Surrogate Models in Evolutionary Single-Objective Optimization: A New Taxonomy and Experimental Study

Hao Tong[a], Changwu Huang[a], Leandro L. Minku[b], Xin Yao[a,*]

[a]*Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China.*
[b]*School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK*

Abstract

Surrogate-assisted evolutionary algorithms (SAEAs), which use efficient surrogate models or meta-models to approximate the fitness function in evolutionary algorithms (EAs), are effective and popular methods for solving computationally expensive optimization problems. During the past decades, a number of SAEAs have been proposed by combining different surrogate models and EAs. This paper dedicates to providing a more systematical review and comprehensive empirical study of surrogate models used in single-objective SAEAs. A new taxonomy of surrogate models in SAEAs for single-objective optimization is introduced in this paper. Surrogate models are classified into two major categories: absolute fitness models, which directly approximate the fitness function values of candidate solutions, and relative fitness models, which estimates the relative rank or preference of candidates rather than their fitness values. Then, the characteristics of different models are analyzed and compared by conducting a series of experiments in terms of time complexity (execution time), model accuracy, parameter influence, and the overall performance when used in EAs. The empirical results are helpful for researchers to select suitable surrogate models when designing SAEAs. Open research questions and future work are discussed at the end of the paper.

Keywords: Evolutionary algorithms, Surrogate models, Absolute fitness models, Relative fitness models, Expensive optimization problems

1. Introduction

Evolutionary Algorithms (EAs) require a so-called fitness function to evaluate the quality of candidate solutions when solving a problem. A large number of fitness evaluations is usually required to obtain a satisfying solution. However, in many real-world problems, the fitness function is very complex and expensive to compute. For example, a single Navier-Stokes fitness evaluation will take several hours [1], and the fitness evaluation of vehicle shape design optimisation [2] involves expensive simulation. Such problems are referred to as computationally expensive problems (CEPs).

Computational cost is a crucial challenge in applying EAs to solve CEPs. Surrogate-assisted evolutionary algorithms (SAEAs) are developed to reduce the computational cost of solving CEPs. In SAEAs, surrogate models or meta-models are approximate models that can simulate the behaviour of the real or true fitness function, but can be evaluated much faster and more cheaply. The easy-to-compute surrogates are used to evaluate some candidate solutions instead of using the real fitness function, so that the computational cost of the optimization procedure is reduced.

The use of surrogate models in EAs has attracted increasing attention over the past decades, and a large number of SAEAs have been introduced. SAEAs have been applied to computationally expensive single-objective optimization [3, 4, 5], multi-objective optimization [6, 7] and combinational optimization [8] problems. In addition, SAEAs

*Corresponding author.
Email addresses: htong6@outlook.com (Hao Tong), xiny@sustech.edu.cn (Xin Yao)

have been applied to problems other than CEPs, including interactive optimization problems, dynamic optimization [5], network architecture search [9] and others. SAEAs have also been successfully adopted to solve expensive real-world optimization problems from different domains, including engineering design [2], health industry [10] and interactive design [11, 12].

Many surrogate modelling approaches have been introduced to SAEAs over the past few years. Surrogate models are usually used to predict new candidate solutions' fitness values by approximating the real fitness function. For this purpose, regression models, such as polynomial regression, Gaussian Process regression (a.k.a. Kriging model), and radial basis functions, are commonly used in SAEAs [13]. This kind of surrogate model that directly predicts candidates' fitness values is referred to as absolute fitness models in this paper. In recent years, some new surrogate models have been introduced to SAEAs by providing the relative rank or preference of candidate solutions rather than predicting their fitness values [14, 15, 16, 17]. In this paper, such surrogate models are referred to as relative fitness models. Both the modelling techniques and how the resulting relative fitness models use within SAEAs are different from commonly used absolute fitness models. However, the existing surveys [13, 3, 5, 8] mostly focus on absolute fitness models that provide absolute fitness value predictions. There is no comprehensive review that systematically discusses both absolute and relative fitness models to the best of our knowledge. In addition, there is still no guideline for choosing suitable surrogate models when designing SAEAs.

This motivates us to give a new and more comprehensive review that discusses both absolute and relative fitness models used in SAEAs for single-objective optimization problems. The remainder of this paper is organized as follows. Section 2 gives a brief introduction to the general SAEA framework and presents our new taxonomy for surrogate models in SAEAs. Sections 3 and 4 review absolute fitness models and relative fitness models, respectively. Section 5 compares different surrogate models from several perspectives, including time complexity, model accuracy, parameter influence, and overall performance, through empirical studies. Section 6 closes the paper with conclusion and future work.

2. General framework of single-objective SAEAs and categorisation of surrogate models

2.1. Overview

The general framework of SAEAs for single-objective optimization is illustrated in Fig. 1. Compared with canonical EAs, SAEAs introduce new steps or issues, including the construction of surrogate models, the interaction between surrogate models and EA, and re-evaluation of some candidate solutions. The construction step involves building or updating surrogate model(s), the interaction concerns the mechanism of incorporating surrogate model(s) into the EA, and the re-evaluation refers to identifying some individuals (candidate solutions) to be evaluated by the real fitness function in the optimization process. The interaction and re-evaluation steps together are also known as model management or evolution control in the literature [3].

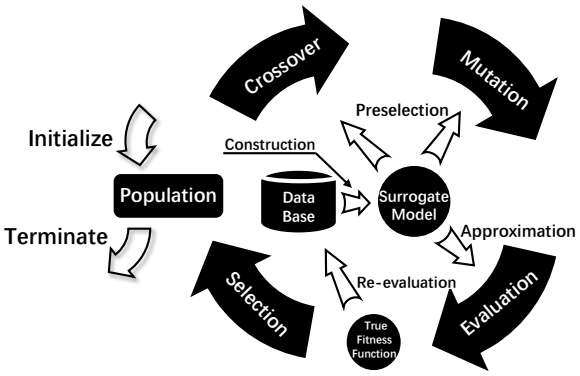


Figure 1: The general framework of SAEAs. The crossover, mutation, evaluation and selection are common operators of a basic EA. The SAEA integrates a surrogate model into the basic EA.

Model Construction. Before using surrogate models in the evolutionary search, SAEAs need to construct such surrogate models based on a set of candidate solutions or samples evaluated by the actual fitness function, which is also called training data set. The initial training data set is used to train the initial surrogate model. It can be collected by running the EA for several generations without using surrogate models or by applying some sampling strategies (also known as Design Of Experiments, DOE), such as Latin Hypercube Sampling (LHS) [18]. Sampling strategies aim at maximizing the amount of information gained from an experimental study while minimizing the amount of data to be collected [19], and thus are commonly adopted in SAEAs. Building high-fidelity surrogate models that fit the real fitness function well usually requires a more extensive training data set and involves a long training time. Hence, models with different levels of fidelity can be used to help search for optimal solutions in SAEAs [20]. Nevertheless, low-fidelity models can potentially misguide the algorithm towards solutions that are not real optimum, but would seem optimal based on the model's judgement. We refer to these solutions as "false optimal solutions or optimums". model selection or model ensemble can be adopted in SAEA to increase the accuracy of models and reduce the risk of converging to a false optimum [21, 22]. Yu *et al.* [23] made a comparison of three model quality measurements used for model selection concerning the model's ability in evaluating and ranking the solutions. They found that the convergence of SAEAs using the model selected by model quality measurements is not as good as SAEAs assisted by the model with the highest fidelity.

Interaction (Pre-selection and Estimation). The constructed surrogate model can be used in the individual creation (i.e., in crossover or mutation) and evaluation steps of an EA, as shown in Fig. 1. Using the surrogate model in the creation of individuals is known as pre-selection, and using the surrogate model in the evaluation can be referred to as estimation of the fitness (or relative fitness) of the solution. Generally, in pre-selection, several new individuals (offspring) are generated by variation operators, and then the surrogate model is used to pre-select the most promising offspring. After the pre-selection, the exact fitness values of the selected individuals are still computed using the actual fitness function in the evaluation step, ensuring that the optimization process can converge to the true optimum. This method reduces the computational cost by improving the quality of individuals evaluated by the true function.

When using surrogate models in the evaluation step of an EA, a portion of the fitness evaluations are provided by the surrogate model rather than the actual fitness function. This is to reduce the high computational cost related to the evaluation of all individuals, which can be very expensive when solving CEPs. By replacing expensive fitness evaluations with efficient surrogate model predictions, the computational cost can be reduced significantly [3]. However, if all the individuals are evaluated by surrogates, the SAEA may converge to a false optimum. Therefore, surrogate models should be used alongside the actual fitness function in fitness evaluation to prevent false convergence..

Re-evaluation. This is a key step that affects the convergence of SAEAs. Individual-based and generation-based evolution control are the two commonly used strategies for determining individuals to be re-evaluated by the true fitness function [5]. Individual-based evolution control re-evaluates some individuals in each generation. The most intuitive strategy is selecting individuals which are the fittest according to the fitness predictions from the surrogate model. For example, in Particle Swarm Optimization (PSO) [24], if the personal best and global best individuals are challenged, the challengers have to be re-evaluated to determine whether the best individual should really be updated or not. The uncertainty information is also considered together with the predicted fitness (or relative fitness) by using methods such as Lower Confidence Bound (LCB) [25].

Similarly, recent re-evaluation strategies [22] also apply methods to determine which solutions are more valuable to be re-evaluated, aiming at achieving a good trade-off between exploitation and exploration. Another different approach is based on clustering techniques [26]. The population is clustered into several groups, and then, in each group, the individual closest to the centre is re-evaluated. Different from individual-based control, generation-based control re-evaluates the whole population (i.e., all individuals) in a generation [27]. The frequency of re-evaluation for generation-based control can be fixed or adaptive. In fixed generation-based control, the whole population is re-evaluated once in a fixed number of generations. In adaptive generation-based control, whether a generation is controlled (re-evaluated) or not is determined based on the fidelity of the surrogate models.

2.2. A New Categorisation of Surrogate Models in SAEAs

It is no doubt that the surrogate model is a key element in SAEAs. The construction, interaction, and re-evaluation steps of SAEAs are all related to the surrogate model. In the literature, plenty of surrogate models have been adopted to

assist EAs. In existing reviews, SAEAs and surrogate models are categorized based on different criteria. For example, Jin *et al.* [13, 3] categorized SAEAs into individual-based, generation-based and population-based evolution control assisted algorithms. Jin *et al.* [5] and Thomas *et al.* [8] discussed single- and multi-model(s) in SAEAs according to the number of models used for fitness approximation. Lim *et al.* [4] researched the generalized evolutionary framework focusing on *curse of uncertainty* and *bless of uncertainty*. Cheng *et al.* [28] discussed models used in EAs in the literature, including models that estimate the distribution, models that map from the objective space to the decision space, and surrogate models. In addition, there also exist many surveys about multi-objective SAEAs. Chugh *et al.* [6] reviewed multi-objective evolutionary algorithms (MOEAs) for computationally expensive problems based on the kind of approximation (i.e., function approximation, problem approximation and fitness approximation) used in MOEAs. In Deb *et al.*'s work [7], six meta-modelling frameworks are summarised based on the cardinality of meta-models for objectives and constraints in MOEAs. Jin *et al.* [29] also discussed SAEAs for some real-world computationally expensive multi-objective problems.

Surrogate models in SAEAs can predict the fitness values or the preferences of candidate solutions generated in evolutionary search. In some practical problems, such as interactive optimization where the fitness is given by humans, the quality of solutions is hard to estimate as numerical value, and is usually captured by relative ranking. In this case, only the surrogate models that predict the preferences or ranking of candidate solutions are suitable. Thus, using surrogate models to predict absolute fitness values or relative fitness values is an important criterion for choosing surrogate models in SAEAs. Moreover, relative fitness models define a new type of surrogates that can assist EAs in solving expensive problems. It also introduces a series of new model management strategies that are different from those used for absolute fitness models, as will be explained in Section 4. Therefore, whether the surrogate model estimates absolute or relative fitness is an important consideration that widely influences the design of the SAEA.

With this in mind, this paper categorizes and systematically compares surrogate models in SAEAs based on the following taxonomy:

- Absolute Fitness Models, which directly predict the fitness function values of candidate solutions in the optimization process.
- Relative Fitness Models, which provide an estimation of rank or preference of candidates rather than the absolute fitness values.

Each main category is further divided into two subcategories, as shown in Fig. 2. Absolute fitness models are divided into regression-based and similarity-based models. Relative fitness models are divided into rank-based and classification-based models. The next two sections will give a brief review of absolute and relative fitness models, respectively.

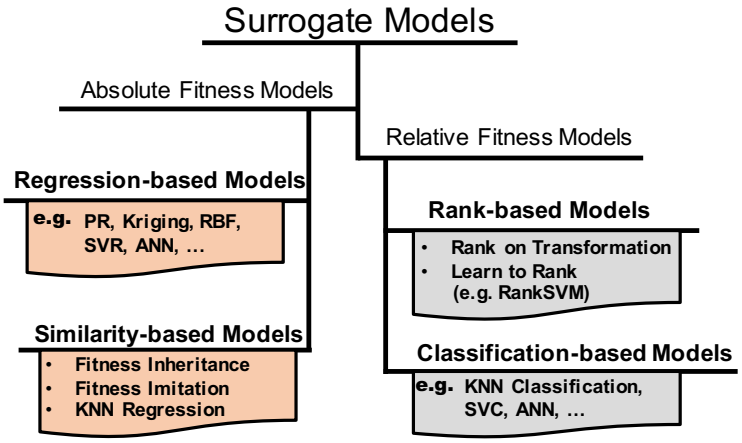


Figure 2: Categorisation of surrogate models in SAEAs

3. Absolute Fitness Models

Absolute fitness models are commonly used in SAEAs. They aim at predicting the fitness values of individuals. Let F denotes the actual fitness function. The goal of constructing an absolute fitness model is to learn a model $\hat{F}$ from some evaluated samples (called training data) to approximate the input-output relationship of the function F, so that the model can predict the fitness value of any new individual. An illustrative flow diagram of SAEAs using absolute fitness models is given in Fig. 3, in which the surrogate model constructed on the training set of evaluated samples is used to predict the absolute fitness of new offspring. Then, some of the offspring are evaluated by the real fitness function in the re-evaluation step and the re-evaluated individuals are added to the training data set. The selection operation is performed based on the predicted and exact fitness values. According to modeling approaches, absolute fitness models can be divided into two subcategories: regression-based and similarity-based models.

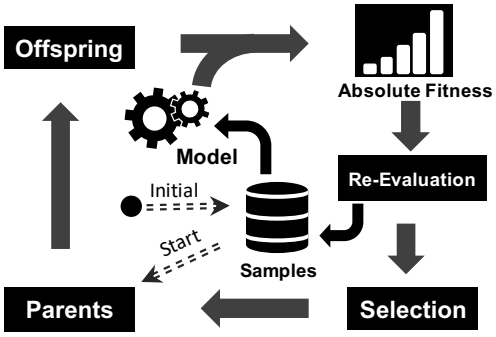


Figure 3: Illustration of SAEA using absolute fitness model.

3.1. Regression-based Models

Regression-based models are widely used to model the relationship between the input and output of a system [30, 13]. In SAEAs, a regression-based model is the approximate model $\hat{F}$, which maps a solution vector (d dimensional candidate solution or individual) $\mathbf{x} \in \mathbb{R}^d$ to its fitness value (output) $y = F(\mathbf{x}) \in \mathbb{R}$. Given the training data set $\mathcal{D} = [(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_n, y_n)]$, the objective of training or constructing a regression model $\hat{F}$ is to minimize the error function (also called loss function) $E_{\hat{F}}$, which represents the discrepancy between real fitness values and model predictions. The most commonly adopted error function is the mean squared error (MSE) defined as

$$E_{\hat{F}} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{F}(\mathbf{x}_i))^2 \tag{1}$$

where n is the number of samples in training data set $\mathcal{D}$, and y_i and $\hat{F}(\mathbf{x}_i)$ denote the true and predicted fitness values of individual $\mathbf{x}_i \in \mathcal{D}$, respectively.

Based on regression techniques, many surrogate models are used to assist EAs in the literature [30], including Polynomial Regression (PR), Support Vector Regression (SVR), Kriging Model, Radial Basis Function (RBF), etc. SVR is an effective surrogate model which has some practical applications, such as the optimization of railway wind barriers [31], but its training process will be expensive when the dimensionality of the problem is very high. Kriging is a popular surrogate model in SAEAs because it can estimate the uncertainty as well as the fitness. RBF models have also been used in SAEAs to solve many real-world problems [32].

However, there is no clear conclusion on which regression model is the best because each model has its own strengths and weaknesses according to experimental studies [33]. Jin *et al.* [34] analyzed PR, Kriging and RBF on fourteen test problems. They showed that RBF is the best for high-order non-linear problems both for small and large scales. For low-order non-linear problems, PR is the best for local modeling, which is only used to predict the fitness in a small area, and the Kriging method is the most suitable for global modeling, which can be used to predict the fitness across the whole search space. The results also showed that the Kriging method is the most sensitive to noisy data. Diaz *et al.* [35] obtained similar results, showing that the Kriging method is relatively better for low dimensional problems. At the same time, RBF is more suitable for high-dimensional problems.

3.2. Similarity-based Models

Similarity-based surrogate models are another sub-category of absolute fitness models [36]. Similar to regression-based models, similarity-based models also involve a function $\hat{F}$ which provides an approximate fitness value $\hat{F}(\mathbf{x}) \in \mathbb{R}$ to a solution $\mathbf{x} \in \mathbb{R}^d$. However, different from regression-based models, the model $\hat{F}$ is constructed based on the correlation between unevaluated and evaluated individuals. A general formulation of similarity-based models is as follows:

$$\hat{F}(\mathbf{x}) = \sum_{i=1}^{N_c} Cor(\mathbf{x}, \mathbf{x}_i) F(\mathbf{x}_i) \tag{2}$$

where N_c denotes the number of individuals used to derive the fitness of a new individual $\mathbf{x}$, $Cor(\mathbf{x}, \mathbf{x}_i)$ represents the correlation between $\mathbf{x}$ and $\mathbf{x}_i$, and $F(\mathbf{x}_i)$ denotes the true fitness of the individual $\mathbf{x}_i$.

In the literature, the commonly used similarity-based modelling techniques include *Fitness Inheritance*, *k-Nearest Neighbours* regression (*k*NN-R) and *Fitness Imitation*. Fitness inheritance, which was first proposed in 1995 [37], estimates the fitness of a new solution based on the fitness of its parents. Several SAEAs using fitness inheritance have been proposed during the last decades. For example, Sun *et al.* [38] proposed an efficient algorithm that combined fitness inheritance and the PSO algorithm to solve large-scale expensive problems. In *k*NN-R, the fitness of a new individual is determined by its k nearest neighbour individuals, which can be seen as a generalized form of fitness inheritance [36]. In fitness imitation, the population is clustered into several groups, and one individual is selected as the representative of each group. The representative individuals are evaluated using the true fitness function, whereas the fitness values of the remaining individuals are estimated based on the similarity measurements between themselves and the representative individuals [3].

Similarity-based models are easy-to-construct since they are essentially based on the similarities between individuals. They do not require to pre-define specific formulas or to optimize the model parameters with training data. Instead, they only need a method to calculate the similarities. Thanks to the limited size of sample data in expensive problems, the time for finding related neighbours to predict offspring's fitness in similarity-based models is not particularly costly. Thus, the computational cost in creating and using similarity-based models is much cheaper than regression-based models in most SAEAs. The similarity measurement is a key issue for similarity-based fitness approximation, which directly affects the performance of surrogates.

Nonetheless, the accuracy of similarity-based models deteriorates significantly when the problem is highly nonlinear and/or the search space is enormous. As a result, it is better to employ similarity-based models as local surrogates. Due to models' structural characteristics, each similarity-based model has its range of applications. Taking the Artificial Immune System as a typical example, fitness inheritance is not appropriate because each offspring only has one parent available [39].

3.3. Discussions

Absolute fitness models are straightforward methods for fitness approximation and play an essential role in SAEAs. Both regression-based and similarity-based models provide approximate fitness values for individuals (solutions), and then the predicted fitness values are used in population creation or individual selection in EAs.

Compared with regression-based models, similarity-based models are more straightforward and more comfortable to construct and use since only the similarity between individuals is considered for predictions. On the contrary, regression-based models are more accurate than similarity-based models to some extent because they optimize parameters in the surrogate model to capture the potential patterns of actual fitness functions. Thus, regression-based models are suitable for non-linear, multi-dimensional problems, while similarity-based models are commonly used as a local approximation in small regions.

Although absolute fitness models are powerful methods to approximate the fitness functions for CEPs, it is not straightforward to use them in SAEAs, since some other aspects like fidelity and/or complexity of the model also need to be taken into account. Therefore, in model management, it is important to balance the accuracy and efficiency of absolute fitness models in SAEAs [5].

4. Relative Fitness Models

Relative fitness models focus on predicting the relative preference between individuals rather than their absolute fitness values. In EAs, the fitness of individuals, which represents the quality of candidate solutions, is the basis for determining whether an individual is selected or eliminated in the selection operation. Generally, fitter individuals, i.e., individuals with better fitness values, are selected to enter the next generation. For some EAs, such as the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [40] and Self-Learning Particle Swarm Optimizer (SLPSO) [41], selection depends on the ranking of individuals. In this case, the result of selection only related to the rank of individuals. In other words, if the absolute fitness values of individuals changes, but their ranking remains the same, the result of selection does not change. Moreover, there are some practical problems in which fitness values are hard to be calculated, such as the human interactive problems [11]. Relative fitness methods, which focus on the comparative result of the current population, are suitable in these cases. A general framework of relative fitness model assisted EAs is shown in Fig. 4, in which the surrogate model predicts the relative fitness of new offspring. Then some individuals are re-evaluated by the actual fitness function.

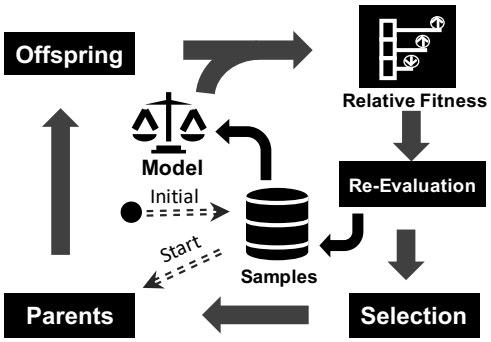


Figure 4: Illustration of SAEA using relative fitness model.

Unlike absolute fitness approximation, a relative fitness model provides the preference of a new individual in a specific group rather than its absolute fitness value. In the existing literature, the relative fitness approximation is given as a rank label $\{r_1, r_2, \cdots, r_N\}$ or a classification label $\{+1, -1\}$. According to this, relative fitness models are further categorized into rank-based and classification-based models.

4.1. Rank-based Models

Rank-based models predict the relative rank of a new population-based on evaluated samples, and then the individuals with a higher rank survive to the next generation. Generally, a population of size N is ranked by a surrogate model $R(\cdot)$ from best to worst as presented in Eq. (3),

$$R(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N) = (\mathbf{x}_{1:N}, \ldots, \mathbf{x}_{\mu:N}, \ldots, \mathbf{x}_{N:N}) \tag{3}$$

where $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N$ represent individuals in a new population (without true fitness) and $\mathbf{x}_{\mu:N}$ represents the μ-th best individual in the population ranked by the model.

For a rank-based model, the predicted order or rank is more important than the fitness value, so that a proper error function named ranking preservation [23] as shown in Eq. (4) can be used as the loss function.

$$E_R = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} \theta(y_i - y_j, R(\mathbf{x}_i) - R(\mathbf{x}_j)) \tag{4}$$

where $\theta(a, b) = (a \leq 0)$ xor $(b \leq 0)$, y_i, y_j are true fitness of $\mathbf{x}_i$ and $\mathbf{x}_j$, and $R(\mathbf{x}_i), R(\mathbf{x}_j)$ are ranks predicted by the surrogate model.

There are two different rank-based models motivated from different perspectives in the literature. The first one, which we named as *Rank on Transformation*, transfers the original objective space into another equivalent but different

space. In the new space, individuals' rank is maintained but it is much easier to be approximated. Gong *et al.* [42] transferred an expensive objective function into a probability density function and then built a cheap rank-based model by approximating the probability density function. The second type of rank-based models directly predicts the rank of individuals by using learning to rank algorithms. The first work was in 2006 when Runarsson [14] proposed a rank-based surrogate model (by using ordinal regression) to assist an evolution strategy. After that, other work followed, e.g., Loshchilov *et al.* [43] adopted RankSVM to enhance CMA-ES for solving CEPs and Lu *et al.* [17] employed RankSVM to pre-select the high quality individuals in differential evolution algorithms. Huang *et al.* [44] adopted the approximate ranking procedure based on a Kriging model to assist the search process of CMA-ES.

In rank-based models, it is particularly important to select appropriate training samples. If the training samples are too close to each other, the discrepancy between them is hard to distinguish. If samples are too far away from each other, the precision of the rank model may significantly deteriorate. As a result, it is hard to obtain a rank-based surrogate model with high accuracy, which affects the convergence of SAEAs. Therefore, rank-based models are usually applied for pre-selection in the literature [17].

4.2. Classification-based Models

Classification-based models were first discussed in 2003 [12] and have developed greatly in recent years. Different from rank-based models, classification-based models just consider the comparison result between individuals and a reference solution. The mechanism of classification-based models is shown in Fig. 5. A classifier is built based on a single reference individual and its neighbours. If the neighbour's fitness is greater than the reference, it is assigned with label +1, otherwise, label -1 is assigned to it. A different classifier is built for each reference point. New individuals are then assigned label +1 or -1 predicted by each classifier.

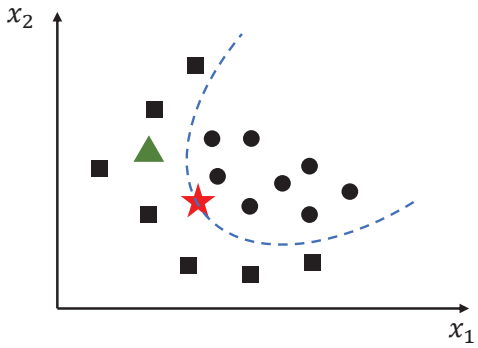


Figure 5: Classification-based model. Star denotes the reference, triangle denotes the new individual, squares represent the samples with label -1, circles represent the samples with label +1.

The task of building a classification-based model can be considered as: train a model for determining whether the new individual is better than the reference solution or not. The classifier $C : \mathbb{R}^d \rightarrow \{+1, -1\}$ provides a label $+1$ or -1 for any new individual $\mathbf{x} \in \mathbb{R}^d$, where label $+1$ and -1 represents that the new individual is better or worse than the reference, respectively [12]. A straightforward error function to measure the quality of the classification model is the following:

$$E_C = \frac{1}{N} \sum_{n=1}^{N} \mathbb{I}(L_n, C_n) \tag{5}$$

where $\mathbb{I}(a, b)$ is an indicator function which has value 1 if a equals b and the value 0 otherwise, L_n is the true label for training data, C_n is the predicted label. Commonly used classification models in the literature include Support Vector Classification [15], kNN classification [45], and etc. It is worth to mention that the kNN classification (kNN-C) model predicts the label +1 or -1 of a new individual, while the kNN regression (kNN-R) model gives an approximate fitness of the new individual.

Lu *et al.* [15] proposed a classical classification-based SAEA for single-objective optimization called classification-assisted differential evolution (CADE). It applied soft-margin support vector classification (SVC) as the classifier and

the parent is taken as the reference individual for each offspring in each generation. The CADE mainly contains three steps, including training set choosing, classifier training and exact re-evaluation. If the predictive result for an individual is better than its parent, this individual will be re-evaluated by the real function, Otherwise, it will be deleted directly. Recently, some researchers applied a fuzzy-classification algorithm (Fuzzy-kNN) to filter out unpromising individuals before the actual evaluation, because they consider fuzzy membership more reliable than only using the labels [46]. Experiments showed that a common algorithm's (DE's) efficacy was significantly boosted after applying the classification-based surrogate model.

In classification-based SAEAs, class imbalanced training data (i.e., the problem of a given class being underrepresented in the training data compared to the other class) is often encountered when creating classifiers. The samples for constructing a classification-based model are selected from the neighbours of a reference solution. As the evaluated points are distributed sparsely in the search space, the situation where all the selected samples are worse or better than the reference point may happen, forming a class-imbalanced dataset. Lu *et al.* [15] employed a threshold-moving method to address the imbalance dataset problem.

4.3. Discussions

We have reviewed the relative fitness approximate models in SAEAs in this section. Compared with regression-based absolute fitness models, relative fitness models are much simpler because they only focus on the rank or ordinal relation of individuals rather than their absolute fitness values. As they are typically not very accurate, they are usually used as local models in SAEAs for pre-selection. In contrast, regression-based models, in general, are more elaborate and precise than relative fitness models. Thus, regression models are usually used as global models in SAEAs.

The application of relative fitness models so far has been limited to some EAs. The rank-based model was mainly used in CMA-ES [43] and the main achievements of classification-based models are still limited to DE [16, 17] and some special MOEAs [47]. It is necessary to mention that relative fitness models cannot describe the global trend and provide information about the whole fitness landscape. They are only used as a local model around querying individuals or solutions in SAEAs.

In summary, relative fitness approximation methods have some advantages and are suitable for some practical problems. For example, in human interactive problems, the fitness strongly depends on the human sense and is difficult to quantify [12], making relative fitness approximation methods more suitable than absolute ones. In constrained CEPs, it is natural to consider the feasibility of solutions as a classification problem and the violation degree as rank problems in relative fitness approximation.

However, there are still many issues under-explored in relative fitness models for SAEAs. For example, the model management strategies for classification-based and rank-based models are currently very limited, which is a potential direction for future research. Moreover, it is also valuable to research relative fitness models in other optimization problems, not only in CEPs, such as constrained optimization problems. In constrained optimization problems, the feasibility of an individual can be formulated as a classification problem with two classes, i.e. feasible or infeasible [48].

5. Experimental Study and Analysis

In this section, we conduct a series of experiments to analyze the performance of absolute and relative fitness models. In the actual application, it is crucial to know the characteristic of candidate models before selecting the surrogates, such as the construction efficiency and prediction accuracy of models. Besides, the performance of the corresponding SAEA is the most important. Therefore, in this section, we will compare the performance of different surrogate models on single-objective optimization benchmark functions concerning the models' construction efficiency, prediction accuracy, the effect of hyper-parameters for each model and the corresponding SAEAs' optimization ability.

5.1. Experimental setting

5.1.1. Surrogate models and model management strategies

To compare different surrogate models, we select one representative model from each category in our experiments. According to a recent paper comparing meta-modelling techniques in EAs [35], RBF is the most robust and scalable regression model. So, we select the RBF model as the representative regression-based model. However, there is

no paper comparing other kinds of surrogate models. Therefore, we choose the most commonly used and typical models, that is, kNN-R, RankSVM, and SVC as the representative similarity-based model, rank-based model, and classification-based model, respectively.

In addition to modelling techniques, there also exist several model management strategies in SAEAs. In our experiments, we choose two typical strategies: *individual-based evolution control* and *pre-selection*. In individual-based evolution control (IB), the number of individuals being re-evaluated by real fitness functions is determined by parameter p_{sm} for an absolute fitness model, and by τ_{th} for a rank-based model, where τ is Kendall's coefficient [14]. For the pre-selection strategy (PS), the parameter generator factor λ_{pre} determines how many offspring are generated by mutation and crossover operators for an absolute fitness model and a rank-based model. Classification-based models are only used for pre-selection in the literature [15], and the generator factor is not necessary, because the classification-based model is just used to select the better individual between an individual and its parent. The surrogate models, model management strategies and associated parameters that are investigated in the experimental study are list in Table 1.

Table 1: The surrogate models, model management strategies and parameters that are studies in experiments.

Model Category	Representative Model	IB	Parameter of IB	PS	Parameter of PS
Regression-based Model	RBF	Yes	p_{sm}	Yes	λ_{pre}
Similarity-based Model	kNN-R	Yes	p_{sm}	Yes	λ_{pre}
Rank-based Model	RankSVM	Yes	τ_{th}	Yes	λ_{pre}
Classification-based Model	SVC	No	–	Yes	–

The genetic algorithm (GA) is selected as the basic EA to be combined with different surrogate models through different model management strategies listed in Table 1. The pseudo-codes of GA assisted with different models and model management strategies are presented in Algorithms 1 and 2. Algorithm 1 uses individual-based evolution control (IB) as the model management strategy. The absolute fitness model predicts the fitness of all offspring obtained from mutation and crossover operators and a portion of them, i.e., $p_{sm} \cdot N$ individuals, are re-evaluated by the real fitness function. The rank-based model re-evaluates the offspring sequentially until the model accuracy, i.e., Kendall's τ, satisfies its threshold τ_{th}. Finally, the best N individuals are selected to form the new generation. The pre-selection (PS) based SAEA, as shown in Algorithm 2, generates λ_{pre} offspring for each individual and select the best one as the new offspring when using the absolute fitness model and rank-based model. The classification model, specifically, is built for each parent and then used to determine if its offspring should be removed or re-evaluated.

5.1.2. Benchmark functions and running condition

Several typical functions selected from CEC2015 expensive optimization test suite [49] were used in the experiments. Specifically, two unimodal (f_1 and f_2), two multimodal (f_4 and f_8) and two composition (f_{13} and f_{15}) test functions with two dimensions $d = 10$ and $d = 20$ are tested. In all SAEA instances, the surrogate model is used as a local model, that is, a model is constructed for each new individual. Nevertheless, the samples in the database are uniformly distributed in the whole space in the early search stage so that the "local area" for such a model is the whole search space. Therefore, the constructed model can also be regarded as a global model in the early search stage to some extent. The model in later search stage is more close to a local model. In our experiments, we execute independently each algorithm on each problem instance for 25 times and the number of fitness evaluations for each run is set to $200d$, where d denotes the dimensionality of the problem instance. A LHS with $5d$ samples is performed to collect the initial training data set for each SAEA. The population size N is set to 40, crossover rate pc is set to 0.7, and mutation rate pm is set to 0.3. The code used in the experiments will be made available in Github[1], where the RBF model is from a widely used toolbox[2], kNN-R was implemented by ourselves based on the formulation in [36], and the Rank-SVM model is from the author of [15], and SVC is provided by Matlab r2016b. All parameters for the models were set to the default values provided by the toolboxes.

[1] https://github.com/HawkTom/CodeforSAEAsurvey
[2] https://sites.google.com/site/srgtstoolbox/

Algorithm 1: SAEA using Individual-based Evolution Control

```
Initialize the first population with N individuals.
Evaluate N individuals.
Set FE = N.
while FE < maxFE do
    • Perform Crossover and Mutation operators to generate N offspring.
    / *** Absolute Fitness Surrogate Model *** /
    if absolute fitness model then
        Predict the fitness of each offspring.
        Re-evaluate p_sm * N best individuals.
        FE = FE + p_sm * N
    / *** Rank-based Model *** /
    if rank-based model then
        Kendall's τ = 0.
        while τ ≤ τ_th do
            Controlled individual ← the highest rank individual.
            Re-evaluate the controlled individual.
            FE = FE + 1.
            Caculate the Kendall's τ.
    • Select N best individuals from parents and offspring to enter into the next generation.
```

5.2. Analysis 1: computation time

First of all, we analyze the execution time of each investigated surrogate model. To this end, we recorded the CPU time from the moment when the algorithm chooses the training data to the moment when it predicts fitness values or ranks the offspring. The experiments were executed in a computer with Intel Core i5 CPU with 3.2 GHz and 8 GB RAM using Matlab r2016b. The sizes of training data were set to $5d$ for all models. The average execution time for training and using the surrogate model in different problems and dimensionalities are presented in Table 3.

Accoridng to the results listed in Table 3, kNN-R model is the most efficient surrogate model. This is because kNN-R is a lazy model that does not learn a discriminative function from the training data but "memorizes" the training data set. The only computational burden is calculating the similarity between a new individual against the memorized training data and then predicting the fitness using the similarities. Usually, the size of training data set used in SAEAs is relatively small, which makes kNN-R model's predictions very quickly. The RBF model is also very efficient. Its computational cost mainly comes from the computation of the distance matrix between training data points and its inversion, which leads to longer execution time than kNN-R. The SVC model takes longer computing time than RBF model. The main reason is that the SVC model involves a time-consuming quadratic programming optimization process. RankSVM is the most expensive model among the four models investigated in this study, because it involves the pairwise operation in training and testing procedures. As a result, the execution time of RankSVM was much longer than the other three models.

5.3. Analysis 2: model accuracy

The performance of SAEAs is affected by how well the surrogate models can perform their predictions. An inaccurate surrogate model will lead to harmful consequences for the SAEA. Existing surveys used the coefficient of determination (R^2) to evaluate the predictions given by different surrogate models [33, 35]. However, the ability to maintain individuals' rank information is much more important than how well the surrogate model fits the real fitness function in EAs. So, we use rank correlation instead of R^2 for comparing the four surrogate models in this section.

Algorithm 2: SAEA using Pre-selection

```
Initialize the first population with N individuals.
Evaluate N individuals.
Set FE = N.
while FE < maxFE do
    / *** Absolute Fitness Surrogate Model *** /
    / *** or Rank-based Model ***/
    if absolute fitness model or rank-based model then
        for each individual do
            Generate λ_pre offspring candidates.
            Build the surrogate model.
            if absolute fitness model then
                Predict the fitness of all candidates.
                Select the best candidate as the offspring.
            if rank-based model then
                Caculate the Kendall's τ.
                if τ > 0.5 then
                    Select the best candidate as the offspring.
                else
                    Randomly select a candidate as the offspring.
            Re-evaluate the offspring.
            FE = FE + 1.
        • Select N best individuals from parents and offspring to enter into the next generation.
    / *** Classification-based Model *** /
    if classification-based model then
        • Perform Crossover and Mutation operators to generate N offspring.
        for each individual do
            Find its parent as the reference individual.
            Build classification-based model.
            Predict the label of offspring.
            if predictive label is positive then
                Re-evaluate the offspring.
                FE = FE + 1.
                Select best individual from parent and offspring to enter new generation.
            else
                Select parent into new generation.
        • Select N best individuals from parents and offspring to enter into the next generation.
```

Table 2: The details of benchmark functions [49]. F^* denotes the optimal fitness of the problem instance.

Modality	Problem	F^*	Dimensionality (d)
Unimodal	f_1	100	10, 20
	f_2	200	
Simple Multimodal	f_4	400	
	f_8	800	
Composition Function	f_{13}	1300	
	f_{15}	1500	

Table 3: The average execution time for training and using the surrogate model when $d = 10, 20$. (Unit: *ms*)

d	Problem	RBF	kNN-R	RankSVM	SVC
10	f_1	0.957	**0.280**	7.852	4.436
	f_2	0.974	**0.280**	6.319	3.165
	f_4	0.966	**0.279**	6.825	3.341
	f_8	0.952	**0.277**	7.526	4.039
	f_{13}	0.957	**0.279**	7.754	4.146
	f_{15}	0.953	**0.278**	7.644	4.323
20	f_1	2.265	**0.383**	65.923	5.184
	f_2	2.279	**0.394**	92.022	3.478
	f_4	2.305	**0.400**	90.002	4.881
	f_8	2.254	**0.384**	81.653	5.006
	f_{13}	2.253	**0.386**	79.875	5.231
	f_{15}	2.192	**0.383**	77.685	5.247

The **bold** values denote the best result among four SAEAs in one problem instance.

In each generation, N parents generate N offspring by variation operators, and the whole set of parents plus offspring is denoted as $P = \{Ind_i | i = 1, 2, ..., 2N\}$. The new population selected based on true fitness function evaluation is denoted as $P_f = \{Ind_i | i = \lambda_1, \lambda_2, ..., \lambda_N\}$. And the new population selected with the use of surrogate model is denoted as $P_m = \{Ind_i | i = \mu_1, \mu_2, ..., \mu_N\}$. Then, the rank correlation R_{corr} of the model is calculated using Eq. (6):

$$R_{corr} = \frac{|P_m \cap P_f|}{N} \tag{6}$$

The metric R_{corr} is calculated for each surrogate model in each generation and the results are showed in Table 4. The value in each cell is the average R_{corr} value of all generations.

Table 4: The average rank correlation of surrogate models across all generations calculated based on Eq. 6 for $d = 10$ and 20.

d	Problem	RBF	kNN-R	RankSVM	SVC
10	f_1	0.836	0.676	**0.937**	0.489
	f_2	**0.921**	0.915	0.585	0.466
	f_4	**0.863**	0.843	0.574	0.501
	f_8	**0.820**	0.758	0.709	0.518
	f_{13}	**0.801**	0.751	0.690	0.539
	f_{15}	**0.826**	0.737	0.760	0.544
20	f_1	0.908	0.600	**0.938**	0.472
	f_2	**0.953**	0.950	0.550	0.460
	f_4	**0.895**	0.829	0.621	0.614
	f_8	**0.852**	0.786	0.707	0.573
	f_{13}	**0.830**	0.655	0.829	0.532
	f_{15}	**0.875**	0.738	0.740	0.575

The **bold** values denote the best result among four SAEAs in one problem instance.

From Table 4, we can easily find that the surrogate models have the following relationship in terms of rank correlation R_{corr}:

$$\text{RBF} \succ k\text{NN-R} \approx \text{RankSVM} \succ \text{SVC}$$

where $a \succ b$ means that model a has larger average R_{corr} than model b , and $a \approx b$ signifies model a and b have similar average R_{corr}.

Overall, the absolute fitness models (RBF and kNN-R) are more accurate than the relative fitness models (RankSVM and SVC) considering the rank correlation, although the relative fitness model intuitively should be better than the absolute fitness model. It is probably because the absolute fitness model aims at approximating the fitness landscape, and if the approximation is accurate enough, the ranking accuracy is also high. From this perspective, we can claim that the RBF model is better than the kNN-R model with respect to R_{corr}. The RankSVM model performed relatively well in general. Compared with the kNN-R model, it is sometimes better and sometimes worse over different benchmark functions. In our experiments, the SVC model has the lowest rank correlation.

Even though the quality of the predictions given by surrogate models directly influences the performance of SAEAs, this impact varies with different model management strategies. In particular, it has more significant influence on the individual-based evolution control than on the pre-selection strategy. This is because poor predictions cause many low-quality individuals to survive to the new generation and high-quality individuals to be obsolete in the individual-based control strategy. In the pre-selection strategy, low-quality does not lead SAEA to false optima but only deteriorates the convergence speed of SAEA, because all individuals entering into the selection operator have actual fitness. Therefore, even though the SVC's accuracy is not outstanding, the SAEA assisted by SVC can obtain acceptable performance.

5.4. Analysis 3: parameter influence

In SAEAs, the use of surrogate models brings in some new parameters, which affect the performance of SAEAs. How to determine these newly introduced parameters is a crucial issue in designing SAEAs. However, to the best of our knowledge, there is no comprehensive study on this issue in the literature. In [36], the authors only analyzed the effect of database size and neighbourhood size in the kNN model. In [35], the authors studied the influence of training samples' size for local and global models, but they only focused on regression-based models. In order to have a comprehensive understanding about the influence of the parameters of different kinds of surrogate models on SAEAs' performance, we conducted a series of experiments in this subsection.

We studied two important parameters for each surrogate model:

- For the individual-based evolution control, one is the size of training data K, and the other is the re-evaluation ratio p_{sm}. In the rank-based model, individuals are re-evaluated until they satisfy a threshold of rank correlation τ_{th}, thus τ_{th} determines the re-evaluation ratio.
- For the pre-selection strategy, one parameter is the size of training data K, and another is the generator factor λ_{pre} that determines how many offspring are generated for one pair of parents. We employed a grid search to study the parameters' influence. The SAEAs and parameter settings investigated in our experiments are listed in Table 5.

Table 5: SAEAs and parameters investigated to determine the effect of parameter choice on SAEAs' performances.

Model Management Strategy	Surrogate Model	SAEA	Parameters
Individual-based Evolution Control	RBF	IB-RBF	K, p_{sm}
	kNN-R	IB-kNN-R	K, p_{sm}
	RankSVM	IB-RankSVM	K, τ_{th}
Pre-Selection Strategy	RBF	PS-RBF	K, λ_{pre}
	kNN-R	PS-kNN-R	K, λ_{pre}
	RankSVM	PS-RankSVM	K, λ_{pre}
	SVC	PS-SVC	K
K	$\{i \times d \mid i = 1, 2, \ldots, 10\}$		
p_{sm}, τ_{th}	$\{0.05(2i-1) \mid i = 1, 2, \ldots, 10\}$		
λ_{pre}	$\{i + 1 \mid i = 1, 2, \ldots, 10\}$		

In order to study the overall performance and to avoid the difference of problems' scale, we employ another performance metric from literature [50]. For each SAEA, denoted as A, we have problem instances $\mathcal{F} = \{f_k | k = 1, 2, ..., n\}$ and parameter settings $C = \{c_j | j = 1, 2, ..., m\}$. The performance of each parameter setting $PM(c_i)$ over the problem instances can be calculated using Eq. (7):

$$PM(c_i) = \frac{1}{m-1} \sum_{j=1, j\neq i}^{m} P(A_{c_i} > A_{c_j}) \tag{7}$$

The $P(A_{c_i} > A_{c_j})$ represents the probability of A_{c_i} outperforming A_{c_j}, which can be calculated by Eq. (8):

$$P(A_{c_i} > A_{c_j}) = \frac{1}{n} \sum_{k=1}^{n} P(q_{i,k} < q_{j,k} | f_k) \tag{8}$$

where $q_{i,k}$ denotes the quality of solution obtained by A_{c_i} on f_k, and $P(q_{i,k} < q_{j,k} | f_k)$ can be estimated by the following equation:

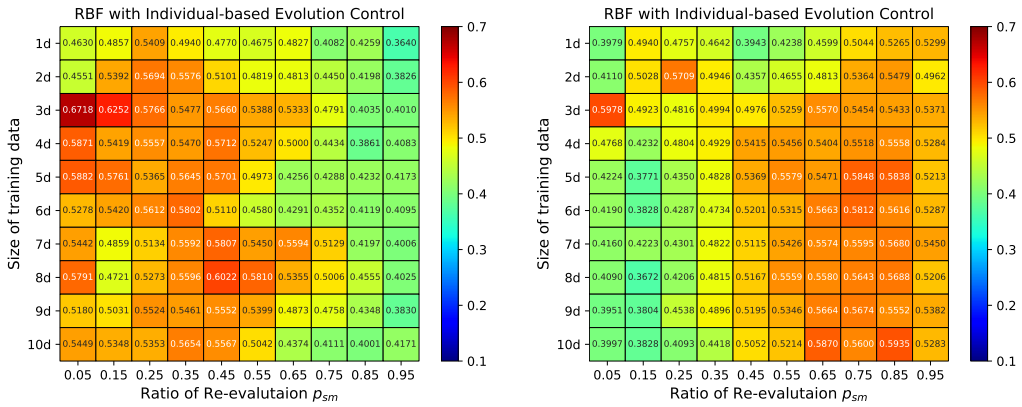
$$P(q_{i,k} < q_{j,k} | f_k) = \frac{\sum_{s=1}^{s_i} \sum_{t=1}^{s_j} \mathbb{I}(y_{i,k,s} < y_{j,k,t})}{s_i \times s_j} \tag{9}$$

where $y_{i,k,s}$ represents the fitness of the solution obtained by A_{c_i} in the s_{th} trial for the problem f_k, s_i and s_j represent the number of trials of each algorithm for one problem, and $\mathbb{I}(\cdot)$ is the indicator function.

The experimental results of different SAEAs using different surrogate models and parameter settings are presented in Fig. 6 $\sim$ 12. The number in each cell of heatmap represents the performance value $PM(c_i)$ of each parameter configuration.

5.4.1. IB-RBF

The performance calculated based on Eq. 7 of IB-RBF (i.e., the SAEA using RBF model and individual-based evolution control) with different values of K and p_{sm} is presented in Fig. 6. It can be observed that a large p_{sm} value gets a worse performance when $d = 10$, but large p_{sm} for $d = 20$ makes the performance much better. It might because the quality of the RBF model is higher in the case of $d = 10$. With the same maximal number of real fitness function evaluations, a small p_{sm} indcates that the SAEA can evolve more generations. And when the surrogate model is accurate enough, more generations make the SAEA obtain better solution. On the contrary, when the surrogate model is not accurate, more generations probably lead SAEA to a false optimum so that the re-evaluation is much more important. On the other hand, more re-evalation solutions could provide more information for constructing more accurate local surrogate model, and it helps a lot in high dimensional cases. Therefore, a high p_{sm} can get a better performance in the case of $d = 20$ for IB-RBF. The size of training data K has weak impact on IB-RBF's performance. According to the result, $3d$ to $4d$ would be the most suitable value for K in IB-RBF.



a) $d = 10$ b) $d = 20$

Figure 6: IB-RBF's performance calculated based on Eq. 7 when using different values of K (size of training data) and p_{sm} (ratio of re-evaluation).

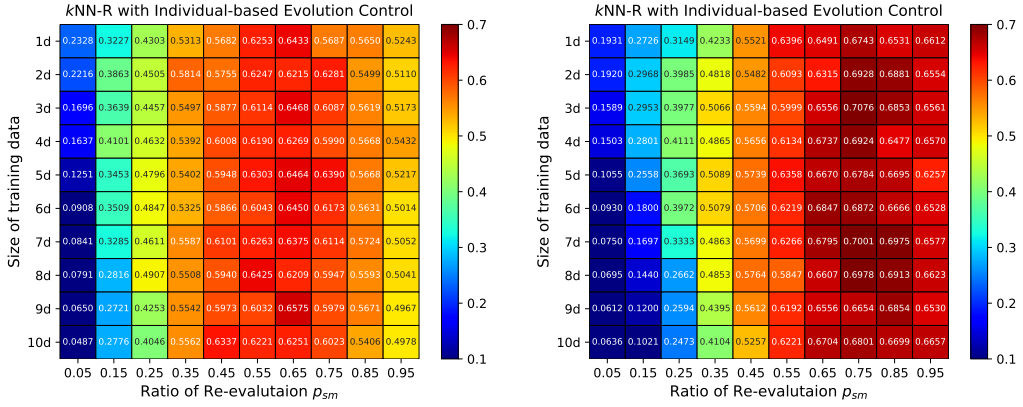
5.4.2. IB-kNN-R

For the IB-kNN-R, i.e., the SAEA using kNN-R model with individual-based evolution control, we can easily find that the size of training data K does not has strong influence on SAEA's performance for both $d = 10$ and $d = 20$, as shown in Fig. 7. We can analyze the reason from the kNN-R predictive model, which is represented by Eq. (10):

$$\hat{F}(\mathbf{x}) = \frac{\sum_{j=1}^{k} s\left(\mathbf{x}, \mathbf{x}_j\right)^2 F\left(\mathbf{x}_j\right)}{\sum_{j=1}^{k} s\left(\mathbf{x}, \mathbf{x}_j\right)^2} \tag{10}$$

where $s\left(\mathbf{x}, \mathbf{x}_j\right) = 1 - \frac{d(\mathbf{x}, \mathbf{x}_j)}{d(\mathbf{x}^U, \mathbf{x}^L)}$ is the similarity between $\mathbf{x}$ and $\mathbf{x}_j$, $d\left(\mathbf{x}, \mathbf{x}_j\right)$ is the distance between $\mathbf{x}$ and $\mathbf{x}_j$, $\mathbf{x}^U$ and $\mathbf{x}^L$ are the upper and lower bounds of $\mathbf{x}$, respectively. From Eq. (10), it can be seen that the sample points $\mathbf{x}_j$ that are far from the new candidate solution $\mathbf{x}$ contribute little to the predicted fitness $\hat{F}(\mathbf{x})$ since the similarity $s\left(\mathbf{x}, \mathbf{x}_j\right)$ is very small. The predicted fitness $\hat{F}(\mathbf{x})$ mainly depends on the near neighbors of $\mathbf{x}$. As a result, IB-kNN-R with the small and large K value obtains similar performance. From the results, it can be found that the generator factor

p_{sm} significantly affects the performance of IB-kNN-R. When p_{sm} takes value between 0.55 and 0.75, IB-kNN-R can achieve good performance on both $d = 10$ and $d = 20$ test problems. This is because that a small p_{sm} is likely to lead the SAEA to false optima and a large p_{sm} will result more individuals being evaluated by real fitness function in each generation so that the number of generation will decrease when the maximum number of the real fitness function evaluation is fixed. The p_{sm} values of 0.55 to 0.75 lead to a good balance between the model accuracy (avoid leading to false optima) and the number of evolution generations.



a) $d = 10$ b) $d = 20$

Figure 7: IB-kNN-R's performance calculated based on Eq. 7 when using different values of K (size of training data) and p_{sm} (ratio of re-evaluation).

5.4.3. IB-RankSVM

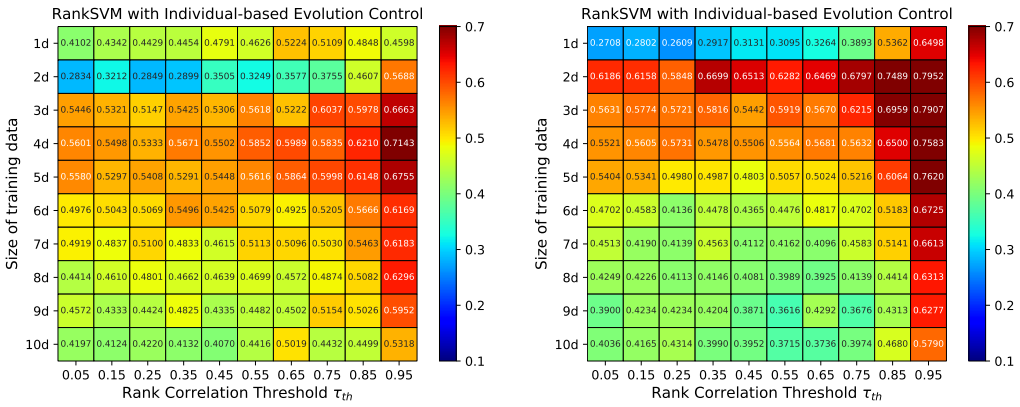
According to the performance calculated by Eq. 7 of IB-RankSVM (the SAEA using RankSVM model with individual-based evolution control) with different values of k and τ_{th} in Fig. 8, the influence of training data size K on SAEA's performance is more significant than that of rank correlation threshold τ_{th}. From the result, the K values of $3d$ to $4d$ are good choices. When K is small, the training data is not sufficient to construct accurate rank models. On the other hand, when K is large and it has a large training data set, the Kendall's τ will be small if the rank model is not accurate enough because the Kendall's τ is computed using the relative ordering of the ranks of all $K(K-1)/2$ possible pairs [14]. Therefore, it requires a larger number of re-evaluation to satisfy the threshold τ_{th} in each generation so that IB-RankSVM's performance deteriorates due to the the decrease of number of evolutionary generations. As a result, a moderate value of K is suitable for the rank-based model. For the rank correlation threshold τ_{th}, a large value is better because it means a more accurate rank-based model. From the results, τ_{th} values bigger than 0.95 might be suitable choices both for $d = 10$ and $d = 20$.

5.4.4. PS-RBF

For the result of PS-RBF (the SAEA using RBF model with pre-selection strategy) in Fig. 9, the upper part of the heatmap is shallower than the lower part for $d = 10$ as well as for $d = 20$. This indicates that a bigger training data size results in better performance of PS-RBF. This is because more data can make the RBF model more accurate and the pre-selection can select potentially better solutions. A K value of $3d$ to $4d$ is enough because more data than $4d$ may not increase the quality of RBF significantly. For pre-selection factor λ_{pre}, a bigger value leads to better performance, but the performance does not increase prominently as the increasing of λ_{pre}. From the results, the pre-selection factor value of $\lambda_{pre} = 8$ is suitable for PS-RBF.

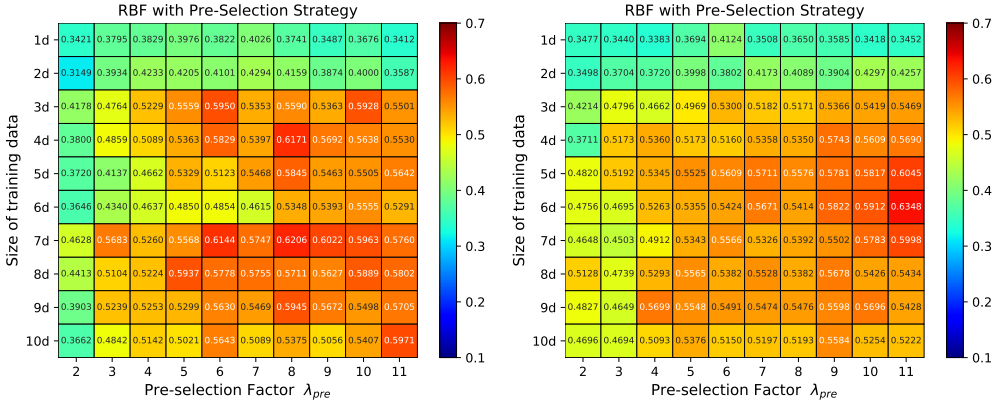
5.4.5. PS-kNN-R

For PS-kNN-R, i.e., the SAEA using kNN-R model with pre-selection strategy, the difference of performance between different parameter settings is very small as shown in Fig. 10. In the case of $d = 10$, the performance of PS-kNN-R is slightly improved when λ_{pre} increases. This is probably because the number of offspring generated by mutation or crossover operators increases λ_{pre} increasing, which results in an increase of diversity in the search



a) $d = 10$ b) $d = 20$

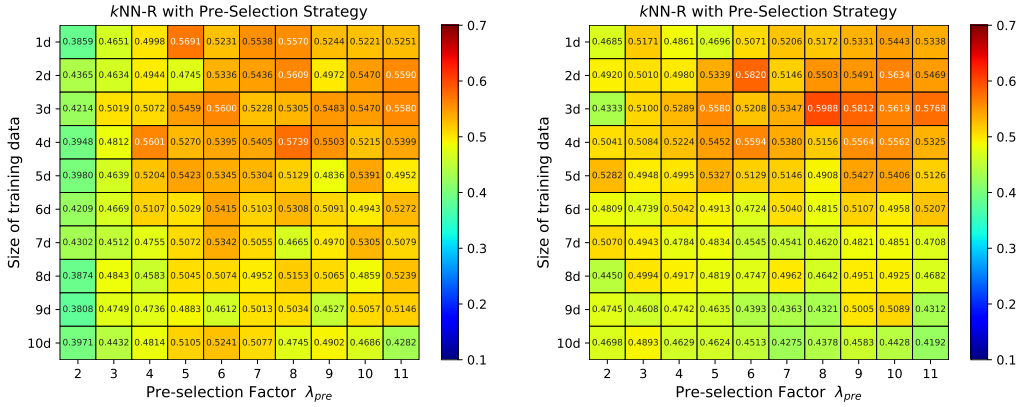
Figure 8: IB-RankSVM's performance calculated based on Eq. 7 when using different values of K (size of training data) and τ_{th} (rank correlation threshold).



a) $d = 10$ b) $d = 20$

Figure 9: PS-RBF's performance calculated based on Eq. 7 when using different values of K (size of training data) and λ_{pre} (pre-selection factor).

process. And, when $d = 20$, the recommended parameter setting for PS-kNN-R would be a small K and a large λ_{pre}, even though the performance does not vary significantly.

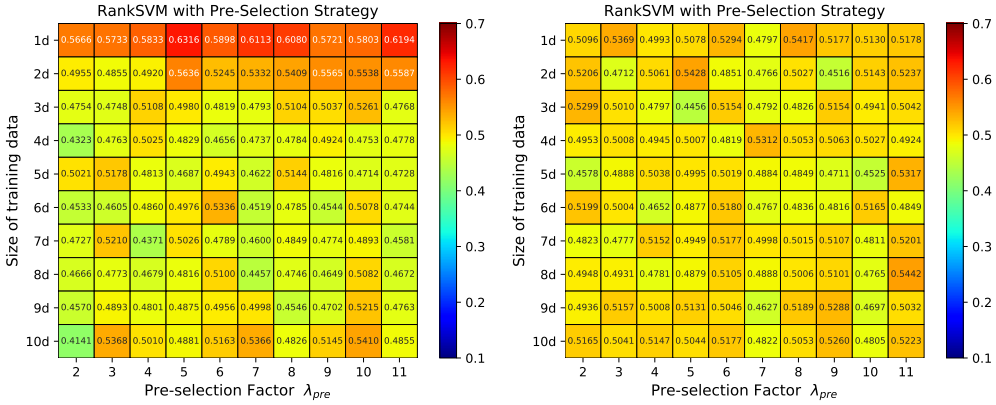


a) $d = 10$ b) $d = 20$

Figure 10: PS-kNN-R's performance calculated based on Eq. 7 when using different values of K (size of training data) and λ_{pre} (pre-selection factor).

5.4.6. PS-RankSVM

The performance of PS-RankSVM (the SAEA using RankSVM model with pre-selection strategy) with different parameter settings is shown in Fig. 11. The performance of different parameters are very similar to each other, especially in the case of $d = 20$. For $d = 10$, a small K, i.e., $K = 1d$, achieves better performance than larger values of K. The main reason for the similar performance is the low accuracy of RankSVM which makes a small value of Kendall's τ. Especially, when K and λ_{pre} take large values, the training data for each rank-based model is $K \cdot \lambda_{pre}$, and the τ which is computed using the relative ordering of the ranks of all $K \cdot \lambda_{pre}(K \cdot \lambda_{pre} - 1)/2$ possible pairs will be very small. Thus, PS-RankSVM will randomly select the candidate as the offspring in most cases since τ is small, i.e., the model accuracy is low. On the other hand, the individuals to be ranked in PS-Rank are generated from the same parents so that these individuals might be close to each other. Therefore, a small number of training data can distinguish these individuals more precisely. The parameter λ_{pre} has almost no influence on the performance of PS-RankSVM.

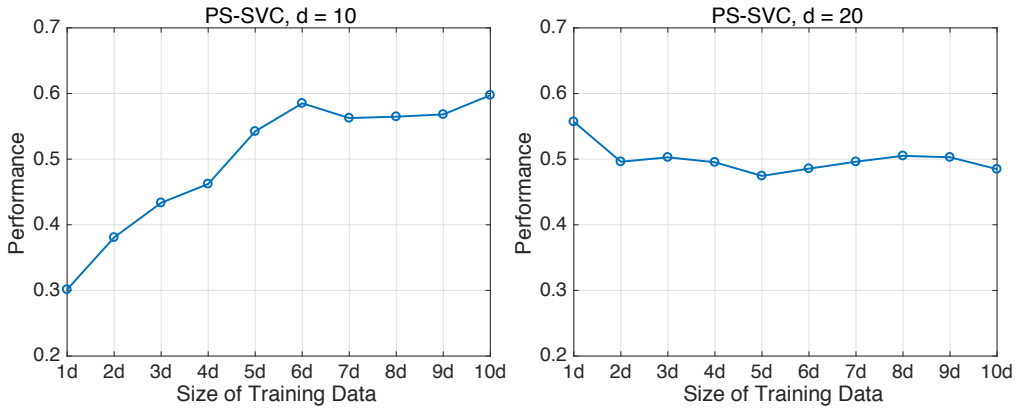


a) $d = 10$ b) $d = 20$

Figure 11: PS-RankSVM's performance calculated based on Eq. 7 when using different values of K (size of training data) and λ_{pre} (pre-selection factor).

5.4.7. PS-SVC

For the SAEA using SVC model with pre-selection strategy (PS-SVC), due to its special algorithm structure, only one parameter (training data size) has an effect on PS-SVC's performance. The performance of PS-SCV with different values of training data size K is presented in Fig. 12. It is very clear that the training data size value of $6d$ is enough in the case of $d = 10$. SAEA's performance is improved slightly when K continues to increase. The training data size K does not have critical influence on SAEA's performance when $d = 20$. The likely reason is that the limited and imbalanced training data cannot satisfy the requirement for training an accurate SVC model in the case of $d = 20$.



a) $d = 10$ b) $d = 20$

Figure 12: PS-SVC's performance calculated based on Eq. 7 when using different values of K (size of training data).

5.5. Analysis 4: performance comparison

In this subsection, we will analyze the performance of SAEAs using different kinds of surrogate models and model management strategies on the test problem instances. For a fair comparison, we compare the performance of each SAEA with its best parameter configuration obtained from the experiments in previous subsection.

5.5.1. Overall performance

The results are presented in Table 6. Numbers in each cell denote average best fitness and standard deviation over 25 independent runs. The bold ones are the best results among all algorithms on the corresponding problem. The last column presents the obtained result of the EA without using surrogate models, i.e., No Surrogate (NS). The ranking of each algorithm instance is calculated on the basis of statistical hypothesis tests (i.e., Wilcoxon rank-sum test with a 0.05 significance level), in which if two results have no significant difference according to the hypothesis test, they will be given the same ranking. The number in brackets of each cell represents the algorithm's ranking for a given problem instance. Besides, the average ranking of algorithms across all problem instances for $d = 10$ and $d = 20$ are also provided in the table.

First of all, it is obvious that PS-RBF and IB-RBF performed best among all algorithms on most problem instances, as reflected by the average ranking. This indicates that the RBF model is the best one among the absolute and relative fitness models that are investigated in our experiments. The kNN-R model is the second-best surrogate model according to the results in Table 6, with the ranking of PS-kNN-R and IB-kNN-R follows that of the RBF model. The RankSVM and SVC models performed much worse than the previous two kinds of models, where the SVC model was the worst in our experiments. Overall, we can get a basic conclusion in terms of the average performance:

$$\text{RBF} \succ k\text{NN-R} \succ \text{RankSVM} \succ \text{SVC}$$

where $M_1 \succ M_2$ represents M_1 performs better than M_2 in terms of the optimization performance. This is the same as the ranking of model accuracy in section 5.3. This indicates that the model's accuracy, with respect to the rank correlation, directly influences the performance of the corresponding SAEAs.

Table 6: Average best fitness and standard deviation (AVR ± STD) for each algorithm on each problem instances over 25 runs. The boldface values are the best fitness for each problem instance according to the Wilcoxon rank-sum test with a 0.05 significance level. The rank obtained by pairwise comparison using Wilcoxon rank-sum test is listed in the bracket after each value.

d	Problem	PS-RBF	PS-kNN-R	PS-Rank	PS-SVC
10	f_1	**6.3189e+03 ± 2.7100e+03 (1)**	**6.2222e+03 ± 3.0985e+03 (1)**	1.3269e+05 ± 1.0201e+05 (4)	5.4414e+05 ± 1.3998e+06 (4)
	f_2	**1.9534e+04 ± 7.8264e+03 (1)**	2.1284e+04 ± 8.9185e+03 (3)	2.2319e+04 ± 7.4356e+03 (3)	2.5320e+04 ± 8.5496e+03 (5)
	f_4	**6.7013e+02 ± 2.0807e+02 (1)**	6.6177e+02 ± 1.5887e+02 (2)	8.3013e+02 ± 2.2133e+02 (5)	9.5162e+02 ± 1.9861e+02 (6)
	f_8	8.0299e+02 ± 8.5518e-01 (4)	8.0247e+02 ± 8.2028e-01 (3)	8.0352e+02 ± 6.9876e-01 (5)	8.0415e+02 ± 8.2992e-01 (6)
	f_{13}	**1.6178e+03 ± 1.3178e+00 (1)**	1.6197e+03 ± 3.0017e+00 (5)	1.6189e+03 ± 1.6564e+00 (3)	1.6235e+03 ± 4.5621e+00 (8)
	f_{15}	**1.6750e+03 ± 1.7918e+02 (1)**	1.8295e+03 ± 7.0078e+01 (3)	1.6913e+03 ± 1.8190e+02 (2)	1.8804e+03 ± 9.5120e+01 (6)
Average-Ranking (d = 10)		1.5	2.83	3.67	5.83
20	f_1	**3.3580e+07 ± 2.7307e+07 (1)**	4.1484e+07 ± 2.5689e+07 (2)	6.3646e+07 ± 4.5747e+07 (3)	1.3027e+08 ± 1.1491e+08 (6)
	f_2	3.2654e+04 ± 7.0460e+03 (2)	3.5837e+04 ± 8.0113e+03 (4)	3.5188e+04 ± 8.6006e+03 (3)	3.8394e+04 ± 8.9980e+03 (5)
	f_4	1.2969e+03 ± 3.7046e+02 (2)	1.3536e+03 ± 3.4626e+02 (3)	1.6349e+03 ± 4.9787e+02 (4)	2.1249e+03 ± 5.7958e+02 (6)
	f_8	**8.0852e+02 ± 3.2504e+00 (1)**	8.0886e+02 ± 3.4550e+00 (2)	8.1621e+02 ± 8.5271e+00 (4)	8.1727e+02 ± 4.5242e+00 (6)
	f_{13}	**1.5958e+03 ± 5.9549e+00 (1)**	1.5998e+03 ± 9.1526e+00 (4)	1.5990e+03 ± 6.9405e+00 (3)	1.6070e+03 ± 1.5230e+01 (6)
	f_{15}	**1.9238e+03 ± 3.5793e+01 (1)**	1.9449e+03 ± 4.1802e+01 (2)	1.9652e+03 ± 3.7121e+01 (4)	2.0253e+03 ± 4.0137e+01 (6)
Average-Ranking (d = 20)		1.33	2.83	3.5	5.83
Average-Ranking		1.42	2.83	3.58	5.83

d	Problem	IB-RBF	IB-kNN-R	IB-Rank	NS
10	f_1	7.8707e+03 ± 5.3762e+03 (2)	3.5295e+04 ± 1.0491e+05 (3)	9.4191e+03 ± 1.0561e+04 (2)	8.8084e+05 ± 8.8456e+05 (5)
	f_2	**1.9742e+04 ± 6.3255e+03 (1)**	2.1131e+04 ± 8.1986e+03 (2)	2.5569e+04 ± 1.0216e+04 (4)	3.0945e+04 ± 1.1908e+04 (6)
	f_4	**5.7690e+02 ± 1.0756e+02 (1)**	6.4679e+02 ± 1.3257e+02 (2)	7.3147e+02 ± 1.6015e+02 (3)	7.5388e+02 ± 1.4102e+02 (4)
	f_8	**8.0180e+02 ± 9.8822e-01 (1)**	**8.0190e+02 ± 1.0296e+00 (1)**	8.0210e+02 ± 1.0909e+00 (2)	8.0403e+02 ± 8.4194e-01 (6)
	f_{13}	1.6184e+03 ± 1.3885e+00 (2)	1.6189e+03 ± 1.7946e+00 (4)	1.6202e+03 ± 2.2969e+00 (7)	1.6204e+03 ± 2.9060e+00 (6)
	f_{15}	**1.6409e+03 ± 1.6444e+02 (1)**	1.8809e+03 ± 4.6203e+01 (5)	1.9048e+03 ± 8.3818e+01 (7)	1.7931e+03 ± 1.9014e+02 (4)
Average-Ranking (d = 10)		1.33	2.83	4.17	5.17
20	f_1	6.1769e+07 ± 5.3520e+07 (3)	8.9753e+07 ± 1.2423e+08 (4)	1.3174e+08 ± 1.0504e+08 (6)	1.0890e+08 ± 9.5198e+07 (5)
	f_2	**3.1528e+04 ± 7.6819e+03 (1)**	3.5311e+04 ± 1.2032e+04 (3)	3.9266e+04 ± 1.3019e+04 (5)	3.8990e+04 ± 8.1845e+03 (5)
	f_4	**1.1292e+03 ± 3.8573e+02 (1)**	1.3336e+03 ± 4.6006e+02 (2)	1.8292e+03 ± 5.0367e+02 (5)	1.9917e+03 ± 4.2165e+02 (6)
	f_8	8.0917e+02 ± 3.9245e+00 (2)	8.1215e+02 ± 5.9572e+00 (3)	8.6065e+02 ± 5.8219e+01 (7)	8.1685e+02 ± 4.9504e+00 (5)
	f_{13}	1.5972e+03 ± 6.1122e+00 (2)	1.5991e+03 ± 8.8844e+00 (3)	1.6113e+03 ± 2.1264e+01 (4)	1.6041e+03 ± 1.5099e+01 (5)
	f_{15}	1.9368e+03 ± 2.2315e+01 (2)	1.9600e+03 ± 4.8161e+01 (3)	2.0581e+03 ± 7.5934e+01 (6)	1.9873e+03 ± 4.1149e+01 (5)
Average-Ranking (d = 20)		1.83	3	5.5	5.17
Average-Ranking		1.58	2.92	4.83	5.17

The **bold** values denote the best result among all SAEAs in one problem instance.

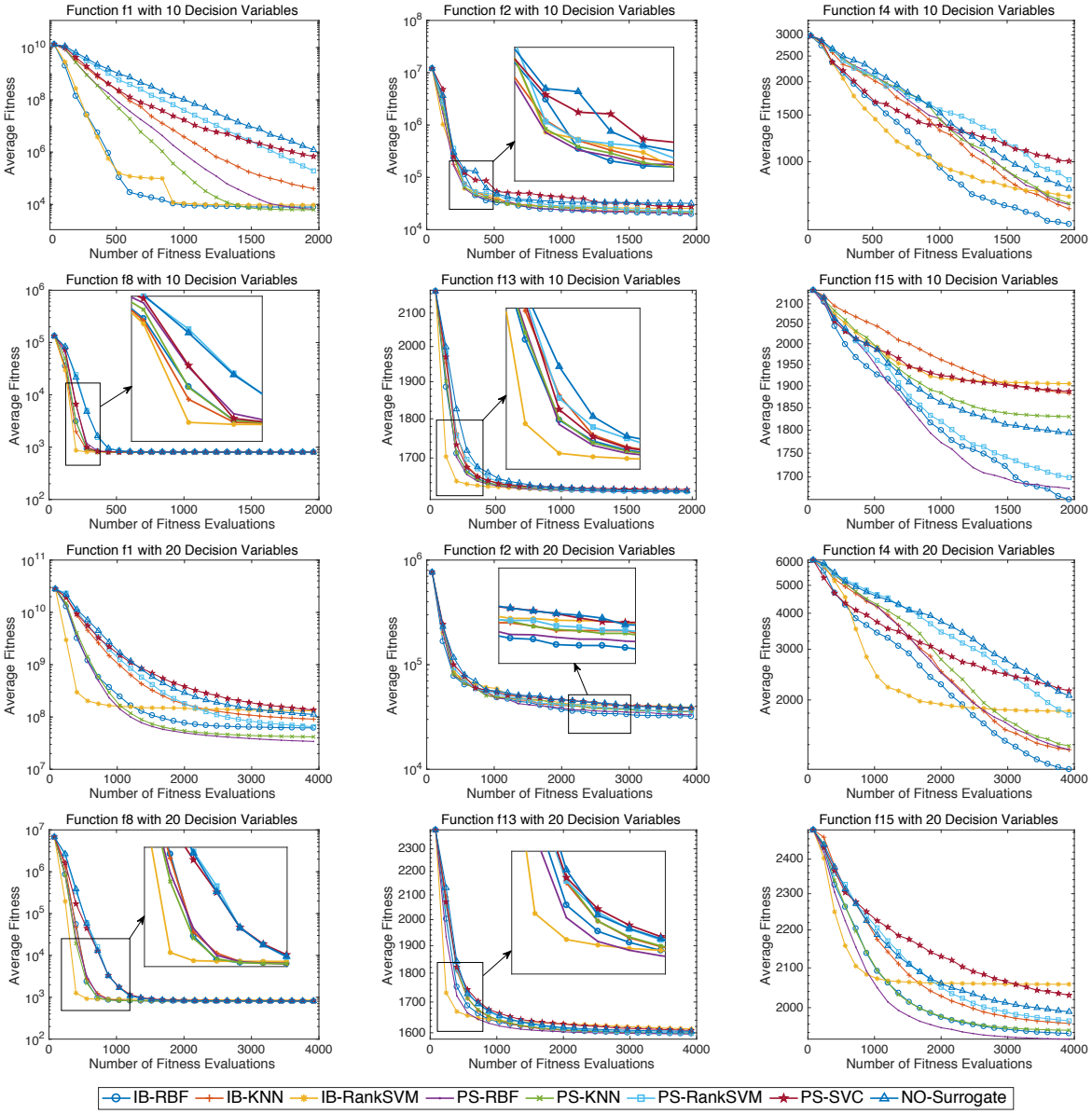


Figure 13: The convergence curves of algorithm instances on problem instances with $d = 10, 20$.

Furthermore, it is easy to see that absolute fitness models (RBF, kNN-R) obtained better results than relative fitness models (RankSVM, SVC) from the perspective of the obtained solution's quality. The accuracy of each model mainly affects the performance of SAEAs. Besides, another important reason is that absolute fitness models are much more robust than relative fitness models. The absolute fitness model is trained to fit the actual fitness function, and even if there is some prediction error, the rank of predicted individuals can be maintained. However, the target of relative fitness models is to predict the rank information and its prediction error will influence the rank directly. Therefore, the absolute fitness model can tolerate a larger prediction error than the relative fitness model. Obviously, the absolute fitness model is much easier to lead to an acceptable model, which can maintain the same amount of rank information, than the relative fitness model. As a result, the absolute fitness model generally has higher accuracy than relative fitness model, and the corresponding SAEAs perform much better.

By analysing the same model with two different model management strategies, we can find that the pre-selection strategy is slightly better than the individual-based evolution control. As discussed in Section 2, the pre-selection strategy reduces the computational cost by improving the quality of the individuals, and it can ensure that the optimization converges to the true optimum. The individual-based evolution control cannot guarantee the right direction of the optimisation process. Therefore, in most cases, the pre-selection strategy is better than individual-based evolution control. In some cases, especially when the model is accurate enough to capture the overall trend of fitness function, SAEAs using individual-based control may perform better than SAEAs using pre-selection since the former can evolve more generations than the latter when the consumed number of fitness function evaluations are the same.

The relative performance of each model for $d = 10$ and $d = 20$ has no big difference. This is probably because the increase of dimensionality has a similar influence on all kinds of surrogate models.

5.5.2. Discussion for each model

We plot the average evolutionary curves of each algorithm on each problem instance in Fig. 13. By using the evolutionary curves together with the numerical results in Table 6, we will discuss the characteristics of each model by analysing the corresponding SAEA's performance in this subsection.

As analyzed before, the RBF model is the best model among the four surrogate models investigated in this study. On almost all problem instances, RBF obtains the best result, as shown in $f_1, f_2, f_4, f_{13}, f_{15}$ with $d = 10$ and f_1, f_8, f_{13}, f_{15} with $d = 20$ in Table 6. From the evolutionary curves, we can see that PS-RBF and IB-RBF show a steady convergence. In most cases, PS-RBF and IB-RBF converge faster than others during the whole optimization process. The main reason for the RBF model's excellent performance is its high accuracy. Therefore, we can easily conclude that SAEAs can benefit from the RBF model when solving CEPs and obtain a good final result.

The kNN-R model also predicts the fitness of individual and obtains relative good performance, as shown in Table 6. However, we can see in the result table that kNN-R model's performance deteriorates on composite functions, i.e., f_{13} and f_{15}. This is because the composite function is highly non-linear, and the accuracy of the kNN-R model decreases dramatically, which is reflected in Table 4. Besides, the convergence speed of kNN-R assisted EA is not very fast, as shown in the convergence curves. This is because the predicted fitness of an individual highly depends on its nearest neighbours, and the prediction error of kNN-R will be large in the area which has not been explored. Therefore, even though a much better individual is generated, which is distant from the mainly explored area, it will probably be predicted as a worse individual and be discarded. As a result, the convergence step of kNN-R assisted EA will be small.

For the SAEA using the RankSVM model, its performance is better than that of the EA without any surrogate model in our experiments. However, its performance is still limited and inferior to that of SAEAs using the RBF and kNN-R models. Unlike the previous two models, i.e., RBF and kNN-R, the performance of PS-RankSVM and IB-RankSVM has a big difference, and PS-RankSVM performs much better than IB-RankSVM. It is probably because the accuracy of the obtained RankSVM model is low and influences individual-based evolution control much more than the pre-selection strategy.

The overall performance of SAEAs using RankSVM is not competitive to SAEAs using RBF and kNN-R. However, we find that IB-RankSVM converges very fast in the early stage of the search process, but the final results are not good on almost all functions according to Fig. 13. To analyze the reason for this phenomenon, We record the number of generations at every 25 real fitness evaluations during the whole optimization process of IB-RankSVM and plot it in Fig. 14. From Fig. 14, it can be found that, in the early stage, few individuals are re-evaluated in each generation of IB-RankSVM so that the search process improves quickly. While in latter stage, more individuals are re-evaluated

in each generation, and thus the convergence speed of IB-RankSVM slows down. This also indicates that in the early stage, the accuracy of RankSVM model is relatively high, and it can guide the search process of SAEAs properly and improve the search efficiency. As the search proceeding, the model accuracy deteriorates, more individuals need to be evaluated to obtain a high-quality RankSVM model. Thus, the efficiency of IB-RankSVM reduces in the latter stage.

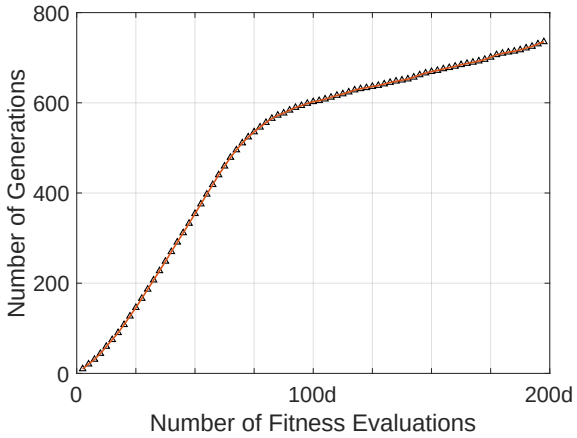


Figure 14: The increase of number of generations during the optimization process of IB-RankSVM on f_1 with $d = 20$.

The SVC model is the worst one among the four models investigated in this study. The performance of SAEAs using SVC is even worse than EAs without any surrogate models in our experiments. Compared with other models with a pre-selection strategy, PS-SVC does not generate additional offspring during mutation and crossover so that the search diversity is much lower than other models with pre-selection strategies. From the evolutionary curves, the SVC model assisted SAEA also obtains good convergence at the early stage but its speed slows down very quickly. This is probably because of the decrease of the SVC model's accuracy in the later stage of optimization, as showed in Fig. 15, in which we plot the change of accuracy in the optimization process.

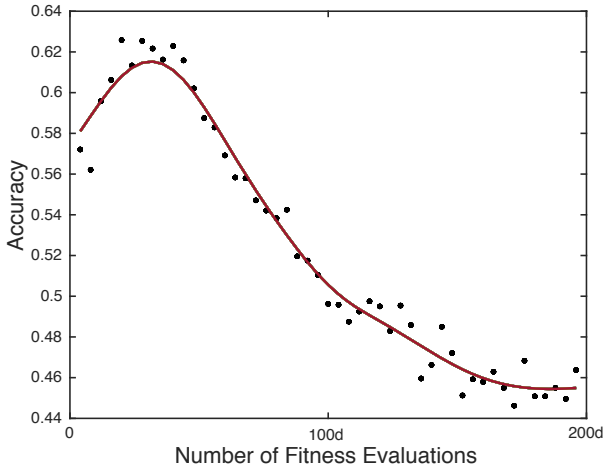


Figure 15: The dynamic change of average accuracy for SVC model in PS-SVC on f_1 with $d = 20$.

It is worth mentioning that the performance of SAEA with PS-SVC is even worse than EAs without any surrogate models. The problem of low diversity is one reason, but the most important is that the SVC model is not suitable in our EA framework. In the first work of classification-based model [15], the authors used SVC in a DE algorithm, in which one parent generates one offspring and the new parent is selected between these two individuals, which

we named as parent-offspring tournament selection (POTS). However, in our experiment, the basic framework for selection is to choose the better individuals in the individual pool ,which includes parent and offspring population. The SVC model is constructed on the basis of a single individual rather than a population. Furthermore, we compared the performance of SVC assisted EA with POTS-based no-surrogate EA (POTS-NSEA) for $d = 10$. The results are showed in Fig. 16, where the blue box shows SVC-SAEA's fitness and the orange box shows POTS-NSEA's fitness. In this case, SVC-SAEA performs better than EA without any surrogate models in almost all functions. Therefore, we can conclude that the classification model is more suitable for POTS-based EAs.

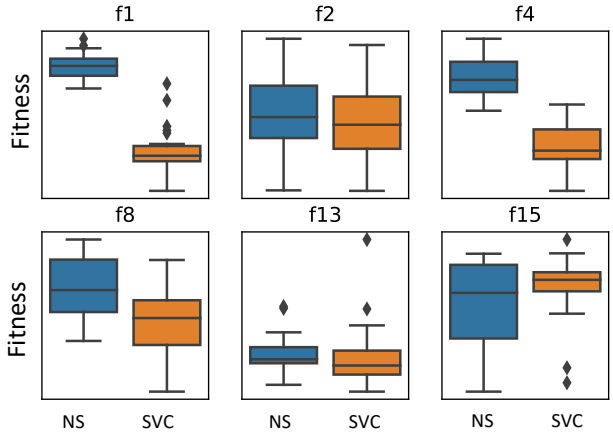


Figure 16: The comparison between SVC-SAEA and POTS-NSEA on test functions with $d = 10$

5.6. Summary

In summary, from the above experiments and analysis, we can obtain a series of conclusions about different models:

- Comparison in terms of efficiency of model construction and prediction (execution time): kNN-R $\succ$ RBF $\succ$ SVC $\succ$ RankSVM;
- Comparison in terms of model predictive performance (rank correlation): RBF $\succ$ kNN-R $\approx$ RankSVM $\succ$ SVC;
- We showed the best parameter settings related to model management when using our algorithms;
- The performance of SAEAs using absolute fitness models are generally better than that using relative fitness models because of the larger prediction error tolerance for absolute fitness model;
- Pre-selection strategy is slightly better than individual-based evolution control in terms of the convergence of SAEAs;
- RBF (regression-based model) is the best model and is recommended for SAEAs to solve CEPs in terms of the convergence of SAEAs;
- The convergence step of kNN-R (similarity-based model) with individual-based evolution control is very small;
- RankSVM (rank-based model) is better to be used as the global model due to its requirements for the uniformly distributed training data;
- SVC (classification-based model) is more suitable to POTS (parent-offspring tournament selection) based EAs.

6. Conclusion and future work

This paper provided a new taxonomy and a comprehensive comparison of surrogate models used in SAEAs for single-objective optimization problems. Different from existing surveys [13, 3, 5, 8], our taxonomy divides the existing surrogate models in SAEAs into two main classes: absolute and relative fitness models. Based on this taxonomy, a comprehensive survey of surrogate models used in SAEAs is given. Then, a systematical comparative study among different models is conducted, which will provide valuable guidance to select suitable surrogate models used in SAEAs.

Absolute fitness models provide approximate fitness values of new individuals to replace the actual expensive fitness evaluation. According to modelling approaches, absolute fitness models could be further divided into regression-based and similarity-based models. Regression-based models construct a mathematical function to model the actual fitness function. They have higher accuracy in approximating the fitness function. Besides, regression-based models could provide comprehensive information about the real fitness function, which is valuable for optimisation and problem analysis. The similarity-based models derive the fitness of unevaluated new individuals by measuring the correlation between the new individuals and their neighbours. It was the most convenient method among the four kinds of models, but it usually has low accuracy, especially when the fitness landscape is complex.

Relative fitness approximation models have recently been introduced to SAEAs. This relatively new kind of method focuses on the preference or rank of individuals rather than their absolute fitness values. Relative fitness models could be categorized into two sub-classes: rank-based and classification-based surrogate models. Rank-based models provide the rank information of the current population for selection, and classification-based methods compare individuals with one reference solution. These two models were both designed explicitly for EAs that only used the rank information in the selection process. Due to the special mechanism, they were only used in limited studies.

Comparison studies of different surrogate models were conducted from different perspectives, including execution time, model accuracy, parameter influence, and the performance of SAEAs. We selected RBF, kNN-R, RankSVM and SVC as the representative surrogate models and embedded them into a general EA, i.e., genetic algorithm. Firstly, the computation time of model construction and the prediction was compared, and kNN-R and RBF models were more efficient than RankSVM and SVC models, while RankSVM was the most time-consuming model. Then, the model accuracy was compared by using rank correlation as the metric. We found that the RBF model was more accurate than the other three surrogate models on most problem instances. Finally, we compared their efficacy by running the SAEAs assisted by different models and model management strategies. The results indicate that the pre-selection strategy is better than individual-based evolution control given the same surrogate model. The RBF model is still the most effective surrogate model among absolute and relative fitness models. In addition, we also found that the SVC model was only suitable for EAs with POTS.

For each surrogate model used in our experiment, we investigated the influence of some critical model parameters. The size of training data K did not make a big difference to the SAEA's performance when using RBF and kNN-R models, but it was very important when using RankSVM and SVC models. For individual-based evolution control, the parameter determining the re-evaluation ratio was important when using any of the four surrogate models. Low p_{sm} is more suitable for the RBF model; moderate p_{sm} is more suitable for the kNN-R model and large τ_{th} is more suitable for the RankSVM model. For the pre-selection strategy, the pre-selection factor's influence was not very obvious, and the results suggest that a moderate λ_{pre} is suitable for all surrogate models.

For future work, it is well worth to review the surrogate models applied in multi-objective optimization, especially the relative fitness surrogate models. Although there have been some reviews about surrogate models in multi-objective evolutionary algorithms [6, 7], there is still no work comparing different kinds of models by experimental analysis. And there is also no research studying the influence of parameter choice for different models in multi-objective evolutionary algorithms. Therefore, it is valuable to do such research on this topic.

Finally, we also found during our literature review that most works in SAEAs are concerned with unconstrained single-objective optimization. However, many real-world problems have constraints, multiple objectives and may be dynamic and contain uncertainty. There are only a few studies on SAEAs for constrained optimization, multi-objective optimization and dynamic optimization. These are interesting directions for future research.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Grant No. 61976111), Guangdong Provincial Key Laboratory (Grant No. 2020B121201001), the Program for Guangdong Introducing Innovative and Enterpreneurial Teams(Grant No. 2017ZT07X386), Shenzhen Science and Technology Program (Grant No. KQTD2016112514355531).

References

[1] Y. S. Ong, P. B. Nair, A. J. Keane, Evolutionary optimization of computationally expensive problems via surrogate modeling, AIAA Journal 41 (2003) 687–696.
[2] M. Urquhart, E. Ljungskog, S. Sebben, Surrogate-based optimisation using adaptively scaled radial basis functions, Applied Soft Computing 88 (2020) 106050.
[3] Y. Jin, A comprehensive survey of fitness approximation in evolutionary computation, Soft Computing 9 (2005) 3–12.
[4] D. Lim, Y. Jin, Y.-S. Ong, B. Sendhoff, Generalizing surrogate-assisted evolutionary computation, IEEE Transactions on Evolutionary Computation 14 (2010) 329–355.
[5] Y. Jin, Surrogate-assisted evolutionary computation: Recent advances and future challenges, Swarm and Evolutionary Computation 1 (2011) 61–70.
[6] T. Chugh, K. Sindhya, J. Hakanen, K. Miettinen, A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms, Soft Computing 23 (2019) 3137–3166.
[7] K. Deb, R. Hussein, P. C. Roy, G. Toscanopulido, A taxonomy for metamodeling frameworks for evolutionary multiobjective optimization, IEEE Transactions on Evolutionary Computation 23 (2019) 104–116.
[8] T. Bartz-Beielstein, M. Zaefferer, Model-based methods for continuous and discrete global optimization, Applied Soft Computing 55 (2017) 154–167.
[9] Y. Sun, H. Wang, B. Xue, Y. Jin, G. G. Yen, M. Zhang, Surrogate-assisted evolutionary deep learning using an end-to-end random forest-based performance predictor, IEEE Transactions on Evolutionary Computation 24 (2020) 350–364.
[10] H. Wang, Y. Jin, J. O. Jansen, Data-driven surrogate-assisted multiobjective evolutionary optimization of a trauma system, IEEE Transactions on Evolutionary Computation 20 (2016) 939–952.
[11] X. Sun, D. Gong, Y. Jin, S. Chen, A new surrogate-assisted interactive genetic algorithm with weighted semisupervised learning, IEEE Transactions on Cybernetics 43 (2013) 685–698.
[12] J. Ziegler, W. Banzhaf, Decreasing the number of evaluations in evolutionary algorithms by using a meta-model of the fitness function, in: European Conference on Genetic Programming, Springer, 2003, pp. 264–275.
[13] Y. Jin, B. Sendhoff, Fitness approximation in evolutionary computation-a survey, in: Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation, Morgan Kaufmann Publishers Inc., 2002, pp. 1105–1112.
[14] T. P. Runarsson, Ordinal regression in evolutionary computation, in: Parallel Problem Solving from Nature-PPSN IX, Springer, 2006, pp. 1048–1057.
[15] X. Lu, K. Tang, X. Yao, Classification-assisted differential evolution for computationally expensive problems, in: IEEE Congress on Evolutionary Computation (CEC), 2011, IEEE, 2011, pp. 1986–1993.
[16] X.-F. Lu, K. Tang, Classification-and regression-assisted differential evolution for computationally expensive problems, Journal of Computer Science and Technology 27 (2012) 1024–1034.
[17] X. Lu, K. Tang, B. Sendhoff, X. Yao, A new self-adaptation scheme for differential evolution, Neurocomputing 146 (2014) 2–16.
[18] F. A. Viana, A tutorial on latin hypercube design of experiments, Quality and Reliability Engineering International 32 (2016) 1975–1985.
[19] S. S. Garud, I. A. Karimi, M. Kraft, Design of computer experiments: A review, Computers & Chemical Engineering 106 (2017) 71–95.
[20] H. Wang, Y. Jin, J. Doherty, A generic test suite for evolutionary multifidelity optimization, IEEE Transactions on Evolutionary Computation 22 (2018) 836–850.
[21] X. Lu, T. Sun, K. Tang, Evolutionary optimization with hierarchical surrogates, Swarm and Evolutionary Computation 47 (2019) 21–32.
[22] H. Wang, Y. Jin, J. Doherty, Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems, IEEE Transactions on Cybernetics 47 (2017) 2664–2677.
[23] H. Yu, Y. Tan, C. Sun, J. Zeng, A comparison of quality measures for model selection in surrogate-assisted evolutionary algorithm, Soft Computing 23 (2019) 12417–12436.
[24] C. Sun, Y. Jin, J. Zeng, Y. Yu, A two-layer surrogate-assisted particle swarm optimization algorithm, Soft Computing 19 (2015) 1461–1475.
[25] B. Liu, Q. Zhang, G. G. Gielen, A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems, IEEE Transactions on Evolutionary Computation 18 (2014) 180–192.
[26] Y. Jin, B. Sendhoff, Reducing fitness evaluations using clustering techniques and neural network ensembles, in: Genetic and Evolutionary Computation Conference, Springer, 2004, pp. 688–699.
[27] L. Bajer, Z. Pitra, J. Repický, M. Holeňa, Gaussian process surrogate models for the CMA evolution strategy, Evolutionary Computation 27 (2019) 665–697.
[28] R. Cheng, C. He, Y. Jin, X. Yao, Model-based evolutionary algorithms: a short survey, Complex & Intelligent Systems 4 (2018) 283–292.
[29] Y. Jin, H. Wang, T. Chugh, D. Guo, K. Miettinen, Data-driven evolutionary optimization: an overview and case studies, IEEE Transactions on Evolutionary Computation 23 (2019) 442–458.
[30] A. I. Forrester, A. J. Keane, Recent advances in surrogate-based optimization, Progress in Aerospace Sciences 45 (2009) 50–79.
[31] X. Llorà, K. Sastry, D. E. Goldberg, A. Gupta, L. Lakshmi, Combating user fatigue in igas: partial ordering, support vector machines, and synthetic fitness, in: Genetic and Evolutionary Computation Conference, GECCO 2005, Proceedings, Washington DC, USA, June 25-29, 2005, 2005, pp. 1363–1370.

[32] R. G. Regis, Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions, IEEE Transactions on Evolutionary Computation 18 (2014) 326–347.
[33] T. Østergård, R. L. Jensen, S. E. Maagaard, A comparison of six metamodeling techniques applied to building performance simulations, Applied Energy 211 (2018) 89–103.
[34] R. Jin, W. Chen, T. W. Simpson, Comparative studies of metamodelling techniques under multiple modelling criteria, Structural and Multidisciplinary Optimization 23 (2001) 1–13.
[35] A. Díaz-Manríquez, G. Toscano, C. A. C. Coello, Comparison of metamodeling techniques in evolutionary algorithms, Soft Computing 21 (2017) 5647–5663.
[36] L. Fonseca, H. Barbosa, A. Lemonge, On similarity-based surrogate models for expensive single-and multi-objective evolutionary optimization, in: Computational Intelligence in Expensive Optimization Problems, Springer, 2010, pp. 219–248.
[37] R. E. Smith, B. A. Dike, S. A. Stegmann, Fitness inheritance in genetic algorithms, in: Proceedings of the 1995 ACM Symposium on Applied Computing, number 6 in SAC '95, ACM, 1995, pp. 345–350.
[38] C. Sun, J. Zeng, J. Pan, S. Xue, Y. Jin, A new fitness estimation strategy for particle swarm optimization, Information Sciences 221 (2013) 355–370.
[39] T. Hildebrandt, J. Branke, On using surrogates with genetic programming, Evolutionary Computation 23 (2015) 343–367.
[40] N. Hansen, The CMA evolution strategy: a comparing review, in: Towards a new evolutionary computation, Springer, 2006, pp. 75–102.
[41] C. Li, S. Yang, T. T. Nguyen, A self-learning particle swarm optimizer for global optimization problems, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42 (2012) 627–646.
[42] W. Gong, A. Zhou, Z. Cai, A multioperator search strategy based on cheap surrogate models for evolutionary optimization, IEEE Transactions Evolutionary Computation 19 (2015) 746–758.
[43] I. Loshchilov, M. Schoenauer, M. Sebag, Self-adaptive surrogate-assisted covariance matrix adaptation evolution strategy, in: Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, 8, ACM, 2012, pp. 321–328.
[44] C. Huang, B. Radi, A. E. Hami, H. Bai, CMA evolution strategy assisted by kriging model and approximate ranking, Applied Intelligence 48 (2018) 4288–4304.
[45] J. Zhang, A. Zhou, K. Tang, G. Zhang, Preselection via classification: A case study on evolutionary multiobjective optimization, Information Sciences 465 (2018) 388–403.
[46] J. Zhang, J. X. Huang, Q. V. Hu, Boosting evolutionary optimization via fuzzy-classification-assisted selection, Information Sciences 519 (2020) 423–438.
[47] L. Pan, C. He, Y. Tian, H. Wang, X. Zhang, Y. Jin, A classification-based surrogate-assisted evolutionary algorithm for expensive many-objective optimization, IEEE Transactions on Evolutionary Computation 23 (2019) 74–88.
[48] S. D. Handoko, C. K. Kwoh, Y.-S. Ong, Feasibility structure modeling: an effective chaperone for constrained memetic algorithms, IEEE Transactions on Evolutionary Computation 14 (2010) 740–758.
[49] Q. Chen, B. Liu, Q. Zhang, J. Liang, P. Suganthan, B. Qu, Problem definitions and evaluation criteria for cec 2015 special session on bound constrained single-objective computationally expensive numerical optimization, Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou, China and Technical Report, Nanyang Technological University (2014).
[50] F. Peng, K. Tang, G. Chen, X. Yao, Population-based algorithm portfolios for numerical optimization, IEEE Transactions on evolutionary computation 14 (2010) 782–800.