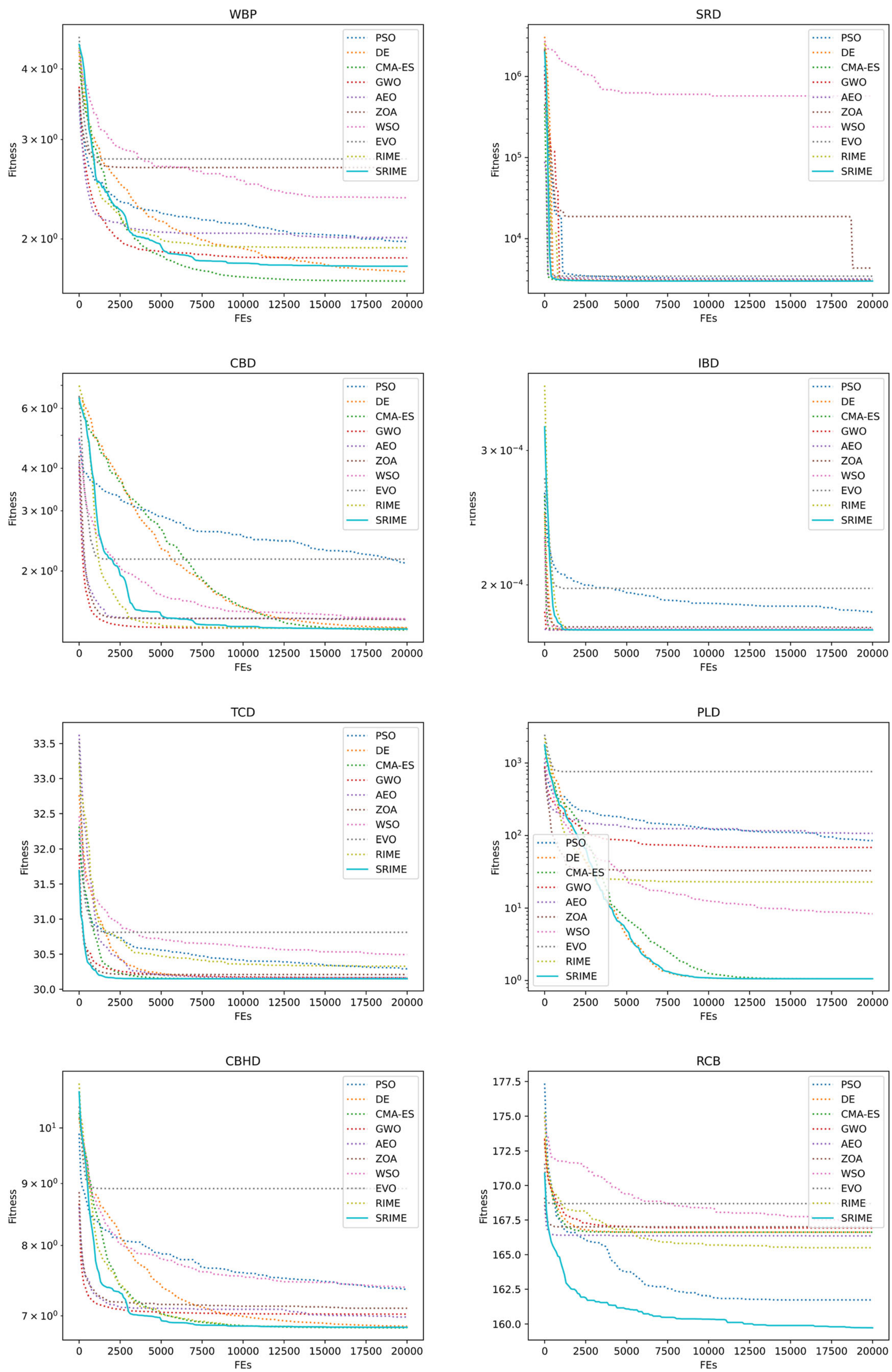
5.1 Performance analysis based on CEC2020 suite

Since benchmark functions in the CEC2020 suite have various characteristics such as shifted, rotated, unimodal, multi-modal, and composite, thus, it can fully evaluate the overall performance of our proposed SRIME. In the unimodal function (i.e., f_1), SRIME inherits the superior

Table 8 Experimental and statistical results on engineering problems

Func.		PSO	DE	CMA-ES	GWO	AEO	ZOA	WSO	EVO	RIME	SRIME
WBP	Mean	1.9782e+00 +	1.7491e+00 ≈	**1.6835e+00** −	1.8507e+00 ≈	2.0093e+00 +	2.6751e+00 +	2.3641e+00 +	2.7706e+00 +	1.9293e+00 +	1.7855e+00
	Std	1.0231e−01	2.5584e−02	3.8103e−04	1.8332e−01	3.7077e−01	3.6408e−01	3.7174e−01	5.3400e−01	2.2111e−01	1.5360e−01
	best	1.8004e+00	1.7205e+00	1.6829e+00	1.6958e+00	1.6845e+00	1.9634e+00	1.7475e+00	1.9008e+00	1.7036e+00	1.6876e+00
	worst	2.1512e+00	1.8252e+00	1.6844e+00	2.3770e+00	3.0495e+00	3.6005e+00	3.2922e+00	4.1217e+00	2.8182e+00	2.3626e+00
SRD	Mean	3.1382e+03 +	2.9883e+03 +	2.9871e+03 +	3.0691e+03 +	3.1745e+03 +	4.3262e+03 +	5.7932e+05 +	3.4347e+03 +	2.9963e+03 +	**2.9869e+03**
	Std	5.2582e+01	1.6237e+00	4.9198e−02	5.6899e+01	5.1034e+02	6.0602e+02	1.3695e+06	3.3585e+02	7.0867e+00	5.6256e−04
	best	3.0381e+03	2.9870e+03	2.9870e+03	2.9983e+03	2.9881e+03	3.2484e+03	3.0268e+03	3.0100e+03	2.9876e+03	2.9869e+03
	worst	3.2474e+03	2.9955e+03	2.9872e+03	3.1891e+03	5.1007e+03	6.0877e+03	5.7820e+06	4.2829e+03	3.0180e+03	2.9869e+03
CBD	Mean	2.1032e+00 +	1.3615e+00 +	**1.3420e+00** −	1.3585e+00 ≈	1.4397e+00 +	1.4426e+00 +	1.4467e+00 +	2.1672e+00 +	1.3574e+00 ≈	1.3503e+00
	Std	2.8919e−01	6.1573e−03	1.2834e−03	2.2945e−02	7.8800e−02	6.6320e−02	5.7351e−02	4.5218e−01	1.5191e−02	8.7103e−03
	best	1.6347e+00	1.3498e+00	1.3402e+00	1.3410e+00	1.3426e+00	1.3551e+00	1.3590e+00	1.4512e+00	1.3409e+00	1.3419e+00
	worst	2.8433e+00	1.3711e+00	1.3455e+00	1.4311e+00	1.6181e+00	1.6095e+00	1.6384e+00	3.2013e+00	1.4100e+00	1.3853e+00
IBD	Mean	1.8405e−04 +	1.7458e−04 ≈	**1.7458e −04** ≈	1.7458e−04 ≈	**1.7458e −04** ≈	1.7588e−04 +	1.7531e−04 +	1.9790e−04 +	1.7458e−04 ≈	1.7458e−04
	Std	3.7286e−06	3.9501e−15	1.3553e−19	1.6823e−11	1.3553e−19	4.0393e−06	5.5047e−07	1.9463e−05	1.1196e−09	1.4328e−10
	best	1.7779e−04	1.7458e−04	1.7458e−04	1.7458e−04	1.7458e−04	1.7458e−04	1.7462e−04	1.7459e−04	1.7458e−04	1.7458e−04
	worst	1.9246e−04	1.7458e−04	1.7458e−04	1.7458e−04	1.7458e−04	1.9339e−04	1.7610e−04	2.6354e−04	1.7459e−04	1.7458e−04
TCD	Mean	3.0291e+01 +	3.0150e+01 +	3.0150e+01 +	3.0162e+01 +	3.0164e+01 +	3.0210e+01 +	3.0495e+01 +	3.0812e+01 +	3.0328e+01 +	**3.0150e+01**
	Std	7.6171e−02	4.4080e−05	2.0787e−09	1.4541e−02	1.4648e−02	8.5008e−02	1.4293e−01	4.2575e−01	2.0913e−01	6.0611e−12
	best	3.0187e+01	3.0150e+01	3.0150e+01	3.0150e+01	3.0150e+01	3.0150e+01	3.0152e+01	3.0176e+01	3.0151e+01	3.0150e+01
	worst	3.0467e+01	3.0150e+01	3.0150e+01	3.0205e+01	3.0213e+01	3.0498e+01	3.0781e+01	3.1774e+01	3.0857e+01	3.0150e+01
PLD	Mean	8.5114e+01 +	1.0574e+00 ≈	1.0575e+00 +	6.8752e+01 +	1.0698e+02 +	3.2786e+01 +	8.3341e+00 +	7.6292e+02 +	2.2975e+01 +	**1.0574e+00**
	Std	3.7737e+01	8.1546e−07	6.4088e−05	1.0936e+02	1.1310e+02	7.5890e+01	6.4679e+00	7.1836e+02	6.5686e+01	6.0867e−07
	best	2.1552e+01	1.0574e+00	1.0574e+00	1.0579e+00	1.0577e+00	4.3765e+00	1.9064e+00	7.7311e+01	1.0623e+00	1.0574e+00
	worst	1.8491e+02	1.0574e+00	1.0576e+00	3.6459e+02	3.1880e+02	3.3513e+02	2.6399e+01	3.0660e+03	2.2998e+02	1.0574e+00
CBHD	Mean	7.3600e+00 +	6.8632e+00 +	**6.8437e+00** −	7.0239e+00 +	6.9846e+00 +	7.1027e+00 +	7.3903e+00 +	8.9145e+00 +	6.8507e+00 ≈	6.8495e+00
	Std	2.3485e−01	9.4051e−03	2.4452e−04	1.2506e−01	9.0480e−02	1.5523e−01	2.3715e−01	8.2938e−01	4.9921e−03	5.3934e−03
	best	6.9522e+00	6.8482e+00	6.8433e+00	6.8721e+00	6.8465e+00	6.9030e+00	7.0632e+00	7.2500e+00	6.8440e+00	6.8436e+00
	worst	7.8107e+00	6.8826e+00	6.8443e+00	7.3503e+00	7.1372e+00	7.4518e+00	8.1239e+00	1.0978e+01	6.8638e+00	6.8661e+00
RCB	Mean	1.6174e+02 +	1.6662e+02 +	1.6662e+02 +	1.6693e+02 +	1.6636e+02 +	1.6701e+02 +	1.6750e+02 +	1.6869e+02 +	1.6550e+02 +	**1.5973e+02**
	Std	2.0897e+00	6.0548e−01	6.0548e−01	9.9594e−01	1.2671e+00	1.0834e+00	1.3314e+00	2.0974e+00	1.3240e+00	3.6992e−01
	best	1.5936e+02	1.6356e+02	1.6356e+02	1.6356e+02	1.6055e+02	1.6356e+02	1.6356e+02	1.6356e+02	1.6386e+02	1.5937e+02
	worst	1.6677e+02	1.6677e+02	1.6677e+02	1.6844e+02	1.6677e+02	1.7021e+02	1.7142e+02	1.7306e+02	1.6713e+02	1.6115e+02
+/ ≈ /− summary:		8/0/0	5/3/0	4/1/3	5/3/0	7/1/0	8/0/0	8/0/0	8/0/0	5/3/0	
Ave. rank		6.8	3.5	**2.0**	5.1	5.6	7.5	8.0	9.8	4.6	**2.0**

Mean and std: the mean and the standard deviation of 30 trial runs; best and worst: the best and the worst final solution found among 30 trial runs


WBP
SRD
CBD
IBD
TCD
PLD
CBHD
RCB
PSO
DE
CMA-ES
GWO
AEO
ZOA
WSO
EVO
RIME
SRIME
Fitness
FEs

◀ **Fig. 8** Convergence curves of eight engineering optimization problems

exploitation ability of RIME and outperforms PSO, DE, CMA-ES, GWO, AEO, ZOA, WSO, EVO, and the original RIME in f_1 with distinct dimensions. As the dimension of the problem increases, the superiority of SRIME is amplified, and the domination between SRIME and other competitor algorithms is not changed, which can be observed from statistical analysis in Tables 4, 5, 6 and 7. The curves of optimization in Fig. 7 also confirm the excellent convergence speed of SRIME. Therefore, we can conclude that SRIME has a strong exploitation ability.

f_2 to f_{10} in CEC2020 benchmark functions are multimodal, hybrid, and composite functions, these functions have multiple optima and complex fitness landscapes so that they are allowed to evaluate the exploration ability and the capacity of escaping from local optima. Experimental and statistical results in Tables 4, 5, 6 and 7 show that our proposed SRIME has better performance on most of instances, and we can infer that the RIME structure-based optimization technique (i.e., RIME and SRIME) is competitive to the state-of-the-art EAs. Besides, the introduction of our proposed three improvements can at least approximately equal to or significantly accelerate the convergence of RIME in most cases. As the dimension of the problem increases, the superiority of SRIME is enhanced, which can be observed from the performance indicator of average rank. Consequently, we conclude that the incorporation of RIME and our proposed three strategies is suitable to deal with these optimization problems.

However, the significant deterioration situations between RIME and SRIME happen on 50-D and 100-D f_{10}, and we reasonably infer that our proposed three techniques are not proper for this specific task. In this case, the NFLT can be applied to explain this degeneration. NFLT states that every pair of optimization algorithms has identical average performance on all possible problems, and if an algorithm performs better on a certain category problem, it must deteriorate on the rest of the problems, since it is the only way to achieve the same average performance. Thus, the inferiority of SRIME in 50-D and 100-D f_{10} is acceptable.

5.2 Performance analysis based on engineering problems

In real-world applications, another characteristic of the optimization algorithm that we are concerned about is stability. If an algorithm has an outstanding average value of trial runs but the standard deviation is also large, it

Table 9 Ablation experiments on 10-D CEC2020 benchmark functions

Func.		RIME	RIME$_{lhs}$	RIME$_{mhs}$	RIME$_{edbs}$	SRIME
f_1	Mean	3.86e+05 +	6.08e+05 +	6.31e+05 +	9.29e+04 ≈	**6.93e+04**
	Std	4.57e+05	8.42e+05	9.81e+05	1.39e+05	1.17e+05
f_2	Mean	2.72e+07 +	2.90e+07 +	3.97e+07 +	1.28e+07 +	**6.25e+06**
	Std	2.44e+07	2.89e+07	4.93e+07	2.40e+07	1.01e+07
f_3	Mean	1.18e+07 +	1.53e+07 +	1.43e+07 +	6.22e+06 +	**3.13e+06**
	Std	1.23e+07	2.00e+07	1.54e+07	7.97e+06	3.53e+06
f_4	Mean	1.90e+03 ≈	1.90e+03 ≈	1.90e+03 ≈	1.90e+03 ≈	**1.90e+03**
	Std	9.00e−01	5.89e−01	5.71e−01	7.13e−01	6.29e−01
f_5	Mean	**1.19e+04** ≈	1.37e+04 ≈	1.42e+04 ≈	1.58e+04 ≈	1.22e+04
	Std	7.46e+03	7.73e+03	8.44e+03	8.71e+03	8.46e+03
f_6	Mean	3.27e+03 ≈	2.72e+03 ≈	2.86e+03 ≈	2.72e+03 ≈	**2.25e+03**
	Std	2.52e+03	1.91e+03	2.11e+03	1.58e+03	1.42e+03
f_7	Mean	**1.83e+04** ≈	2.01e+04 ≈	2.10e+04 ≈	2.28e+04 ≈	2.13e+04
	Std	1.04e+04	1.48e+04	1.28e+04	1.31e+04	1.61e+04
f_8	Mean	2.30e+03 +	2.30e+03 +	2.30e+03 +	2.30e+03 +	**2.30e+03**
	Std	2.15e+00	1.66e+00	2.46e+00	1.68e+00	1.43e+00
f_9	Mean	2.63e+03 +	2.63e+03 +	2.62e+03 +	2.61e+03 +	**2.59e+03**
	Std	4.07e+01	4.77e+01	2.85e+01	5.35e+01	3.38e+01
f_{10}	Mean	2.99e+03 ≈	3.00e+03 ≈	**2.99e+03** ≈	3.00e+03 ≈	2.99e+03
	Std	2.15e+01	2.72e+01	1.15e+01	2.48e+01	1.99e+01
+/ ≈ /− summary:		5/5/0	5/5/0	5/5/0	4/6/0	
Ave. rank		3.3	3.5	3.6	3.0	**1.6**

Table 10 Ablation experiments on 30-D CEC2020 benchmark functions

Func.		RIME	RIME$_{lhs}$	RIME$_{mhs}$	RIME$_{edbs}$	SRIME
f_1	Mean	2.56e+07 +	2.52e+07 +	2.54e+07 +	2.46e+06 ≈	**1.70e+06**
	Std	1.06e+07	8.72e+06	1.12e+07	1.90e+06	1.55e+06
f_2	Mean	2.88e+09 +	2.65e+09 +	2.75e+09 +	2.62e+08 ≈	**1.84e+08**
	Std	1.16e+09	1.21e+09	1.12e+09	2.95e+08	2.12e+08
f_3	Mean	8.73e+08 +	9.36e+08 +	1.00e+09 +	**7.85e+07** ≈	9.07e+07
	Std	4.15e+08	5.00e+08	4.47e+08	6.70e+07	1.24e+08
f_4	Mean	1.92e+03 +	1.92e+03 +	1.92e+03 +	1.92e+03 +	**1.91e+03**
	Std	3.54e+00	4.67e+00	4.76e+00	4.74e+00	2.57e+00
f_5	Mean	9.33e+05 ≈	7.11e+05 ≈	1.02e+06 ≈	7.61e+05 ≈	**6.37e+05**
	Std	6.25e+05	4.44e+05	6.42e+05	4.35e+05	3.56e+05
f_6	Mean	4.87e+04 ≈	5.80e+04 +	3.87e+04 ≈	4.52e+04 ≈	**3.82e+04**
	Std	1.91e+04	2.48e+04	1.99e+04	2.32e+04	2.25e+04
f_7	Mean	1.40e+06 ≈	1.50e+06 ≈	1.24e+06 ≈	1.26e+06 ≈	**1.07e+06**
	Std	6.66e+05	8.47e+05	7.63e+05	6.78e+05	7.06e+05
f_8	Mean	2.40e+03 +	2.40e+03 +	2.40e+03 +	2.39e+03 ≈	**2.38e+03**
	Std	1.39e+01	1.06e+01	1.24e+01	8.83e+00	8.46e+00
f_9	Mean	2.97e+03 +	2.98e+03 +	2.98e+03 +	2.70e+03 ≈	**2.70e+03**
	Std	8.47e+01	9.23e+01	9.62e+01	9.31e+01	7.07e+01
f_{10}	Mean	2.95e+03 +	2.94e+03 +	2.96e+03 +	2.93e+03 ≈	**2.93e+03**
	Std	2.85e+01	2.17e+01	3.54e+01	1.07e+01	1.51e+01
+/ ≈ /− summary:		7/3/0	8/2/0	7/3/0	1/9/0	
Ave. rank		4.0	3.8	3.9	2.2	**1.1**

Table 11 Ablation experiments on 50-D CEC2020 benchmark functions

Func.		RIME	RIME$_{lhs}$	RIME$_{mhs}$	RIME$_{edbs}$	SRIME
f_1	Mean	2.22e+08 +	2.02e+08 +	2.12e+08 +	1.76e+07 ≈	**1.28e+07**
	Std	6.35e+07	5.67e+07	7.42e+07	1.64e+07	8.72e+06
f_2	Mean	2.63e+10 +	2.63e+10 +	2.50e+10 +	2.17e+09 +	**1.10e+09**
	Std	8.67e+09	8.48e+09	7.95e+09	1.13e+09	5.87e+08
f_3	Mean	8.56e+09 +	8.76e+09 +	7.37e+09 +	5.71e+08 +	**3.69e+08**
	Std	2.52e+09	2.32e+09	2.46e+09	3.62e+08	2.32e+08
f_4	Mean	1.95e+03 +	1.96e+03 +	1.96e+03 +	1.94e+03 +	**1.93e+03**
	Std	1.10e+01	1.21e+01	1.16e+01	8.01e+00	8.32e+00
f_5	Mean	6.89e+06 +	7.09e+06 +	6.45e+06 ≈	4.98e+06 ≈	**4.93e+06**
	Std	2.84e+06	2.88e+06	3.09e+06	1.95e+06	2.44e+06
f_6	Mean	4.49e+05 +	3.87e+05 +	4.09e+05 +	**6.93e+04** ≈	7.91e+04
	Std	4.49e+05	3.69e+05	3.98e+05	2.76e+04	5.97e+04
f_7	Mean	8.38e+06 ≈	9.62e+06 +	9.54e+06 +	8.46e+06 ≈	**6.14e+06**
	Std	4.46e+06	4.32e+06	3.87e+06	4.02e+06	2.69e+06
f_8	Mean	2.52e+03 −	**2.52e+03** −	2.71e+03 +	2.62e+03 ≈	2.63e+03
	Std	2.33e+01	2.61e+01	1.01e+03	7.77e+02	8.87e+02
f_9	Mean	3.85e+03 +	3.83e+03 +	3.81e+03 +	**2.96e+03** ≈	2.96e+03
	Std	2.26e+02	1.93e+02	2.34e+02	1.91e+02	2.41e+02
f_{10}	Mean	3.71e+03 +	3.56e+03 +	3.70e+03 +	3.46e+03 +	**3.41e+03**
	Std	1.85e+02	1.42e+02	1.87e+02	9.75e+01	1.00e+02
+/ ≈ /− summary:		8/1/1	9/0/1	9/1/0	4/6/0	
Ave. rank		4.0	3.8	3.7	2.0	**1.5**

Table 12 Ablation experiments on 100-D CEC2020 benchmark functions

Func.		RIME	RIME$_{lhs}$	RIME$_{mhs}$	RIME$_{edbs}$	SRIME
f_1	Mean	3.19e+09 +	2.97e+09 +	3.00e+09 +	1.85e+08 ≈	**1.48e+08**
	Std	6.62e+08	6.92e+08	9.12e+08	9.44e+07	7.70e+07
f_2	Mean	2.96e+11 +	2.97e+11 +	2.80e+11 +	2.08e+10 +	**1.32e+10**
	Std	6.43e+10	5.92e+10	6.21e+10	8.75e+09	7.51e+09
f_3	Mean	1.08e+11 +	1.06e+11 +	1.10e+11 +	8.41e+09 +	**5.66e+09**
	Std	2.26e+10	1.92e+10	2.58e+10	4.21e+09	3.17e+09
f_4	Mean	2.15e+03 +	2.16e+03 +	2.18e+03 +	**2.06e+03** ≈	2.08e+03
	Std	8.49e+01	9.47e+01	1.21e+02	2.59e+01	4.84e+01
f_5	Mean	6.28e+07 +	7.00e+07 +	7.58e+07 +	3.86e+07 ≈	**3.37e+07**
	Std	2.08e+07	2.32e+07	2.40e+07	8.23e+06	9.38e+06
f_6	Mean	4.35e+05 ≈	6.24e+05 ≈	7.27e+05 ≈	**2.33e+05** ≈	3.42e+05
	Std	2.83e+05	7.00e+05	7.98e+05	8.90e+04	2.66e+05
f_7	Mean	5.42e+07 +	5.96e+07 +	5.56e+07 +	2.39e+07 ≈	**2.32e+07**
	Std	2.02e+07	2.43e+07	2.14e+07	1.09e+07	7.00e+06
f_8	Mean	**2.70e+03** −	3.28e+03 +	4.37e+03 +	3.23e+03 ≈	3.26e+03
	Std	5.94e+01	3.19e+03	5.02e+03	2.50e+03	2.69e+03
f_9	Mean	1.34e+04 +	1.31e+04 +	1.26e+04 +	5.02e+03 +	**4.65e+03**
	Std	1.17e+03	1.53e+03	1.36e+03	6.17e+02	6.25e+02
f_{10}	Mean	4.14e+03 +	4.10e+03 +	4.14e+03 +	3.68e+03 +	**3.55e+03**
	Std	1.65e+02	1.79e+02	1.98e+02	1.04e+02	6.87e+01
+/ ≈ /− summary:		8/1/1	9/1/0	9/1/0	4/6/0	
Ave. rank		3.5	3.9	4.4	1.8	**1.4**

cannot be regarded as a successful approach in real-world scenarios, since the real-world optimization problem is usually computationally expensive, and multiple trial runs may not be afforded due to the limitation of computational resources. Table 8 shows the results of eight engineering optimization problems including mean, std, best, worst, and statistical tests, which can fully reflect the performance of our proposed SRIME and compared algorithm. From these results, we summarize that our proposed SRIME is competitive with competitor algorithms, and the optimization results in the speed reducer problem (SRD), tubular column problem (TCD), piston lever problem (PLD), and reinforced concrete beam problem (RCB) reveal the excellent global search capacity and stability of SRIME. Additionally, the metric of average rank reveals the competitiveness of SRIME with state-of-the-art EAs such as CMA-ES. However, the inferiority of SRIME can be observed in some specific tasks such as in the welded beam problem (WBP) and cantilever beam problem (CBD) compared with CMA-ES. Similarly, the NFLT can also be used to explain this degeneration reasonably.

5.3 Performance analysis based on ablation experiments

The ablation experiments are implemented to evaluate our proposed three strategies independently. From the experimental and statistical results in Tables 9, 10, 11 and 12, the combination of three strategies can accelerate the convergence of optimization in most cases. Based on the summarized statistical results, the embedded distance-based selection plays the most important role in contributing to the improvement.

5.4 Open topics for future research

The above experiments and analysis reveal the efficiency and scalability of our proposed SRIME. However, there are still some improvements that can work on SRIME. Here, we list some open topics for future research.

Hybridize with advanced search techniques: One topic to further develop SRIME is to hybridize it with advanced EAs, and many works on hybridizing EA techniques have been reported: Gao et al. [42] hybridized the gray wolf optimization (GWO) and slime mould algorithm (SMA) to deal with numerical simulation optimization problems. Singh et al. [43] proposed a hybrid slime mould algorithm (SMA) and naked mole-rat algorithm (NMRA) to solve the above-said problem of the wireless sensor network. Fadheel et al. [44] proposed a hybrid sparrow search algorithm with gray wolf optimizer (SSAGWO) for frequency control of the RE integrated system. Xie et al. [45] proposed a hybrid Henry gas solubility optimization algorithm based on the Harris hawk optimization (HHO-HGSO) to deal with real engineering design problems. Thus, it is promising to hybridize our proposed RIME with some state-of-the-art algorithms.

Extend to various optimization tasks: SRIME is an efficient and powerful optimization technique that has been fully proven by our numerical experiments in practice, thus, it is necessary to extend SRIME to other problems such as large-scale global optimization, multi- and many-objective optimization, traveling salesman problems, feature selection tasks, neural architecture search tasks, and real-world applications [46].

6 Conclusion

This paper focuses on improving the overall optimization performance of RIME and proposes a strengthened RIME (SRIME). Three search strategies are introduced to the original RIME algorithm: (1) Latin hypercube initialization, (2) modified hard rime search scheme, and (3) embedded distance-based selection mechanism. To evaluate the performance of our proposed SRIME, we test it in CEC2020 benchmark functions and eight engineering optimization problems, the experimental and statistical results reveal the efficiency and effectiveness of our proposed SRIME. Ablation experiments are also provided to analyze the performance of each component in our proposed three strategies.

At the end of this paper, we list some open topics for future research. In the future, we will focus on further improving the performance of SRIME and extending it to other optimization tasks.

Acknowledgements This work was supported by JSPS KAKENHI Grant Number JP20K11967, 21A402, and JST SPRING Grant Number JPMJSP2119.

Author contributions RZ: Conceptualization, Methodology, Investigation, Writing—original draft, Writing—review & editing, and Funding acquisition. JY: Investigation, Methodology, Formal Analysis, and Writing—review & editing. CZ: Conceptualization and Writing—review & editing. MM: Writing—review & editing, and Project administration.

Data availability This research code can be downloaded from https://github.com/RuiZhong961230/SRIME.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Azizi M (2021) Atomic orbital search: a novel metaheuristic algorithm. Appl Math Model 93:657–683. https://doi.org/10.1016/j.apm.2020.12.021
2. Rana N, Latiff MSA, Abdulhamid SM, Misra S (2022) A hybrid whale optimization algorithm with differential evolution optimization for multi-objective virtual machine scheduling in cloud computing. Eng Optim 54(12):1999–2016. https://doi.org/10.1080/0305215X.2021.1969560
3. Zhong R, Zhang E, Munetomo M (2023) Cooperative coevolutionary surrogate ensemble-assisted differential evolution with efficient dual differential grouping for large-scale expensive optimization problems. Complex Intell Syst. https://doi.org/10.1007/s40747-023-01262-6
4. Zhong R, Zhang E, Munetomo M (2023) Cooperative coevolutionary differential evolution with linkage measurement minimization for large-scale optimization problems in noisy environments. Complex Intell Syst 9:4439–4456. https://doi.org/10.1007/s40747-022-00957-6
5. Neggaz N, Houssein EH, Hussain K (2020) An efficient henry gas solubility optimization for feature selection. Expert Syst Appl 152:113364. https://doi.org/10.1016/j.eswa.2020.113364
6. Zhong R, Peng F, Zhang E, Yu J, Munetomo M (2023) Vegetation evolution with dynamic maturity strategy and diverse mutation strategy for solving optimization problems. Biomimetics 8(6):454. https://doi.org/10.3390/biomimetics8060454
7. Deng L, Liu S (2023) Snow ablation optimizer: a novel metaheuristic technique for numerical optimization and engineering design. Expert Syst Appl 225:120069. https://doi.org/10.1016/j.eswa.2023.120069
8. De Jong K (1988) Learning with genetic algorithms: an overview. Mach Learn 3:121–138. https://doi.org/10.1007/BF00113894
9. Storn R, Price K (1997) Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces. J Glob Optim 11:341–359. https://doi.org/10.1023/A:1008202821328
10. Koza JR (1994) Genetic programming as a means for programming computers by natural selection. Stat Comput 4:87–112. https://doi.org/10.1007/BF00175355
11. Beyer H-G, Schwefel H-P (2002) Evolution strategies: a comprehensive introduction. Nat Comput 1:3–52. https://doi.org/10.1023/A:1015059928466
12. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95—international conference on neural networks, vol 4, pp 1942–19484. https://doi.org/10.1109/ICNN.1995.488968
13. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput Intell Mag 1(4):28–39. https://doi.org/10.1109/MCI.2006.329691
14. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
15. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
16. Reynolds R (1994) An introduction to cultural algorithms. In: Evolutionary programming—proceedings of the third annual conference, pp 131–139. https://doi.org/10.1142/9789814534116
17. Acharya D, Das D (2022) A novel human conception optimizer for solving optimization problems. Sci Rep 12:21631. https://doi.org/10.1038/s41598-022-25031-6
18. Rahkar Farshi T (2021) Battle royale optimization algorithm. Neural Comput Appl 33:1139–1157. https://doi.org/10.1007/s00521-020-05004-4

19. Shi Y (2011) Brain storm optimization algorithm. In: Tan Y, Shi Y, Chai Y, Wang G (eds) Advances in swarm intelligence. Springer, Berlin, Heidelberg, pp 303–309. https://doi.org/10.1007/978-3-642-21515-5_36
20. Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133. https://doi.org/10.1016/j.knosys.2015.12.022
21. Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680. https://doi.org/10.1126/science.220.4598.671
22. Lam A, Li V (2012) Chemical reaction optimization: a tutorial. Memetic Comput 4:3–17. https://doi.org/10.1007/s12293-012-0075-1
23. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82. https://doi.org/10.1109/4235.585893
24. Su H, Zhao D, Heidari AA, Liu L, Zhang X, Mafarja M, Chen H (2023) RIME: a physics-based optimization. Neurocomputing 532:183–214. https://doi.org/10.1016/j.neucom.2023.02.010
25. Stein M (1987) Large sample properties of simulations using Latin hypercube sampling. Technometrics 29(2):143–151. https://doi.org/10.1080/00401706.1987.10488205
26. Tharwat A, Schenck W (2021) Population initialization techniques for evolutionary algorithms for single-objective constrained optimization problems: deterministic vs. stochastic techniques. Swarm Evol Comput 67:100952. https://doi.org/10.1016/j.swevo.2021.100952
27. Tsang KKT (2018) Basin of attraction as a measure of robustness of an optimization algorithm. In: 2018 14th international conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD), pp 133–137. https://doi.org/10.1109/FSKD.2018.8686850
28. Ghosh A, Das S, Mallipeddi R, Das AK, Dash SS (2017) A modified differential evolution with distance-based selection for continuous optimization in presence of noise. IEEE Access 5:26944–26964
29. Bouamama S, Jlifi B, Ghedira K (2003) D2g2a: a distributed double guided genetic algorithm for max_csps, vol 2773, pp 422–429. https://doi.org/10.1007/978-3-540-45224-9_59
30. Van Thieu N, Mirjalili S (2023) MEALPY: an open-source library for latest meta-heuristic algorithms in Python. J Syst Archit 139:102871. https://doi.org/10.1016/j.sysarc.2023.102871
31. Nguyen T (2020) A framework of optimization functions using Numpy (OpFuNu) for optimization problems. Zenodo. https://doi.org/10.5281/zenodo.3620960
32. Thieu NV (2023) ENOPPY: a Python library for engineering optimization problems. Zenodo. https://doi.org/10.5281/zenodo.7953206
33. Yue CT, Price PNSKV (2020) Problem definitions and evaluation criteria for the CEC 2020 special session and competition on single objective bound constrained numerical optimization. In: Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore
34. Ezugwu A, Agushaka O, Abualigah L, Mirjalili S, Gandomi A (2022) Prairie dog optimization algorithm. Neural Comput Appl 34:20017–20065. https://doi.org/10.1007/s00521-022-07530-9
35. Hansen N, Ostermeier A (2001) Completely derandomized self-adaptation in evolution strategies. Evol Comput 9(2):159–195. https://doi.org/10.1162/106365601750190398
36. Zhao W, Wang L, Zhang Z (2020) Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm. Neural Comput Appl 32:1–43. https://doi.org/10.1007/s00521-019-04452-x
37. Trojovská E, Dehghani M, Trojovský P (2022) Zebra optimization algorithm: a new bio-inspired optimization algorithm for solving optimization algorithm. IEEE Access 10:49445–49473. https://doi.org/10.1109/ACCESS.2022.3172789
38. Ayyarao TSLV, Ramakrishna NSS, Elavarasan RM, Polumahanthi N, Rambabu M, Saini G, Khan B, Alatas B (2022) War strategy optimization algorithm: a new effective metaheuristic algorithm for global optimization. IEEE Access 10:25073–25105. https://doi.org/10.1109/ACCESS.2022.3153493
39. Azizi M, Aickelin U, Khorshidi H, Baghalzadeh Shishehgarkhaneh M (2023) Energy valley optimizer: a novel metaheuristic algorithm for global and engineering optimization. Sci Rep 13:226. https://doi.org/10.1038/s41598-022-27344-y
40. Coello Coello CA (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Comput Methods Appl Mech Eng 191(11):1245–1287. https://doi.org/10.1016/S0045-7825(01)00323-1
41. Holm S (1979) A simple sequentially rejective multiple test procedure. Scand J Stat 6(2):65–70
42. Gao Z-M, Zhao J, Yang Y, Tian X-J (2020) The hybrid grey wolf optimization-slime mould algorithm. J Phys: Conf Ser 1617(1):012034. https://doi.org/10.1088/1742-6596/1617/1/012034
43. Singh S, Singh U (2023) A novel self-adaptive hybrid slime mould naked mole-rat algorithm for numerical optimization and energy-efficient wireless sensor network. Concurr Comput Pract Exp 35:e7809. https://doi.org/10.1002/cpe.7809
44. Fadheel BA, Wahab NIA, Mahdi AJ, Premkumar M, Radzi MABM, Soh ABC, Veerasamy V, Irudayaraj AXR (2023) A hybrid grey wolf assisted-sparrow search algorithm for frequency control of RE integrated system. Energies 16(3):1177. https://doi.org/10.3390/en16031177
45. Xie W, Xing C, Wang J, Guo S, Guo M-W, Zhu L-F (2020) Hybrid Henry gas solubility optimization algorithm based on the Harris Hawk optimization. IEEE Access 8:144665–144692. https://doi.org/10.1109/ACCESS.2020.3014309
46. Zhong R, Peng F, Yu J, Munetomo M (2024) Q-learning based vegetation evolution for numerical optimization and wireless sensor network coverage optimization. Alex Eng J 87:148–163. https://doi.org/10.1016/j.aej.2023.12.028