A Surrogate-Assisted Differential Evolution With Knowledge Transfer for Expensive Incremental Optimization Problems

Yuanchao Liu, Jianchang Liu, Jinliang Ding, *Senior Member, IEEE*, Shangshang Yang, *Member, IEEE*, and Yaochu Jin, *Fellow, IEEE*

***Abstract*—In some real-world applications, the optimization problems may involve multiple design stages. At each design stage, the objective is incrementally modified by incorporating more decision variables and optimized. In addition, the fitness evaluations (FEs) are often highly costly. Such optimization problems can be called expensive incremental optimization problems (EIOPs). Despite their importance, EIOPs have not attracted much attention over the past few years. Since the objectives of different design stages are different but related, reusing the search experience from the past design stages is beneficial to the evolutionary search of the current design stage. Therefore, a surrogate-assisted differential evolution with knowledge transfer (SADE-KT) is proposed in this work, which aims to fill the current gap in solving EIOPs. The major merit of the proposed SADE-KT is its ability to seamlessly integrate knowledge transfer and the surrogate-assisted evolutionary search. In SADE-KT, a surrogate-based hybrid knowledge transfer strategy is first proposed. This strategy makes it possible to reuse the knowledge captured from the past design stages by leveraging different knowledge transfer techniques. As a result, the convergence for the current design stage can be speeded up. Then, a two-level surrogate-assisted evolutionary search is developed to search for the optimum. Comprehensive empirical studies have demonstrated that the proposed algorithm works efficiently on EIOPs.**

***Index Terms*—Differential evolution (DE), expensive incremental optimization problems (EIOPs), knowledge transfer, surrogate-assisted evolutionary algorithm (SAEA).**

Manuscript received 18 March 2023; revised 22 May 2023; accepted 24 June 2023. Date of publication 4 July 2023; date of current version 1 August 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62273080; in part by the National Key Research and Development Plan Project under Grant 2022YFB3304700; in part by the National Natural Science Foundation of China under Grant 61988101; in part by the 111 Project 2.0 under Grant B08015; and in part by the Liaoning Province Central Leading Local Science and Technology Development Special Project under Grant 2022JH6/100100055. The work of Yaochu Jin was supported by the Alexander von Humboldt Professorship for Artificial Intelligence Endowed by the German Federal Ministry of Education and Research. *(Corresponding authors: Jianchang Liu; Yaochu Jin.)*

Yuanchao Liu and Jianchang Liu are with the State Key Laboratory of Synthetical Automation for Process Industries, and the College of Information Science and Engineering, Northeastern University, Shenyang 110004, China (e-mail: Yuanchaoliu@126.com; liujianchang@ise.neu.edu.cn).

Jinliang Ding is with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110004, China (e-mail: jlding@mail.neu.edu.cn).

Shangshang Yang is with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Artificial Intelligence, Anhui University, Hefei 230039, China (e-mail: yangshang0308@gmail.com).

Yaochu Jin is with the Chair of Nature Inspired Computing and Engineering, Faculty of Technology, Bielefeld University, 33619 Bielefeld, Germany, and also with the State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110004, China (e-mail: yaochu.jin@uni-bielefeld.de).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TEVC.2023.3291697.

Digital Object Identifier 10.1109/TEVC.2023.3291697

I. Introduction

In EXPENSIVE optimization, it can be challenging to formulate an explicit mathematical expression for the fitness evaluation (FE) [1], [2]. Moreover, the FEs may require time-consuming simulations or expensive physical experiments, such as computational fluid dynamics (CFDs) [3], crash simulations [4], and finite element analysis (FEA) [5]. Therefore, a limited number of FEs can be afforded for the algorithm when dealing with expensive optimization.

Despite the success of evolutionary algorithms (EAs) in solving diverse complex optimization problems [6], [7], they may not work well for expensive optimization. Because EAs typically rely on the assumption of having access to plenty of FEs in the search process. To effectively address expensive optimization, most of the existing work incorporates the computationally efficient surrogate models into the framework of EAs [8], [9]. That is, surrogate-assisted EAs (SAEAs) have been widely regarded as one of the most popular approaches for solving expensive optimization over the past decades.

Recently, a variety of SAEAs have been proposed for different kinds of expensive optimization in the literature, including expensive single-objective optimization [10], [11], [12], expensive multi/many-objective optimization [13], [14], [15], and expensive multitask optimization [16], just to name a few. Note that there is only one objective at each design stage for expensive incremental optimization problems (EIOPs). Therefore, the following discussions are limited to expensive optimization with single objective. In general, single-objective SAEAs can be roughly divided into two categories.

In the first category, Kriging-based infill criteria are applied in the SAEAs. For example, an efficient global optimization was proposed in [10], where an expected improvement (EI) infill criterion is developed. A multimodal EI infill criterion was proposed in [17], which can select multiple promising points at each cycle. Zhan et al. [18] developed a fast multipoint EI infill criterion for parallel expensive optimization. In addition, the lower confidence bound (LCB) infill criterion [19] and the probability of improvement (PoI) infill criterion [20] are also widely used. Moreover, for expensive constrained optimization, a constrained EI infill criterion was suggested in [21]. Durantin et al. [22] studied that the algorithm can obtain a competitive performance by simultaneously considering EI, probability of feasibility and prediction variance of constraints. In [23], a probability of constrained improvement criterion was developed, which considers both the objective and constraints. Dong et al. [24] introduced a multistart knowledge mining-based infilling criterion for the discrete design domain. Recently, a multitask Gaussian process-based constrained EI was developed for expensive highly constrained optimization [25]. In addition, some efforts have been made to enhance the scalability of Kriging-based infill criteria to the dimension of the decision space. For instance, the LCB infill criterion with dimension reduction was applied to solve medium-scale expensive optimization [26]. Li et al. [27] incorporated a variable dropout technique into the Bayesian optimization for high-dimensional expensive optimization. A multiobjective infill criterion was proposed in [28], which considers the predicted fitness and the uncertainty information. To extent Kriging to the high-dimensional space, an incremental Kriging model was developed [29], and therefore Kriging-based infill criteria can be employed. In addition to the above deterministic optimization problems, some work has been done for optimization problems with uncertain environments, such as noise optimization [30] and robust optimization [31].

The second category of SAEAs employs heuristic surrogate management strategies. For instance, based on neural networks, an individual-based evolution control and a generation-based evolution control were introduced [11]. In [12], an ensemble model-assisted memetic algorithm was developed for the local search stage. A novel fitness estimation strategy was proposed to assist the search process of particle swarm optimizer (PSO) [32]. Ji et al. [33] introduced several useful strategies into SAEAs for finding multiple optima. For high-dimensional expensive optimization, some researchers have combined surrogate models and swarm intelligence optimization algorithms [34], [35], [36], while some SAEAs introduced evolutionary sampling strategies [37], [38]. Moreover, Cui et al. [39] incorporated surrogate models into the autoencoder-embedded evolutionary optimization framework. In addition, some efforts have been done for large-scale expensive optimization. For example, a fitness approximation strategy was proposed based on the updating rule in competitive swarm optimizer [40]. In [41], a progressive sampling strategy and social learning PSO were cooperated. Furthermore, Regis [42] developed a radial basis function (RBF) model-assisted $(\mu+\mu)$-evolutionary programming, which can perform well on expensive constrained optimization. In [43], a surrogate-assisted partial-evaluation strategy was suggested to select the constraints for real FEs. An expensive constrained optimization problem was decomposed into several subproblems in [44], each of which was solved by a surrogate-assisted differential evolution (DE). An on-demand evaluation technique was employed to address distributed expensive constrained optimization [45]. In addition, some existing work has aimed to deal with mixed-variable expensive optimization [46] and expensive dynamic optimization [47], [48].

There are also some SAEAs that combine Kriging-based infill criteria and heuristic surrogate management strategies. For example, Zhou et al. [49] employed the PoI infill criterion for the global search stage, and adopted the RBF model to assist the local search stage. A simplified Kriging-based EI infill criterion and a surrogate-based trust region local search were cooperated for high-dimensional expensive optimization [50]. Wang et al. [51] built the RBF model to assist the global search phase, while the LCB infill criterion was introduced for the local search phase. In addition, inspired by the multitasking optimization framework, Ding et al. [52] transferred the knowledge from computationally cheap tasks to the expensive task.

According to the above literature review, one can find that SAEAs have achieved great successes over the past few years. However, to the best of our knowledge, most existing SAEAs mainly focus on expensive optimization with a fixed number of decision variables. In other words, little work has been done for EIOPs. In EIOPs, there are multiple design stages. At each design stage, the objective is incrementally modified by adding more decision variables and optimized. Furthermore, the FEs at each design stage are highly costly. Therefore, the main purpose for solving EIOPs is to locate the optimum at each design stage within a limited number of FEs.

In fact, EIOPs are commonly found in real-world applications. For instance, in truss structure optimization, the topology structure of truss can be incrementally optimized by adding nodes and bars [53], where the FEA simulation is required for FEs. Another practical scenario is the aerodynamic shape optimization, where the aerodynamic shape can be incrementally optimized by adding design variables [54], [55]. In addition, the CFD simulation is required for FEs. Therefore, it is of great practical significance and necessity to study how to effectively deal with EIOPs.

Taking the above considerations, this work attempts to fill the current gap in solving EIOPs by proposing a surrogate-assisted DE with knowledge transfer (SADE-KT). The main contributions of this work can be summarized as follows.

1) A surrogate-based hybrid knowledge transfer strategy (SHKTS) is proposed. This strategy can transfer the optima found in the past design stages to the current design stage by leveraging different knowledge transfer techniques. Additionally, the promising points are selected from the transferred points for real FEs. Finally, the exactly evaluated promising points will be injected into the current population. Therefore, on the one hand, the convergence for the current design stage can be speeded up. On the other hand, negative knowledge transfer can be reduced to some extent by leveraging different knowledge transfer techniques.

2) A two-level surrogate-assisted evolutionary search is developed to search for the optimum, where two coordinate systems are adaptively tuned to enhance the search ability on different function landscapes.
3) A novel SAEA, termed SADE-KT, is developed on the basis of the above strategies. The major advantage of the proposed SADE-KT is the seamless integration of knowledge transfer and the surrogate-assisted evolutionary search.

The remainder of this article is structured as follows. Some preliminaries and the motivation of this work are given in Section II. Section III presents the details of the proposed SADE-KT. Extensive empirical studies are performed and discussed in Section IV. Finally, Section V provides the conclusions and future work.

II. Preliminaries and Motivation

In this section, some relevant preliminaries are first given, followed by the motivation of this work.

A. Expensive Incremental Optimization Problems

Similarly with the incremental optimization problems in [56], an EIOP with T design stages can be formulated as follows:

$$\begin{aligned} &\min\ f^t(\mathbf{x}^t) \\ &\text{s.t.}\ \ \mathbf{x}^t = (x_1, x_2, \ldots, x_{d^t}) \in \mathbf{D}^t \end{aligned} \tag{1}$$

where $t = 1, \ldots, T$, $f^t(\mathbf{x}^t)$ denotes the objective to be optimized at the t-th design stage, d^t is the number of decision variables at the t-th design stage, and $\mathbf{x}^t$ and $\mathbf{D}^t$ are a vector of decision variables and the decision space at the t-th design stage, respectively.

In EIOPs, $f^t(\mathbf{x}^t)$ is incrementally modified by adding more decision variables. Therefore, given the t-th design stage, the number of decision variables d^t can be calculated as follows:

$$d^t = d^{t-1} + \Delta d^t \tag{2}$$

where $t \geq 2$, d^{t-1} is the number of decision variables at the $(t-1)$th design stage, and Δd^t is the number of decision variables newly added to the t-th design stage.

In addition, the FEs for the objective are highly costly at each design stage. Thus, the main aim of dealing with EIOPs is to find the optimum at each design stage within a limited number of FEs.

EIOPs can be considered a type of expensive dynamic optimization problems, as both the decision space and the objective function are changed at adjacent stages. In recent years, some work has been done for expensive dynamic optimization. For instance, Yang et al. [57] proposed a knowledge transfer-based surrogate model adaptation for the offline data-driven optimization problems in nonstationary systems. Luo et al. [47] investigated the performance of different SAEAs in dealing with data-driven dynamic optimization. The multipopulation technique is applied to solve expensive dynamic optimization in [48] and [58]. Chen and Li [59] proposed a transfer Bayesian optimization, where the exactly evaluated points from the pervious environments are used to constructed the Gaussian process model for the current environment. Moreover, some SAEAs have been proposed for expensive dynamic multiobjective optimization problems (MOPs) [60], [61], [62]. Nevertheless, little research has been dedicated to EIOPs.

B. Knowledge Transfer in Evolutionary Optimization

In recent years, several studies have shown that knowledge transfer is able to enhance the search efficiency of EAs [63], [64]. For example, Gupta et al. [7] introduced evolutionary multitasking, where the knowledge can be transferred among different tasks. Yang et al. [65] suggested that the knowledge from the assistant models can be transferred into the accurate model in dealing with multitasking MOPs. In [66] and [67], the knowledge from the source tasks can be projected to the target task by a mapping matrix. To deal with dynamic MOPs, the objectives from the previous environment and the current environment are mapped into a latent space, and an initial population can be generated in the latent space for the current environment [68], [69].

In addition, knowledge transfer has been used to transfer the knowledge across heterogeneous objectives in some work. For instance, a single-layer denoising autoencoder is introduced to map the population from the past problems into the current problem [70]. In [71], a kernelized autoencoder is constructed to capture the nonlinearity among different problems. Xue et al. [72] adopted the convergence transfer technique and the diversity transfer technique to transfer the knowledge from convergence-related variables and diversity-related variables, respectively. The solutions from the cheap objective are transferred into the expensive objective in [73].

C. Motivation

A naive strategy for solving EIOPs is to independently optimize the objective at each design stage. However, this naive strategy can lead to a low search efficiency for the algorithm, as it fails to reuse the knowledge from the past design stages. According to the definition of EIOPs, the objective in the past design stages and the objective in the current design stage are different but related. Thus, it becomes possible to reuse the knowledge captured from the past design stages to enhance the search efficiency of the algorithm at the current design stage.

In EIOPs, the following two cases are existed at the t-th design stage [56], where $t \geq 2$.

1) The newly added decision variables at the t-th design stage do not interact with the previous decision variables.
2) The newly added decision variables at the t-th design stage interact with the previous decision variables.

In the first case, the optima found in the past design stages can be directly transferred to the t-th design stage, and we can search the newly added decision variables with the assistance of the surrogate model. As a result, several promising points, which contain the knowledge captured from the past design stages, can be obtained. These promising points are expected to speed up the convergence of the algorithm at the t-th design stage.

However, for the second case, directly reusing the optima found in the past design stages may result in negative knowledge transfer, which can mislead the search of the algorithm at the t-th design stage. Note that the objective in the past design stages and the objective at the t-th design stage are different but related. According to the pervious studies in [68], [70], and [72], domain adaptation can build the connections between the target optimization problem and the source optimization problems. Thus, one can consider projecting the optima found in the past design stages (which can be viewed as the source optimization problems) to the t-th design stage (which can be viewed as the target optimization problem) via domain adaptation for improving the search efficiency.

Due to the lack of prior knowledge, it is difficult to know the relationship between the newly added decision variables at the current design stage and the previous decision variables. Therefore, it can be challenging to adopt the most suitable transfer mechanism for the algorithm. However, if an improper transfer mechanism is used, it may influence the effect of knowledge transfer and the performance of algorithm. To this end, an SHKTS is proposed in this work. The main idea of the SHKTS is to reuse the knowledge captured from the past design stages by leveraging the reusing technique and domain adaptation. Thus, the SHKTS not only can improve the search efficiency of the current design stage but also can reduce negative knowledge transfer to some extent.

Moreover, the search technique should keep a good scalability, since the number of decision variables is gradually increased in EIOPs. The hierarchical surrogate-assisted framework performs well on expensive optimization with different dimensions [34], [35], [36], [37]. Thus, based on the hierarchical surrogate-assisted framework, a two-level surrogate-assisted evolutionary search is proposed to search for the optimum at each design stage. In the proposed search technique, the first-level search is employed for exploration, where the original coordinate system and the eigen coordinate system are adaptively tuned for enhancing the search ability on different function landscapes. The second-level search aims to exploit the promising local region. In addition, the RBF model is introduced as the surrogate model in this work. Because the RBF model can be used to approximate the regular and irregular surfaces with high efficiency and accuracy, and the performance of the RBF model is less sensitive to the number of decision variables for the approximated objective [74]. Furthermore, many studies have verified that the RBF model is well suited for expensive optimization, in particular for high-dimensional expensive optimization [34], [36], [38]. The details of the RBF model are given in Section I of the Supplemental material.

III. Proposed SADE-KT

In this section, SADE-KT is proposed, which aims to fill the current gap in solving EIOPs. This work adopts DE as the search algorithm. Because DE has a good global search ability, and many previous work has demonstrated the efficiency of DE in different optimization problems [75], [76], [77]. The details of DE can be found in Section II of the Supplemental material.

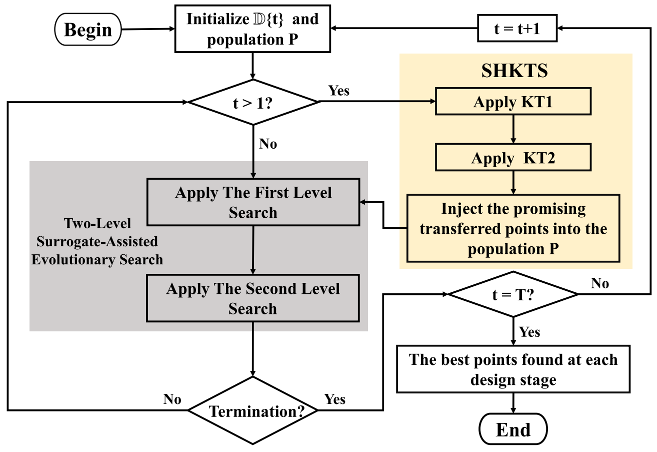

Begin
Initialize $\mathbb{D}\{t\}$ and population P
t = t+1
SHKTS
Apply KT1
Apply KT2
Inject the promising transferred points into the population P
t > 1?
Yes
No
Two-Level Surrogate-Assisted Evolutionary Search
Apply The First Level Search
Apply The Second Level Search
t = T?
No
Yes
The best points found at each design stage
Termination?
No
Yes
End

Fig. 1. Diagram of SADE-KT.

Algorithm 1 Framework of SADE-KT

```
Input: N (The population size), NI (The number of initial sampling points), λ (The number of trial offspring generated by each mutation strategy), T (The total number of design stages);
Output: $\mathbb{B}$ (The best points found at each design stage);
1:  for $t = 1 : T$ do
2:      Sample NI initial points PI by LHS in the decision space for real FEs;
3:      $\mathbb{D}\{t\} \leftarrow PI$;
4:      Select the best N points from PI as the search population $P = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$, and the fitness $f^t = \{f^t(\mathbf{x}_1), f^t(\mathbf{x}_2), \dots, f^t(\mathbf{x}_N)\}$;
5:      $g \leftarrow 0$;
6:      while The stopping condition of the $t$-th design stage is not met do
7:          if $t \geq 2$ then
8:              Apply the Surrogate Based Hybrid Knowledge Transfer Strategy;
9:          end if
10:         /*Two-Level Surrogate-Assisted Evolutionary Search*/
11:         Apply The First Level Search;
12:         Apply The Second Level Search;
13:         $g \leftarrow g + 1$;
14:     end while
15:     $\mathbb{B}(t) \leftarrow$ The best point in $\mathbb{D}\{t\}$;
16: end for
```

A. General Framework of SADE-KT

The pseudocode of the framework and diagram for SADE-KT are given in Algorithm 1 and Fig. 1, respectively. At each design stage t, NI initial points PI are first sampled in the decision space by Latin hypercube sampling (LHS) [78] for real FEs. These exactly evaluated initial points PI will be added into the database $\mathbb{D}\{t\}$. In addition, similar to [26], the best N points in PI are selected to form the search population

$P = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$. If the stopping condition of this design stage is not met, SADE-KT will enter into the search loop. In the search loop, an SHKTS is proposed to reuse the knowledge captured from the past design stages, when $t \geq 2$. Thus, the convergence of the population at the current design stage can be accelerated. Next, a two-level surrogate-assisted evolutionary search is developed to search for the decision space. To be specific, the first-level search applies multiple mutation strategies in DE for exploration. In addition, the original coordinate system and the eigen coordinate system are adaptively tuned for enhancing the search ability on different function landscapes. In the second-level search, a surrogate-based local search is employed for exploitation. As a result, a good balance between exploration and exploitation can be maintained.

In the following, the details of the two-level surrogate-assisted evolutionary search and the SHKTS will be described.

B. Two-Level Surrogate-Assisted Evolutionary Search

At each design stage, a two-level surrogate-assisted evolutionary search, which contains the first-level search and the second-level search, is performed to search for the optimum.

1) First-Level Search: The first-level search aims to explore the decision space. To maintain the exploration ability, it can be effective to employ multiple operators to generate candidates [38], [76], [77]. In addition, the eigen coordinate system can reflect the features of the function landscape to some extent by using the population distribution information [79], which is suited for different function landscapes. To this end, the first-level search applies multiple mutation strategies in DE to generate a set of promising candidates, and adaptively tunes the original coordinate system and the eigen coordinate system. The first-level search works as follows.

Step 1: If the first-level search is first conducted at the t-th design stage, the matrices and parameters for the eigen coordinate system are initialized.

Step 2: All points in $\mathbb{D}\{t\}$ are used to build an RBF model for the objective.

Step 3: **For** each point $\mathbf{x}_i$ in P, **do**

Step 3.1: Multiple mutation strategies in DE are used to generate several mutant vectors $\mathbf{v}_i$.

Step 3.2: If a random number sampled on the interval [0, 1] is no larger than p_i^g, the eigen coordinate system is used to generate offspring $\mathbf{u}_i$; otherwise, the original coordinate system is used to generate the offspring $\mathbf{u}_i$.

Step 3.3: The built RBF model is used to predict the fitness of each offspring in $\mathbf{u}_i$.

Step 3.4: Selecting the best offspring u_{best} in $\mathbf{u}_i$ based on the approximated fitness.

Step 3.5: u_{best} is exactly evaluated and added into $\mathbb{D}\{t\}$.

Step 3.6: If the exact fitness of u_{best} is better than that of $\mathbf{x}_i$, u_{best} replaces $\mathbf{x}_i$; otherwise, $\mathbf{x}_i$ is preserved.

End For

Step 4: The matrices and parameters for the eigen coordinate system are updated for the next iteration.

The following remarks on the first-level search can be made.

1) The eigen coordinate system is built based on the method proposed in ACoS, where ACoS is an adaptive framework for tuning the coordinate systems proposed in [79]. The details of ACoS can be found in Section III of the Supplemental material or [79].
2) In step 3.1, any mutation strategies in DE can be applied to generate mutant vectors. In this work, for simplicity and efficiency, DE/rand/1, DE/best/1, DE/current-to-rand/1, and DE/current-to-best/1 are employed, each of which generates λ mutant vectors. Therefore, $4 \times \lambda$ mutant vectors $\mathbf{v}_i = \mathbf{v}_{i,1} \cup \mathbf{v}_{i,2} \cup \mathbf{v}_{i,3} \cup \mathbf{v}_{i,4}$ for the $\mathbf{x}_i$ can be obtained, where $\mathbf{v}_{i,1} = \{v_{i,1}, v_{i,2}, \ldots, v_{i,\lambda}\}$, $\mathbf{v}_{i,2} = \{v_{i,\lambda+1}, v_{i,\lambda+2}, \ldots, v_{i,2\lambda}\}$, $\mathbf{v}_{i,3} = \{v_{i,2\lambda+1}, v_{i,2\lambda+2}, \ldots, v_{i,3\lambda}\}$, and $\mathbf{v}_{i,4} = \{v_{i,3\lambda+1}, v_{i,3\lambda+2}, \ldots, v_{i,4\lambda}\}$ are generated by the above four mutation strategies, respectively.
3) In step 3.2, when the eigen coordinate system is used, each offspring $u_{i,k}$ in $\mathbf{u}_i$ can be calculated as follows:

$$u_{i,k} = \mathbf{x}_i + \left(\mathbf{B}^g\right)\mathbf{M}\left(\mathbf{B}^g\right)^T\left(v_{i,k} - \mathbf{x}_i\right) \quad (3)$$

where $\mathbf{B}^g$ is the orthogonal matrix used for constructing the eigen coordinate system at the gth iteration, $v_{i,k}$ is the kth mutant vector in $\mathbf{v}_i$, $\mathbf{M} = \text{diag}(s_1, s_2, \ldots, s_{d^t})$, and

$$s_j = \begin{cases} 1, \text{ if } \left(\text{rand}_j \leq CR\right) \quad \text{or} \quad \left(j = j_{\text{rand}}\right) \\ 0, \text{ otherwise} \end{cases} \quad (4)$$

where $j = 1, 2, \ldots, d^t$, d^t is the number of decision variables at the t-th design stage, rand_j is randomly selected from [0, 1], j_{rand} is a random integer in $\{1, 2, \ldots, d^t\}$, and $CR \in [0, 1]$ denotes the crossover rate. When the original coordinate system is used, each offspring $u_{i,k}$ in $\mathbf{u}_i$ can be calculated as follows:

$$u_{i,k} = \mathbf{x}_i + \mathbf{M}\left(v_{i,k} - \mathbf{x}_i\right). \quad (5)$$

4) Similar to GPEME proposed in [26], the first-level search generates multiple offspring and selects the best one based on the surrogate model for real FE in each iteration.

The pseudocode of the first-level search is listed in Algorithm S2 in the Supplemental material.

2) Second-Level Search: After exploring the decision space, the algorithm will exploit the promising local region. Therefore, a surrogate-based local search is introduced in the second-level search, which works as follows.

Step 1: If the size of $\mathbb{D}\{t\}$ is less than τ, all points in $\mathbb{D}\{t\}$ are stored in an archive $\mathbb{P}$; otherwise, the points in $\mathbb{D}\{t\}$ whose fitness belongs to the first τ are stored in an archive $\mathbb{P}$.

Step 2: Building a local region as follows:

$$x_i \in [xl_i, xu_i], \quad i = 1, 2, \ldots, d^t \quad (6)$$

where xl_i and xu_i are the minimum and the maximum values of the ith decision variable of the points in $\mathbb{P}$, respectively, and d^t is the number of decision variables at the t-th design stage.

Step 3: All points in $\mathbb{P}$ are used to construct a local RBF model for the objective.

Step 4: Finding the optimum $\mathbf{x}_o$ of the local RBF model in the local region formed by (6).

Step 5: $\mathbf{x}_o$ is exactly evaluated and added into $\mathbb{D}\{t\}$.

Step 6: The exactly evaluated $\mathbf{x}_o$ is used to replace the worst point in P.

The following remarks on the surrogate-based local search can be made.

1) In step 1, the maximum number of points in $\mathbb{P}$ is limited to τ. All points in $\mathbb{P}$ will be used to construct a local RBF model in step 3. It is difficult to construct an accurate local RBF model when there is no enough training data. Previous studies in [36], [46], and [50] suggest that setting the maximum number of training data to $5 \times d^t$ is a good choice for the local search phase in SAEAs. Thus, we set τ to $5 \times d^t$ in this work.
2) In step 4, as recommended by [37], DE/best/1 is adopted to find $\mathbf{x}_o$.

The pseudocode of the second-level search is listed in Algorithm S3 in the Supplemental material.

C. Surrogate-Based Hybrid Knowledge Transfer Strategy

In SADE-KT, reusing the useful knowledge captured from the past design stages can improve the search efficiency at the t-th design stage, where the t-th design stage denotes the current design stage and $t \geq 2$. Therefore, an SHKTS is proposed in this work. In the SHKTS, two effective knowledge transfer techniques, namely, KT1 and KT2, are introduced. In KT1, the optima found in the past design stages will be directly transferred into the current design stage, and then searching the newly added decision variables with the assistance of the surrogate model. KT2 aims to adapt the knowledge captured from the past design stages to the current design stage. In addition, the promising points will be selected from the transferred points for real FEs. Finally, the exactly evaluated promising points are injected into the current population. Algorithm 2 gives the pseudocode of the SHKTS.

In the following, the details of each technique in the SHKTS will be described, namely, KT1 (lines 4–8 of Algorithm 2), KT2 (lines 10–16 of Algorithm 2), and injection of the transferred points (lines 24–26 of Algorithm 2).

1) KT1: KT1 aims to directly transfer the optima found in the past design stages to the t-th design stage. Note that all the optima found in the past design stages are stored in an archive $\mathbb{B}$. In other words, given a point $\mathbf{x}_j^b$ in $\mathbb{B}$, $\mathbf{x}_j^b$ is the optimum found at the jth design stage.

In KT1, we first use all the points in $\mathbb{D}\{t\}$ to build an RBF model, where $\mathbb{D}\{t\}$ stores all the exactly evaluated points at the t-th design stage. Then, given a point $\mathbf{x}_j^b = (x_{j,1}^b, x_{j,2}^b, \ldots, x_{j,d^j}^b)$ in $\mathbb{B}$, it will be inherited by the t-th design stage, where $j < t$ and d^j is the number of decision variables at the jth design stage. Thereafter, we will search for the remaining $d^t - d^j$ decision variables with the assistance of the built RBF model, where d^t is the number of decision variables at the t-th design stage. To this end, the following optimization problem can be defined:

$$\begin{aligned} &\min\ \hat{f}(\mathbf{x}^t) \\ &\text{s.t.}\ \mathbf{x}^t = \left(x_{j,1}^b, x_{j,2}^b, \ldots, x_{j,d^j}^b, x_{d^j+1}, x_{d^j+2}, \ldots, x_{d^t}\right) \\ &\qquad l_i \leq x_i \leq u_i, \qquad i = d^j+1, d^j+2, \ldots, d^t \end{aligned} \tag{7}$$

Algorithm 2 SHKTS

```
Input: $P$ (The population), $\mathbb{D}$ (All exactly evaluated points), $d^t$ (The number of decision variables at the $t$-th design stage), $\mathbb{B}$ (The best points found at each design stage);
Output: $P$ (The population), $\mathbb{D}$ (All exactly evaluated points);
1: $\mathbb{T} \leftarrow \emptyset$;
2: for $j = 1 : t-1$ do
3:     /*KT1*/
4:     Use all points in $\mathbb{D}\{t\}$ to build an RBF model for the objective, $\hat{f}(x)$;
5:     $\mathbf{x}_j^b \leftarrow \mathbb{B}(j)$, where $\mathbf{x}_j^b = (x_{j,1}^b, x_{j,2}^b, \ldots, x_{j,d^j}^b)$;
6:     Define the optimization problem by (7);
7:     Obtain the transferred point $T_j^1$ by optimizing the defined optimization problem;
8:     $\hat{f}(T_j^1) \leftarrow$ the approximated fitness of $T_j^1$;
9:     /*KT2*/
10:    $X_s = [x_1^s, x_2^s, \ldots, x_{n_s}^s] \leftarrow \mathbb{D}\{j\}$, $X_t = [x_1^t, x_2^t, \ldots, x_{n_t}^t] \leftarrow \mathbb{D}\{t\}$;
11:    Obtain $\bar{X}_s = [\bar{x}_1^s, \bar{x}_2^s, \ldots, \bar{x}_{n_s}^s]$ and $\bar{X}_t = [\bar{x}_1^t, \bar{x}_2^t, \ldots, \bar{x}_{n_t}^t]$ by (8);
12:    Obtain $\mathbf{P}_s$ and $\mathbf{P}_t$ by optimizing (15);
13:    Calculate the mapping matrix $\mathbf{M}$ by (16);
14:    Obtain the mapped point $\bar{T}_t^b$ by (17);
15:    Obtain the transferred point $T_j^2$ by standardizing $\bar{T}_t^b$ back to the scale of the $t$-th design space;
16:    $\hat{f}(T_j^2) \leftarrow$ the approximated fitness of $T_j^2$;
17:    if $\hat{f}(T_j^1)$ is better than $\hat{f}(T_j^2)$ then
18:        $\mathbb{T} \leftarrow \mathbb{T} \bigcup T_j^1$;
19:    else
20:        $\mathbb{T} \leftarrow \mathbb{T} \bigcup T_j^2$;
21:    end if
22: end for
23: /*injection of the transferred points*/
24: Exactly evaluate each point in $\mathbb{T}$;
25: Add the exactly evaluated $\mathbb{T}$ into $\mathbb{D}\{t\}$;
26: $P \leftarrow$ select the best $N$ points in $P \bigcup \mathbb{T}$;
```

where $\hat{f}(\mathbf{x}^t)$ is the objective approximated by the RBF model at the t-th design stage, and l_i and u_i denote the lower bound and upper bound for the ith decision variable, respectively. Next, an optimizer is employed to address the optimization problem defined by (7). Similar to the second-level search, DE/best/1 is adopted as the optimizer. As a result, an optimum T_j^1, which can be regarded as the transferred point, can be obtained. In T_j^1, the first d^j decision variables inherit from $\mathbf{x}_j^b$, and the remaining $d^t - d^j$ decision variables are the optimum of the optimization problem defined by (7). Therefore, on the one hand, the knowledge from the jth design stage can be effectively reused. On the other hand, the transferred point is an approximated optimum at the t-th design stage.

To have a clear understanding of KT1, Fig. 2 presents an illustrative example. In this example, $d^j = 3$, $d^t = 5$, and $\mathbf{x}_j^b = (2, 1, 3)$. First, each decision variable in $\mathbf{x}_j^b$ is inherited by $\mathbf{x}^t$, that is, $\mathbf{x}^t = (2, 1, 3, x_4, x_5)$. Then, the optimization problem defined by (7) can be minimized when $x_4 = 7$ and

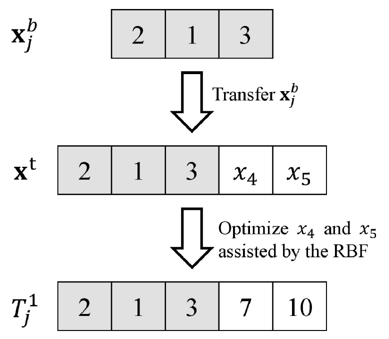

$\mathbf{x}_j^b$
2 1 3
Transfer $\mathbf{x}_j^b$
$\mathbf{x}^t$
2 1 3 x_4 x_5
Optimize x_4 and x_5 assisted by the RBF
T_j^1
2 1 3 7 10

Fig. 2. Illustration of KT1. In this example, $d^j = 3$, $d^t = 5$, and $\mathbf{x}_j^b = (2, 1, 3)$. Based on KT1, the transferred point $T_j^1 = (2, 1, 3, 7, 10)$.

$x_5 = 10$. Finally, the transferred point $T_j^1 = (2, 1, 3, 7, 10)$ can be obtained.

2) KT2: In EIOPs, the objective at the jth design stage and the objective at the t-th design stage are different but related. Domain adaptation can reveal the hidden correlations between the target task and the source task [80], where the target task and the source task are different but related. Thus, one can consider employing domain adaptation to construct the connections between the t-th design stage and the jth design stage. Note that although domain adaptation is widely applied in different optimization problems, such as evolutionary sequential transfer optimization [70], [72], multitasking optimization [66], and dynamic optimization [68], little work has been reported to employ domain adaptation for EIOPs.

Domain adaptation by covariance matching is an effective domain adaptation method in transfer learning [81], and its effectiveness in transferring the convergence-related variables of source optimization tasks to the target optimization task has been studied in [72]. To this end, KT2 borrows the ideas from the domain adaptation by covariance matching for projecting the optimum found at the jth design stage to the t-th design stage.

In KT2, the jth design stage is regarded as the source optimization task, and the t-th design stage is viewed as the target optimization task. The archive $\mathbb{D}$ stores all the exactly evaluated points from each design stage. That is, $\mathbb{D}\{j\}$ and $\mathbb{D}\{t\}$ store each exactly evaluated point at the jth and the t-th design stages, respectively. Therefore, the points in $\mathbb{D}\{j\}$ and $\mathbb{D}\{t\}$ are used to form the source data $X_s = [x_1^s, x_2^s, \ldots, x_{n_s}^s]$ and the target data $X_t = [x_1^t, x_2^t, \ldots, x_{n_t}^t]$, respectively, where n_s is the number of source data and n_t is the number of target data. Note that, at the t-th design stage, *NI* initial points *PI* are first sampled in the decision space by LHS for real FEs, and then the exactly evaluated initial points *PI* are added into $\mathbb{D}\{t\}$ (lines 2 and 3 of Algorithm 1). In addition, the algorithm can obtain several exactly evaluated points in each iteration, which are also added into $\mathbb{D}\{t\}$. Thus, $\mathbb{D}\{t\}$ always contains at least *NI* exactly evaluated points. As a result, there are no less than *NI* points in X_t. Thereafter, the centered matrices of X_s and X_t can be calculated as follows:

$$\begin{aligned}\bar{X}_s &= [\bar{x}_1^s, \bar{x}_2^s, \ldots, \bar{x}_{n_s}^s] = X_s - \bar{x}_s\mathbf{1}^T\\ \bar{X}_t &= [\bar{x}_1^t, \bar{x}_2^t, \ldots, \bar{x}_{n_t}^t] = X_t - \bar{x}_t\mathbf{1}^T\end{aligned} \tag{8}$$

where $\bar{x}_s$ and $\bar{x}_t$ denote the mean values of the source data and target data in each column, respectively, $\mathbf{1}$ is a column vector where each element is one.

Furthermore, $\bar{X}_s$ and $\bar{X}_t$ will be projected into a common low-dimensional space $\mathbb{R}^{d_c}$ by the affine transformations, where d_c is the dimensionality of the common low-dimensional space. That is

$$\begin{aligned}Z_s &= [z_1^s, z_2^s, \ldots, z_{n_s}^s] = \mathbf{P}_s^T\bar{X}_s \in \mathbb{R}^{d_c\times n_s}\\ Z_t &= [z_1^t, z_2^t, \ldots, z_{n_t}^t] = \mathbf{P}_t^T\bar{X}_t \in \mathbb{R}^{d_c\times n_t}\end{aligned} \tag{9}$$

where Z_s and Z_t are the projected samples of $\bar{X}_s$ and $\bar{X}_t$, respectively, $\mathbf{P}_s \in \mathbb{R}^{d_s\times d_c}$, $\mathbf{P}_t \in \mathbb{R}^{d_t\times d_c}$, d_s and d_t represent the number of decision variables for the source optimization task and the target optimization task, respectively.

In the common low-dimensional space, Z_s and Z_t should consider the following two aspects. The first one is that Z_s and Z_t should have a similar distribution. According to [81], higher order moments are suited for measuring the distribution of Z_s and Z_t, where the mean and the covariance are two moments considered in this work. Therefore, minimizing the following objective function can reduce the gap between Z_s and Z_t:

$$\arg\min_{\mathbf{P}_s,\mathbf{P}_t} \quad D(Z_s, Z_t) = \|\bar{Z}_s - \bar{Z}_t\|_F^2 + \left\|\frac{Z_sZ_s^T}{n_s} - \frac{Z_tZ_t^T}{n_t}\right\|_F^2 \tag{10}$$

where $\bar{Z}_s$ and $\bar{Z}_t$ are the mean value of Z_s and Z_t, respectively. From (8) and (9), one can find that $\bar{Z}_s = \bar{Z}_t = \mathbf{0}_{d_c\times 1}$. Thus, according to (8) and (9), (10) can be reformulated as follows:

$$\arg\min_{\mathbf{P}_s,\mathbf{P}_t} \quad D(Z_s, Z_t) = \left\|\frac{\mathbf{P}_s^T\bar{X}_s\bar{X}_s^T\mathbf{P}_s}{n_s} - \frac{\mathbf{P}_t^T\bar{X}_t\bar{X}_t^T\mathbf{P}_t}{n_t}\right\|_F^2. \tag{11}$$

The second aspect is the consistent discriminative information. In other words, the fitness ranks of each point in $\bar{X}_s$ and $\bar{X}_t$ should be aligned in the common low-dimensional space. To this end, the points in $\bar{X}_s$ and $\bar{X}_t$ are ranked based on fitness values. Moreover, r_s and r_t are used to store the rank labels of each point in $\bar{X}_s$ and $\bar{X}_t$, respectively. In addition, for each element in r_s and r_t, if the value of this element is larger than n, this element will be set as n, where $n = \min(n_s, n_t)$. Thereafter, the rank matching function can be defined as follows:

$$\arg\min_{\mathbf{P}_s,\mathbf{P}_t} \quad F(Z_s, Z_t) = \frac{1}{n_sn_t}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t} w_{i,j}\|z_i^s - z_j^t\|_2^2 \tag{12}$$

where $w_{i,j}$ is calculated as follows:

$$w_{i,j} = \begin{cases}1, \text{ if } \bar{x}_i^s \text{ and } \bar{x}_j^t \text{ have the same rank label}\\ 0, \text{ if } \bar{x}_i^s \text{ and } \bar{x}_j^t \text{ have different rank labels.}\end{cases} \tag{13}$$

In addition, according to (9), $z_i^s = \mathbf{P}_s^T\bar{x}_i^s$ and $z_j^t = \mathbf{P}_t^T\bar{x}_j^t$. Thus, (12) can be rewritten as follows:

$$\arg\min_{\mathbf{P}_s,\mathbf{P}_t} \quad F(Z_s, Z_t) = \frac{1}{n_sn_t}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t} w_{i,j}\|\mathbf{P}_s^T\bar{x}_i^s - \mathbf{P}_t^T\bar{x}_j^t\|_2^2. \tag{14}$$

By considering the above two aspects simultaneously, the following model can be obtained:

$$\arg\min_{\mathbf{P}_s,\mathbf{P}_t} \quad \{D(Z_s, Z_t) + \alpha F(Z_s, Z_t)\} \tag{15}$$

where α is a weight. By optimizing (15), we can achieve $\mathbf{P}_s$ and $\mathbf{P}_t$. Finally, a mapping matrix $\mathbf{M}$ can be calculated as follows:

$$\mathbf{M} = \left(\mathbf{P}_t\mathbf{P}_t^T\right)^{-1}\mathbf{P}_t\mathbf{P}_s^T. \tag{16}$$

After obtaining $\mathbf{M}$, the best point $\bar{x}_s^b$ in $\bar{\mathrm{X}}_s$ will be mapped into the target space by the follows:

$$\bar{T}_t^b = \mathbf{M}\bar{x}_s^b \tag{17}$$

where $\bar{T}_t^b$ is the mapped point. Thus, the transferred point T_j^2 can be obtained by standardizing $\bar{T}_t^b$ back to the scale of the target space. Furthermore, the fitness of T_j^2 will be approximated by the RBF model, denoted as $\hat{f}(T_j^2)$.

After performing KT1 and KT2, two transferred points, namely, T_j^1 and T_j^2, from the jth design stage can be obtained. In addition, the one with a better approximated fitness between T_j^1 and T_j^2 will be preserved in $\mathbb{T}$. Note that the knowledge from each past design stage will be transferred into the t-th design stage by the above transfer processes. Thus, $\mathbb{T}$ contains $t-1$ promising transferred points from each past design stage.

3) Injection of the Transferred Points: Since each point in $\mathbb{T}$ is the promising transferred point from the past design stages, we will exactly evaluate each point in $\mathbb{T}$. In addition, the exactly evaluated points in $\mathbb{T}$ will be added to the DS. Furthermore, each point in $\mathbb{T}$ will be injected into the population P. To be specific, the best N points in $P \bigcup \mathbb{T}$ will be selected and retained in P for the next iteration.

The SHKTS not only can reuse the knowledge captured from the past design stages to improve the search efficiency of SADE-KT at the current design stage but also can reduce negative knowledge transfer to some extent. This is due to the following three reasons.

1) The SHKTS adopts KT1 and KT2 simultaneously. In KT1, the current design stage can inherit the optima found in the past design stages. In addition, KT2 builds the connections between the current design stage and the past design stages, which can map the optima found in the past design stages to the current design stage.
2) In the SHKTS, only the better one between the transferred point from KT1 and the transferred point from KT2 can be exactly evaluated and injected into P.
3) Each point in P and each transferred point in $\mathbb{T}$ have been exactly evaluated. Therefore, a misleading comparison result that may be caused by the prediction error of RBF model can be avoided, when injecting $\mathbb{T}$ into P. In other words, the best N exactly evaluated points will be retained in P.

In the following section, the ablation study will be conducted to verify the effect of the SHKTS.

IV. Experimental Studies

This section aims to test the performance of the proposed SADE-KT by conducting a set of experiments. Due to the page limit, some tables and figures are presented in the Supplemental material. In addition, the computational complexity of SADE-KT is analyzed in Section X of the Supplemental material.

In the following tables that list the comparative results, "+," "−," and " =" mean that SADE-KT is significantly better than, significantly worse than, and similar to the compared method, respectively, according to the Wilcoxon rank sum test with a significance level of 0.05.

A. Test Problems

In this work, 26 test problems are selected to test the performance of the proposed SADE-KT in solving EIOPs. Table S1 in the Supplemental material gives the basic information of these test problems.

F1–F19 are collected in CEC2005 [82], F20–F26 are proposed by Cheng et al. [56]. In addition, there are three design stages for each test problem, where $d^1 = 30$, $d^2 = 50$ and $d^3 = 100$. Furthermore, according to [56] and [82], these test problems have different features, such as unimodal, multimodal, shifted, rotated, separable, and nonseparable.

B. Parameter Settings

The parameters for SADE-KT are outlined as follows.

1) The population size $N = 10$ for each test problem.
2) The number of trial offspring generated by each mutation strategy: $\lambda = 70$. The sensitivity analysis of λ is given in Section VI of the Supplemental material.
3) The scaling factor F and the crossover rate CR for the first-level search: $F = randn(\mu_F, 0.1)$ and $CR = randn(\mu_{CR}, 0.1)$, where $randn(a, b)$ is a normal distribution with mean a and standard deviation b. In addition, μ_F and μ_{CR} are initialized to be 0.5 and updated in each iteration according to the update rules in [83].
4) The parameters for the optimizer used to address (6) and (7): As recommended by [37], DE/best/1 is adopted in this work. In addition, the parameters of DE/best/1 are set as the same as recommended in [37].
5) The parameters for KT2: As recommended by [72] and [81], $d_c = 4$ and $\alpha = 0.1$.

Remark: In SADE-KT, we set the population size N to a small number based on the following two reasons. First, each candidate in the population will generate $4 \times \lambda = 280$ offspring by four mutation strategies in the first-level search. It means that each candidate can keep a good exploration ability. Second, for each candidate in the population, the offspring with the best predicted fitness is exactly evaluated. Thus, N real FEs will be consumed after performing the first-level search. A small N value can save a lot of FEs in each iteration, which is important for expensive optimization. Furthermore, our simulation results and the results in [42] and [55] can verify that algorithms with a small population size are able to obtain a good performance on the high-dimensional optimization problems.

The general parameters for the experiments are given as follows.

1) *Number of Initial Sampling Points NI:* For each compared algorithm, *NI* is set to 100 at each design stage for a fair comparison.
2) *Stopping Condition for Each Design Stage:* The maximum number of FEs, denoted as MaxFEs, is used as

TABLE I
STATISTICAL RESULTS AT THE FIRST DESIGN STAGE OBTAINED BY SADE-KT OVER 31 INDEPENDENT RUNS

Function No.	The first design stage ($d^1 = 30$)			
	Best	Worst	Average	Std.
F1	-4.48E+02	-4.15E+02	-4.39E+02	9.56E+00
F2	2.15E+04	7.55E+04	5.28E+04	1.28E+04
F3	4.24E+07	2.65E+08	1.11E+08	5.95E+07
F4	3.91E+04	1.11E+05	6.90E+04	2.00E+04
F5	6.47E+03	1.53E+04	1.12E+04	2.35E+03
F6	3.04E+07	1.71E+08	7.33E+07	3.30E+07
F7	4.51E+03	4.52E+03	4.52E+03	4.68E-01
F8	-1.19E+02	-1.19E+02	-1.19E+02	5.91E-02
F9	-2.22E+02	-7.19E+01	-1.54E+02	4.22E+01
F10	-1.30E+02	1.79E+01	-6.88E+01	4.44E+01
F11	1.20E+02	1.39E+02	1.33E+02	4.51E+00
F12	1.03E+06	1.61E+06	1.34E+06	1.64E+05
F13	-1.06E+02	-9.64E+01	-1.02E+02	2.91E+00
F14	-2.86E+02	-2.85E+02	-2.86E+02	1.05E-01
F15	6.34E+02	9.36E+02	8.07E+02	6.48E+01
F16	4.09E+02	6.52E+02	5.11E+02	6.98E+01
F17	9.55E+02	1.05E+03	9.99E+02	2.73E+01
F18	8.95E+02	1.54E+03	1.07E+03	2.10E+02
F19	6.69E+02	1.58E+03	1.40E+03	1.89E+02
F20	6.67E-01	6.15E+01	6.68E+00	1.35E+01
F21	8.78E+01	2.69E+02	1.60E+02	5.07E+01
F22	1.67E+07	1.36E+08	4.71E+07	2.72E+07
F23	7.48E+01	2.61E+02	1.69E+02	5.43E+01
F24	1.36E+07	1.57E+08	4.81E+07	3.32E+07
F25	4.24E+01	2.95E+02	1.73E+02	7.20E+01
F26	1.84E+02	5.19E+02	3.85E+02	8.83E+01

TABLE II
STATISTICAL RESULTS AT THE SECOND DESIGN STAGE OBTAINED BY SADE-KT OVER 31 INDEPENDENT RUNS

Function No.	The second design stage ($d^2 = 50$)			
	Best	Worst	Average	Std.
F1	7.21E+02	4.44E+03	1.90E+03	9.28E+02
F2	1.10E+05	2.69E+05	1.53E+05	4.23E+04
F3	4.88E+08	1.53E+09	8.53E+08	2.64E+08
F4	1.17E+05	3.30E+05	1.85E+05	5.59E+04
F5	2.21E+04	3.47E+04	2.77E+04	3.31E+03
F6	2.47E+08	1.15E+09	5.55E+08	2.08E+08
F7	6.05E+03	6.17E+03	6.09E+03	3.52E+01
F8	-1.19E+02	-1.19E+02	-1.19E+02	5.56E-02
F9	9.26E+00	2.36E+02	9.87E+01	5.93E+01
F10	1.52E+02	2.93E+02	2.36E+02	3.47E+01
F11	1.63E+02	1.76E+02	1.72E+02	2.98E+00
F12	4.58E+06	8.09E+06	6.41E+06	1.02E+06
F13	-8.38E+01	-4.73E+01	-7.25E+01	9.53E+00
F14	-2.76E+02	-2.75E+02	-2.76E+02	1.76E-01
F15	6.82E+02	1.06E+03	9.58E+02	8.01E+01
F16	5.69E+02	7.77E+02	6.43E+02	6.12E+01
F17	1.08E+03	1.30E+03	1.15E+03	5.56E+01
F18	1.31E+03	1.51E+03	1.44E+03	4.87E+01
F19	1.45E+03	1.72E+03	1.57E+03	7.29E+01
F20	1.27E+02	7.36E+02	3.36E+02	1.34E+02
F21	2.04E+02	5.03E+02	3.71E+02	6.98E+01
F22	4.56E+07	4.15E+08	1.54E+08	1.05E+08
F23	2.84E+02	5.12E+02	4.06E+02	6.00E+01
F24	5.20E+07	3.67E+08	1.16E+08	7.44E+07
F25	3.02E+02	5.67E+02	4.15E+02	6.31E+01
F26	4.21E+02	2.57E+03	1.39E+03	5.82E+02

the stopping condition for each design stage. To be specific, MaxFEs $= 11 \times d^1 = 11 \times 30 = 330$ for the first design stage, MaxFEs $= 11 \times (d^2 - d^1) = 11 \times (50 - 30) = 220$ for the second design stage, and MaxFEs $= 11 \times (d^3 - d^2) = 11 \times (100 - 50) = 550$ for the third design stage.

3) *Number of Runs:* Each compared algorithm is independently run 31 times on each test problem.

C. Behavior Study of SADE-KT

The statistical results at each design stage obtained by SADE-KT over 31 independent runs are given in Tables I–III, respectively. Figs. S1–S3 in the Supplemental material display the convergence curves at each design stage achieved by SADE-KT, respectively.

From Tables I–III and Figs. S1–S3 in the Supplemental material, SADE-KT demonstrates good performance and convergence at each design stage on most test problems, especially on F1, F10, F13, and F20. However, SADE-KT almost stagnates in the whole search stage on F8 and F14. Because F8 is the multimodal function with the global optimum on the bound. On the other hand, F14 is expanded by various complex multimodal functions. Thus, searching for the optimum on the these two test problems poses a challenge task for the algorithm.

In SADE-KT, several strategies are introduced. In the following, we will analyze the effect of each strategy.

1) Effect of Knowledge Transfer: The proposed SHKTS aims to transfer the knowledge captured from the past design stages to the current design stage by applying KT1 and KT2. This raises two interesting questions, namely, whether knowledge transfer can improve the performance of the algorithm and SHKTS outperforms KT1 and KT2.

TABLE III
STATISTICAL RESULTS AT THE THIRD DESIGN STAGE OBTAINED BY SADE-KT OVER 31 INDEPENDENT RUNS

Function No.	The third design stage ($d^3 = 100$)			
	Best	Worst	Average	Std.
F1	3.50E+02	1.55E+03	8.09E+02	3.02E+02
F2	2.57E+05	6.69E+05	4.61E+05	9.70E+04
F3	6.51E+09	1.46E+10	1.13E+10	1.80E+09
F4	4.44E+05	9.52E+05	6.39E+05	1.25E+05
F5	4.49E+04	6.16E+04	5.19E+04	4.34E+03
F6	9.57E+08	1.91E+09	1.39E+09	2.37E+08
F7	1.25E+04	1.82E+04	1.49E+04	1.79E+03
F8	-1.19E+02	-1.18E+02	-1.19E+02	3.24E-02
F9	3.34E+02	7.32E+02	5.74E+02	9.25E+01
F10	8.28E+02	1.49E+03	9.98E+02	1.44E+02
F11	2.55E+02	2.71E+02	2.65E+02	3.86E+00
F12	2.42E+07	3.47E+07	3.01E+07	3.11E+06
F13	-2.44E+01	3.03E+00	-1.24E+01	7.13E+00
F14	-2.51E+02	-2.50E+02	-2.51E+02	1.89E-01
F15	6.19E+02	1.10E+03	9.70E+02	1.11E+02
F16	5.46E+02	7.32E+02	6.15E+02	5.67E+01
F17	1.00E+03	1.54E+03	1.31E+03	1.47E+02
F18	1.40E+03	1.57E+03	1.46E+03	4.32E+01
F19	1.55E+03	1.72E+03	1.67E+03	3.71E+01
F20	3.19E+01	1.04E+02	6.29E+01	1.87E+01
F21	5.48E+02	8.94E+02	7.14E+02	1.07E+02
F22	1.83E+07	7.06E+07	4.48E+07	1.57E+07
F23	3.68E+02	8.83E+02	6.95E+02	1.20E+02
F24	2.29E+07	9.00E+07	4.23E+07	1.78E+07
F25	3.97E+02	8.64E+02	6.60E+02	1.47E+02
F26	3.75E+02	8.22E+02	5.94E+02	1.24E+02

TABLE IV
WILCOXON RANK SUM TEST RESULTS BETWEEN SADE-KT AND SADE-WKT

SADE-KT VS.	Wilcoxon rank sum test	SADE-WKT
The second design stage ($d^2 = 50$)	+	20
	-	0
	=	6
The third design stage ($d^3 = 100$)	+	15
	-	1
	=	10

To answer the first question, a comparison is made between SADE-KT and SADE-WKT. In SADE-WKT, knowledge transfer is removed, that is, the objective is independently optimized at each design stage. SADE-KT and SADE-WKT are independently run 31 times on each test problem. As knowledge transfer will not be applied at the first design stage, we only show the statistical results at the second design stage and the third design stage obtained by each method in this section. The Wilcoxon rank sum test results between SADE-KT and SADE-WKT are shown in Table IV. Table S5 in the Supplemental material gives the detailed statistical results obtained by SADE-KT and SADE-WKT.

From Table IV, SADE-KT shows superior or comparable performance to SADE-WKT across each test problem at the second design stage. Moreover, SADE-WKT performs better than SADE-KT only on one out of 26 test problems at the third design stage. It can indicate that knowledge transfer plays a crucial role in enhancing the performance of the algorithm.

To answer the second question, the following two variants are designed.

1) *SADE-KT$_{v1}$:* It removes KT1, and only employs KT2 for knowledge transfer.
2) *SADE-KT$_{v2}$:* It removes KT2, and only employs KT1 for knowledge transfer.

SADE-KT and each variant are independently run 31 times on each test problem. The Wilcoxon rank sum test results between SADE-KT and each variant are shown in Table V. Table S6 in the Supplemental material gives the detailed statistical results obtained by SADE-KT, SADE-KT$_{v1}$, and SADE-KT$_{v2}$.

From Table V, SADE-KT outperforms SADE-KT$_{v1}$ on 20 and 14 out of 26 test problems at the second design stage and the third design stage, respectively. In addition, SADE-KT$_{v2}$ only surpasses SADE-KT on three out of 26 test problems at the second design stage and two out of 26 test problems at the third design stage. Thus, we can consider that SHKTS is better than KT1 and KT2 in the proposed SADE-KT. Because the SHKTS can reduce negative knowledge transfer to some extent compared with KT1 and KT2 as shown in the next section.

In addition, to display the role of KT1 and KT2 in SHKTS, Fig. 3 shows the average proportion of the exactly evaluated points in SHKTS generated by KT1 and KT2 on some test problems over 31 independent runs. To be specific, the proportion of the exactly evaluated points in SHKTS generated by KT1 is defined as $N_1/(N_1+N_2)$, while $N_2/(N_1+N_2)$ is the proportion of the exactly evaluated points in SHKTS generated by KT2, where N_1 and N_2 are the number of the exactly evaluated points generated by KT1 and KT2, respectively. From Fig. 3, one can find that both KT1 and KT2 can make contribution for SADE-KT.

TABLE V
WILCOXON RANK SUM TEST RESULTS BETWEEN SADE-KT AND TWO VARIANTS

SADE-KT VS.	Wilcoxon rank sum test	SADE-KT$_{v1}$	SADE-KT$_{v2}$
The second design stage ($d^2 = 50$)	+	20	11
	-	1	3
	=	5	12
The third design stage ($d^3 = 100$)	+	14	13
	-	7	2
	=	5	11

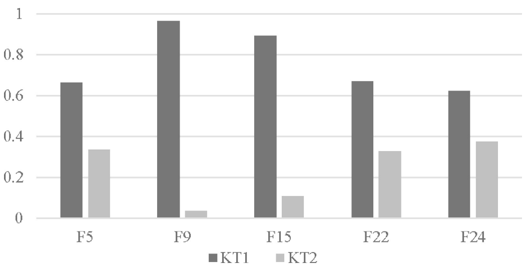


Fig. 3. Average proportion of the exactly evaluated points in SHKTS generated by KT1 and KT2 over 31 independent runs.

2) Effect of Each Level: In SADE-KT, a two-level surrogate-assisted evolutionary search is developed to search for the optimum at each design stage. Thus, to study the effectiveness of each level, two variants are designed, namely, SADE-KT$_1$ and SADE-KT$_2$. In SADE-KT$_1$, the algorithm only adopts the first-level search, while the second-level search is discarded. SADE-KT$_2$ only applies the second-level search, while the first-level search is removed. SADE-KT$_1$, SADE-KT$_2$, and SADE-KT are independently run 31 times on each test problem. The statistical results obtained by SADE-KT$_1$, SADE-KT$_2$ and SADE-KT are given by Table S7 in the Supplemental material.

Based on the statistical results shown in Table S7 in the Supplemental material, the overall performance of SADE-KT is better than that of SADE-KT$_1$ and SADE-KT$_2$ on most test problems. Because it is hard to obtain a good exploitation ability for SADE-KT$_1$. In addition, due to the lack of the first-level search, the exploration ability of SADE-KT$_2$ may be influenced. Thus, one can consider that each level is important for SADE-KT.

3) Effect of Adaptively Using Two Coordinate Systems: In SADE-KT, two coordinate systems are adaptively tuned in the first-level search. To study whether adaptively using two coordinate systems is better than using one, we compare SADE-KT with the following two variants.

1) *SADE-KT$_O$*: It only employs the original coordinate system in the first-level search.
2) *SADE-KT$_E$*: It only applies the eigen coordinate system in the first-level search.

SADE-KT and each variant are independently run 31 times on each test problem. The statistical results obtained by SADE-KT$_O$, SADE-KT$_E$ and SADE-KT are given by Table S8 in the Supplemental material. As shown in Table S8 in the

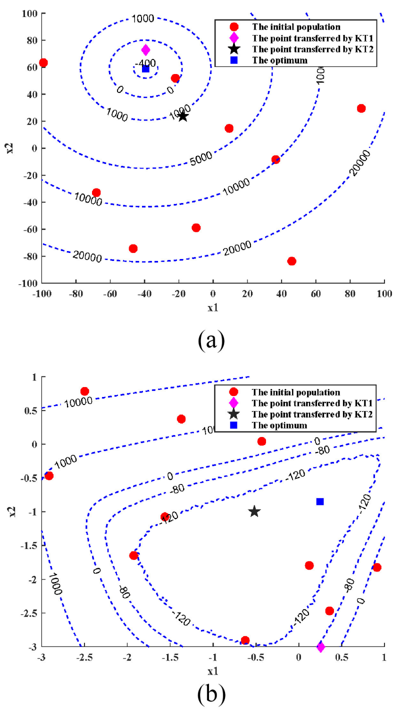


Fig. 4. Distributions of each point in the first generation of the second design stage on F1 and F13. (a) Distributions of the initial population, the transferred points and the optimum on F1. (b) Distributions of the initial population, the transferred points and the optimum on F13.

Supplemental material, SADE-KT$_O$ and SADE-KT$_E$ perform better than SADE-KT on some test problems. Because it is difficult to effectively deal with each test problem for a given algorithm. Nevertheless, the overall performance of SADE-KT is better than that of SADE-KT$_O$ and SADE-KT$_E$ on most test problems, in particular on F20. This phenomenon indicates that adaptively using two coordinate systems can increase the probability for creating the promising offspring. Therefore, we can conclude that adaptively using two coordinate systems is beneficial for SADE-KT.

D. *Transfer Behavior of SHKTS*

In this section, we show the transfer behavior of SHKTS on F1 and F13. The properties of F1 and F13 are separable and nonseparable, respectively. In this experiment, there are two design stages for F1 and F13, where $d^1 = 1$ and $d^2 = 2$. The parameter *NI* and the maximum number of FEs for each design stage are set to 10 and 100, respectively. The specific parameters for SADE-KT are set based on Section IV-B. Fig. 4 displays the distributions of each point in the first generation of the second design stage on F1 and F13. Table VI gives the predicted fitness values of each transferred point in the first generation of the second design stage on F1 and F13.

From Fig. 4(a), it can be found that the point transferred by KT1 is distributed closer to the optimum than the point transferred by KT2 on F1. According to Table VI, the point transferred by KT1 can be preserved. In addition, from Fig. 4(b), the point transferred by KT2 is distributed near the optimum of F13, whereas the point transferred by KT1 is far from the optimum. Because F13 is a nonseparable test problem, KT1 is not well suited for such case. Based on the predicted fitness value, the point transferred by KT2 can be preserved. Thus, the convergence of the population can be speeded up after performing SHKTS on F1 and F13.

TABLE VI
PREDICTED FITNESS VALUES OF EACH TRANSFERRED POINT IN THE FIRST GENERATION OF THE SECOND DESIGN STAGE ON F1 AND F13

Function No.	The predicted fitness value of the point transferred by KT1	The predicted fitness value of the point transferred by KT2
F1	**-3.01E+02**	3.44E+02
F13	-1.10E+02	**-2.38E+02**

TABLE VII
STATISTICAL RESULTS ON THE CANTILEVER PROBLEM OBTAINED BY SADE-WKT AND SADE-KT

The first design stage ($n^1 = 10$)		
Algorithm	SADE-WKT	SADE-KT
Best	**0.0062**	**0.0062**
Average	**0.0066**	**0.0066**
Worst	0.0076	**0.0071**
Std.	0.0004	**0.0003**
The second design stage ($n^2 = 20$)		
Algorithm	SADE-WKT	SADE-KT
Best	0.0575	**0.0568**
Average	0.0626	**0.0608**
Worst	**0.0676**	0.0692
Std.	0.0037	**0.0032**
The third design stage ($n^3 = 30$)		
Algorithm	SADE-WKT	SADE-KT
Best	**0.2178**	0.2314
Average	(28)	**0.2871**
Worst	NA	**0.3917**
Std.	NA	**0.0395**

However, only performing KT1 makes it difficult to improve the convergence of the population on F13. Similarly, if KT2 is employed only, the improvement in population convergence on F1 may be limited. This implies that relying solely on KT1 or KT2 can result in negative knowledge transfer in some cases. In this work, negative knowledge transfer refers to the inability of the transferred point to improve the convergence of the population. Thus, one can consider that SHKTS can alleviate negative knowledge transfer to some extent compared with KT1 and KT2.

E. *Application of SADE-KT on the Cantilever Problem*

The cantilever problem can be incrementally optimized by adding more decision variables [53]. Thus, we test the effectiveness of SADE-KT on the cantilever problem in this section. In the cantilever problem, a cantilever beam has n^t steps at each design stage and bears $P = 50kN$ force on the tip [84]. There are three different variables, i.e., width (b_i), height (h_i) and length (l_i), on each beam step. Thus, $3 \times n^t$ different design variables exist at each design stage and are arrayed in the following:

$$X = [b_1, h_1, l_1, b_2, h_2, l_2, \ldots, b_{n^t}, h_{n^t}, l_{n^t}]. \tag{18}$$

TABLE VIII
STATISTICAL RESULTS OF E_{BBC} VALUES OBTAINED BY EACH METHOD ON THE MPB WITH DIFFERENT SHIFT SEVERITIES

	CPSO	APCPSO	SA-CPSO	SADE-KT
s=1	2.64E+01(7.52E+00)+	2.49E+01(4.49E+00)+	2.42E+01(5.65E+00)+	**1.65E+01(5.91E+00)**
s=2	**2.23E+01(7.22E+00)-**	2.58E+01(5.24E+00)+	2.62E+01(4.93E+00)+	2.42E+01(5.94E+00)
s=3	2.56E+01(6.67E+00)+	2.56E+01(3.27E+00)+	2.80E+01(4.24E+00)+	**2.32E+01(6.18E+00)**
s=4	2.48E+01(6.19E+00)+	2.91E+01(4.98E+00)+	2.85E+01(4.38E+00)+	**2.25E+01(7.36E+00)**
s=5	3.02E+01(7.87E+00)+	2.83E+01(4.51E+00)+	3.13E+01(2.99E+00)+	**2.77E+01(5.59E+00)**
s=6	**2.54E+01(7.46E+00)-**	2.61E+01(3.44E+00)-	3.14E+01(3.75E+00)+	2.68E+01(4.37E+00)
s=7	2.64E+01(7.85E+00)+	2.81E+01(3.98E+00)+	3.25E+01(2.93E+00)+	**2.39E+01(4.35E+00)**
+/-/=	5/2/0	6/1/0	7/0/0	

The properties of the used material are set as $E = 200GPa$ and $\delta_{\text{allow}} = 350MPa$. This cantilever problem aims to minimize the tip deflection (δ^t) at each design stage. The details can be seen Section IX of the Supplemental material.

In this study, there are three design stages, where $n^1 = 10$, $n^2 = 20$ and $n^3 = 30$. The maximum number of FEs MaxFEs $= 11 \times 3 \times 10 = 330$ for the first design stage, MaxFEs $= 11 \times 3 \times (20 - 10) = 330$ for the second design stage and MaxFEs $= 11 \times 3 \times (30 - 20) = 330$ for the third design stage. We compare SADE-KT with SADE-WKT to verify that knowledge transfer can be beneficial for the cantilever problem. The specific parameters are set based on Section IV-B. The number of initial sampling points at each design stage is set to 100 for each compared method. To satisfy the constraints, a penalty function approach is employed to penalize the infeasible points. Table VII gives the statistical results on this cantilever problem obtained by each method over 31 independent runs. Note that $(*)$ is the number of successful runs out of 31 independent runs, where a successful run denotes that at least one feasible point is found in this run.

From Table VII, SADE-KT and SADE-WKT can obtain almost the same good results at the first design. However, at the second design stage, the average performance of SADE-KT is slightly better than that of SADE-WKT. At the third design stage, SADE-WKT only obtains 28 successful runs over 31 independent runs, while SADE-KT shows a good overall performance. Thus, one can consider that the search efficiency of the second design stage and the third design stage can be improved by introducing knowledge transfer. Overall, SADE-KT shows competitiveness in dealing with the cantilever problem compared with SADE-WKT.

F. Comparative Studies on Expensive Dynamic Optimization

The expensive dynamic optimization problems have the following two features [48]. First, the objective, constraints, or the number of objectives or variables may change over time. Second, FEs are highly costly in each environment. Thus, the algorithm should continuously track the dynamic optimum within a limited number of FEs. In this section, the moving peaks benchmark (MPB) is selected as the test problem, where the MPB is a classical dynamic optimization test problem proposed in [85]. Moreover, the best-error-before-change (E_{BBC}) [86] is used as the performance indicator. The details of MPB and E_{BBC} can be found in Section XI of the Supplemental material. The parameters for MPB are given in Table S9 in the Supplemental material. Note that the change frequency U is set to 300, which indicates that the environment changes every 300 FEs.

SADE-KT is compared with three state-of-the-art dynamic optimization methods, namely, CPSO [87], APCPSO [88], and SA-CPSO [48]. CPSO and APCPSO are designed for dynamic optimization problems, and SA-CPSO is developed for expensive dynamic optimization problems. The specific parameters for SADE-KT are set based on Section IV-B. The population size for CPSO and APCPSO is set to 20 due to $U = 300$. The other specific parameters for each compared method are kept the same as in the corresponding papers. The Wilcoxon rank sum test with a significance level of 0.05 is used to compare the significant difference between the compared methods. The statistical results of E_{BBC} values obtained by each method on the MPB with different shift severities are given in Table VIII.

From Table VIII, CPSO performs better than SADE-KT when $s = 2$ and $s = 6$, and the overall performance of APCPSO is better than that of SADE-KT on $s = 6$. This is because SADE-KT only applies one population to search for the decision space, which may perform worse than the multipopulation methods in some cases. Nevertheless, SADE-KT can obtain the best overall performance on five out of seven test problems. Thus, one can consider that SADE-KT is competitive in solving expensive dynamic optimization problems compared with CPSO, APCPSO, and SA-CPSO.

V. CONCLUSION

In this work, SADE-KT is proposed, which aims to fill the current gap in solving EIOPs. In SADE-KT, a two-level surrogate-assisted evolutionary search is designed to search for the optimum at each design stage. Moreover, an SHKTS is proposed, which contains two effective knowledge transfer techniques. Thus, the search efficiency of the current design stage can be improved by reusing the knowledge captured from the past design stages. The effectiveness of SADE-KT in solving EIOPs has been verified on 26 test problems and a cantilever problem. We also investigate the performance of SADE-KT on the expensive dynamic optimization.

There are some limitations in the proposed SADE-KT. First, transferring the knowledge from all the past design stages may not always be beneficial for the current stage, particularly when there are many design stages. Thus, it may be worth selecting the helpful design stages for knowledge transfer. Second, KT2 only applies the exactly evaluated points to form the source data and the target data. However, the number of the exactly evaluated points is limited in EIOPs. Therefore, it is interesting to study whether semisupervised knowledge transfer can be introduced to the KT2. Third, the computational burden of the SADE-KT is slightly higher compared to some sate-of-the-art SAEAs due to the application of mutation strategies and knowledge transfer in each iteration. A possible way to reduce the computational burden is to perform knowledge transfer every certain number of iterations.

REFERENCES

[1] Y. Jin, “Surrogate-assisted evolutionary computation: Recent advances and future challenges,” *Swarm Evol. Comput.*, vol. 1, no. 2, pp. 61–70, 2011.

[2] S. Yang, Y. Tian, X. Xiang, S. Peng, and X. Zhang, “Accelerating evolutionary neural architecture search via multifidelity evaluation,” *IEEE Trans. Cogn. Develop. Syst.*, vol. 14, no. 4, pp. 1778–1792, Dec. 2022.

[3] A. I. J. Forrester, N. W. Bressloff, and A. J. Keane, “Optimization using surrogate models and partially converged computational fluid dynamics simulations,” *Proc. Roy. Soc. A, Math. Phys. Eng. Sci.*, vol. 462, no. 2071, pp. 2177–2204, 2006.

[4] G. De Gaetano, D. Mundo, C. Maletta, M. Kroiss, and L. Cremers, “Multi-objective optimization of a vehicle body by combining gradient-based methods and vehicle concept modelling,” *Case Stud. Mech. Syst. Signal Process.*, vol. 1, pp. 1–7, Jul. 2015.

[5] A. G. Manca and C. M. Pappalardo, “Topology optimization procedure of aircraft mechanical components based on computer-aided design, multibody dynamics, and finite element analysis,” in *Design, Simulation, Manufacturing: The Innovation Exchange*. Cham, Switzerland: Springer, 2020, pp. 159–168.

[6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[7] A. Gupta, Y.-S. Ong, and L. Feng, “Multifactorial evolution: Toward evolutionary multitasking,” *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 343–357, Jun. 2016.

[8] Y. Jin, H. Wang, T. Chugh, D. Guo, and K. Miettinen, “Data-driven evolutionary optimization: An overview and case studies,” *IEEE Trans. Evol. Comput.*, vol. 23, no. 3, pp. 442–458, Jun. 2019.

[9] J.-Y. Li, Z.-H. Zhan, and J. Zhang, “Evolutionary computation for expensive optimization: A survey,” *Mach. Intell. Res.*, vol. 19, no. 1, pp. 3–23, 2022.

[10] D. R. Jones, M. Schonlau, and W. J. Welch, “Efficient global optimization of expensive black-box functions,” *J. Global Optim.*, vol. 13, no. 4, pp. 455–492, 1998.

[11] Y. Jin, M. Olhofer, and B. Sendhoff, “A framework for evolutionary optimization with approximate fitness functions,” *IEEE Trans. Evol. Comput.*, vol. 6, no. 5, pp. 481–494, Oct. 2002.

[12] D. Lim, Y. Jin, Y.-S. Ong, and B. Sendhoff, “Generalizing surrogate-assisted evolutionary computation,” *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 329–355, Jun. 2010.

[13] J. Knowles, “ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems,” *IEEE Trans. Evol. Comput.*, vol. 10, no. 1, pp. 50–66, Feb. 2006.

[14] Q. Zhang, W. Liu, E. Tsang, and B. Virginas, “Expensive multiobjective optimization by MOEA/D with Gaussian process model,” *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 456–474, Jun. 2010.

[15] Q. Liu, R. Cheng, Y. Jin, M. Heiderich, and T. Rodemann, “Reference vector-assisted adaptive model management for surrogate-assisted many-objective optimization,” *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 12, pp. 7760–7773, Dec. 2022.

[16] S. Huang, J. Zhong, and W.-J. Yu, “Surrogate-assisted evolutionary framework with adaptive knowledge transfer for multi-task optimization,” *IEEE Trans. Emerg. Topics Comput.*, vol. 9, no. 4, pp. 1930–1944, Oct.–Dec. 2021.

[17] A. Sóbester, S. J. Leary, and A. J. Keane, “A parallel updating scheme for approximating and optimizing high fidelity computer simulations,” *Struct. Multidiscip. Optim.*, vol. 27, no. 5, pp. 371–383, 2004.

[18] D. Zhan, Y. Meng, and H. Xing, “A fast multipoint expected improvement for parallel expensive optimization,” *IEEE Trans. Evol. Comput.*, vol. 27, no. 1, pp. 170–184, Feb. 2023.

[19] J. E. Dennis and V. Torczon, “Managing approximation models in optimization,” in *Multidisciplinary Design Optimization State of the Art*, vol. 5. Philadelphia, PA, USA: SIAM, 1997, pp. 330–347.

[20] H. Ulmer, F. Streichert, and A. Zell, “Evolution strategies assisted by Gaussian processes with improved preselection criterion,” in *Proc. Congr. Evol. Comput. (CEC)*, vol. 1, 2003, pp. 692–699.

[21] M. Schonlau, W. J. Welch, and D. R. Jones, *Global Versus Local Search in Constrained Optimization of Computer Models* (Lecture Notes-Monograph Series), vol. 34. Hayward, CA, USA: Inst. Math. Stat., 1998, pp. 11–25.

[22] C. Durantin, J. Marzat, and M. Balesdent, “Analysis of multi-objective Kriging-based methods for constrained global optimization,” *Comput. Optim. Appl.*, vol. 63, no. 3, pp. 903–926, 2016.

[23] R. Shi, L. Liu, T. Long, Y. Wu, and Y. Tang, “Filter-based adaptive Kriging method for black-box optimization problems with expensive objective and constraints,” *Comput. Methods Appl. Mech. Eng.*, vol. 347, pp. 782–805, Apr. 2019.

[24] H. Dong, P. Wang, B. Song, Y. Zhang, and X. An, “Kriging-assisted discrete global optimization (KDGO) for black-box problems with costly objective and constraints,” *Appl. Soft Comput.*, vol. 94, Sep. 2020, Art. no. 106429.

[25] R. Jiao, B. Xue, and M. Zhang, “Investigating the correlation amongst the objective and constraints in Gaussian process-assisted highly constrained expensive optimization,” *IEEE Trans. Evol. Comput.*, vol. 26, no. 5, pp. 872–885, Oct. 2022.

[26] B. Liu, Q. Zhang, and G. G. E. Gielen, “A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems,” *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 180–192, Apr. 2014.

[27] C. Li, S. Gupta, S. Rana, V. Nguyen, S. Venkatesh, and A. Shilton, “High dimensional Bayesian optimization using dropout,” in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 2096–2102.

[28] J. Tian, Y. Tan, J. Zeng, C. Sun, and Y. Jin, “Multiobjective infill criterion driven Gaussian process-assisted particle swarm optimization of high-dimensional expensive problems,” *IEEE Trans. Evol. Comput.*, vol. 23, no. 3, pp. 459–472, Jun. 2019.

[29] D. Zhan and H. Xing, “A fast Kriging-assisted evolutionary algorithm based on incremental learning,” *IEEE Trans. Evol. Comput.*, vol. 25, no. 5, pp. 941–955, Oct. 2021.

[30] D. Huang, T. T. Allen, W. I. Notz, and N. Zeng, “Global optimization of stochastic black-box systems via sequential Kriging meta-models,” *J. Global Optim.*, vol. 34, no. 3, pp. 441–466, 2006.

[31] J. W. Kruisselbrink, M. T. M. Emmerich, A. H. Deutz, and T. Bäck, “A robust optimization approach using Kriging metamodels for robustness approximation in the CMA-ES,” in *Proc. IEEE Congr. Evol. Comput.*, 2010, pp. 1–8.

[32] C. Sun, J. Zeng, J. Pan, S. Xue, and Y. Jin, “A new fitness estimation strategy for particle swarm optimization,” *Inf. Sci.*, vol. 221, pp. 355–370, Feb. 2013.

[33] X. Ji, Y. Zhang, D. Gong, and X. Sun, “Dual-surrogate-assisted cooperative particle swarm optimization for expensive multimodal problems,” *IEEE Trans. Evol. Comput.*, vol. 25, no. 4, pp. 794–808, Aug. 2021.

[34] C. Sun, Y. Jin, R. Cheng, J. Ding, and J. Zeng, “Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems,” *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 644–660, Aug. 2017.

[35] Y. Haibo, T. Ying, Z. Jianchao, S. Chaoli, and J. Yaochu, “Surrogate-assisted hierarchical particle swarm optimization,” *Inf. Sci.*, vols. 454–455, pp. 59–72, Jul. 2018.

[36] Y. Liu, J. Liu, and Y. Jin, “Surrogate-assisted multipopulation particle swarm optimizer for high-dimensional expensive optimization,” *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 7, pp. 4671–4684, Jul. 2022.

[37] X. Wang, G. G. Wang, B. Song, P. Wang, and Y. Wang, “A novel evolutionary sampling assisted optimization method for high-dimensional expensive problems,” *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 815–827, Oct. 2019.

[38] H. Zhen, W. Gong, and L. Wang, “Evolutionary sampling agent for expensive problems,” *IEEE Trans. Evol. Comput.*, vol. 27, no. 3, pp. 716–727, Jun. 2023.

[39] M. Cui, L. Li, M. Zhou, and A. Abusorrah, “Surrogate-assisted autoencoder-embedded evolutionary optimization algorithm to solve high-dimensional expensive problems,” *IEEE Trans. Evol. Comput.*, vol. 26, no. 4, pp. 676–689, Aug. 2022.

[40] C. Sun, J. Ding, J. Zeng, and Y. Jin, "A fitness approximation assisted competitive swarm optimizer for large scale expensive optimization problems," *Memetic Comput.*, vol. 10, no. 2, pp. 123–134, 2018.

[41] H.-R. Wang, C.-H. Chen, Y. Li, J. Zhang, and Z.-H. Zhan, "Progressive sampling surrogate-assisted particle swarm optimization for large-scale expensive optimization," in *Proc. Genet. Evol. Comput. Conf.*, 2022, pp. 40–48.

[42] R. G. Regis, "Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 326–347, Jun. 2014.

[43] K. H. Rahi, H. K. Singh, and T. Ray, "Partial evaluation strategies for expensive evolutionary constrained optimization," *IEEE Trans. Evol. Comput.*, vol. 25, no. 6, pp. 1103–1117, Dec. 2021.

[44] G. Li and Q. Zhang, "Multiple penalties and multiple local surrogates for expensive constrained optimization," *IEEE Trans. Evol. Comput.*, vol. 25, no. 4, pp. 769–778, Aug. 2021.

[45] F.-F. Wei, W.-N. Chen, Q. Li, S.-W. Jeon, and J. Zhang, "Distributed and expensive evolutionary constrained optimization with on-demand evaluation," *IEEE Trans. Evol. Comput.*, vol. 27, no. 3, pp. 671–685, Jun. 2023.

[46] J. Liu, Y. Wang, G. Sun, and T. Pang, "Multisurrogate-assisted ant colony optimization for expensive optimization problems with continuous and categorical variables," *IEEE Trans. Cybern.*, vol. 52, no. 11, pp. 11348–11361, Nov. 2022.

[47] W. Luo, R. Yi, B. Yang, and P. Xu, "Surrogate-assisted evolutionary framework for data-driven dynamic optimization," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 3, no. 2, pp. 137–150, Apr. 2019.

[48] Y. Liu, J. Liu, T. Zheng, and Y. Yang, "A surrogate-assisted clustering particle swarm optimizer for expensive optimization under dynamic environment," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2020, pp. 1–8.

[49] Z. Zhou, Y. S. Ong, P. B. Nair, A. J. Keane, and K. Y. Lum, "Combining global and local surrogate models to accelerate evolutionary optimization," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 37, no. 1, pp. 66–76, Jan. 2007.

[50] X. Cai, L. Gao, and X. Li, "Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 365–379, Apr. 2020.

[51] W. Wang, H.-L. Liu, and K. C. Tan, "A surrogate-assisted differential evolution algorithm for high-dimensional expensive optimization problems," *IEEE Trans. Cybern.*, vol. 53, no. 4, pp. 2685–2697, Apr. 2023.

[52] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized multitasking for evolutionary optimization of expensive problems," *IEEE Trans. Evol. Comput.*, vol. 23, no. 1, pp. 44–58, Feb. 2019.

[53] I. Y. Kim and O. L. De Weck, "Variable chromosome length genetic algorithm for progressive refinement in topology optimization," *Struct. Multidiscipl. Optim.*, vol. 29, no. 6, pp. 445–456, 2005.

[54] M. Olhofer, Y. Jin, and B. Sendhoff, "Adaptive encoding for aerodynamic shape optimization using evolution strategies," in *Proc. Congr. Evol. Comput.*, vol. 1, 2001, pp. 576–583.

[55] Y. Jin, M. Olhofer, and B. Sendhoff, "On evolutionary optimization of large problems using small populations," in *Proc. Int. Conf. Natural Comput.*, 2005, pp. 1145–1154.

[56] R. Cheng, M. N. Omidvar, A. H. Gandomi, B. Sendhoff, S. Menzel, and X. Yao, "Solving incremental optimization problems via cooperative coevolution," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 762–775, Oct. 2019.

[57] C. Yang, J. Ding, Y. Jin, and T. Chai, "Incremental data-driven optimization of complex systems in nonstationary environments," *Sci. China Inf. Sci.*, vol. 61, no. 12, pp. 1–3, 2018.

[58] L. Zhao, Y. Hu, B. Wang, X. Jiang, C. Liu, and C. Zheng, "A surrogate-assisted evolutionary algorithm based on multi-population clustering and prediction for solving computationally expensive dynamic optimization problems," *Expert Syst. Appl.*, vol. 223, Mar. 2023, Art. no. 119815.

[59] R. Chen and K. Li, "Transfer Bayesian optimization for expensive black-box optimization in dynamic environment," in *Proc. IEEE Int. Conf. Syst. Man Cybern. (SMC)*, 2021, pp. 1374–1379.

[60] X. Fan, K. Li, and K. C. Tan, "Surrogate assisted evolutionary algorithm based on transfer learning for dynamic expensive multi-objective optimisation problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2020, pp. 1–8.

[61] Z. Liu and H. Wang, "A data augmentation based Kriging-assisted reference vector guided evolutionary algorithm for expensive dynamic multi-objective optimization," *Swarm Evol. Comput.*, vol. 75, Dec. 2022, Art. no. 101173.

[62] X. Zhang, G. Yu, Y. Jin, and F. Qian, "An adaptive Gaussian process based manifold transfer learning to expensive dynamic multi-objective optimization," *Neurocomputing*, vol. 538, Jun. 2023, Art. no. 126212.

[63] A. Gupta, Y.-S. Ong, and L. Feng, "Insights on transfer optimization: Because experience is the best teacher," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 51–64, Feb. 2018.

[64] K. C. Tan, L. Feng, and M. Jiang, "Evolutionary transfer optimization—A new frontier in evolutionary computation research," *IEEE Comput. Intell. Mag.*, vol. 16, no. 1, pp. 22–33, Feb. 2021.

[65] C. Yang, J. Ding, Y. Jin, C. Wang, and T. Chai, "Multitasking multiobjective evolutionary operational indices optimization of beneficiation processes," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1046–1057, Jul. 2019.

[66] Z. Chen, Y. Zhou, X. He, and J. Zhang, "Learning task relationships in evolutionary multitasking for multiobjective continuous optimization," *IEEE Trans. Cybern.*, vol. 52, no. 6, pp. 5278–5289, Jun. 2022.

[67] Y. Guo, G. Chen, M. Jiang, D. Gong, and J. Liang, "A knowledge guided transfer strategy for evolutionary dynamic multiobjective optimization," *IEEE Trans. Evol. Comput.*, early access, Nov. 16, 2022, doi: 10.1109/TEVC.2022.3222844.

[68] M. Jiang, Z. Huang, L. Qiu, W. Huang, and G. G. Yen, "Transfer learning-based dynamic multiobjective optimization algorithms," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 501–514, Aug. 2018.

[69] J. Li, T. Sun, Q. Lin, M. Jiang, and K. C. Tan, "Reducing negative transfer learning via clustering for dynamic multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 26, no. 5, pp. 1102–1116, Oct. 2022.

[70] L. Feng, Y.-S. Ong, S. Jiang, and A. Gupta, "Autoencoding evolutionary search with learning across heterogeneous problems," *IEEE Trans. Evol. Comput.*, vol. 21, no. 5, pp. 760–772, Oct. 2017.

[71] L. Zhou, L. Feng, A. Gupta, and Y.-S. Ong, "Learnable evolutionary search across heterogeneous problems via kernelized autoencoding," *IEEE Trans. Evol. Comput.*, vol. 25, no. 3, pp. 567–581, Jun. 2021.

[72] X. Xue et al., "Evolutionary sequential transfer optimization for objective-heterogeneous problems," *IEEE Trans. Evol. Comput.*, vol. 26, no. 6, pp. 1424–1438, Dec. 2022.

[73] X. Wang, Y. Jin, S. Schmitt, M. Olhofer, and R. Allmendinger, "Transfer learning based surrogate assisted evolutionary bi-objective optimization for objectives with different evaluation times," *Knowl. Based Syst.*, vol. 227, Sep. 2021, Art. no. 107190.

[74] R. L. Hardy, "Multiquadric equations of topography and other irregular surfaces," *J. Geophys. Res.*, vol. 76, no. 8, pp. 1905–1915, 1971.

[75] Z.-J. Wang, J.-R. Jian, Z.-H. Zhan, Y. Li, S. Kwong, and J. Zhang, "Gene targeting differential evolution: A simple and efficient method for large scale optimization," *IEEE Trans. Evol. Comput.*, early access, Jun. 23, 2022, doi: 10.1109/TEVC.2022.3185665.

[76] G. Li, Z. Wang, and M. Gong, "Expensive optimization via surrogate-assisted and model-free evolutionary optimization," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 5, pp. 2758–2769, May 2023.

[77] Y. Liu, J. Liu, Y. Jin, F. Li, and T. Zheng, "A surrogate-assisted two-stage differential evolution for expensive constrained optimization," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 7, no. 3, pp. 715–730, Jun. 2023.

[78] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[79] Z.-Z. Liu, Y. Wang, S. Yang, and K. Tang, "An adaptive framework to tune the coordinate systems in nature-inspired optimization algorithms," *IEEE Trans. Cybern.*, vol. 49, no. 4, pp. 1403–1416, Apr. 2019.

[80] F. Zhuang et al., "A comprehensive survey on transfer learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021.

[81] L. Li and Z. Zhang, "Semi-supervised domain adaptation by covariance matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2724–2739, Nov. 2019.

[82] P. Suganthan et al., "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Dept. Elect. Electron. Eng., Nanyang Technol. Univ., Singapore, Rep. 2005005, 2005.

[83] J. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[84] G. Cheng, A. Younis, K. H. Hajikolaei, and G. G. Wang, "Trust region based mode pursuing sampling method for global optimization of high dimensional design problems," *J. Mech. Des.*, vol. 137, Feb. 2015, Art. no. 21407.

[85] J. Branke, “Memory enhanced evolutionary algorithms for changing optimization problems,” in *Proc. Congr. Evol. Comput.*, vol. 3, Feb. 1999, pp. 1875–1882.

[86] T. T. Nguyen, S. Yang, and J. Branke, “Evolutionary dynamic optimization: A survey of the state of the art,” *Swarm Evol. Comput.*, vol. 6, pp. 1–24, Oct. 2012.

[87] S. Yang and C. Li, “A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments,” *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 959–974, Dec. 2010.

[88] Y. Liu, J. Liu, Y. Jin, F. Li, and T. Zheng, “An affinity propagation clustering based particle swarm optimizer for dynamic optimization,” *Knowl. Based Syst.*, vol. 195, May 2020, Art. no. 105711.

Yuanchao Liu received the B.S. degree in automation from Northeastern University at Qinhuangdao, Qinhuangdao, China, in 2017, and the master’s degree in control theory and control engineering and the Ph.D. degree in control science and engineering from Northeastern University, Shenyang, China, in 2019 and 2023, respectively.

He is currently a Lecturer with the College of Information Science and Engineering, Northeastern University. His current research interests include multiobjective optimization, robust optimization, dynamic optimization, and data-driven optimization.

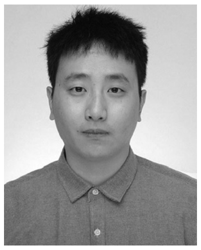
Jianchang Liu was born in Heishan, Liaoning, China, in 1960. He received the B.S., M.S., and Ph.D. degrees from Northeastern University, Shenyang, China, in 1980, 1989, and 1998, respectively.

He is currently a Professor with the College of Information Science and Engineering, Northeastern University. His research interests include multiobjective optimization, modeling, control and optimization for complex process, intelligent control theory and application, and fault diagnosis.

Jinliang Ding (Senior Member, IEEE) received the bachelor’s, master’s, and Ph.D. degrees in control theory and control engineering from Northeastern University, Shenyang, China, in 2001, 2004, and 2012, respectively.

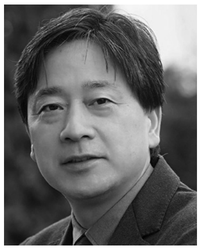
He is a Professor with the State Key Laboratory of Synthetical Automation for Process Industry, Northeastern University. He has authored or coauthored over 200 refereed journal papers and refereed papers at international conferences. He has also invented or co-invented over 50 patents. His research interests include modeling, plant-wide control and optimization for the complex industrial systems, machine learning, industrial artificial intelligence, and computational intelligence and application.

Dr. Ding was a recipient of the Young Scholars Science and Technology Award of China in 2016, the National Science Fund for Distinguished Young Scholars in 2015, the National Technological Invention Award in 2013, and three First-Prize of Science and Technology Awards of the Ministry of Education in 2006, 2012, and 2018, respectively. One of his articles published on *Control Engineering Practice* was selected for the Best Paper Award of 2011–2013. He is an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS.



Shangshang Yang (Member, IEEE) received the B.Sc. and Ph.D. degrees from Anhui University, Hefei, China, in 2017 and 2022, respectively.

He is currently a Postdoctoral Fellow with the School of Artificial Intelligence, Anhui University. His current research interests include evolutionary multiobjective optimization, neural architecture search, intelligent education, and graph learning.



Yaochu Jin (Fellow, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 1988, 1991, and 1996, respectively, and the Dr.-Ing. degree from Ruhr University Bochum, Bochum, Germany, in 2001.

He is an Alexander von Humboldt Professor of Artificial Intelligence endowed by the German Federal Ministry of Education and Research, with the Faculty of Technology, Bielefeld University, Bielefeld, Germany. He is also a Distinguished Chair and a Professor of Computational Intelligence with the Department of Computer Science, University of Surrey, Guildford, U.K. He was a Finland Distinguished Professor with the University of Jyväskylä, Jyväskylä, Finland; a Changjiang Distinguished Visiting Professor with the Northeastern University, Shenyang, China; and a Distinguished Visiting Scholar with the University of Technology Sydney, Ultimo, NSW, Australia. His main research interests include multiobjective and data-driven evolutionary optimization, evolutionary multiobjective learning, trustworthy AI, and evolutionary developmental AI.

Prof. Jin is the recipient of the 2018 and 2021 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award, and the 2015, 2017, and 2020 *IEEE Computational Intelligence Magazine* Outstanding Paper Award. He was named by the Web of Science as “a Highly Cited Researcher” from 2019 to 2022 consecutively. He is currently the President-Elect of the IEEE Computational Intelligence Society and the Editor-in-Chief of *Complex & Intelligent Systems*. He was the Editor-in-Chief of the IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS, an IEEE Distinguished Lecturer in 2013–2015 and 2017–2019, and the Vice-President for Technical Activities of the IEEE Computational Intelligence Society from 2015 to 2016. He is a Member of Academia Europaea.