Scaling Up Radial Basis Function for High-Dimensional Expensive Optimization Using Random Projection

Daofu Guo
School of Automation Science and Engineering
Xi'an Jiaotong University
Xi'an, China
gdf19951205@stu.xjtu.edu.cn

Zhigang Ren*
School of Automation Science and Engineering
Xi'an Jiaotong University
Xi'an, China
renzg@xjtu.edu.cn

Yongsheng Liang
School of Automation Science and Engineering
Xi'an Jiaotong University
Xi'an, China
liangyongsheng@stu.xjtu.edu.cn

An Chen
School of Automation Science and Engineering
Xi'an Jiaotong University
Xi'an, China
chenan123@stu.xjtu.edu.cn

***Abstract*—Surrogate model assisted evolutionary algorithms (SAEAs) have attracted much research attention in solving computationally expensive optimization problems. They show excellent performance on low-dimensional optimization problems by saving a large number of real fitness evaluations, but generally fail on high-dimensional problems due to the contradiction between the huge solution space and the limited computational resources. To alleviate this issue, this study attempts to scale up radial basis function (RBF), which is a kind of widely used surrogate model, by taking advantage of the random projection (RP) technique, and thus develops a RP-based RBF (RP-RBF). Different from existing methods that directly train RBF in the original solution space, RP-RBF first randomly projects the original high-dimensional solution space onto many low-dimensional subspaces, and then trains an RBF in each subspace. The resulting low-dimensional RBFs are finally used together to approximate the fitness values of new candidate solutions. The introduction of RP greatly reduces the number of training samples required by RBF on the one hand, and helps RBF still capture the main characteristics of the original problems on the other hand. To verify the effectiveness of RP-RBF, this study integrates it with a differential evolution (DE) and develops a novel SAEA named RP-RBF-DE. Experimental results on a set of 12 benchmark functions demonstrate that RP-RBF significantly improves the accuracy of the traditional RBF and RP-RBF-DE outperforms the traditional DE and a general RBF-assisted DE.**

Keywords—high-dimensional expensive optimization, random projection, radial basis function, differential evolution

I. INTRODUCTION

With the rapid development of technology, optimization problems are becoming increasingly complex. Traditional optimization algorithms based on gradient information , such as Newton's method [1] and conjugate gradient method [2], have been unable to meet actual needs in many scenarios because of the black-box characteristic most optimization problems possess and are gradually replaced by some potential evolutionary algorithms (EAs), such as differential evolution (DE) [3], genetic algorithms [4] and particle swarm optimization [5]. These EAs have been applied to many real world optimization problems, ranging from gene recognition [6], drug design [7] to wireless networks [8], and air traffic management [9]. A common characteristic of most EAs is that they generally require a large number of fitness evaluations (FEs) to find a promising solution. This would greatly restrict the performance of EAs in solving expensive optimization problems, such as antenna design [10], structural optimization [11] and circuit design [12], since a single real fitness evaluation of these problems would be very time-consuming and resource-consuming.

Surrogate model assisted evolutionary algorithms (SAEAs) have received increasing attention in solving expensive optimization problems. The main idea of SAEAs is to build an approximate model for the original computationally expensive problem, and then evaluate candidate solutions based on the built model. Only those solutions that are identified as promising ones will be evaluated again by the real fitness function. This approach can effectively reduce the real fitness evaluations and thus improve the overall performance. The most commonly used surrogate models include polynomial regression [13], support vector machines [14], radial basis function (RBF) [15] and Gaussian processes [16]. Based on these surrogate models, many SAEAs have been proposed to solve expensive optimization problems, including surrogate-assisted DE [17], surrogate-assisted genetic algorithm [18] and surrogate-assisted particle swarm optimization [19].

At present, most SAEAs are mainly used to solve low-dimensional or medium-dimensional problems with a dimension of 50 or less, and they can hardly be employed to solve high-dimensional problems with more than a hundred variables. One of the main reason is the so called "curse of dimensionality" [20]. The search space of an optimization problem will increase exponentially with the increase of the problem dimension, resulting in the EAs cannot fully explore the solution space within acceptable computational time. SAEAs also will need a large number of training samples to build sufficiently accurate approximate models with the increase of problem dimension, which is impracticable for

* Corresponding author: Zhigang Ren (renzg@xjtu.edu.cn).

Research supported in part by the National Natural Science Foundation of China under Grant 61873199, and in part by the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2020JM-059, and in part by the Fundamental Research Funds for the Central Universities under Grant xzy022019028.

expensive optimization problems with limited computational resources.

In recent years, there are a handful of attempts on solving high-dimensional expensive problems with SAEAs. Sun *et al.* [21] proposed a surrogate-assisted cooperative swarm optimization algorithm (SA-COSO) to solve high-dimensional computationally expensive problems. SA-COSO built a global RBF surrogate to capture the profile of problem with the aim of locating promising solution regions, which would be further exploited by a local surrogate model. Cai *et al.* [22] introduced a generalized surrogated-assisted evolutionary algorithm (GSGA), which uses surrogate models to help guide the updating mechanism and prescreen promising solutions. Liu *et al.* [23] introduced Sammon mapping technique into a Gaussian process surrogate model which could reduce the dimension of solution space to make it easy for building surrogate models.

In this paper, we propose a random projection-based [24] RBF surrogate model (RP-RBF) for fitness approximation of high-dimensional problems. Different from most existing methods that directly train the surrogate model in the original space, RP-RBF first projects the samples in high-dimensional space into many low-dimensional subspaces using RP technique, and then trains an RBF model in each subspace based on the projected samples to capture the characteristics of the original problem. Finally, these surrogate models in all subspaces will be integrated together to approximate the fitness values of new candidate solutions. Combining RP-RBF and a DE algorithm, we further develop a novel SAEA named RP-RBF-assisted DE (RP-RBF-DE) to improve the performance of DE in solving high-dimensional expensive optimization problems. In order to demonstrate the performance of our method, RP-RBF-DE was tested on a set of standard benchmark functions with 100 and 200 dimensions and compared with a general RBF assisted DE (RBF-DE) and a typical DE. Experimental results indicate that RP-RBF could effectively improve the approximation accuracy of RBF and RP-RBF-DE can have an edge over RBF-DE and DE on different kinds of high-dimensional expensive problems with limited real fitness evaluations.

The rest of this paper is organized as follows. Section II briefly introduces the RBF model and the random projection technique. Section III describes the procedures of RP-RBF and RP-RBF-DE. Section IV gives the experimental results and discussions. The conclusion is finally presented in Section V.

II. Background

A. Radial Basis Function

Hardy [25] first proposed RBF as an approximation model. Compared with other models, RBF model is easy to train and not sensitive to the problem dimension. Therefore, the RBF model is employed as the basic surrogate model in our method. Our study employs the RBF model given in [26], which is defined as follows:

Given n distinct points $\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n \in \mathbb{R}^D$, where their fitness values are $f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \cdots, f(\boldsymbol{x}_n)$. In this paper, we use the following interpolating form:

$$\hat{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \omega_i \phi(\| \boldsymbol{x} - \boldsymbol{x}_i \|) + \boldsymbol{b}^T \boldsymbol{x} + a, \boldsymbol{x} \in \mathbb{R}^D \quad (1)$$

where $\phi(\cdot)$ denotes the basis function, $\|\cdot\|$ denotes the Euclidean norm, and $(\omega_1, \omega_2, \cdots, \omega_n)^T = \boldsymbol{\omega} \in \mathbb{R}^n$, $\boldsymbol{b} \in \mathbb{R}^D$ and $a \in \mathbb{R}$ are corresponding parameters. There are different choices for the basis function, such as the Gaussian basis function, thin plate spline basis function and cubic basis function [27]. In this paper, we adopt the cubic basis function: $\phi(r) = r^3$ as the basis function in RP-RBF.

With the given distinct points, the unknown parameters $\boldsymbol{\omega} \in \mathbb{R}^n$, $\boldsymbol{b} \in \mathbb{R}^D$ and $a \in \mathbb{R}$ can be obtained by solving the following linear equations:

$$\begin{pmatrix} \Phi & P \\ P^T & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{\omega} \\ \boldsymbol{\gamma} \end{pmatrix} = \begin{pmatrix} F \\ \mathbf{0} \end{pmatrix} \quad (2)$$

where $\boldsymbol{\Phi}$ is an $n \times n$ matrix with $\Phi_{ij} = \Phi(\| \boldsymbol{x}_i - \boldsymbol{x}_j \|)$ and $P^T = \begin{pmatrix} \boldsymbol{x}_1^T & \boldsymbol{x}_2^T & \cdots & \boldsymbol{x}_n^T \\ 1 & 1 & \cdots & 1 \end{pmatrix}$, $\boldsymbol{\omega} = (\omega_1, \omega_2, \cdots, \omega_n)^T$, $\boldsymbol{\gamma} = (\boldsymbol{b}^T, a)^T$ and $F = \left(f(\boldsymbol{x}_1), f(\boldsymbol{x}_2), \cdots, f(\boldsymbol{x}_n) \right)^T$.

If and only if rank $(P) = D + 1$, the (2) has a unique solution [26]. As a result, in order to get a unique RBF model, the number of distinct points must be greater than or equal to $D + 1$.

B. Random Projection

A very natural idea to solve high-dimensional problems is to perform dimensionality reduction. Random projection [24], as a dimensionality reduction method, projects the samples in original high-dimensional space into many uniformly and randomly generated low-dimensional subspaces. RP has been used in many fields, including computer science, signal processing and machine learning. It enjoys nice theoretical characteristics, among which the most attractive is the low distortion of the Euclidean geometry. The reason for its nice characteristics is that the theory of RP is based on Johnson-Lindenstrauss Lemma [28] as follows:

Johnson-Lindenstrauss Lemma: For an $\varepsilon (0 < \varepsilon < 0.5)$ and an integer n, let $k(n, \varepsilon) = \lfloor 9(\varepsilon^2 - 2\varepsilon^2/3)^{-1} \log n \rfloor + 1$. If $n > k(n, \varepsilon)^2$, then for any n-point set S in $\mathbb{R}^n$, there exists a map $f : S \to \mathbb{R}^{k(n,\varepsilon)}$ such that $(1-\varepsilon)\|u - v\|^2 < \|f(u) - f(v)\|^2 < (1+\varepsilon)\|u - v\|^2$ for all u, v in S.

The specific proof of this lemma can refer to [28]. This Lemma guarantees that the relative distance of a set of points in the high-dimensional Euclidean space can be maintained in a certain error range when mapped to a low-dimensional space. This property makes RP suitable for models like RBF whose calculation and analysis are based on the distances among points since RP could guarantee the training accuracy of RBF in low-dimensional subspaces to a certain extent. Besides, an appropriate combination of estimates from a set of random subspaces also have smooth effect, which is helpful to ensure

the approximation accuracy even with a small number of training samples.

The specific process of random projection method can be described as follows:

Given a set of samples DB including N points: $DB = (\boldsymbol{x}_i \in \mathbb{R}^d)_{i=1:N} \in \mathbb{R}^{N\times d}$, and then we randomly generate M different random matrices $(R_i \in \mathbb{R}^{k\times d})_{i=1:M}$ with entries drawn i.i.d from a univariate Gaussian $N(0,\sigma^2)$. Then we can get M new population $DB_i = (R_i \bullet DB^T)_{i=1:M}$. In this way, we project sample points from $\mathbb{R}^d$ to $\mathbb{R}^k$, where k represents the dimension of the low-dimensional subspace and M represents the number of the low-dimensional subspaces. In this paper, we choose $\sigma^2 = 1/d$ as recommended in [24] to make R well approximate an orthonormal matrix on condition that k is much lower than d.

III. Method

In this section, the random projection-based RBF (RP-RBF) is first proposed. Then RP-RBF is integrated into a typical DE and the complete procedure of the resultant RP-RBF-DE is presented.

A. RP-RBF

As mentioned in Section II (B), RP can guarantee the low distortion of the Euclidean geometry when projecting the samples in high-dimensional space into low-dimensional space. According to the interpolating form equation of RBF presented in Section II (A), we can find that the establishment of RBF is related to the distance between distinct points to some extent. Based on the above idea, we build RBF models in a set of low-dimensional subspaces generated by RP.

Given a set of training samples with their real fitness values and a set of candidate solutions waiting for evaluation in the original space, RP-RBF first generates a number of low-dimensional subspaces using the RP technique and projects the training samples and candidate solutions into every subspace. Then in each subspace, RP-RBF trains a RBF model based on these projected training samples and evaluate the fitness values of the projected candidate solutions with the trained RBF model. After the subspace training and evaluation process, the fitness value of a candidate solution in the original space is computed as the average of its corresponding fitness values in all the subspaces. The reason we use the average value lies in that these subspaces are all generated randomly using the RP technique and thus they possess the same weight. By training surrogate models in low-dimensional subspaces, fewer training samples are required and the approximation accuracy may also be improved. The detailed procedure of RP-RBF is presented in **Algorithm 1**. In step 1, RP-RBF initializes the parameters including the low-dimensional subspace dimensionality k and the number of low-dimensional subspaces M. In steps 3 to 7, training samples DB and population PB are mapped to M $k-$dimensional subspaces with random projection and then a RBF model is built in each $k-$dimensional space. Projected solutions in PB are evaluated by these models and the results are stored in PB_value. It should be noted that the training and evaluation operations in different subspaces are independent of each other and thus they can be implemented in parallel to quicken the overall process. In step 9, the average of $PB\_value_i, i = 1,2,\cdots M$ will be calculated as the final approximated fitness values of solutions in PB.

Algorithm1: Procedure of RP-RBF

Input: training samples $DB = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_N\} \in \mathbb{R}^{N\times D}$ with their real fitness evaluations $F = \{f(\boldsymbol{X}_1), f(\boldsymbol{X}_2), \cdots, f(\boldsymbol{X}_N)\}$ and solutions to be evaluated $PB = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_L\}$;

Output: the approximated fitness values PB_value of solutions in PB;

1. Initialize parameters, including low-dimensional subspace dimensionality k, number of low-dimensional subspaces M;
2. **for** $i = 1 : M$
3. Generate a random projection matrix $R_i \in \mathbb{R}^{k\times D}$;
4. Project DB into k-dimensions: $DB_i = [R_i \cdot DB^T]^T \in \mathbb{R}^{N\times k}$;
5. Project PB into k-dimensions: $PB_i = [R_i \cdot PB^T]^T \in \mathbb{R}^{L\times k}$;
6. Build a RBF surrogate $\hat{f}_i$ with DB_i and F;
7. Evaluate PB_i with $\hat{f}_i$ and store the values into $PB\_value_i \in \mathbb{R}^{1\times L}$;
8. **end**
9. Calculate $PB\_value \in \mathbb{R}^{1\times L}$ as the average of $PB\_value_i, i = 1, 2, \cdots M$.

B. RP-RBF-DE

To solve high-dimensional expensive optimization problems, we further integrate RP-RBF with a DE and develop a novel SAEA named RP-RBF-assisted DE (RP-RBF-DE). DE has shown great success in many real-world optimization problems but it generally requires a large number of real fitness evaluations to find promising solutions. RP-RBF can help DE save many real evaluations and RP-RBF-DE has the potential to obtain better optimization results with limited computational resources.

DE shares a similar framework with most evolutionary algorithms. It begins with an initial population $P = \{\boldsymbol{x}_i \,|\, i = 1,2,\cdots,NP\}$ with size NP. In this paper, we use a variant of DE called DE/best/1, which uses the best solution in each generation as the base vector to speed up generating more promising candidates.

Let $\boldsymbol{x}^{best}$ be the best solution in P and $\boldsymbol{x} = (x_1, x_2, \cdots, x_D) \in \mathbb{R}^D$ be a solution in P. An intermediate vector is first produced by mutation operator:

$$\boldsymbol{v} = \boldsymbol{x}^{best} + F \cdot (\boldsymbol{x}_{r1} - \boldsymbol{x}_{r2}) \tag{3}$$

where $\boldsymbol{x}_{r1}$ and $\boldsymbol{x}_{r2}$ are two distinct solutions selected from P randomly and are different form $\boldsymbol{x}^{best}$. F is the scaling factor.

After mutation, a binary crossover operator generates the final child solution $\boldsymbol{u} = (u_1, u_2, \cdots, u_D) \in \mathbb{R}^D$ as follows:

$$u_j = \begin{cases} v_j, & \text{if } (rand_j \le CR) \text{ or } j = j_{rand} \\ x_j, & \text{otherwise} \end{cases} \tag{4}$$

where $rand_j$ is a uniformly distributed random number within (0,1), $CR \in [0,1]$ is called the crossover rate.

In RP-RBF-DE, the trained RP-RBF model is used to evaluate the fitness values of the new solutions in each generation. Only some high-quality solutions selected by RP-RBF will be further evaluated by the real fitness function. All the solutions evaluated by real functions and their fitness values will be maintained in a database as an archive for updating the RP-RBF model. As a result, the RP-RBF model will be more and more accurate and the performance of DE can be improved with limited computational resources. The detailed procedure of RP-RBF-DE is shown in **Algorithm 2**. In step 1, RP-RBF-DE initializes the parameters including population size NP, number of training samples λ, number of new solutions added in the database in each generation τ, scaling factor F and crossover rate CR. In steps 2 and 3, the database is initialized randomly. In steps 5 and 6, NP best solutions from the database are selected to form the population P and NP child solutions are generated by applying crossover and mutation operators. Then in step 7, λ newest solutions in the database and their fitness values are used to build RP-RBF surrogate models and evaluate the child solutions generated in step 6. In steps 8 and 9, the best τ solutions are selected by their approximated fitness values. These solutions will be evaluated again by the real fitness function and added to the database together with their real fitness values. RP-RBF-DE repeats steps 4-9 until meeting the stopping criterion.

Algorithm 2:Procedure of RP-RBF-DE

1. Initialize parameters, including population size NP, number of training samples λ, number of new solutions added in the database in each generation τ, scaling factor F and crossover rate CR.
2. Randomly initialize N solutions $DB = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_N\}$;
3. Evaluate all solutions in DB with the real function. Put these solutions and their real fitness values in the initial database;
4. **while** the stopping criterion is not met **do**
5. Select NP best solutions from the database to form a population P;
6. Apply the mutation and crossover operators of DE on P to generate NP child solutions;
7. Take the λ newest solutions from the database and their real fitness evaluations as the training samples to evaluate the NP child solutions generated in step 6 by applying **Algorithm 1**;
8. Select the best τ child solutions according to their approximated fitness values;
9. Evaluate the best τ solutions get in step 8 with the real fitness function. Add these solutions and their real function values to the database;
10. **end**
11. Output the obtained best solution.

IV. Experimental Studies

In this section, the effectiveness of RP-RBF and RP-RBF-DE were tested on a set of 12 benchmark functions (f_1-f_{12}) from IEEE CEC 2005 test suite [29]. These functions can be classified into two classes according to their characteristics, including 5 unimodal functions f_1-f_5 and 7 multimodal functions f_6-f_{12}. More details of them can be found in [29].

In our experiments, each algorithm was tested on these functions with 100 dimensions and 200 dimensions. An algorithm would be terminated when a maximum number of FEs was reached. The maximum number of FEs for functions with 100 dimensions and 200 dimensions were set to 20000 and 30000, respectively. And 20 independent runs were conducted on each function. At each run, we recorded the function error value (FEV), i.e. $f(\boldsymbol{x}) - f(\boldsymbol{x}^*)$, to evaluate the performance of an algorithm, where $\boldsymbol{x}$ is the obtained best solution and $\boldsymbol{x}^*$ is the optimal one.

A. Parameter Settings

There are seven parameters need to be set in advance. Their settings are given as follows.

1) The population size NP. Population size has great influence on the exploration and exploitation ability of an algorithm. A small size may cause premature convergence easily and a large size would bring slow convergence rate. According to our prior experiments, we set NP=50 in the following experiments, which is consistent with the setting in [23].

2) The scaling factor F and the crossover rate CR of DE. The setting of parameters in DE has been fully studied by many works, and we set both F and CR to 0.8 according to the suggestion in [30].

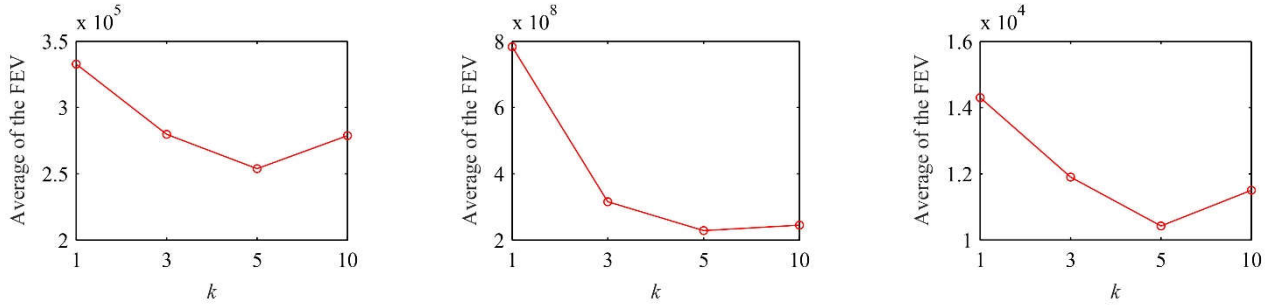
3) The number of training samples λ for building RP-RBF models. A large number of training samples can improve the model quality, but will also consume too much computational resources. Considering that the samples are all projected into the low-dimensional subspaces in RP-RBF, a smaller number of training samples would be sufficient for building models in subspaces. Based on the above considerations, we set λ=110 for 100-dimensional problems and λ=210 for 200-dimensional problems.

4) The number of new solutions added to the database in each generation τ. Based on the above settings, our preliminary experiments on τ with different values taken from $\{1, 10, 20, 30\}$ showed that a smaller number of τ would bring a slight improvement to optimization results and more computational cost. Finally, we suggest setting τ =20 in our experiments.

The rest of this subsection mainly focuses on analyzing the influence of the parameters in the random projection technique, i.e. dimensionality of the projected low-dimensional subspace k and the number of low-dimensional subspaces M. According to [24], in order to cover as much solution space as possible, the number of M must be set above the minimum $M_{\min} = [D/k]$, where D is the dimensionality of the original search space. In this paper, we set $M = \text{ceil}(4 * D/k)$. Next we focus on setting the value of k which should be much smaller than D so that the RP technology can take effect. To find a proper value for k, we conducted experiments on benchmark functions with 100D when varying the value of k from $\{1, 3, 5, 10\}$ to observe the performance of RP-RBF-DE. For brevity, we only reported the results on three functions, i.e. unimodal functions f_2, f_3 and multimodal function f_7.

Fig.1 presents the results obtained by RP-RBF-DE on the three functions with different k values. It can be vividly

observed from Figs. 1(a)-(c) that when the value of k changes, the results obtained by RP-RBF-DE don't have significant difference since they are all within the same order of magnitude. It indicates that RP-RBF-DE is relatively robust to k in a sense. RP-RBF-DE performs best when $k = 5$ and its performance would generally degenerate when k is too small or too large. For a small value of k, the subspace is too narrow to reveal the characteristics of the original space and over-fitting may also occur when training a low-dimensional RBF model with too much samples in a subspace. On the other hand, if k is set too large, the dimensional reduction effect of RP technique would decrease and thus cause performance deterioration. Based on the experimental results and to balance the performance on different functions, we suggest setting $k = 5$ in the following experiments.



(a) f_2 (b) f_3 (c) f_7

Fig. 1. Performance of RP-RBF-DE with different k values

B. *Performance of RP-RBF-DE*

In this section, the performance of RP-RBF-DE was evaluated on 12 benchmark functions from CEC 2005 test suite. As comparison, a basic DE and a RBF-assisted DE (RBF-DE) which directly trains RBF model in the original space are also included in our experiments. To make a fair comparison, the common parameters of RP-RBF-DE, RBF-DE and DE were set to the same values. The only difference between RP-RBF-DE and RBF-DE lies in the adopted surrogate model, their comparison results can reflect the effectiveness of the RP-RBF model. The basic DE works as a baseline.

Tables I and II report the final results obtained by RP-RBF-DE, RBF-DE and DE. Cohen's d effect size [31] was used to measure the performance difference between RP-RBF-DE and the other two algorithms, where "+", "−" and "≈" in Tables I and II denote that the result of the corresponding algorithm is better than, worse than and similar to that of RP-RBF-DE, respectively. Furthermore, the best results in Tables are highlighted in bold.

From Tables I and II, we can summarize that:

1) For all the 12 functions with 100D or 200D, RP-RBF-DE performs no worse than RBF-DE on all the functions and they both have an edge over DE except f_8 (a complex shifted rotated Ackley's function with global optimum on bounds), where they have similar performance on. More concretely, for unimodal functions f_1-f_5 with 100D and 200D, RP-RBF-DE could obtain better results than RBF-DE on all the 5 functions. As for the 7 multimodal functions, whatever with 100D or 200D, RP-RBF-DE performs significantly better than RBF-DE on 6 functions. The results demonstrate that surrogate-assisted DE could effectively improve the performance of DE on different kinds of functions within limited real fitness evaluations. And the proposed RBF-DE could improve the performance of RBF on high-dimensional problems.

TABLE I. THE MEAN AND STANDARD DEVIATION (MEAN ± STANDARD DEVIATION) OF FEVS OBTAINED BY DE, RBF-DE AND RP-RBF-DE ON 12 CEC 2005 BENCHMARK FUNCTIONS WITH 100D

Fun.	DE	RBF-DE	RP-RBF-DE
f_1	2.55E+04 ± 3.37E+03 −	1.53E+04 ± 2.72E+03 −	**3.33E+03 ± 1.05E+03**
f_2	4.93E+05 ± 4.43E+04 −	3.37E+05 ± 3.88E+04 −	**2.65E+05 ± 5.45E+04**
f_3	2.90E+09 ± 3.80E+08 −	9.79E+08 ± 2.56E+08 −	**2.95E+08 ± 9.17E+07**
f_4	6.07E+05 ± 5.73E+04 −	4.48E+05 ± 4.97E+04 −	**3.82E+05 ± 6.84E+04**
f_5	5.08E+04 ± 3.59E+03 −	3.22E+04 ± 3.38E+03 −	**2.76E+04 ± 5.30E+03**
f_6	4.74E+09 ± 2.10E+09 −	2.61E+09 ± 9.46E+08 −	**3.71E+08 ± 1.92E+08**
f_7	1.48E+04 ± 1.02E+02 −	1.23E+04 ± 3.55E+00 −	**1.11E+04 ± 2.38E-02**
f_8	2.14E+01 ± 2.67E-02 ≈	2.14E+01 ± 2.44E-02 ≈	**2.14E+01 ± 2.28E-02**
f_9	1.19E+03 ± 1.15E+01 −	7.52E+02 ± 5.59E+01 −	**5.23E+02 ± 1.12E+02**
f_{10}	1.38E+03 ± 6.62E+01 −	1.16E+03 ± 5.51E+01 −	**1.14E+03 ± 4.56E+01**
f_{11}	2.01E+02 ± 5.83E-14 −	1.68E+02 ± 1.49E+00 −	**1.61E+02 ± 1.30E+01**
f_{12}	2.82E+07 ± 1.49E+06 −	2.25E+07 ± 2.33E+06 −	**1.71E+07 ± 3.57E+06**
No. of +/≈/-	0/1/11	0/1/11	−

2) When the problem dimension changes from 100 to 200, the performance of the three algorithms would all degenerate. Even so, RP-RBF-DE could successfully find better solutions than the other two algorithms which means that the proposed RP-RBF has good scalability to the problem dimension.

The last rows of Tables I and II report the overall comparison results, from which we can conclude that RP-RBF-DE indicates significant superiority over RBF-DE on this set of benchmark functions with different dimensions. The success of RP-RBF-DE mainly benefits from the random projection technique, which could help to obtain more accurate RBF model with small number of samples in subspaces.

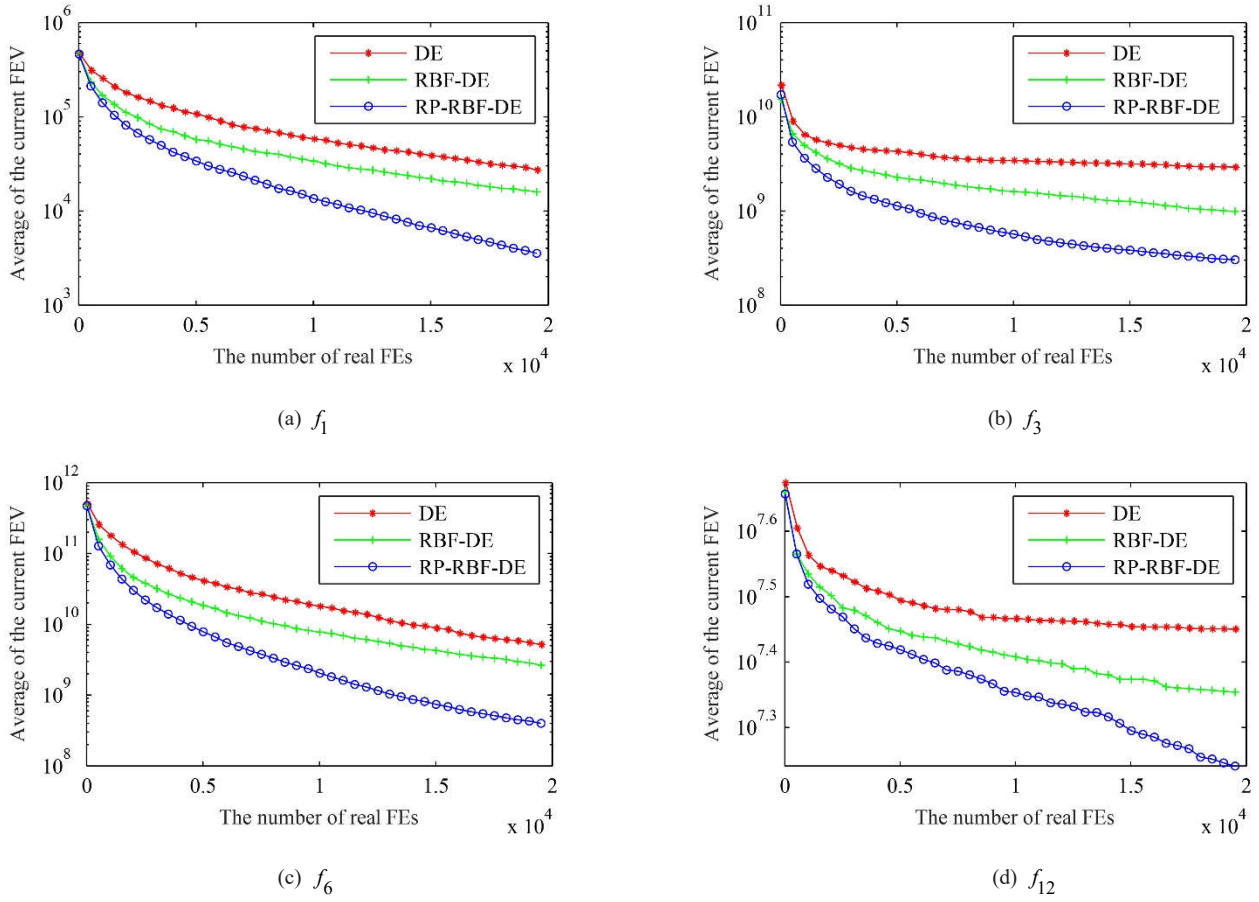
TABLE II. THE MEAN AND STANDARD DEVIATION (MEAN ± STANDARD DEVIATION) OF FEVS OBTAINED BY DE, RBF-DE AND RP-RBF-DE ON 12 CEC 2005 BENCHMARK FUNCTIONS WITH 200D

Fun.	DE	RBF-DE	RP-RBF-DE
f_1	9.92e+04 ± 6.97e+03 −	8.56e+04 ± 1.13e+04 −	**2.78e+04 ± 7.05e+03**
f_2	1.99e+06 ± 1.58e+05 −	1.36e+06 ± 1.58e+05 −	**9.99e+05 ± 1.15e+05**
f_3	1.04e+10 ± 9.60e+08 −	3.58e+09 ± 6.81e+08 −	**1.27e+09 ± 4.95e+08**
f_4	2.42e+06 ± 2.94e+05 −	1.74e+06 ± 2.16e+05 −	**1.61e+06 ± 2.41e+05**
f_5	1.09e+05 ± 4.85e+03 −	7.79e+04 ± 6.59e+03 −	**6.98e+04 ± 6.15e+03**
f_6	2.97e+10 ± 5.04e+09 −	2.79e+10 ± 5.82e+09 −	**7.92e+09 ± 2.50e+09**
f_7	8.73e+04 ± 1.03e+03 −	6.55e+04 ± 2.61e+01 −	**6.39e+04 ± 2.41e+00**
f_8	**2.15e+01 ± 1.56e-02** ≈	2.15e+01 ± 1.90e-02 ≈	2.15e+01 ± 1.59e-02
f_9	2.49e+03 ± 2.74e+01 −	1.97e+03 ± 1.34e+02 −	**1.48e+03 ± 1.78e+02**
f_{10}	3.35e+03 ± 7.80e+01 −	2.82e+03 ± 1.17e+02 −	**2.66e+03 ± 8.94e+01**
f_{11}	4.03e+02 ± 1.17e-13 −	3.53e+02 ± 2.70e+00 −	**3.51e+02 ± 3.88e+00**
f_{12}	1.50e+08 ± 3.78e+06 −	1.19e+08 ± 7.50e+06 −	**7.35e+07 ± 1.68e+07**
No. of +/ ≈ /-	0/1/11	0/1/11	−

To further reveal the performance of RP-RBF-DE, its convergence curves on different test functions were recorded. For brevity, only the results on four functions, i.e. two unimodal functions f_1, f_3 and two multimodal functions f_6, f_{12}, were presented. Figs. 2 and 3 present the convergence curves of RP-RBF-DE, RBF-DE and DE on the four functions with 100D and 200D, respectively. It can be observed from Figs. 2 and 3 that RP-RBF-DE generally has faster convergence rate than RBF-DE and DE on different functions. This results indicate that the proposed RP-RBF model performs well on high-dimensional problems and could effectively reduce the number of real fitness evaluations. Besides, the performance of RP-RBF-DE is also very stable on different functions with different dimensions, which is consistent with the results in Tables I and II.

V. CONCLUSION

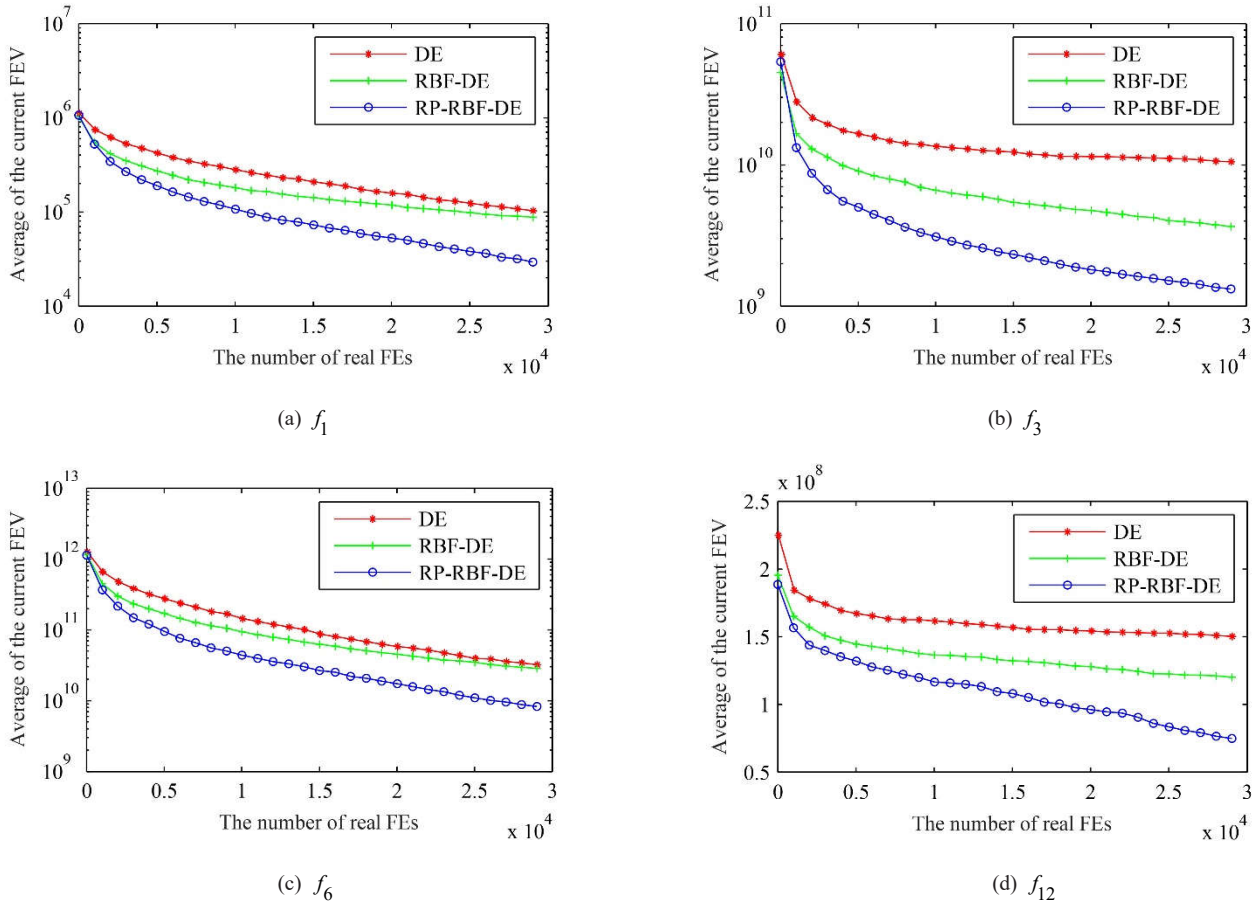
In this paper, a random projection-based radial basis function (RP-RBF) surrogate model is proposed for fitness approximation of high-dimensional problems. Different from existing methods that directly train surrogate model in the original space, RP-RBF randomly projects the samples into low-dimensional subspaces and trains an RBF model based on the projected samples for each subspace. Then the models in all subspaces are synthetically used to approximate the fitness values of new candidate solutions. By transforming the



(a) f_1 (b) f_3

(c) f_6 (d) f_{12}

Fig. 2. Convergence curves on 4 functions with 100D



(a) f_1 (b) f_3

(c) f_6 (d) f_{12}

Fig. 3. Convergence curves on 4 functions with 200D

model training and evaluation process into subspaces through random projection, the required number of training samples and the training difficulty are both reduced, which leads to more accurate approximation accuracy and more efficient performance. Moreover, the proposed RP-RBF is combined with a DE and a SAEA named RP-RBF-assisted DE (RP-RBF-DE) is developed for solving high-dimensional expensive optimization problems. Experimental results on a set of 12 benchmark functions demonstrate that RP-RBF-DE performs very stable on different functions with different dimensions and can always achieve better results than RBF-DE and DE with limited real fitness evaluations.

Our future work will focus on investigating the robustness and efficiency of the RP-RBF model on higher-dimensional problems. It is also interesting to combine different surrogate models with RP technique to develop different models for high-dimensional problems. Moreover, we will put more emphasis on real-world optimization problems.

REFERENCES

[1] D.K.R. Babajee, M.Z. Dauhoo, “An analysis of the properties of the variants of Newton’s method with third order convergence,” Applied Mathematics & Computation, vol. 183, no.1 pp. 659-684, 2006.

[2] M.R. Hestenes , E.L. Stiefel, “Methods of Conjugate Gradients for Solving Linear Systems,” Journal of Research of the National Bureau of Standards (United States), vol. 49, pp. 409-436, 1952.

[3] K.V. Price, R.M. Storn and J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Berlin, Germany, Springer, 2005.

[4] J. Stender, "An Introduction to genetic algorithms," IEE Colloquium on Applications of Genetic Algorithms, London, UK, pp. 1-4, 1994.

[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," Proceedings of ICNN'95 - International Conference on Neural Networks, Perth, WA, Australia, vol.4, pp. 1942-1948, 1995.

[6] J.Y. Sun, J.M. Garibaldi and C. Hodgman, "Parameter estimation using meta-heuristics in systems biology: A comprehensive review," IEEE/ACM Transactions on Computational Biology & Bioinformatics, vol.9, no.1, pp. 185-202, 2012.

[7] D. Douguet, "e-LEA3D: a computational-aided drug design web server", Nucleic Acids Research, vol.38, pp. 615-621, 2010.

[8] Y. Yoon and Y. Kim, "An Efficient Genetic Algorithm for Maximum Coverage Deployment in Wireless Sensor Networks," in IEEE Transactions on Cybernetics, vol. 43, no. 5, pp. 1473-1483, Oct. 2013.

[9] A. Pasini, P. Notry and D. Delahaye, "Direct Route Optimization for Air Traffic Management Improvement," 2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC), London, pp. 1-9, 2018.

[10] T. Milligan and J. Wiley, Modern Antenna Design. New York NY, USA:Wiley Online Library, 2005.

[11] Y. Jin and B. Sendhoff, "A systems approach to evolutionary multiobjective structural optimization and beyond," in IEEE Computational Intelligence Magazine, vol. 4, no. 3, pp. 62-76, Aug. 2009.

[12] D. Allstot, K. Choi, and J. Park, Parasitic-Aware Optimization of CMOSRF Circuits. The Netherlands: Springer, 2003.

[13] G. G. Wang, “Adaptive response surface method using inherited latin hypercube design points,”Journal of Mechanical Design, vol. 125, no.2, pp. 210-220, 2003.

[14] A. J. Smola, and B. Schölkopf, “A tutorial on support vector regression,” Statistics and computing, vol. 14, no. 3, pp. 199-222, 2004.

[15] R. L. Hardy, “Multiquadric Equations of Topography and Other Irregular Surfaces,”Journal of Geophysical Research Atmospheres, vol. 76, pp. 1905–1915, 1971.

[16] C. Rasmussen, “Gaussian processes in machine learning,” in Advanced Lectures on Machine Learning, Cambridge, MA, USA: MIT Press, 2004.

[17] R. Mallipeddi and M. Lee, "Surrogate model assisted ensemble differential evolution algorithm," 2012 IEEE Congress on Evolutionary Computation, Brisbane, QLD, pp. 1-8, 2012.

[18] X. Y. Sun, D. W. Gong and X. P. Ma, "Directed fuzzy graph-based surrogate model-assisted interactive genetic algorithms with uncertain individual's fitness," 2009 IEEE Congress on Evolutionary Computation, Trondheim, pp. 2395-2402, 2009.

[19] R. G. Regis, “Particle swarm with radial basis function surrogates for expensive black-box optimization,” Journal of Computational Science, vol. 5, no. 1, pp. 12–23, 2014.

[20] F. van den Bergh and A. P. Engelbrecht, "A Cooperative approach to particle swarm optimization," in IEEE Transactions on Evolutionary Computation, vol. 8, no. 3, pp. 225-239, June 2004.

[21] C. Sun, Y. Jin, R. Cheng, J. Ding and J. Zeng, "Surrogate-Assisted Cooperative Swarm Optimization of High-Dimensional Expensive Problems," in IEEE Transactions on Evolutionary Computation, vol. 21, no. 4, pp. 644-660, Aug. 2017.

[22] X. Cai, L. Gao and X. Li, "Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems," in IEEE Transactions on Evolutionary Computation, in press.

[23] B. Liu, Q. Zhang and G. G. E. Gielen, "A Gaussian Process Surrogate Model Assisted Evolutionary Algorithm for Medium Scale Expensive Optimization Problems," in IEEE Transactions on Evolutionary Computation, vol. 18, no. 2, pp. 180-192, April 2014.

[24] A. Kabán, J. Bootkrajang and R. J. Durrant, "Toward Large-Scale Continuous EDA: A Random Matrix Theory Perspective," in Evolutionary Computation, vol. 24, no. 2, pp. 255-291, June 2016.

[25] R. L. Hardy, "Multiquadric equations of topography and other irregular surfaces," Journal of Geophysical Research, vol. 76, no.8, pp. 1905–1915, 1971.

[26] M. Powell, The theory of radial basis function approximation in 1990, Oxford , UK, Oxford University Press pp. 105-2210.

[27] S. Wild and C.A. Shoemaker, "Global convergence of radial basis function trust region derivative-free algorithms," SIAM Review, vol. 21, no.3, pp. 761-781, 2013.

[28] P. Frankl , H. Maehara, "The Johnson-Lindenstrauss lemma and the sphericity of some graphs," Journal of Combinatorial Theory, Series B, vol. 44, no.3, pp. 355–362, 1988.

[29] P.N. Suganthan, N, Hansen, J.J. Liang, K. Deb, Y.-P. Chen, A. Auger and S.Tiwari, "Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization," Tech. Report, Nanyang Technological University, 2013.

[30] K. Price, R. Storn, and J. Lampinen, Differential Evolution: A Practical Approach to Global Optimization. New York, NY, USA. Springer, 2005.

[31] J. Cohen, Statistical Power Analysis for the Behavioral Sciences Lawrence Earlbaum Associates, Hillsdale, NJ, pp. 20–26, 1988.