Swarm and Evolutionary Computation 48 (2019) 288–311

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Surrogate-guided differential evolution algorithm for high dimensional expensive problems

Xiwen Cai, Liang Gao *, Xinyu Li, Haobo Qiu

The State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science & Technology, Wuhan 430074, PR China

ARTICLE INFO

Keywords:
Differential evolution
Surrogate-guided mutation
Surrogate-guided differential evolution optimization
High dimensional expensive problems

ABSTRACT

Engineering optimization problems usually involve computationally expensive simulations and massive design variables. Solving these problems in an efficient manner is still a big challenge. Recently, surrogate-assisted metaheuristic algorithms have been widely studied and are considered to have potential to solve such engineering optimization problems. In this paper, a surrogate-guided differential evolution algorithm is proposed to further improve the optimization efficiency for these problems. Unlike other surrogate-assisted metaheuristic algorithms, it makes a fusion between the differential evolution algorithm and surrogates, which are not just taken as an additional tool to accelerate the convergence of metaheuristic algorithms. Specifically, the proposed algorithm combines the optima predicted by the global and local surrogates with the mutation operator and makes them guide the mutation direction of the differential evolution algorithm, which thus makes the differential evolution algorithm converge fast. A simple surrogate prescreening strategy is also proposed to further improve its optimizing efficiency. In order to validate the proposed algorithm, it is tested by a lot of high dimensional numerical benchmark problems whose dimensions vary from 20 to 200 and is applied to an optimal design of a stepped cantilever beam and an optimal design of bearings in an all-direction propeller. An overall comparison between the proposed algorithm and other optimization algorithms has been made. The results show that the proposed algorithm is promising for optimizing the high dimensional expensive problems especially for the problems whose dimensions are more than 30.

1. Introduction

Metaheuristic algorithms such as the genetic algorithm (GA) [1], particle swarm optimization (PSO) [2], and differential evolution (DE) [3] are popular and widely applied in engineering optimization. Previous studies [4–10] show that these algorithms can handle high dimensional optimization problems well. In the optimizing process, however, the function calls are usually very high. They could reach hundreds of thousands. This can be acceptable because these problems are usually very cheap and the time to make a function call can be negligible. However, if these metaheuristic algorithms are applied to the engineering optimization problems which involve computationally expensive simulations, the computational cost will be tremendous and even prohibitive. According to Simpson et al. [11], Ford Motor Company spends about 36–160 h running one car crash simulation, which means a function call in the numerical optimization problems. One promising approach to reduce computation time for optimizing highly time-consuming problems is to employ computationally cheap approximation models (surrogates) to replace in part the computationally expensive exact function evaluations. However, with the “curse of dimensionality”, the common surrogates such as kriging [12], the radial basis function (RBF) [13], support vector regression (SVR) [14] and polynomial response surface (PRS) [15] usually cannot obtain a high accuracy for problems whose dimensions are more than ten according to the previous studies [16–18]. The optimizing methods [19–21] based on them are thus not suitable for handling high dimensional problems. Besides, the huge metamodeling time of kriging could be another reason to make the kriging-based optimizing methods unsuitable for high dimensional problems [22,23].

Recently, surrogate-assisted metaheuristic algorithms [24,25] have received increasing attention for addressing such high dimensional expensive optimization problems whose dimensions are usually more than 20. These algorithms usually take the metaheuristic algorithms as the main optimizing framework and take the surrogates as an additional tool to accelerate the convergence of the basic metaheuristic algorithms. They usually do not have a high accuracy requirement for surrogates.

* Corresponding author.
E-mail addresses: 864533724@qq.com (X. Cai), gaoliang@mail.hust.edu.cn (L. Gao), lixinyu@mail.hust.edu.cn (X. Li), hobbyqiu@163.com (H. Qiu).

https://doi.org/10.1016/j.swevo.2019.04.009
Received 9 May 2018; Received in revised form 18 September 2018; Accepted 17 April 2019
Available online 10 May 2019

According to the ways of accelerating convergence, the surrogate-assisted metaheuristic algorithms can be mainly classified into three types: metaheuristic algorithms assisted by surrogate prescreening, metaheuristic algorithms assisted by the optimum of the global surrogate and metaheuristic algorithms assisted by surrogate-based local search.

Metaheuristic algorithms assisted by surrogate prescreening are widely used for optimizing expensive high dimensional problems. These algorithms are mainly based on the kriging surrogate (also called as Gaussian model). They usually utilize the kriging-based sampling infilling criteria such as expected improvement (EI) [26,27], probability of improvement (PoI) [28], and lower confidence bound (LCB) [29–32] to prescreen out some promising candidate offspring points produced by the basic metaheuristic algorithms. Among these algorithms, the recently proposed LCB-assisted DE algorithm (called as GPEME) [29] shows high efficiency in optimization for medium-scale expensive problems. Besides, some algorithms are based on predicted response of surrogates. They usually use the surrogate's response function to sort the candidate offspring points produced by basic metaheuristic algorithms and then select the promising points as the offspring points. For example, Fonseca et al. [33] used the surrogate as fitness inheritance to assist a genetic algorithm (GA) in solving optimization problems with a limited computational budget. Regis [34] utilized an RBF surrogate to identify the most promising trial position for each particle in the swarm. Mallipeddi and Lee [35] used surrogates to generate the competitive offspring points among the trial offspring points. Gong et al. [36] used a cheap density function model to select the most promising candidate offspring point among a set of candidate offspring points produced by multiple offspring reproduction operators. Vincenzi and Gambarelli [37] used a scoring function which considers both the surrogate's predicted responses and the distance of the candidate points to select the potential offspring points of the DE algorithm. Moreover, some approaches are based on comparing the candidate point's predicted response provided by surrogates with its parent point's exact response and then deciding whether the candidate points should be evaluated by the exact response function. For example, Praveen and Duvigneau [38] proposed an RBF-assisted PSO algorithm for aerodynamic shape design. It can use the surrogate to screen some more promising particles among the candidate offspring swarm particles in each optimization iteration. A similar algorithm was also proposed by Sun et al. [39]. The difference is that they use an accurate two-layer surrogate construction scheme for improving the optimizing efficiency. Jin et al. [40] used the local ensemble surrogate to generate the competitive trial vectors in the DE optimization process. Elasayad et al. [41] used the kriging surrogate to select the parameters of the DE algorithm suitably in order to accelerate its convergence. What's more, Sun et al. [42] proposed a surrogate-assisted cooperative swarm optimization algorithm (SA-COSO). It uses the global RBF surrogate to screen the best candidate point of the social learning particle swarm optimization. The SA-COSO algorithm shows high optimization efficiency for high dimensional problems whose dimensions can reach 200. It can be learnt from the metaheuristic algorithms assisted by surrogate prescreening that their optimization efficiency could be highly related to that of basic metaheuristic algorithms. This is because the surrogate-prescreening strategy does not change the search mechanism of basic metaheuristic algorithms, but it improves the quality of offspring population points. Surrogates are usually taken as an additional tool to accelerate the convergence of metaheuristic algorithms.

The metaheuristic algorithms assisted by the optimum of the global surrogate usually use the predicted optimum provided by the global surrogate to replace the current best population point if the predicted optimum is better than the current best population point. For example, in the research of Ong et al. [43], the global RBF surrogate can help to accelerate the convergence of the GA to a certain degree if it is accurately constructed. Parno et al. [44] used a Kriging surrogate to improve the efficiency of PSO. Tang et al. [45] used a hybrid global surrogate model consisting of a quadratic polynomial and an RBF model to develop a surrogate-based PSO algorithm. Safari et al. [46] used the optimum of high dimensional model representation to guide PSO to search in a fast way. Tsoukalas et al. [47] used the optimum of an acquisition function, which accounts for the surrogate's response as well as the spread of all population points, to accelerate the convergence of evolutionary annealing simplex algorithm. Because it could be very difficult to find the accurate optimum of a surrogate in high dimensional space, some researchers including Regis [34] and Yu et al. [48] tried to build a local surrogate around the current best point and used the predicted optimum of the surrogate found in the local region to accelerate the convergence of PSO. The metaheuristic algorithms assisted by surrogate-based local search usually use the surrogate-based local search firstly and then use the search mechanism of basic metaheuristic algorithms for global optimization. For example, Ong et al. [43] employed a trust-region method for an interleaved use of exact models for the objective and constrained functions with computationally cheap RBF surrogates during the local search. Then the GA operator was run for global optimization. The local search based on trust region method and surrogates is very common in surrogate-assisted metaheuristic algorithms. They can be read elsewhere [49–55]. The recently proposed GA assisted by surrogate-based trust region local search (GS-SOMA) can be seen previous research [54]. It uses the ensemble of multiple surrogates to obtain a robust and accurate approximation in the optimizing process. Besides, the research also shows that the GA assisted by surrogate-based trust region local search can be developed to solve multi-objective optimization problems. The optimization efficiency of the metaheuristic algorithms assisted by the optima of surrogates could be highly related to the accuracy of the built surrogates. Generally, an accurate surrogate tends to provide useful optimum information and it thus directs the metaheuristic algorithms to search in an efficient way. It should be noted that only a brief summary of the surrogate-assisted metaheuristic algorithms is made in this research. The more comprehensive overview of surrogate-assisted algorithms can be found elsewhere [24,56–58].

Though the current surrogate-assisted metaheuristic algorithms can handle the high dimensional expensive problems well, most of them in the optimization process still need a large number of function evaluations usually higher than thousands in order to obtain a good optimizing result. Besides, they are usually developed for optimizing problems whose dimensions are usually lower than 30. For example, the GS-SOMA algorithm proposed by Lim et al. [54] needs 8000 exact function evaluations for 30-dimensional problems. The surrogate-assisted differential evolution algorithm (ESMDE) proposed by Mallipeddi and Lee [35] needs more than 10,000 function evaluations for 30 dimensional problems. Therefore, how to optimize high dimensional problems especially for the problems whose dimensions are more than 50 in an efficient way is the main motivation of this study. In order to further improve the optimization efficiency for high dimensional expensive problems, an efficient surrogate-guided differential evolution algorithm is proposed. Unlike other surrogate-assisted metaheuristic algorithms, it makes a fusion between surrogates and the differential evolution algorithm. Surrogates are not just taken an additional tool to accelerate the convergence of basic metaheuristic algorithms. Specifically, the optima predicted by the global and local surrogates are combined with the mutation operator, and they guide the mutation direction of the differential evolution algorithm. Besides, a simple surrogate prescreening strategy is employed to further improve the optimizing efficiency. To validate the performance of the proposed algorithm, it is tested by a lot of problems whose dimensions vary from 20 to 200. Results show the proposed algorithm is promising for optimizing high dimensional expensive problems especially for those with dimensions of more than 30.

The remainder of this work is organized as follows. In Section 2, the background theories involved in the proposed algorithm are presented. In Section 3, the proposed surrogate-guided DE algorithm is presented. The experimental results and discussions are presented in Section 4 and Section 5 concludes the paper.

2. Background theory

2.1. Radial basis function

In this study, the RBF surrogate [20] is used in the optimization process of the proposed algorithm. Some studies [16,59,60] show that RBF usually can obtain a more accurate approximation for high dimensional problems compared with other common surrogates including PRS, kriging and SVR. Another merit of RBF is that its metamodeling speed is fast when compared with the kriging or Gaussian model. The RBF surrogate is defined as follows:

Given n distinct points $x_1, x_2, \cdots, x_n \in R^D$ and the function values $f(x_1), f(x_2), \cdots f(x_n)$, the interpolating form of RBF can be expressed as:

$$\widehat{f}(x) = \sum_{i=1}^{n} \lambda_i \Phi_i(x - x_i) + p(x) \tag{1}$$

where $\Phi_i(\cdot)$ is the ith basis function, $\|\cdot\|$ is the Euclidean norm, and λ_i denotes the weight of the ith basis function. The common types of radial basis functions include the Linear ($\psi(r) = r$), Cubic ($\psi(r) = r^3$), Gaussian ($\psi(r) = exp(-\xi r^2)$) and Multiquadric ($\psi(r) = \sqrt{r^2 + \xi^2}$). ξ is a positive constant. $p(x)$ is a linear polynomial function ($b^T x + a$). $r = x - x_i$ is the Euclidean distance between the predicted point x and the existing sample point x_i.

The unknown parameters ($\lambda_1, \lambda_2, \cdots, \lambda_n \in R^D$, $b \in R^D$, $a \in R$) of the radial basis function can be obtained as the solution of the linear equations:

$$\begin{pmatrix} \Phi & P \\ P^T & 0 \end{pmatrix} \begin{pmatrix} \lambda \\ c \end{pmatrix} = \begin{pmatrix} F \\ 0 \end{pmatrix} \tag{2}$$

where Φ is the $n \times n$ matrix with $\Phi_{ij} = \Phi(x_i - x_j)$ and

$$P = \begin{pmatrix} x_1{}^T & 1 \\ x_2{}^T & 1 \\ \vdots & \vdots \\ x_n{}^T & 1 \end{pmatrix}, \lambda = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{pmatrix}, c = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_d \\ a \end{pmatrix}, F = \begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_n) \end{pmatrix} \tag{3}$$

If rank $(P) = D + 1$, the matrix $\begin{pmatrix} \Phi & P \\ P^T & 0 \end{pmatrix}$ is nonsingular and system (2) has a unique solution [61,62]. Thus a unique radial basis function model is obtained.

2.2. JADE algorithm

As a popular and efficient optimizing algorithm, DE is widely studied and applied in engineering practice. The details and development of the DE algorithm can be found elsewhere [25,63]. In this section, the JADE algorithm [64] is introduced. It is an improved version of basic DE algorithm. JADE has three main steps including current-to-pbest mutation, crossover and selection procedures.

Firstly, the initial population of JADE is defined as $\{x_{i,0} = x_{1,i,0}, x_{2,i,0}, \cdots, x_{D,i,0} | i = 1, 2, \cdots, N\}$, which is generated by Latin hypercube sampling method in MATLAB. $x_{j,i,0}$ is constrained in the range of $[x_j^{low}, x_j^{up}]$ for $j = 1, 2, \cdots, D$. D is the dimension of the problem, and N is the population size. After the initialization, the mutation vector of JADE without archive in generation g can be expressed as follows:

$$\boldsymbol{v}_{i,g} = \boldsymbol{x}_{i,g} + F_i \cdot \left(\boldsymbol{x}^p_{best,g} - \boldsymbol{x}_{i,g}\right) + F_i \cdot (\boldsymbol{x}_{r1,g} - \boldsymbol{x}_{r2,g}) \tag{4}$$

where $\boldsymbol{x}^p_{best,g}$ is randomly chosen as one of the top 100 p% individuals in the current population with $p \in (0, 1]$. $r1$ and $r2$ are distinct integers uniformly chosen from the set $\{1, 2, \cdots, NP\} \setminus \{i\}$. At each generation g, the mutation factor F_i of each individual x_i is independently generated according to a Cauchy distribution with location parameter μ_F and scale parameter 0.1.

$$F_i = randc_i(\mu_F, 0.1) \tag{5}$$

where F_i is truncated to be 1 if $F_i \geq 1$ or regenerated if $F_i \leq 0$. Denote S_F as the set of all successful mutation factors in generation g. The location parameter μ_F of the Cauchy distribution is initialized to be 0.5 and then updated at the end of each generation as:

$$\mu_F = (1 - c) \cdot \mu_F + c \cdot mean_L(S_F) \tag{6}$$

where c is a constant number set as 0.1, and $mean_L(\cdot)$ is the Lehmer mean as:

$$mean_L(S_F) = \frac{\sum_{F \in S_F} F^2}{\sum_{F \in S_F} F} \tag{7}$$

Secondly, the crossover operation forms the final trial/offspring vector $\boldsymbol{u}_{i,g} = (u_{1,i,g}, u_{2,i,g}, \cdots, u_{D,i,g})$

$$u_{j,i,g} = \begin{cases} v_{j,i,g}, & if\ rand(0,1) \leq CR_i\ or\ j = j_{rand} \\ x_{j,i,g}, & else \end{cases} \tag{8}$$

where $rand(a, b)$ is a uniform random number on the interval [a, b] and independently generated for each j and each i, and $j_{rand} = randint(1, D)$ is an integer randomly chosen from 1 to D and newly generated for each i. At each generation g, the mutation factor CR_i of each individual x_i is independently generated according to a normal distribution with location parameter μ_{CR} and standard deviation 0.1:

$$CR_i = randn_i(\mu_{CR}, 0.1) \tag{9}$$

where CR_i is truncated to be [0, 1]. Denote S_{CR} as the set of all successful crossover probabilities CR_is at generation g. The mean μ_{CR} is initialized to be 0.5 and updated at the end of each generation as:

$$\mu_{CR} = (1 - c) \cdot \mu_{CR} + c \cdot mean_A(S_{CR}) \tag{10}$$

where c is a constant number set as 0.1, $mean_A(\cdot)$ is the usual arithmetic mean.

Thirdly, the selection operation is to select the better one from the parent vector $\boldsymbol{x}_{i,g}$ and the trial vector $\boldsymbol{u}_{i,g}$ according to their fitness values $f(\cdot)$. Given that the optimizing problem is a minimizing problem, the selected vector is given by:

$$\boldsymbol{x}_{i,g+1} = \begin{cases} \boldsymbol{u}_{i,g}, & if\ f(\boldsymbol{u}_{i,g}) < f(\boldsymbol{x}_{i,g}) \\ \boldsymbol{x}_{i,g}, & else \end{cases} \tag{11}$$

More details of JADE can be found in previous research [64].

3. Surrogate-guided differential evolution optimization

In this section, a generalization of the surrogate-guided differential evolution optimization algorithm for high dimensional expensive problems is presented. The framework is depicted in Fig. 1. The framework illustrated here is based on the JADE algorithm without optional external archive. The proposed algorithm is named as S-JADE. The main differences between S-JADE and JADE are that S-JADE uses the surrogate-guided mutation and surrogate-guided selection mechanisms in the optimizing process. The specific descriptions and explanations about S-JADE optimizing procedures are given in the following sections.

3.1. Surrogate-guided mutation in S-JADE

Different from JADE, the mutation vector of S-JADE in generation g can be expressed as follows:

$$\boldsymbol{v}_{i,g} = \boldsymbol{x}_{i,g} + F_i \cdot \left(\boldsymbol{x}^p_{best,g} - \boldsymbol{x}_{i,g}\right) + F_i \cdot r \cdot \left(\boldsymbol{x}^*_{r1,g} - \boldsymbol{x}_{r2,g}\right) \tag{12}$$

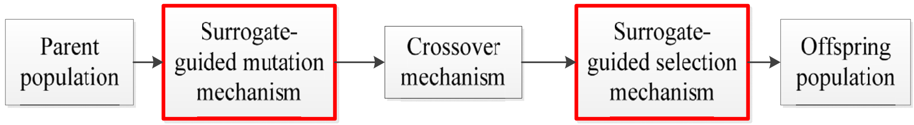


Fig. 1. Optimizing process of S-JADE.

$$x^*_{r1,g} = \begin{cases} X_{Nbest,r1}; & if\ \hat{f}_{L,r1}(X_{Nbest,r1}) < f(x_{r1,g}) \\ x_{r1,g}; & else \end{cases},\ X_{Nbest,r1} = \mathrm{argmin}(\hat{f}_{L,r1}(x)) \quad (13)$$

where the setting of p and F_i is the same as JADE. r is a random number to enhance the converging speed and diversity of population points. It is set in the range of $[0, 1.25]$. $\hat{f}_{L,r1}(x)$ is the predicted response function of the local surrogate built by all the evaluated points in the neighbor region of $x_{r1,g}$. According to the RBF's modeling process in Section 2.1, it can be expressed as: $\hat{f}_{L,r1}(x) = \sum_{i=1}^{n_{local}} \lambda_i \Phi_i(x - x_i) + p(x)$, n_{local} is the number of evaluated points in the neighbor region. To construct a surrogate accurately, the sample size should be not small. Therefore, n_{local} is set larger than 5D in the research. If the value of n_{local} is smaller than 5D, the evaluated points surrounding this local region should be added until its value reaches 5D or reaches the same size of the whole evaluated points in the design space. $X_{Nbest,r1}$ is the predicted minimum of $\hat{f}_{L,r1}(x)$ in the neighbor region (the red square in Fig. 2). The reason why the local surrogate is used is that it could approximate the local region in an accurate way and then provide an accurate local optimum to guide the search of S-JADE.

The mutation principle of S-JADE is shown in Fig. 2. The difference vector $x_{r1,g} - x_{r2,g}$ is a specific characteristic in DE algorithms. It has a great impact on the mutation direction of $x_{i,g}$ in the optimizing process. Compared with JADE, S-JADE changes the difference vector by using the optimum information provided by local surrogates. The difference vector $x_{r1,g} - x_{r2,g}$ becomes $x^*_{r1,g} - x_{r2,g}$. If the response of the predicted optimum $x^*_{r1,g}$ is smaller than that of $x_{r1,g}$, $x_{i,g}$ could drop closer to the global optimum. Therefore, the convergence speed of JADE can be improved. As is shown in Fig. 2, the trial vector $v_{i-S-JADE}$ (red dot) is closer to the global optimum than v_{i-JADE} (black dot) after mutation.

To determine the size of the neighbor region of $x_{i,g}$ in the proposed surrogate-guided mutation strategy, an algorithm, which was proposed by Kitayama et al. [65] to determine the widths of radial basis functions, is introduced to obtain a suitable size of the local region. The radius r_i of the local region around population point $x_{i,g}$ is determined by:

$$r_i = 0.5 \times \frac{d_{i,max}}{\sqrt{D}\sqrt[D]{N-1}}, i = 1, 2, \cdots N. \quad (14)$$

where $d_{i,max}$ denotes the maximum distance between the ith population point and the other population points. D is the dimension of the problem and N is the population size. Then the local search region of $x_{i,g}$ can be defined as: $[x_{i,g} - r_i, x_{i,g} + r_i] \cap [\mathbf{lb}, \mathbf{ub}]$.**lb** and **ub** are respectively the lower bound and the upper bound of the design space. The minimum predicted response point $X_{Nbest,i}$ can be obtained by using the global optimizing algorithm to minimize $\hat{f}_{L,i}(x)$ in the local region. To note it, Eq. (14) is very suitable to enhance the diversity of the transformed population. This is because, according to Kitayama et al. [65], if the population points are uniformly distributed, the local regions belonging to them can be partitioned without being duplicated. As is shown in Fig. 3, the nine points with black dots are produced by the uniform full factorial design in a two dimensional design space. Through partition according to Eq. (14), the local red square regions belonging to the nine points are thus not duplicated. Then the predicted $x^*_{r1,g}$ points of S-JADE in the local regions will be not duplicated. The diversity of population points thus can be kept.

In addition, the global surrogate is also used in S-JADE to improve its optimizing efficiency. The reason why the global surrogate is used is that, through smoothing out local optima, it could provide an accurate global optimum for an optimization problem. One can imagine that, if the global surrogate is constructed very accurately, the predicted global optimum will be the real global optimum of the optimization problem. S-JADE thus finds the global optimum in an accelerating way. For each generation, all evaluated points are used to construct a global surrogate. Then a predicted optimum can be obtained by optimizing the global surrogate. After that, whether the best point $x_{best,g}$ in the current population should be replaced by the predicted optimum is evaluated. The evaluation method is defined as:

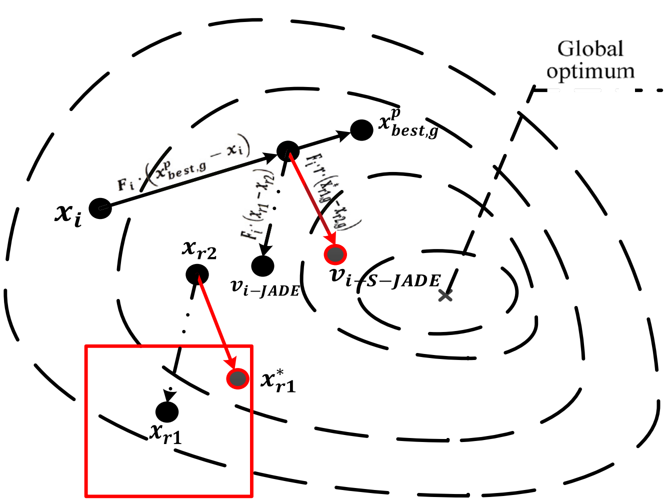


Fig. 2. Illustration of S-JADE's mutation process.

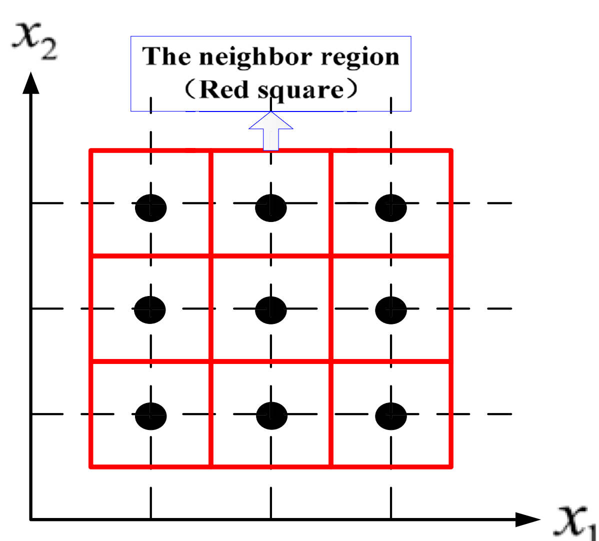


Fig. 3. Distribution of the full factorial design points and their neighbor regions in the two dimensional design space.

Table 1
Characteristics of fifty-four benchmark problems.

Fun.	Name	D	Design space	Global optimum	Characteristic
F1–F5	Ellipsoid [29,30,42]	20, 30, 50, 100, 200	$[-5.12,\ 5.12]^D$	0	Unimodal
F6–F10	Rosenbrock [29,30,42,54]	20, 30, 50, 100, 200	$[-2.048,\ 2.048]^D$	0	Multimodal with narrow valley
F11–F15	Ackley [29,30,42,54]	20, 30, 50, 100, 200	$[-32.768,\ 32.768]^D$	0	Multimodal
F16–F20	Griewank [29,30,42,54]	20, 30, 50, 100, 200	$[-600,\ 600]^D$	0	Multimodal
F21–F24	Shifted Rotated Rastrigin [29,42,54,66]	30, 50, 100, 200	$[-5,\ 5]^D$	−330	Very complicated multimodal
F25	Rotated Hybrid Composition Function with a Narrow Basin for the Global Optimum [29,54,66]	30	$[-5,\ 5]^D$	10	Very complicated multimodal
F26–F28	Rotated hybrid Composition Function [29,66]	50, 100, 200	$[-5,\ 5]^D$	120	Very complicated multimodal
F29	Rastrigin [30]	20	$[-5.12,\ 5.12]^D$	0	Very complicated multimodal
F30–F32	Rotated High Conditioned Elliptic Function [67]	30,50,100	$[-100,\ 100]^D$	100	Unimodal
F33–F35	Shifted and Rotated Weierstrass Function [67]	30,50,100	$[-100,\ 100]^D$	600	Multimodal
F36–F38	Hybrid Function 1 [67]	30,50,100	$[-100,\ 100]^D$	1700	Very complicated multimodal
F39–F40	Shifted Sphere [68]	20,30	$[-20,\ 20]^D$	0	Unimodal
F41–F42	Shifted Ellipsoid [68]	20,30	$[-20,\ 20]^D$	0	Unimodal
F43–F44	Shifted Rotated Ellipsoid [68]	20,30	$[-20,\ 20]^D$	0	Unimodal
F45–F46	Shifted Step [68]	20,30	$[-20,\ 20]^D$	0	Unimodal
F47–F48	Shifted Ackley [68]	20,30	$[-32,\ 32]^D$	0	Typical multimodal
F49–F50	Shifted Griewank [68]	20,30	$[-600,\ 600]^D$	0	Typical multimodal
F51–F52	Shifted Rotated Rosenbrock [68]	20,30	$[-20,\ 20]^D$	0	Very complicated multimodal
F53–F54	Shifted Rotated Rastrigin [68]	20,30	$[-20,\ 20]^D$	0	Very complicated multimodal

$$\boldsymbol{x}_{best,g} = \begin{cases} \boldsymbol{x}_{Gbest}; & if\ d\left(\boldsymbol{x}_{Gbest}, \boldsymbol{x}_{best,g}\right) > \varepsilon,\ and\ f(\boldsymbol{x}_{Gbest}) < f\left(\boldsymbol{x}_{best,g}\right), \\ \boldsymbol{x}_{Gbest} = \operatorname{argmin}\left(\widehat{f}_G(x)\right); & \\ \boldsymbol{x}_{best,g}; & otherwise; \end{cases} \tag{15}$$

where $\widehat{f}_G(x)$ is the predicted response function of the global surrogate built by all the evaluated points $[\boldsymbol{X}_{all}, f(\boldsymbol{X}_{all})]$ in the whole design space. It can be expressed as: $\widehat{f}_G(x) = \sum_{i=1}^{n_{all}} \lambda_i \Phi_i(x - x_i) + p(x)$, n_{all} is the number of all evaluated points in the design space. $\boldsymbol{x}_{Gbest}$ is the minimum predicted response point of $\widehat{f}_G(x)$ in the design space. $d(\cdot)$ is the Euclidean distance. ε is a threshold to make the two points avoid being too close to waste real fitness function evaluations. The default value is set as 0.01 in this study.

3.2. Crossover in S-JADE

The crossover operation of S-JADE is almost the same as that of JADE. The only difference is the setting of the initial value of μ_{CR}. It is suggested to be set as 0.75 in S-JADE. Because the experimental studies show that this setting could be more helpful than the common setting which sets the initial value of μ_{CR} as 0.5 to speed up the convergence of S-JADE. The analysis of the effect of the initial value of μ_{CR} on S-JADE can be found in Section 4.1. The crossover operation of S-JADE forms the final trial/offspring vector as:

$$u_{j,i,g} = \begin{cases} v_{j,i,g}, & if\ rand(0,1) \le CR_i\ or\ j = j_{rand} \\ x_{j,i,g}, & else \end{cases} \tag{16}$$

3.3. Surrogate-guided selection in S-JADE

In S-JADE, the surrogate prescreening strategy is also used to further improve the optimizing efficiency. It firstly uses the global RBF surrogate expressed as $\widehat{f}_G(x) = \sum_{i=1}^{n_{all}} \lambda_i \Phi_i(x - x_i) + p(x)$ to sort the candidate trial vectors $\{\boldsymbol{u}_{i,g} | i = 1, 2, \cdots N\}$ according to their predicted responses $\widehat{f}_G(\boldsymbol{u}_{i,g})$, $i = 1, 2 \cdots N$. Then several trial vectors $\{\boldsymbol{u}_{o,g} | o = 1, 2, \cdots L\}$ with the smallest predicted response values are selected for exact fitness function evaluations. L is the number of the selected trial vectors. $\boldsymbol{u}_{o,g}$ corresponds to the candidate trial vector $\boldsymbol{u}_{io,g}$, which is a vector among the candidate trial vectors $\{\boldsymbol{u}_{i,g} | i = 1, 2, \cdots N\}$. Like JADE, S-JADE uses the selection operation to select the better one from the parent vector $\boldsymbol{x}_{io,g}$ and the trial vector $\boldsymbol{u}_{o,g}$ according to their fitness values $f(\cdot)$. The selected vector is given by:

$$\boldsymbol{x}_{io,g+1} = \begin{cases} \boldsymbol{u}_{o,g}, & if\ f\left(\boldsymbol{u}_{o,g}\right) < f\left(\boldsymbol{x}_{io,g}\right) \\ \boldsymbol{x}_{io,g}, & else \end{cases} \tag{17}$$

The reason why the surrogate-prescreening strategy is useful to accelerate the convergence of S-JADE can be attributed to that, the selected candidate offspring vectors with small predicted responses are considered as more promising to be the global optimum than those with large predicted responses and the time costs for evaluating them by the real function are thus saved.

3.4. The pseudo code of S-JADE

Based on the above description, the pseudo code of S-JADE is given as follows:

Pseudo code of S-JADE:
//Initialization
1. g = 0;
2. Design the initial population P: $\{\boldsymbol{x}_{i,0} = x_{1,i,0}, x_{2,i,0}, \cdots, x_{D,i,0} | i = 1, 2, \cdots, N\}$ by using the Latin hypercube sampling method and evaluate them with exact function $f(\cdot)$, NFE=NFE + N;
3. Set the parameters μ_F and μ_{CR} as 0.5 and 0.75 respectively;
4. Archives $S_F = \varnothing$ and $S_{CR} = \varnothing$;
//Main loop
5. **While** *termination criterion* (NFE $< N_{max}$ *or* g $< G_{max}$) *are not met* **do**
//Construct the global RBF surrogate

(*continued on next page*)

(*continued*)

6. Construct the global surrogate $\widehat{f}_G(x)$ by using RBF model and all evaluated points in the whole design space
7. Obtain the predicted global optimum point x_{Gbest} by using JADE algorithm to optimize $\widehat{f}_G(x)$ in the whole design space and use the exact function evaluation $f(x_{Gbest})$, NFE= NFE+1;
8. Use the criterion of Eq. (15) to update the current best population point and obtain the improved best population point $x_{best,g}$;

//Construct the local RBF surrogates

9. For $i = 1$ to N **do**

10. Based on the region partition strategy of Eq. (14), the local region around $x_{i,g}$ is defined as $[x_{i,g} - r_i, x_{i,g} + r_i] \cap [\mathbf{lb}, \mathbf{ub}]$;
11. Obtain all evaluated points in this local region; If the size of them is smaller than 5D, the surrounding evaluated points should be added until their size reaches the 5D requirement or reaches the same size of the whole evaluated points in the design space. The points in the local region are expressed as $[X_{local}, f(X_{local})]$;
12. Construct the local surrogate $\widehat{f}_{L,i}(x)$ by using RBF model and $[X_{local}, f(X_{local})]$
13. Use the JADE algorithm to optimize $\widehat{f}_{L,i}(x)$ in the local region; Obtain the predicted minimum point $X_{Nbest,i}$;

14. End For

//Use the S-JADE optimizing procedures: Mutation, Crossover and Selection

15. For i = 1 to N **do**

//Mutation

16. Generate $F_i = randc_i(\mu_F, 0.1)$ and $CR_i = randn_i(\mu_{CR}, 0.1)$;
17. Randomly choose $x^p_{best,g}$ from the top 100 p% individuals in the current population;
18. Randomly choose $x_{r1,g} \neq x_{r2,g} \neq x_{i,g}$ from the current population P;

19. If $\widehat{f}_{L,r1}(X_{Nbest,r1}) < f(x_{r1,g})$

20. $x^*_{r1,g} = X_{Nbest,r1}$;

21. Else

22. $x^*_{r1,g} = x_{r1,g}$;

23. End If

(*continued on next column*)

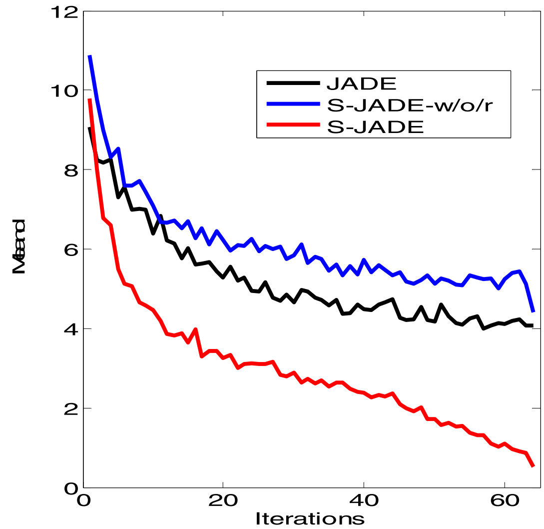


Fig. 5. Change of mean distance between the parent and offspring points.

(*continued*)

24. $v_{i,g} = x_{i,g} + F_i \cdot (x^p_{best,g} - x_{i,g}) + F_i \cdot r \cdot (x^*_{r1,g} - x_{r2,g})$;

//Crossover

25. For i = 1 to D **do**

26. Generate $j_{rand} = randint(1, D)$;

27. If $rand(0,1) \le CR_i$ *or* $j = j_{rand}$

28. $u_{j,i,g} = v_{j,i,g}$;

29. Else

(*continued on next page*)

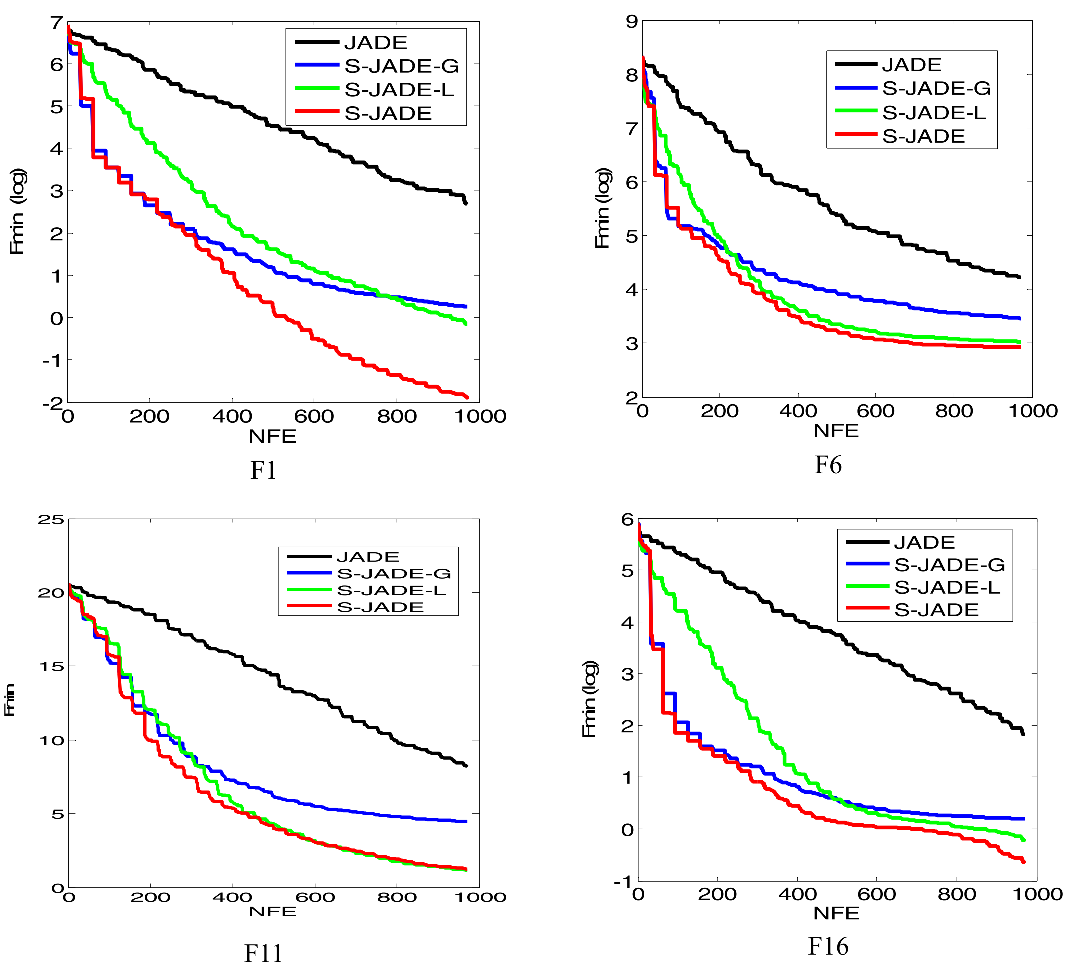


Fig. 4. Convergence curves of JADE, S-JADE-G, S-JADE-L and S-JADE for different functions.

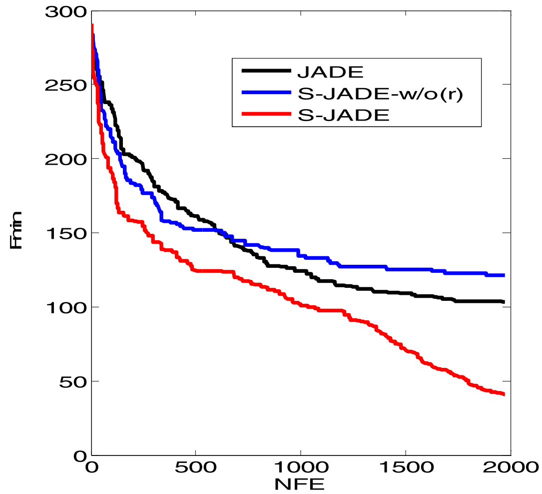


Fig. 6. Convergence curves of S-JADE-w/o/r, S-JADE and JADE for F27.

(*continued*)

30. $u_{j,i,g} = x_{j,i,g}$;
31. End If
32. End For
33. End For
//Selection

(*continued on next column*)

(*continued*)

34. Sort $\hat{f}_G(u_{i,g}), i = 1, 2, \cdots N$ in an ascending order and select out L points $\{u_{o,g} | o = 1, 2, \cdots L\}$ from $\{u_{i,g} | i = 1, 2, \cdots N\}$ with the smallest $\hat{f}_G(\cdot)$ values.
35. For $o = 1$ to L **do**
36. Evaluate $u_{o,g}$ using the exact function $f(\cdot)$, NFE=NFE+1;
37. End For
38. For $i = 1$ to N **do**
39. $x_{i,g+1} = x_{i,g}$;
40. End For
41. For $o = 1$ to L **do**
42. If $f(u_{o,g}) < f(x_{io,g})$
//$u_{o,g} = u_{io,g}$, $u_{io,g}$ is a vector among $\{u_{i,g} | i = 1, 2, \cdots N\}$, $x_{io,g}$ is the parent vector of $u_{io,g}$//
43. $x_{io,g+1} = u_{o,g}$;
44. Else
45. $x_{io,g+1} = x_{io,g}$; $F_{io} \to S_F$, $CR_{io} \to S_{CR}$.
46. End If
47. End For
48. $\mu_F = (1 - c) \cdot \mu_F + c \cdot mean_L(S_F)$;
49. $\mu_{CR} = (1 - c) \cdot \mu_{CR} + c \cdot mean_A(S_{CR})$;
50. $g = g+1$;
51. End While

4. Experimental study and discussion

In order to evaluate the performance of the proposed algorithm, several widely used unimodal and multimodal benchmark problems are adopted. The dimension (D) of these problems varies from 20 to 200. Their characteristics are listed in Table 1. They are from "CEC 2005''

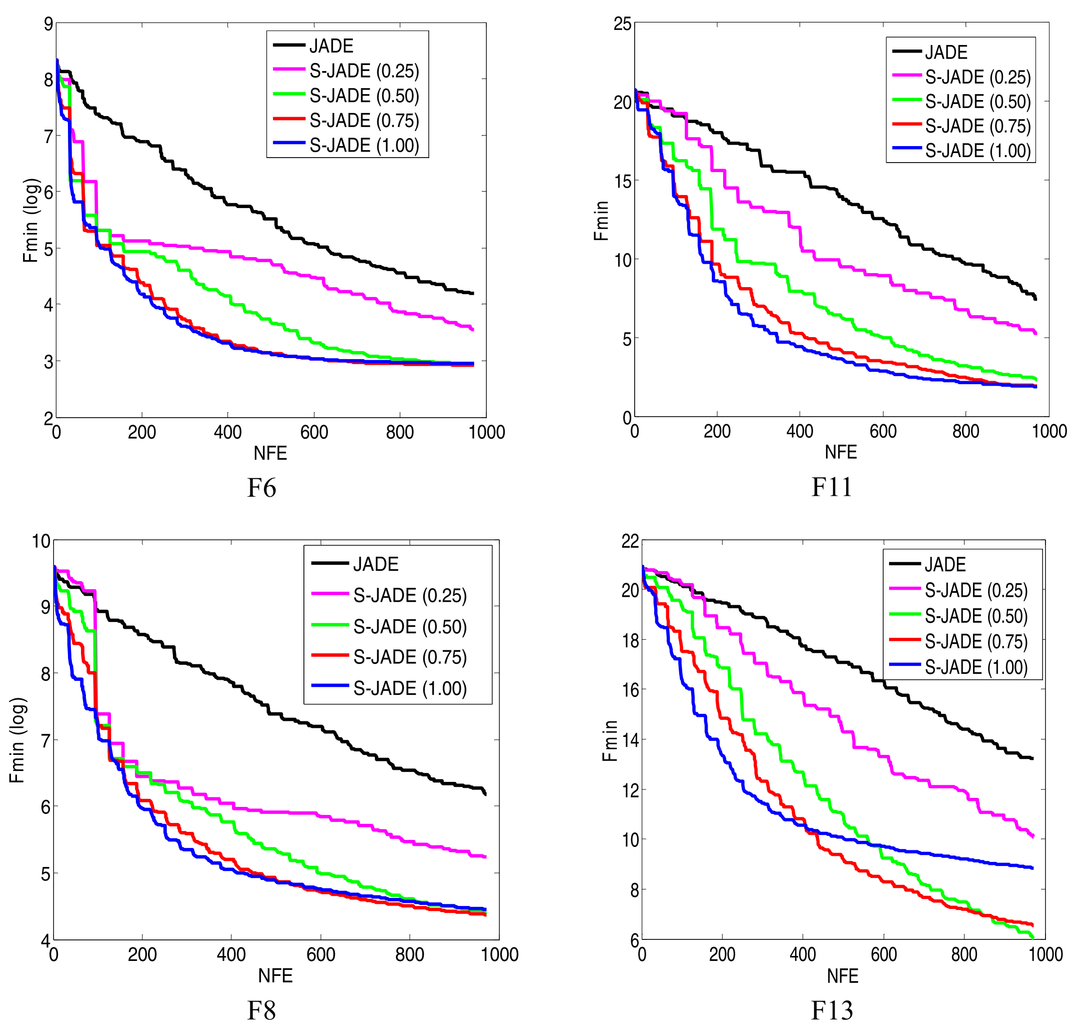


Fig. 7. Convergence curves of S-JADE with different μ_{CR} for F6, F8, F11 and F13.

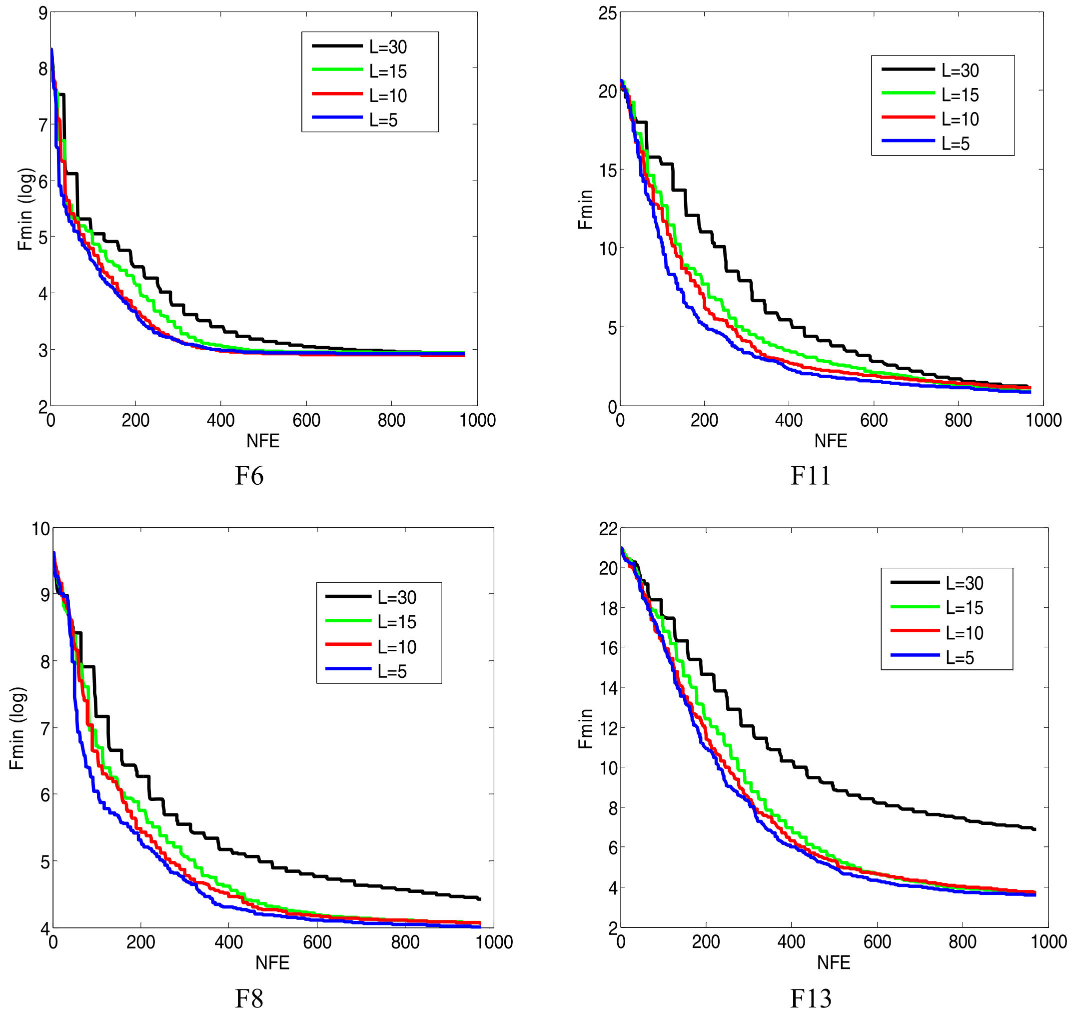


Fig. 8. Convergence curves of S-JADE with different L values for F6, F8, F11 and F13.

[66], part A of "CEC 2014'' [67], part B of "CEC 2014'' [68] and the research papers of Lim et al. [54], Liu et al. [29], Liu et al. [30], and Sun et al. [42].

4.1. Parameter setting and discussion in S-JADE

The proposed S-JADE has some special parameters which are different from the basic JADE algorithm. The setting of these parameters is discussed by experimental studies. In the experiments, the initial population size of S-JADE and JADE is set as $N = 30$. Different optimizing algorithms perform 25 independent runs on each test problem in MATLAB 2011a in order to obtain a robust optimizing result.

In terms of the mutation operator of Eq. (12), two parts are different from JADE. One is that S-JADE uses the predicted optima obtained by the global and local surrogates to replace the current population points. To show the effects of the global and local surrogates on S-JADE, the four common functions with different characteristics are tested. They are the 20-D unimodal Ellipsoid (F1), multimodal Rosenbrock (F6), multimodal Ackley (F11) and multimodal Griewank (F16) functions. Compared with JADE, the S-JADE which only uses the global surrogate or local surrogates is tested. The S-JADE which only uses the global surrogate is named as S-JADE-G and the S-JADE which only use the local surrogates is named as S-JADE-L. The other parameters of S-JADE-G and S-JADE-L are set as the same as those in S-JADE. The convergence curves of JADE, S-JADE-G, S-JADE-L and S-JADE are plotted in Fig. 4. The global and local surrogates all help to speed up the convergence speed of JADE. The reason why the surrogates can improve the optimization efficiency of JADE is that the surrogates could be built well. The predicted optima of the surrogates then lead JADE to search in a relatively accurate way. In the initial stage of optimization, the global surrogate plays a more important role than the local surrogates. In the later optimization, the local surrogates play a more important role than the global surrogate. The S-JADE which combines both the global and local surrogates always performs robustly and efficiently. Therefore, the way of combining surrogates in S-JADE is reasonable and efficient.

Another different part in the mutation operator of Eq. (12) is that S-JADE uses a parameter $r \in [0, 1.25]$. The effects of r can be explained in two aspects. Firstly, it can enrich the diversity of the candidate offspring points. As one can imagine, for the S-JADE without r (S-JADE-w/o/r), the neighbor point X_{Nbest} could not change even though the corresponding population point changes successively in a few iterations, thus possibly lowering the diversity of offspring population points and influencing the global search ability of S-JADE-w/o/r. Secondly, r could speed up the convergence of S-JADE, because a small value of r could make S-JADE bias to local search. The fast convergence property of r could be useful for optimizing the high dimensional expensive problems in engineering practice because designers might really care about the huge computational cost involved in the optimizing process and they usually only need an approximate global optimal solution. Then, the 20-D Rastring function (F27) is tested for an exemplification. The corresponding results are shown in Figs. 5 and 6. The vertical axis in Fig. 5 means the mean distance between the parent and offspring points (*Meand*). Fig. 6 shows the convergence curves of S-JADE, S-JADE-w/o/r and JADE algorithms. It can be seen that the S-JADE algorithm can converge to a better optimum

 Swarm and Evolutionary Computation 48 (2019) 288–311

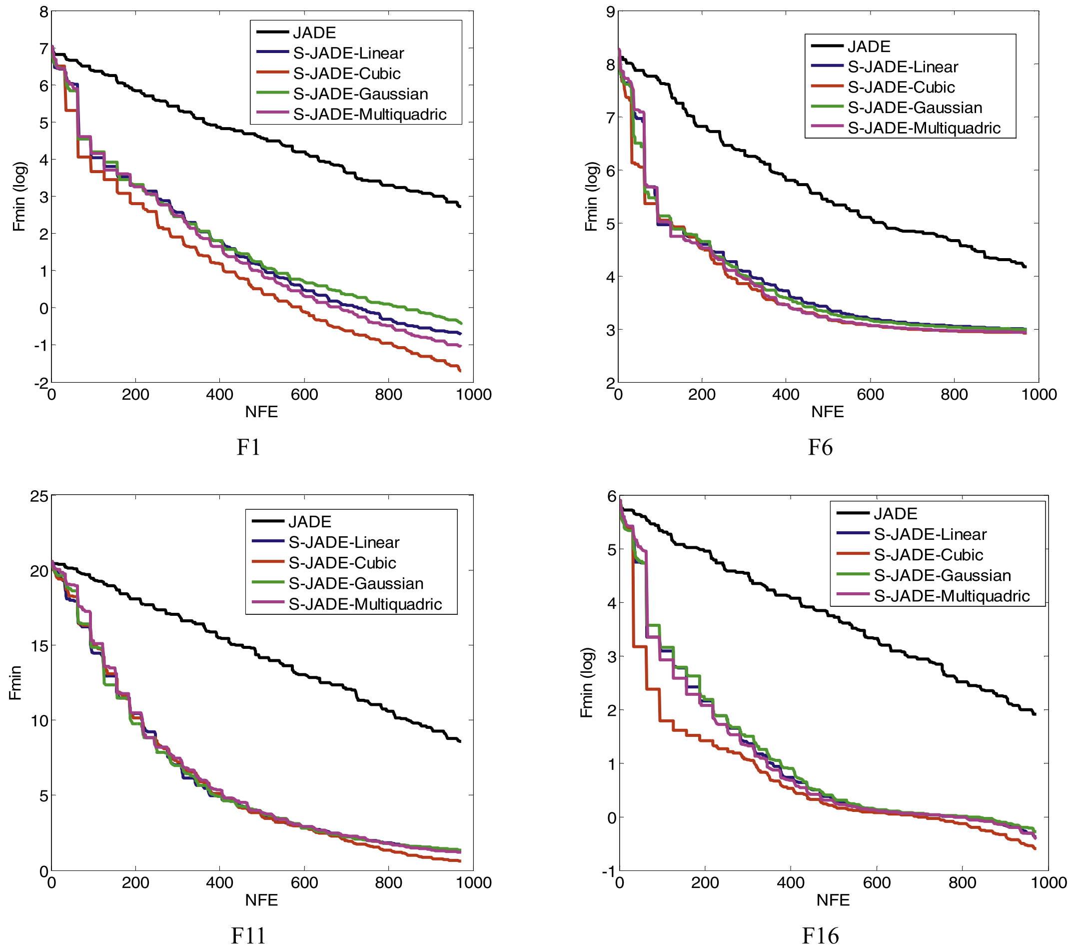


Fig. 9. Convergence curves of S-JADE with different types of RBF for F1, F6, F11 and F16.

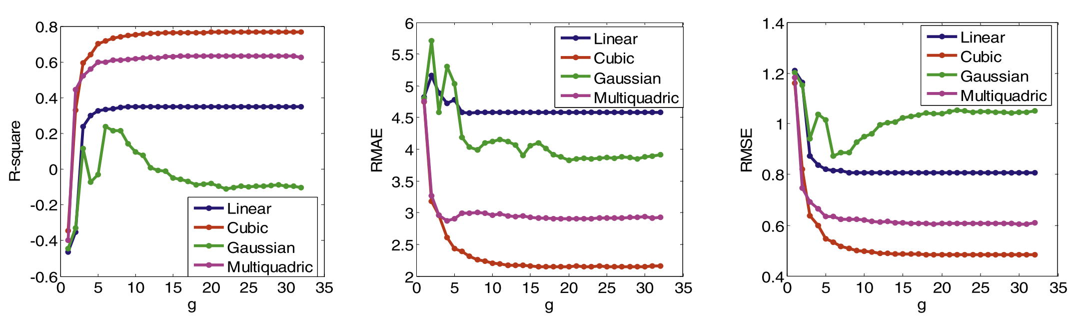


Fig. 10. Accuracy of RBF models with different types of basis function for F1.

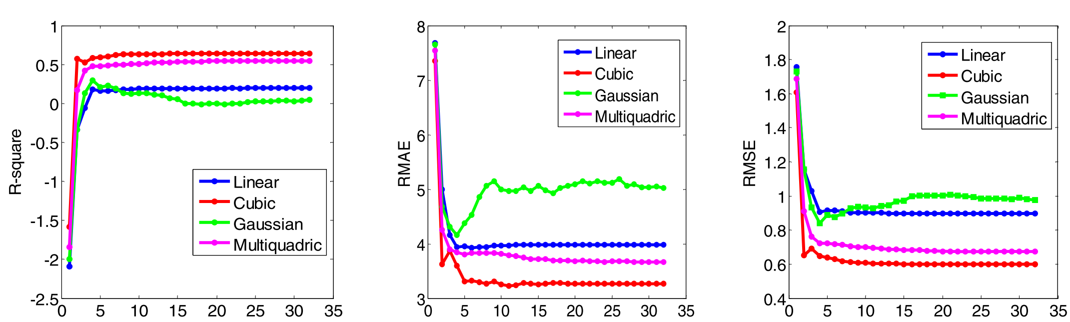


Fig. 11. Accuracy of RBF models with different types of basis function for F6.

with a fast speed when compared with S-JADE-w/o/r and JADE. The S-JADE-w/o/r even performs worse than the basic JADE algorithm. According to Fig. 5, *Meand* of S-JADE drops the fastest. Therefore, the local search effect of parameter *r* is validated.

In the crossover operation of S-JADE, the initial mean crossover rate μ_{CR} is set as 0.75. This is different from JADE whose initial μ_{CR} is set as 0.5. To evaluate the effect of μ_{CR} on S-JADE, the 20-D Rosenbrock (F6), 20-D Ackley (F11), 50-D Rosenbrock (F8) and 50-D Ackley (F13) functions are tested by using S-JADE with different initial μ_{CR} values. The corresponding convergence curves are plotted in Fig. 7. The converging speed of S-JADE gradually increases with the increase of μ_{CR} for F6, F11 and F8. When μ_{CR} is larger than 0.75, the change of the converging speed of S-JADE is small. However, according to the convergence curves for F13, S-JADE with $\mu_{CR} = 1$ performs worse than the S-JADE algorithms with $\mu_{CR} = 0.75$ and $\mu_{CR} = 0.5$. This shows that the value of μ_{CR} should not be set too large. The reason could be attributed to that too large a value of μ_{CR} will lower the crossover between the parent and candidate offspring points. This will lower the diversity of population points and thus weaken the global search ability of S-JADE. Therefore, in order to obtain a good global optimum with a fast speed, μ_{CR} is suggested to be set as 0.75 in S-JADE.

Besides, in the surrogate-guided selection process of S-JADE, a parameter L is added to represent the number of selected offspring points compared with JADE. The effects of L on S-JADE are shown by testing the 20-D Rosenbrock (F6), 20-D Ackley (F11), 50-D Rosenbrock (F8) and 50-D Ackley (F13) functions. The S-JADE algorithms with $L = 5,\ 10,\ 15$ and 30 are tested respectively. The corresponding convergence curves are plotted in Fig. 8. It can be seen that the convergence speed of S-JADE increases with the decrease of L. However, when L is smaller than 10, the convergence speed of S-JADE is slightly improved. For F6 and F13, the convergence speeds of S-JADE almost have no difference. Within the same converging condition, a smaller value of L also means that the optimizing process of S-JADE is longer. In this study, $L = 10$ is suggested to be used in S-JADE in order to obtain a good optimal result and make the experimental study convenient.

Moreover, the proposed S-JADE algorithm could not rely heavily on the accuracy of surrogates. This is because it only requires surrogates to provide an approximate dropping tendency for the parent point $x_{i,g}$ in the mutation process. To validate this viewpoint, different types of RBF surrogate are used to test the performance of S-JADE for 20-D Ellipsoid, Rosenbrock, Ackley and Griewank functions. For the tested S-JADE, the parameter L is set as 30. Four common types of RBF including Linear, Cubic, Gaussian and Multiquadric are used. The convergence curves for S-JADE with different types of RBF are plotted in Fig. 9. It can be seen that the performance of S-JADE is not sensitive to the use of different types of RBF. However, the Cubic RBF seems to be more effective in speeding up the convergence of S-JADE when compared with the other types of RBF. Also, for F1 and F6, the accuracy changes of different types of global RBF surrogate in optimization iterations of S-JADE are given in Figs. 10 and 11, where R-square, RMAE and RMSE [18,69] are the metrics to measure the accuracy of surrogates. It can be seen that different RBF models perform differently in the optimization process. Therefore, the performance of S-JADE could not be highly related to the accuracy of surrogates. Once the used surrogate can provide certain accuracy, the efficiency of S-JADE could be enhanced. In this research, the Cubic RBF is suggested to be used in S-JADE.

What's more, S-JADE actually provides a general way of combining surrogates with DE algorithms. The proposed innovation in S-JADE could be applicable to other DE optimizing frameworks. Take the commonly used DE/rand/1 and DE/best/1 in Ref. [64] for an example. Their mutation strategies are expressed as:

$$\begin{aligned} &\text{DE/rand/1:}\ \ \boldsymbol{v}_{i,g} = \boldsymbol{x}_{r0,g} + F_i \cdot (\boldsymbol{x}_{r1,g} - \boldsymbol{x}_{r2,g}) \\ &\text{DE/best/1:}\ \ \boldsymbol{v}_{i,g} = \boldsymbol{x}_{best,g} + F_i \cdot (\boldsymbol{x}_{r1,g} - \boldsymbol{x}_{r2,g}) \end{aligned}$$

Through integration with such surrogates as S-JADE, they can be expressed as:

$$\begin{aligned} &\text{S- DE/rand/1:}\ \ \boldsymbol{v}_{i,g} = \boldsymbol{x}_{r0,g} + F_i \cdot \boldsymbol{r} \cdot (\boldsymbol{x}^*_{r1,g} - \boldsymbol{x}_{r2,g}) \\ &\text{S- DE/best/1:}\ \ \boldsymbol{v}_{i,g} = \boldsymbol{x}^*_{best,g} + F_i \cdot \boldsymbol{r} \cdot (\boldsymbol{x}^*_{r1,g} - \boldsymbol{x}_{r2,g}) \end{aligned}$$

The other procedures in S- DE/rand/1 and S- DE/best/1 including crossover and selection are the same as basic DE/rand/1 and DE/best/1 algorithms. To validate their performance and make a comparison between them and S-JADE, several functions including the 30-D Ellipsoid (F2), Rosenbrock (F7) and Ackley (F12) are tested. Their parameters are set as follows: F and CR are set as 0.5 and 0.9 respectively in DE/rand/1 and S-DE/rand/1 according to Ref. [64]. F and CR are set as 0.8 and 0.8 respectively in DE/best/1 and S-DE/best/1 according to Ref. [29]. In S-JADE, the parameter L is set as 30. The convergence curves of different algorithms are plotted in Fig. 12. It can be seen that the surrogate enhanced DE algorithms including S-JADE, S-DE/rand/1 and S-DE/best/1 always perform better than the basic DE algorithms including JADE, DE/rand/1 and DE/best/1. This shows that the proposed innovation could also be applicable to other DE optimizing frameworks. Besides, S-JADE can perform better than S- DE/rand/1 and S- DE/best/1 for these tested functions. One reason for the performance superiority of S-JADE over S- DE/rand/1 and S- DE/best/1 can be attributed to that it can adaptively update the mutation and crossover parameters in the optimizing process. To sum up, JADE is a good choice for the proposed optimization framework.

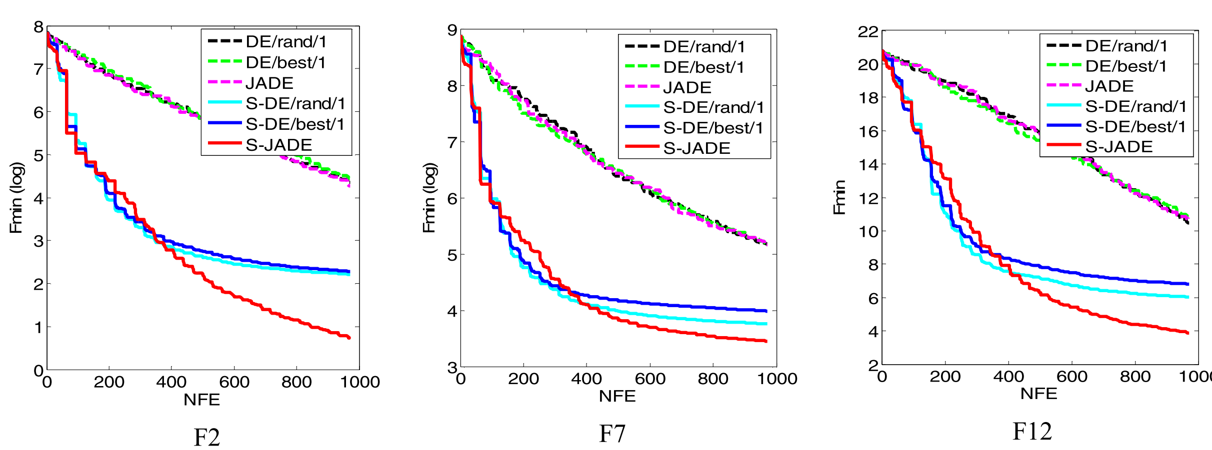


Fig. 12. Convergence curves of different optimizing algorithms for F2, F7 and F12.

Table 2
Optimizing results tested by basic DE variants and S-JADE.

Fun	D	JADE (2009)				CoBiDE (2014)				MPEME (2016)				S-JADE			
		Best	Mean	Worst	Std.	Best	Mean	Worst	Std.	Best	Mean	Worst	Std.	Best	Mean	Worst	Std.
F1	20	1.10E+01	1.68E+01	2.50E+01	4.02E+00	6.99E+01	1.76E+02	3.01E+02	5.51E+01	1.08E+01	2.10E+01	4.14E+01	6.82E+00	1.25E-02	**5.52 E-02** (+++)	1.08E-01	3.43E-02
F2	30	4.27E+01	7.45E+01	1.20E+02	2.24E+01	2.56E+02	6.00E+02	8.73E+02	1.57E+02	6.03E+01	1.14E+02	1.79E+02	3.22E+01	9.27E-01	**2.77E+00** (+++)	5.11E+00	1.36E+00
F3	50	3.28E+02	4.34E+02	5.47E+02	6.52E+01	1.49E+03	2.25E+03	3.03E+03	4.20E+02	3.74E+02	5.56E+02	9.88E+02	1.28E+02	9.93E+00	**1.73E+01** (+++)	2.90E+01	5.08E+00
F4	100	2.80E+03	3.27E+03	3.96E+03	3.46E+02	5.28E+03	1.19E+04	1.76E+04	2.87E+03	2.31E+03	3.80E+03	5.20E+03	7.10E+02	1.02E+02	**1.50E+02** (+++)	3.15E+02	6.07E+01
F5	200	1.52E+04	2.02E+04	2.40E+04	2.77E+03	3.39E+04	4.67E+04	6.32E+04	7.75E+03	1.66E+04	2.13E+04	2.58E+04	2.66E+03	1.41E+03	**1.90E+03** (+++)	2.33E+03	3.00E+02
F6	20	3.68E+01	5.98E+01	8.18E+01	1.48E+01	2.37E+02	4.60E+02	7.65E+02	1.44E+02	4.46E+01	7.72E+01	1.76E+02	2.75E+01	1.75E+01	**1.84E+01** (+++)	1.96E+01	6.30E-01
F7	30	1.03E+02	1.54E+02	1.94E+02	2.85E+01	5.26E+02	1.10E+03	1.94E+03	4.26E+02	1.29E+02	1.97E+02	2.86E+02	4.45E+01	2.87E+01	**2.96 + 01** (+++)	3.05E+01	5.43E-01
F8	50	3.23E+02	4.51E+02	6.55E+02	1.09E+02	1.02E+03	2.81E+03	5.17E+03	8.56E+02	3.75E+02	5.73E+02	9.42E+02	1.40E+02	5.40E+01	**5.67E+01** (+++)	6.04E+01	2.34E+00
F9	100	1.10E+03	1.59E+03	2.14E+03	3.14E+02	3.61E+03	7.29E+03	1.06E+04	2.16E+03	1.29E+03	1.96E+03	3.06E+03	4.18E+02	1.29E+02	**1.40E+02** (+++)	1.81E+02	1.51E+01
F10	200	3.92E+03	4.98E+03	6.10E+03	8.42E+02	7.76E+03	1.59E+04	2.50E+04	5.18E+03	3.70E+03	5.19E+03	6.78E+03	8.01E+02	3.22E+02	**3.84E+02** (+++)	4.51E+02	4.17E+01
F11	20	5.91E+00	7.73E+00	9.11E+00	1.04E+00	1.32E+01	1.65E+01	1.80E+01	1.12E+00	7.98E+00	9.48E+00	1.15E+01	9.52E-01	3.19E-01	**1.03E+00** (+++)	1.70E+00	4.88E-01
F12	30	9.52E+00	1.09E+01	1.30E+01	1.04E+00	1.55E+01	1.78E+01	1.90E+01	9.09E-01	9.53E+00	1.18E+01	1.45E+01	1.07E+00	1.80E+00	**2.34E+00** (+++)	3.24E+00	4.94E-01
F13	50	1.17E+01	1.32E+01	1.40E+01	6.70E-01	1.54E+01	1.81E+01	1.95E+01	1.02E+00	1.18E+01	1.39E+01	1.58E+01	1.02E+00	3.21E+00	**3.78E+00** (+++)	4.22E+00	3.41E-01
F14	100	1.47E+01	1.52E+01	1.58E+01	3.21E-01	1.60E+01	1.86E+01	2.03E+01	9.52E-01	1.44E+01	1.57E+01	1.69E+01	6.31E-01	4.79E+00	**5.29E+00** (+++)	5.73E+00	2.70E-01
F15	200	1.56E+01	1.63E+01	1.73E+01	5.97E-01	1.81E+01	1.91E+01	2.03E+01	5.41E-01	1.50E+01	1.63E+01	1.75E+01	6.29E-01	6.73E+00	**8.42E+00** (+++)	1.16E+01	1.78E+00
F16	20	4.53E+00	7.77E+00	1.24E+01	2.67E+00	3.94E+01	7.41E+01	1.21E+02	2.00E+01	3.77E+00	9.45E+00	1.64E+01	3.43E+00	4.32E-01	**7.94E-01** (+++)	1.02E+00	1.66E-01
F17	30	1.64E+01	2.11E+01	2.63E+01	2.97E+00	8.74E+01	1.49E+02	2.24E+02	3.87E+01	1.81E+01	2.92E+01	4.49E+01	7.22E+00	1.06E+00	**1.20E+00** (+++)	1.38E+00	1.01E-01
F18	50	5.61E+01	7.38E+01	8.72E+01	9.95E+00	2.06E+02	2.94E+02	4.47E+02	6.77E+01	5.23E+01	9.42E+01	1.46E+02	2.48E+01	1.96E+00	**2.27E+00** (+++)	2.78E+00	3.14E-01
F19	100	2.31E+02	2.65E+02	3.01E+02	2.47E+01	4.32E+02	7.53E+02	1.04E+03	1.35E+02	2.06E+02	3.13E+02	4.07E+02	5.42E+01	5.72E+00	**8.33E+00** (+++)	1.04E+01	1.38E+00
F20	200	5.73E+02	7.48E+02	8.97E+02	9.20E+01	8.55E+02	1.60E+03	2.50E+03	4.23E+02	6.62E+02	7.95E+02	1.29E+03	1.33E+02	2.84E+01	**4.72E+01** (+++)	7.63E+01	1.75E+01
F21	30	−4.51E+01	1.80E+00	3.46E+01	2.60E+01	6.49E+01	1.46E+02	2.18E+02	3.78E+01	−6.97E+01	−1.11E+01	8.17E+01	3.49E+01	−1.75E+02	**−8.67E+01** (+++)	−2.03E+01	4.31E+01
F22	50	3.02E+02	3.56E+02	4.28E+02	4.01E+01	5.23E+02	6.71E+02	8.25E+02	7.74E+01	2.50E+02	3.57E+02	4.91E+02	6.04E+01	1.22E+02	**2.25E+02** (+++)	2.93E+02	4.79E+01
F23	100	1.42E+03	1.65E+03	1.97 E+03	1.64E+02	1.94E+03	2.15E+03	2.60E+03	1.63E+02	1.43E+03	1.69E+03	1.86E+03	1.18E+02	8.48E+02	**9.21E+02** (+++)	1.03 E+03	6.98E+01
F24	200	5.51E+03	5.76E+03	6.03E+03	1.74E+02	5.38E+03	6.03E+03	6.75E+03	3.27E+02	4.84E+03	5.28E+03	5.60E+03	1.88E+02	4.61E+03	**4.90E+03** (+++)	5.11E+03	1.65E+02
F25	30	1.00E+03	1.03E+03	1.06 E+03	1.99E+01	1.09E+03	1.18E+03	1.30E+03	4.66E+01	9.61E+02	9.96E+02	1.10E+03	2.95E+01	9.24E+02	**9.29E+02** (+++)	9.36E+02	4.31E+00
F26	50	5.52E+02	5.99E+02	6.91E+02	4.73E+01	7.57E+02	8.67E+02	1.04E+03	7.11E+01	5.25E+02	6.11E+02	7.15E+02	5.18E+01	4.69E+02	**5.53E+02** (+++)	6.61E+02	7.58E+01
F27	100	6.46E+02	7.08E+02	8.71E+02	7.29E+01	8.62E+02	1.05E+03	1.17E+03	8.12E+01	6.53E+02	7.32E+02	9.10E+02	5.85E+01	4.65E+02	**5.48E+02** (+++)	6.09E+02	4.09E+01
F28	200	1.66E+03	1.72E+03	1.77E+03	3.72E+01	1.63E+03	1.74E+03	1.83E+03	5.05E+01	1.62E+03	1.70E+03	1.75E+03	3.04E+01	1.59E+03		1.69E+03	2.94E+01

(*continued on next page*)

Table 2 (*continued*)

Fun	D	JADE (2009)				CoBiDE (2014)				MPEME (2016)				S-JADE			
		Best	Mean	Worst	Std.	Best	Mean	Worst	Std.	Best	Mean	Worst	Std.	Best	Mean	Worst	Std.
															1.65E+03 (+++)		
F29	20	9.92E+01	1.19E+02	1.32E+02	1.27E+01	1.47E+02	1.71E+02	1.93E+02	1.19E+01	7.28E+01	1.26E+02	1.57E+02	1.86E+01	2.28E+01	**4.88E+01** (+++)	6.69E+01	1.48E+01
F30	30	2.89E+08	4.12E+08	5.60E+08	1.09E+08	4.39E+08	6.63E+08	1.03E+09	1.52E+08	1.39E+08	2.98E+08	5.32E+08	1.03E+08	8.04E+07	**1.51E+08** (+++)	3.41E+08	7.60E+07
F31	50	3.77E+08	7.72E+08	1.14E+09	2.58E+08	1.14E+09	2.02E+09	2.91E+09	4.39E+08	3.39E+08	6.91E+08	1.06E+09	1.90E+08	1.55E+08	**3.03E+08** (+++)	4.47E+08	9.44E+07
F32	100	2.54E+09	3.39E+09	5.05E+09	8.76E+08	3.95E+09	6.55E+09	9.39E+09	1.23E+09	1.62E+09	2.60E+09	3.66E+09	5.71E+08	9.51E+08	**1.17E+09** (+++)	1.49E+09	1.80E+08
F33	30	6.39E+02	6.40E+02	6.44E+02	1.70E+00	6.39E+02	6.41E+02	6.44E+02	1.24E+00	6.28E+02	6.37E+02	6.43E+02	3.07E+00	6.19E+02	**6.24E+02** (+++)	6.30E+02	3.07E+00
F34	50	6.69E+02	6.73E+02	6.78E+02	2.75E+00	6.71E+02	6.75E+02	6.78E+02	1.87E+00	6.60E+02	6.69E+02	6.75E+02	3.68E+00	6.43E+02	**6.50E+02** (+++)	6.60E+02	5.55E+00
F35	100	7.56E+02	7.62E+02	7.67E+02	3.38E+00	7.56E+02	7.64E+02	7.69E+02	3.11E+00	7.46E+02	7.57E+02	7.65E+02	5.10E+00	7.17E+02	**7.30E+02** (+++)	7.40E+02	7.20E+00
F36	30	1.09E+07	2.51E+07	4.83E+07	1.02E+07	1.05E+07	3.37E+07	6.61E+07	1.66E+07	4.13E+06	1.31E+07	3.80E+07	8.95E+06	1.64E+06	**7.74E+06** (+++)	1.53E+07	4.45E+06
F37	50	4.96E+07	1.05E+08	2.00E+08	4.58E+07	7.78E+07	1.80E+08	3.07E+08	6.30E+07	1.94E+07	5.86E+07	1.08E+08	2.45E+07	2.00E+07	**3.28E+07** (+++)	5.36E+07	9.91E+06
F38	100	1.98E+08	4.38E+08	6.90E+08	1.53E+08	4.89E+08	8.87E+08	1.51E+09	2.15E+08	1.59E+08	3.19E+08	5.87E+08	1.03E+08	1.09E+08	**1.47E+08** (+++)	1.89E+08	2.63E+07
F39	20	2.39E+01	3.49E+01	4.81E+01	8.50E+00	2.10E+02	3.04E+02	4.12E+02	5.70E+01	2.20E+01	4.91E+01	1.16E+02	2.18E+01	4.40E-04	**1.16E-03** (+++)	1.81E-03	5.39E-04
F40	30	1.81E+01	4.44E+01	7.72E+01	2.01E+01	2.25E+02	4.88E+02	6.92E+02	1.02E+02	2.63E+01	6.66E+01	1.56E+02	2.83E+01	1.53E-04	**2.63E-04** (+++)	5.00E-04	1.17E-04
F41	20	1.20E+02	2.78E+02	4.60E+02	1.05E+02	1.05E+03	2.76E+03	4.01E+03	7.32E+02	1.97E+02	4.31E+02	1.00E+03	1.94E+02	1.05E-02	**4.37E-02** (+++)	1.68E-01	4.88E-02
F42	30	2.96E+02	6.41E+02	1.21E+03	2.54E+02	3.60E+03	6.19E+03	9.22E+03	1.60E+03	3.32E+02	9.86E+02	1.60E+03	3.28E+02	1.38E-02	**2.72E-02** (+++)	5.22E-02	1.33E-02
F43	20	2.15E+02	3.94E+02	5.71E+02	1.06E+02	2.07E+03	3.17E+03	5.05E+03	7.70E+02	1.45E+02	4.57E+02	7.87E+02	1.67E+02	2.22E-02	**6.35E-02** (+++)	1.11E-01	2.39E-02
F44	30	9.01E+02	1.47E+03	2.30E+03	5.02E+02	5.76E+03	1.02E+04	1.53E+04	2.42E+03	7.97E+02	2.08E+03	4.46E+03	9.28E+02	8.97E-02	**4.18E-01** (+++)	6.66E-01	2.18E-01
F45	20	2.30E+01	3.80E+01	5.00E+01	1.02E+01	2.02E+02	3.41E+02	5.59E+02	8.39E+01	2.00E+01	4.78E+01	1.11E+02	2.00E+01	0.00E+00	**3.00E-01** (+++)	1.00E+00	4.83E-01
F46	30	2.80E+01	4.03E+01	5.40E+01	8.82E+00	2.98E+02	4.61E+02	6.82E+02	1.03E+02	3.60E+01	7.99E+01	1.74E+02	3.07E+01	0.00E+00	**0.00E+00** (+++)	0.00E+00	0.00E+00
F47	20	7.17E+00	8.46E+00	9.49E+00	6.81E-01	1.39E+01	1.65E+01	1.81E+01	1.02E+00	6.91E+00	9.19E+00	1.17E+01	1.09E+00	2.47E-01	**5.29E-01** (+++)	9.45E-01	2.93E-01
F48	30	6.10E+00	7.05E+00	8.95E+00	8.00E-01	1.52E+01	1.62E+01	1.76E+01	6.89E-01	6.09E+00	8.98E+00	1.18E+01	1.19E+00	7.82E-03	**4.70E-01** (+++)	1.57E+00	7.30E-01
F49	20	4.32E+00	5.86E+00	7.30E+00	1.04E+00	2.48E+01	7.07E+01	1.12E+02	2.06E+01	5.39E+00	9.16E+00	1.48E+01	2.44E+00	1.39E-01	**3.17E-01** (+++)	4.40E-01	1.21E-01
F50	30	3.09E+00	6.55E+00	8.17E+00	1.59E+00	3.52E+01	9.96E+01	1.55E+02	2.83E+01	5.86E+00	1.19E+01	1.72E+01	2.70E+00	2.97E-01	**5.04E-01** (+++)	8.12E-01	1.59E-01
F51	20	1.05E+02	1.59E+02	2.17E+02	3.93E+01	2.70E+02	6.70E+02	1.27E+03	2.40E+02	1.06E+02	1.44E+02	2.03E+02	2.66E+01	1.60E+01	**1.79E+01** (+++)	1.95E+01	1.24E+00
F52	30	2.02E+02	3.28E+02	4.03E+02	7.14E+01	1.45E+03	2.30E+03	4.96E+03	7.64E+02	1.94E+02	3.21E+02	5.27E+02	9.56E+01	2.40E+01	**7.86E+01** (+++)	1.36E+02	2.75E+01
F53	20	1.21E+02	1.37E+02	1.56E+02	1.32E+01	1.51E+02	1.80E+02	2.12E+02	1.52E+01	1.11E+02	1.36E+02	1.64E+02	1.40E+01	4.11E+01	**9.22E+01** (+++)	1.38E+02	2.74E+01
F54	30	2.11E+02	2.65E+02	2.91E+02	2.25E+01	2.88E+02	3.20E+02	3.46E+02	1.83E+01	2.03E+02	2.52E+02	2.88E+02	2.40E+01	3.49E+01	**6.85E+01** (+++)	9.36E+01	1.83E+01

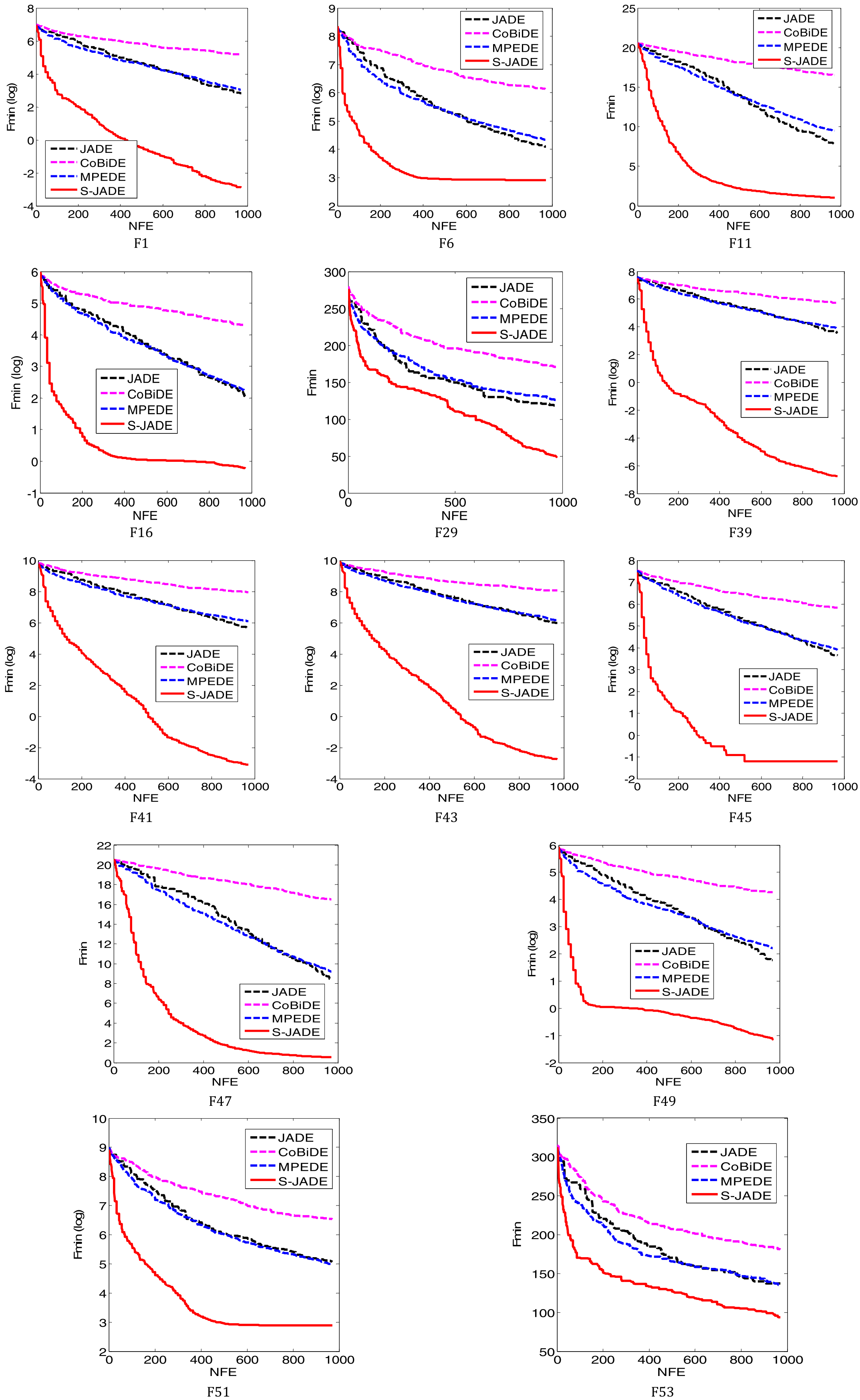


Fig. 13. Convergence curves of JADE, CoBiDE, MPEDE and S-JADE for 20-D functions.

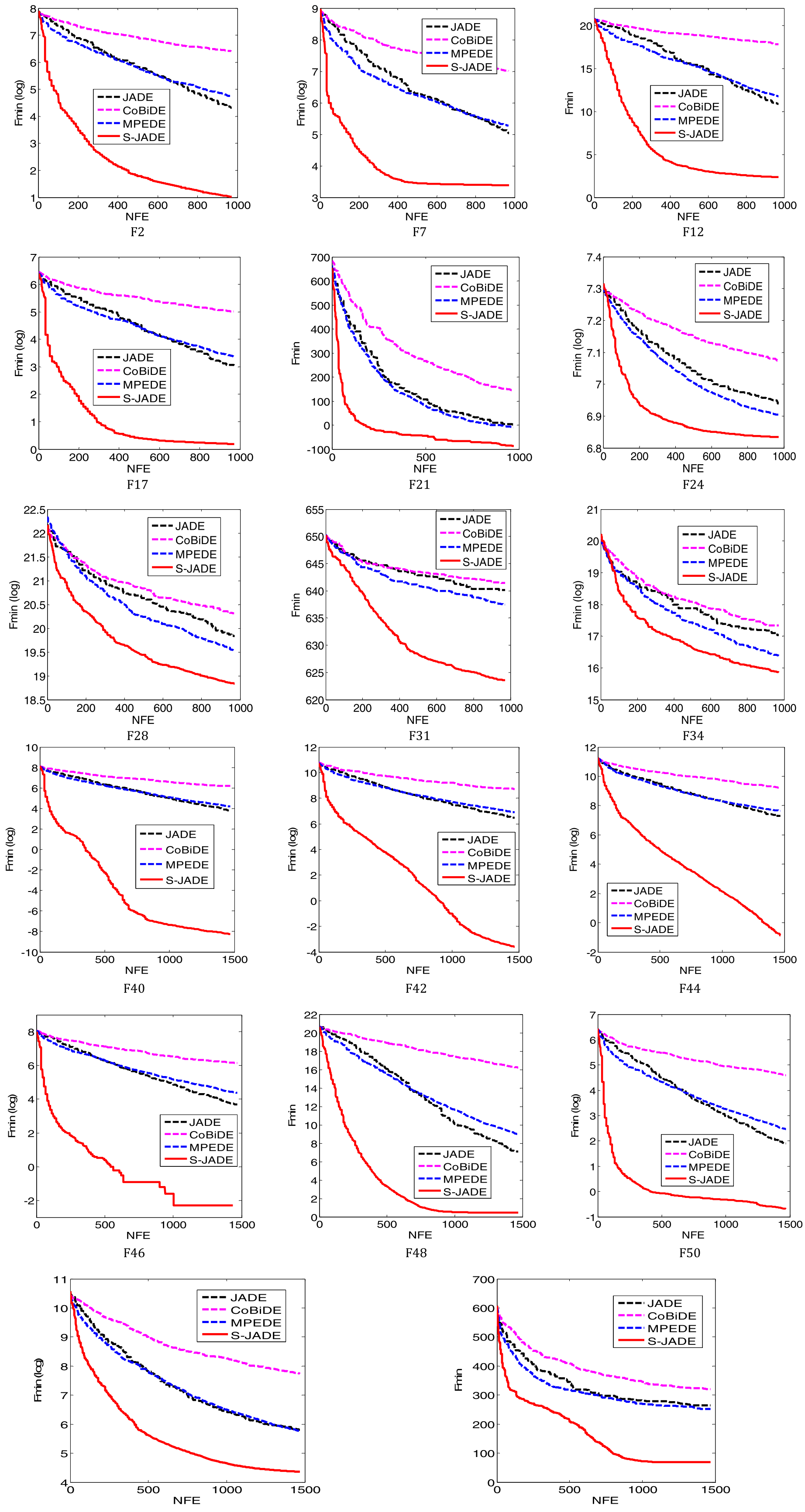


Fig. 14. Convergence curves of JADE, CoBiDE, MPEDE and S-JADE for 30-D functions.

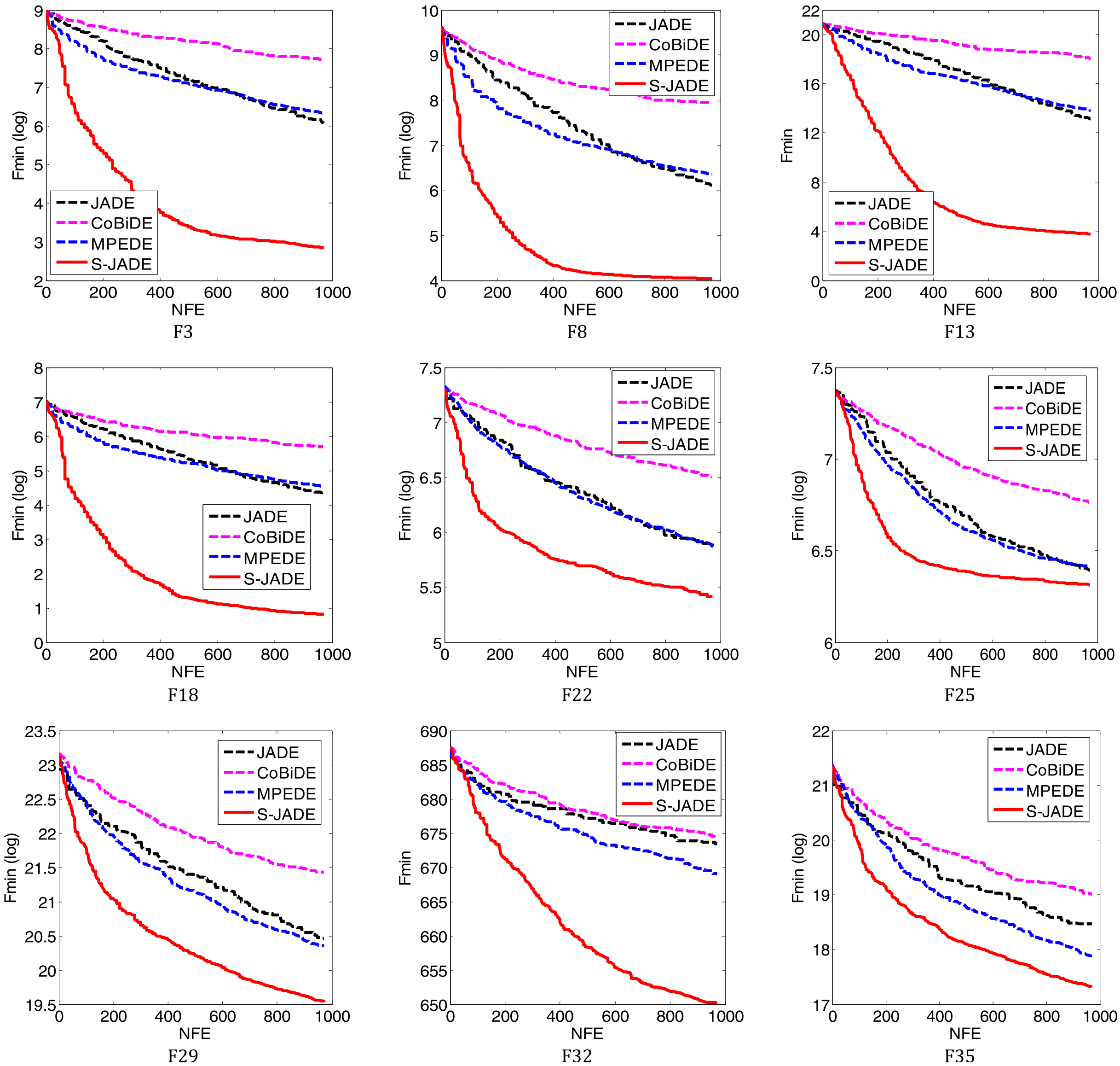


Fig. 15. Convergence curves of JADE, CoBiDE, MPEDE and S-JADE for 50-D functions.

4.2. Comparison between S-JADE and the basic DE algorithms

To validate the performance of the proposed S-JADE algorithm, three DE algorithms including JADE and the recently proposed CoBiDE [70] and MPEDE [71] are tested for comparison. In the experiments, the parameters of JADE are set according to Zhang et al. [64]. Specifically, the population size is $N = 30$. The initial values of μ_F and μ_{CR} are 0.5 and p is 0.05. The JADE's Matlab code can be available online. The parameters of S-JADE are set as follows: the population size is $N = 30$. The initial value of μ_F and μ_{CR} are respectively 0.5 and 0.75. p is set as 0.05. r is set in $[0, 1.25]$. L is set as 10. The type of RBF is Cubic. For JADE and S-JADE, the same initial population points are used. They are obtained by the Latin hypercube sampling method in Matlab 2011a. The MATLAB codes of CoBiDE and MPEDE are available online. To make a fair comparison, the size of population points for these two algorithms ($N = 30$) is set the same as those in JADE and S-JADE. The maximum number of exact function evaluation (NFE) is set as 1000 for all problems except the eight 30D problems in part B of "CEC 2014'' [68], which defines a stopping criterion for optimization of computationally expensive problems. According to "CEC 2014'' [68], NFE is set as 1500 to optimize these 30D problems. In order to obtain a robust optimizing result, all tested optimizing algorithms perform 25 independent runs on each test problem in MATLAB 2011a. In obtaining the predicted optima of global and local surrogates in S-JADE, the basic JADE algorithm is used. The testing results of JADE, CoBiDE, MPEDE and S-JADE are listed in Table 2. The bold values in Table 2 and in other tables mean that S-JADE obtains better optimization results than the other compared algorihtms. The performance of these algorithms is also validated by the Wilcox rank sum test. The testing results are listed after the mean optimizing results of S-JADE in Table 2, where "+'' means S-JADE performs better than the compared algorithm, "≈'' means S-JADE performs similarly to the compared algorithm and "-'' means S-JADE performs worse than the compared algorithm. The convergence curves of different algorithms for all test problems are plotted in Figs. 13–17. According to the testing results, S-JADE has a vivid performance advantage over JADE, CoBiDE and MPEDE within 1000 function evaluations. The high efficiency of S-JADE can be attributed to the way of combining surrogates in the proposed algorithm. The surrogates usually can provide useful predicted response information of a problem. This could direct S-JADE to search in a relatively accurate way. As a result, the optimization efficiency of S-JADE is greatly improved. Besides, S-JADE has good global search ability because it inherits the search mechanism of JADE. For example, S-JADE can

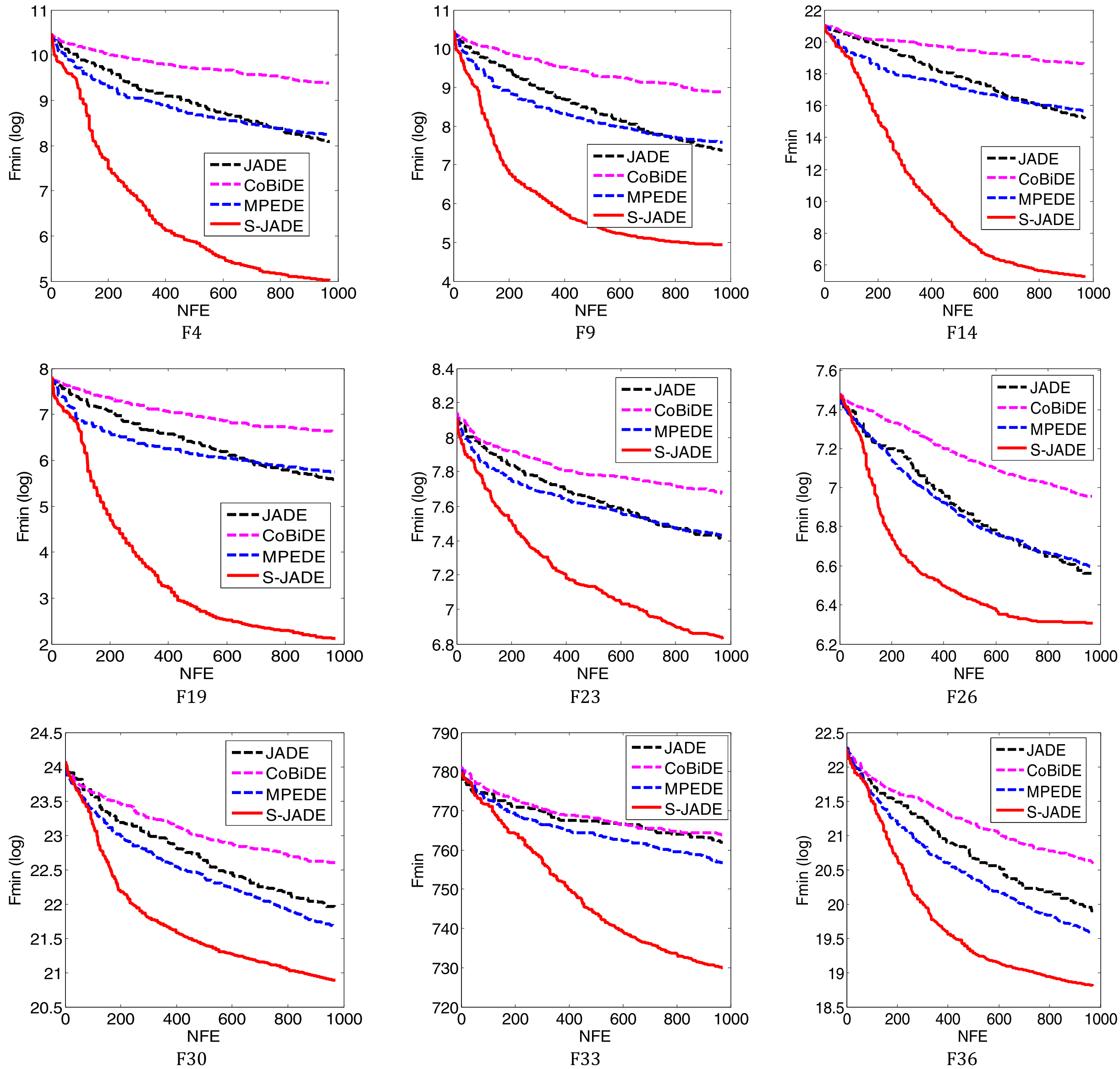


Fig. 16. Convergence curves of JADE, CoBiDE, MPEDE and S-JADE for 100-D functions.

almost reach the global optimum for F1, F2, F9, F10, F13 and F14. Moreover, among the basic DE algorithms, JADE seems not always to performs the best. For example, MPEDE can perform better than JADE for functions F24, F28 and so on. This shows that the JADE algorithm may be not the best choice for the combination with surrogates. In the future study, it will be worthwhile to apply the proposed innovation in S-JADE to other DE optimization framework in order to further improve the efficiency for optimization of expensive problems.

4.3. Comparison between S-JADE and other surrogate-assisted metaheuristic algorithms in the literature

To further demonstrate the performance of S-JADE, three surrogate-assisted algorithms GS-SOMA [54], GPEME [29,30] and SA-COSO [42] in other studies are used for comparison.

GS-SOMA is a typical metaheuristic algorithm assisted by surrogate-based local search. It can achieve a good balance between utilizing the prediction ability of surrogates and the global search ability of the GA algorithm in the optimization process. Researchers [43,54] call this search mode as Lamarckian learning. A performance comparison between the GS-SOMA and S-JADE is made. The same problems including F7, F12, F17, F21 and F24 in Table 1 were also tested previously [54]. The results of GS-SOMA, extracted from the converging figures in that study, are used for comparison. The comparison results are listed in Table 3. It can be seen that S-JADE performs much better than GS-SOMA for these problems. For F7, F12 and F24, it obtains a better optimal result than GS-SOMA within 2000 exact function evaluations. GS-SOMA needs to carry out more than 8000 exact function evaluations to obtain the same optimal result as S-JADE. A similar case also happens on F17 and F21. The reason why S-JADE performs better than GS-SOMA could be that different surrogates are used in their optimization processes.

To make a fair comparison between GS-SOMA and S-JADE, GS-SOMA is tested by using the same type of RBF in S-JADE. To be consistent with S-JADE, the size of training points used to build an accurate local surrogate is set as 5D in the trust region local search (STR) of GS-SOMA. The setting of other parameters in GS-SOMA such as the GA parameters and iterations of STR local search is following the previous study [54]. The testing results for the corresponding functions are listed in Table 4. The performance of GS-SOMA and S-JADE is also validated by the Wilcox rank sum test. The testing results are listed after the mean optimizing results of S-JADE in Table 4. According to Table 4, S-JADE has a clear performance advantage over GS-SOMA for most of the tested problems.

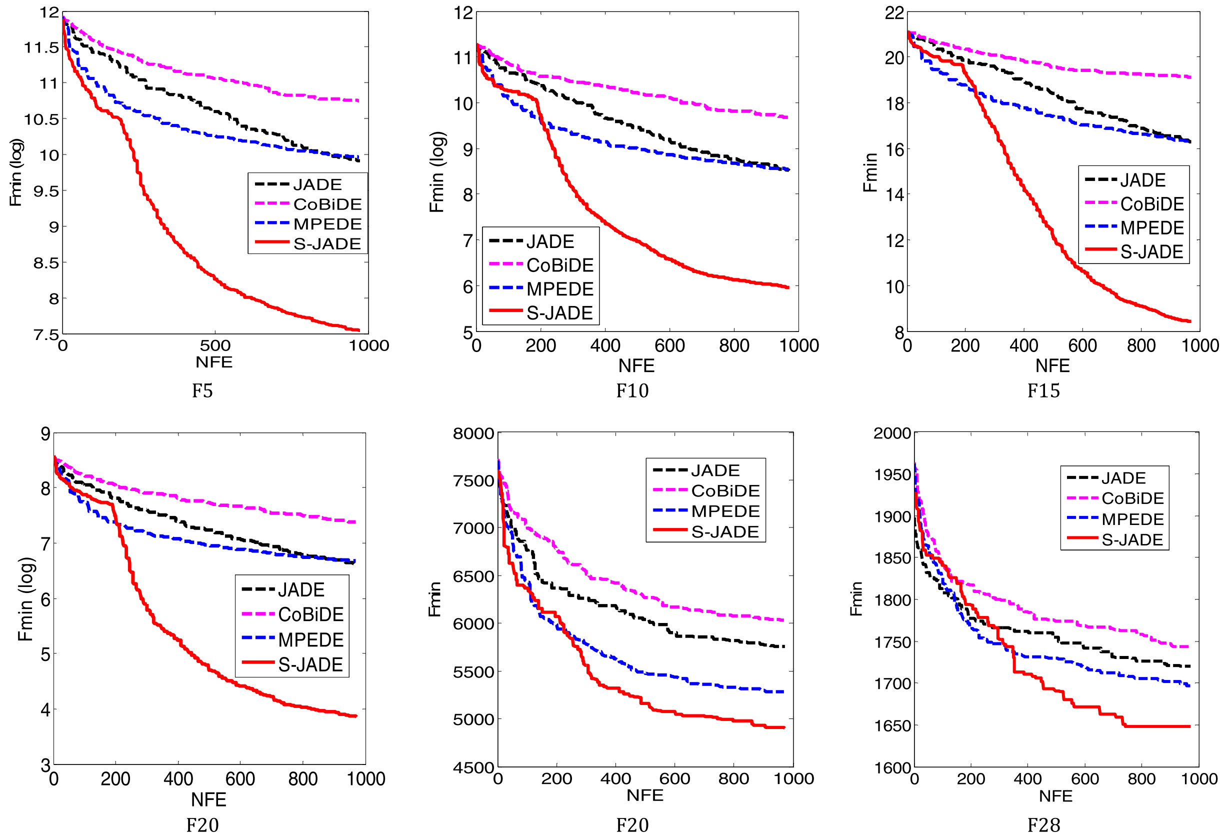


Fig. 17. Convergence curves of JADE, CoBiDE, MPEDE and S-JADE for 200-D functions.

Table 3
Comparison between GS-SOMA's optimizing results [54] and S-JADE's optimizing results.

Fun.	GS-SOMA (2010)		S-JADE
	Mean, NFE = 1000	NFE, $Mean_{(STR\text{-}GA)} \approx Mean_{(ESPSO)}$	Mean, NFE = 1000
F7	2981	>8000	**29.5605**
F12	20	>8000	**2.3425**
F17	330	2500	**1.2028**
F21	>50	3500	**−86.7159**
F25	>1119	>8000	**929.0540**

To show the characteristics of GS-SOMA and S-JADE in their optimization processes, the convergence curve for F13 is plotted in Fig. 18. It can be seen that GS-SOMA has a faster converging speed initially but performs worse than S-JADE finally. This can be explained. The surrogate-based trust region method (STR) is a local search method. It can help GS-SOMA converge very fast to a local optimum. However, running STR could make the diversity of population points decrease quickly, thus weakening the global search ability of GS-SOMA. For S-JADE, because of the neighbor region partition strategy, it has a more reasonable control of the diversity of population points. Therefore, S-JADE has a higher global optimizing efficiency than GS-SOMA.

GPEME is proposed to handle medium-scale problems whose di-

Table 4
Comparative results tested by STR-GA and ESPSO.

Fun	D	GS-SOMA (2010, Self-coded), NFE = 1000				S-JADE, NFE = 1000			
		Best	Mean	Worst	Std.	Best	Mean	Worst	Std.
F2	30	0.3244	1.8414	3.7616	0.9046	0.9273	2.7688 (≈)	5.1138	1.3573
F3	50	44.9224	72.7770	99.5215	14.8557	9.9299	**17.2949** (+)	29.0006	5.0822
F4	100	503.8746	803.8367	917.9952	126.1017	101.7501	**149.8095** (+)	315.0916	60.7157
F7	30	28.6285	30.3899	33.8795	1.1596	28.6683	**29.5605** (+)	30.4839	0.5427
F8	50	83.2610	100.2787	116.7138	9.1403	53.9776	**56.7418** (+)	60.4051	2.3352
F9	100	265.0226	392.0748	478.5143	54.3845	128.6549	**139.7993** (+)	180.6933	15.1350
F12	30	2.2288	2.8987	3.5217	0.3734	1.8012	**2.3425** (+)	3.2367	0.4935
F13	50	5.4811	7.1454	8.2451	0.6251	3.2091	**3.7818** (+)	4.2156	0.3412
F14	100	8.3842	10.0316	10.7161	0.5880	4.7851	**5.2881** (+)	5.7285	0.2699
F17	30	1.0198	1.1322	1.3061	0.0669	1.0611	1.2028 (≈)	1.3832	0.1012
F18	50	7.6278	12.5668	17.5349	2.4589	1.9587	**2.2667** (+)	2.7755	0.3142
F19	100	35.6921	57.0523	67.4822	7.8194	5.7248	**8.3286** (+)	10.3589	1.3760
F21	30	−84.0932	−49.4336	−10.6352	18.9261	−174.5859	**−86.7159** (+)	−20.3397	43.0988
F22	50	165.1355	223.8222	268.9377	27.0030	122.3341	224.7417 (≈)	292.7608	47.9480
F23	100	1.18E+03	1.28E+03	1.38E+03	57.7904	848.0980	**921.4770** (+)	1.03 E+03	69.8446
F25	30	925.0172	935.0925	948.0079	7.0349	923.6931	**929.0540** (+)	936.2157	4.3111

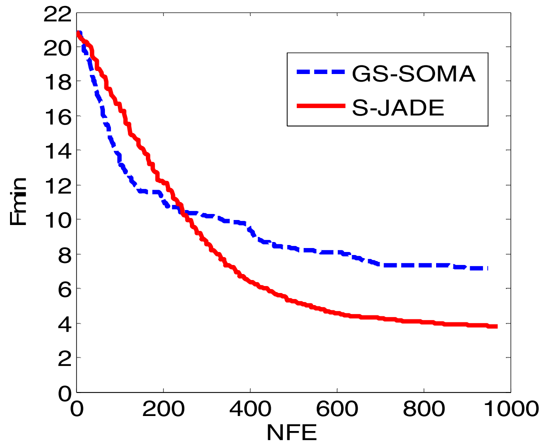


Fig. 18. Convergence curves of GS-SOMA and S-JADE for the 50-D Ackley function (F13).

mensions are smaller than 50, because the kriging surrogate used is very costly to build for high dimensional problems. The GPEME algorithm is also a surrogate-assisted differential evolution algorithm. It uses the LCB samples' infilling criteria of kriging to prescreen out some potential offspring points among the candidate offspring points produced by basic DE algorithm. The optimizing results of GPEME and S-JADE are listed in Table 5. The optimizing results of GPEME are taken from the previous studies [29,30]. Because some problems are commonly tested in the two references, the better and more recent optimizing results of them are thus cited. The performance of GPEME and S-JADE is validated by *t*-test. The testing results are listed after the mean optimizing results of S-JADE in Table 5, where "+'' means S-JADE performs better than the compared algorithm, "≈'' means S-JADE performs similarly to the compared algorithm and "-'' means S-JADE performs worse than the compared algorithm. According to Table 5, it can be found that for some relatively low dimensional problems such as the 20-D functions of F1, F6 and F16 and the 30-D functions of F2 and F17, GPEME has a higher optimum accuracy than S-JADE. However, S-JADE can also search close to the global optimum and has a faster converging speed in the initial searching stage. The corresponding converging curves of S-JADE and GPEME for F1 and F16 are plotted in Fig. 19, where the optimum results of GPEME are extracted every 100 evaluations from the figures in the earlier study [30]. The reasons why S-JADE cannot obtain a highly accurate global optimum can be explained in two aspects. Firstly, the surrogates are only required to provide a mutation direction in S-JADE. They cannot guarantee the mutation accuracy. The predicted difference vector $x^*_{r1,g} - x_{r2,g}$ could be not accurate, which thus influences the optimizing accuracy of S-JADE. Secondly, the fast convergence of S-JADE could make it miss the global optimum information in the optimizing process. Besides, for some complex functions such as the 30-D functions of F21 and F24, S-JADE performs better than GPEME. For the 50-D dimensional problems such as F3, F8, F13 and F18, S-JADE has a vivid performance advantage over GPEME. The reason why S-JADE performs better than GPEME for these problems with higher dimensions could be contributed to the way of using surrogates in the optimization process. GPEME only applies a prescreening strategy based on the global kriging surrogate to select

Table 5
Optimizing results tested by GPEME and S-JADE.

Fun.	D	NFE	GPEME (2014)				S-JADE			
			Best	Mean	Worst	Std.	Best	Mean	Worst	Std.
F1 [30]	20	1500	5.4E-13	5.79E-12	3.89E-11	1.18E-11	0.0040	0.0136 (−)	0.0261	0.0108
F2 [30]	30	2000	4.38E-10	2.50E-09	7.10E-09	2.23E-09	0.0889	0.3755 (−)	0.9290	0.3226
F3 [29]	50	1000	134.0681	221.0774	372.5567	81.6123	9.9299	**17.2949** (+)	29.0006	5.0822
F6 [30]	20	1500	4.0519	13.8642	17.0733	3.7562	15.1592	17.0821 (−)	18.6645	1.4894
F7 [30]	30	2000	22.5733	31.1038	78.7295	16.8010	27.7620	28.2624 (≈)	29.0185	0.4695
F8 [29]	50	1000	172.3547	258.2787	401.4187	80.1877	53.9776	**56.7418** (+)	60.4051	2.3352
F11 [30]	20	1500	0.0001	0.2580	1.4235	0.5472	0.1530	0.3783 (≈)	0.8209	0.2817
F12 [30]	30	2000	0.0001	1.4354	5.2219	1.5602	0.0038	1.4177 (≈)	2.1103	0.8401
F13 [29]	50	1000	9.2524	13.2327	14.9343	1.5846	3.2091	**3.7818** (+)	4.2156	0.3412
F16 [30]	20	1500	0.0000	0.0039	0.0172	0.0066	0.2204	0.3654 (−)	0.6713	0.1825
F17 [30]	30	2000	0.0000	0.0012	0.0123	0.0039	0.8507	1.0014 (−)	1.0522	0.0851
F18 [29]	50	1000	22.5456	36.6459	64.9767	13.1755	1.9587	**2.2667** (+)	2.7755	0.3142
F21 [29]	30	1000	−57.0678	−21.8610	18.0327	36.4492	−174.5859	**−86.7159** (+)	−20.3397	43.0988
F25 [29]	30	1000	933.1601	958.5939	992.8618	25.6946	923.6931	**929.0540** (+)	936.2157	4.3111
F29 [30]	20	1500	15.0031	37.8471	75.2502	16.5290	29.8958	35.2001 (≈)	43.1380	5.9678

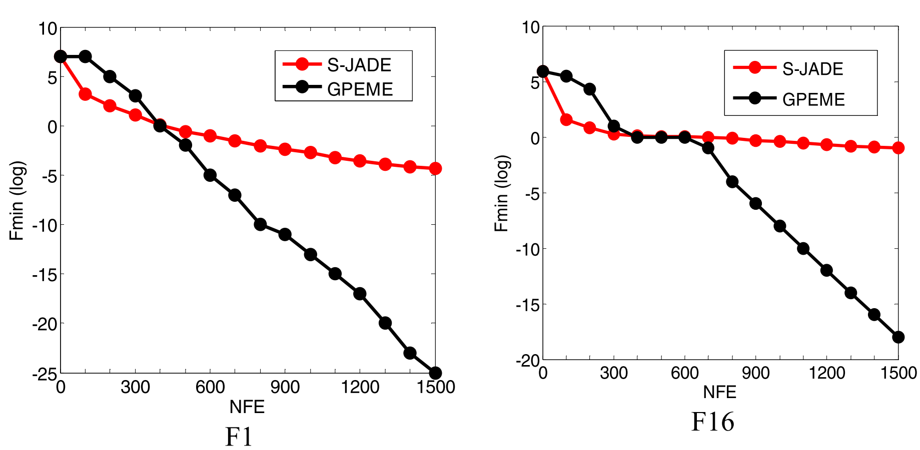


Fig. 19. Convergence curves of S-JADE and GPEME for F1 and F16.

Table 6
Optimizing results tested by SA-COSO and S-JADE.

Fun.	D	SA-COSO(2017),NFE = 1000		S-JADE, NFE = 1000		S-JADE, NFE = 1000 (Use the same type of RBF in SA-COSO)	
		Mean	Std.	Mean	Std.	Mean	Std.
F3	50	51.475	16.246	**17.2949** (+)	5.0822	**26.0185** (+)	12.1986
F4	100	1033.2	317.18	**149.8095** (+)	60.7157	**216.3782** (+)	85.6406
F8	50	252.58	40.744	**56.7418** (+)	2.3352	**78.5170** (+)	7.6379
F9	100	2714.2	117.02	**139.7993** (+)	15.1350	**180.6055** (+)	22.2965
F13	50	8.9318	1.0668	**3.7818** (+)	0.3412	**4.6257** (+)	0.4144
F14	100	15.756	0.5025	**5.2881** (+)	0.2699	**6.5778** (+)	0.6587
F18	50	6.0062	1.1043	**2.2667** (+)	0.3142	**4.4288** (+)	1.4854
F19	100	63.353	19.021	**8.3286** (+)	1.3760	**15.8400** (+)	4.6655
F22	50	197.16	30.599	224.7417 (−)	47.9480	440.2699 (−)	85.2273
F23	100	1.2731E+03	117.19	**921.4770** (+)	69.8446	1.5553E+03 (−)	77.0500

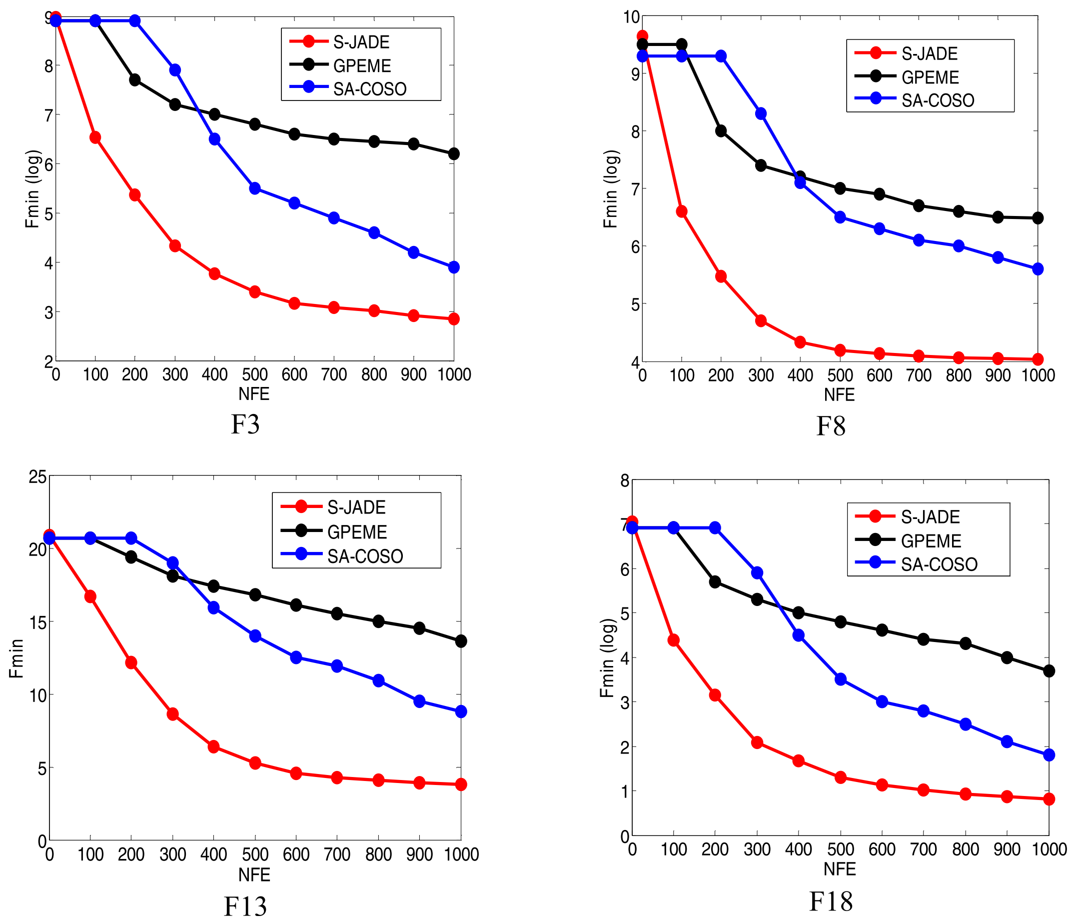


Fig. 20. Convergence curves of S-JADE, GPEME and SA-COSO for the 50-D functions.

Table 7
Time cost of S-JADE for Rosenbrock functions.

Fun.	Dimension	T_{comp} (s)	F (s)	Rate ($F/(F + T_{comp})$)
Rosenbrock (NFE = 1000)	20D	1.4E+03	2000*300 = 6E+05	99.77%
	30D	1.8E+03		99.70%
	50D	2.0E+03		99.67%
	100D	4.3E+03		99.29%
	200D	1.5E+04		97.56%

some promising offspring points under the DE optimization framework. However, S-JADE not only applies a prescreening strategy but also applies the optima of local surrogates to guide the search of the JADE algorithm. To sum up, S-JADE has a fast convergence speed and good global search ability. But its optimizing accuracy still needs to be improved in the future.

SA-COSO is a newly proposed surrogate-assisted PSO algorithm to handle expensive high dimensional problems whose dimension can reach 100. The compared results of S-JADE and SA-COSO for the 50-D and 100-D functions are listed in Table 6. The performance of SA-COSO and S-JADE is validated by *t*-test. The testing results are listed after the mean optimizing results of S-JADE in Table 6. The optimizing results of SA-COSO are taken from a previous study [42]. The convergence curves of S-JADE and SA-COSO and GPEME for the 50-D functions including F3, F8, F13 and F18 are plotted in Fig. 20, where the optimal results of GPEME and SA-COSO are extracted every 100 evaluations from the figures in earlier studies [29,42]. It can be seen that S-JADE performs better than SA-COSO for all tested functions except F22. Besides, S-JADE can converge to a better global optimum with a fast speed when compared

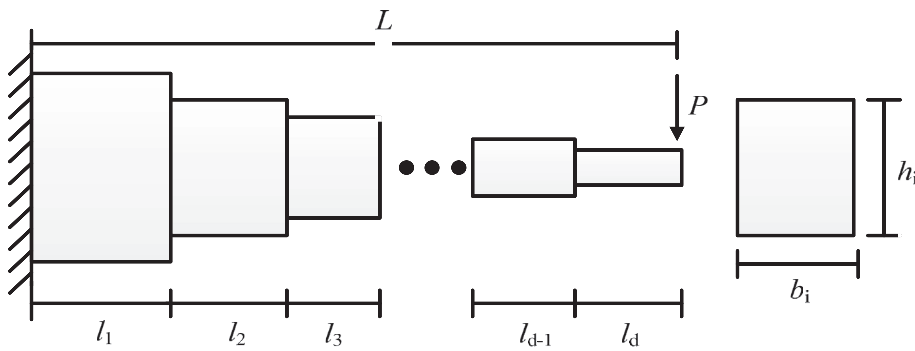


Fig. 21. Stepped cantilever beam with d steps.

Table 8
Optimizing results of different algorithms.

Metrics	Algorithms (NFE = 990)							
	ABC (2005)	TLBO (2011)	MFO (2015)	TRMPS2 (2015)	S-JADE			
					$N = L = 90$	$N = L = 60$	$N = L = 30$	$N = 30, L = 10$
Best	0.0171	0.0104	0.0180	N/A	0.0083	0.0081	0.0079	**0.0063**
Mean	0.0272	0.0191	0.0247	0.0153	0.0110	0.0109	0.0101	**0.0066**
Median	0.0251	0.0188	0.0221	N/A	0.0109	0.0107	0.0101	**0.0063**
Worst	0.0468	0.0290	0.0623	N/A	0.0135	0.0137	0.0117	**0.0079**
Std.	0.0064	0.0047	0.0091	0.00212	0.0019	0.0016	0.0013	**0.0006**

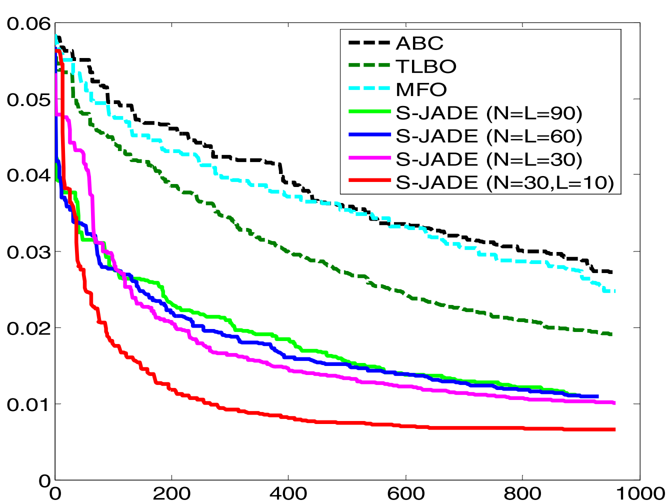


Fig. 22. Convergence curves of different optimization algorithms.

with GPEME and SA-COSO for all the commonly tested 50-D functions. The reason could be the difference of optimization theories and used surrogates. To validate whether different types of RBF models cause the performance difference between S-JADE and SA-COSO, the S-JADE algorithm is tested by using the same type of RBF used in SA-COSO. The RBF model in SA-COSO is expressed as: $\widehat{f}(x) = \lambda_0 + \sum_{i=1}^{n} \lambda_i \Phi_i(x - x_i)$. Compared with the RBF used in S-JADE, it does not add the polynomial term. The type of its basis function is Gaussian. The testing results of S-JADE using the same type of RBF in SA-COSO are listed in Table 6. The types of RBF models do have an effect on the performance difference between S-JADE and SA-COSO. However, the effect is slight. S-JADE with the same type RBF used in SA-COSO also performs much better for most of the testing problems expect for the 50-D and 100-D shifted rotated Rastrigin functions (F22 and F23). The reason why SA-COSO performs well for this function could be that it is a hybrid algorithm of combining different PSO algorithms. It makes the diversity of swarm particles enhanced in the optimizing process and thus searches a more accurate global optimum than S-JADE for this function. The performance superiority of S-JADE over SA-COSO could be attributed to its optimizing mechanism. It can make a good balance between utilizing the prediction ability of surrogates and the global search ability of PSO in the optimization process.

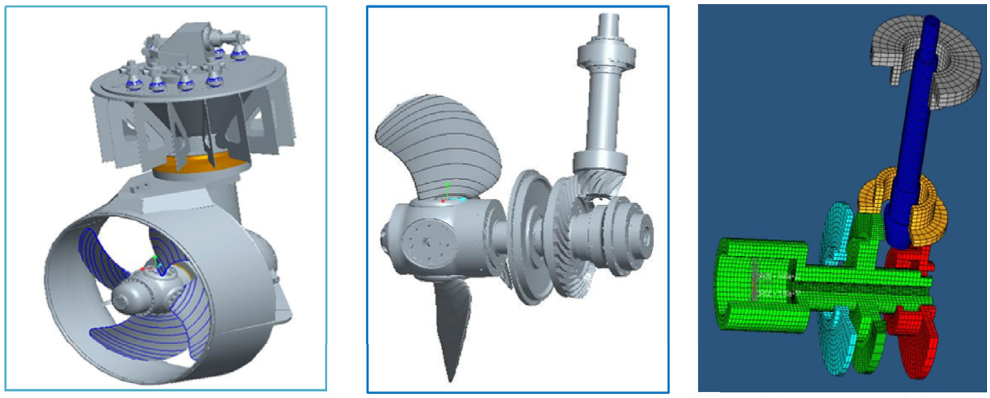
4.4. Analysis of S-JADE's computational complexity

In this subsection, an analytical study on the computational complexity of the S-JADE is presented when it is applied for optimization of expensive problems. The total computational effort, referred to here as T_{comp}, of S-JADE is formulated as follows:

$$T_{comp} = \left(T_{exp} + \sum_{i=1}^{N} F_i \right) + (g - 1) \cdot \left(\sum_{i=1}^{L} F_i + F_{gbest} + \sum_{i=1}^{N} FLS_i + FGS + T_{addtional} \right) \tag{18}$$

where:

T_{exp} Time of experiment design to generate the initial population points, which is often negligible
F Exact function evaluation cost
g Generations of S-JADE
L Number of exact function evaluations in each generation
FLS Time to build the local RBF surrogates
FGS Time to build the global RBF surrogate
$T_{addtional}$ Additional time costs such as response predictions and population updating procedures, which are often negligible



(a). Contour model (b). UG model (c). FEM model

Fig. 23. Different models of the propeller.

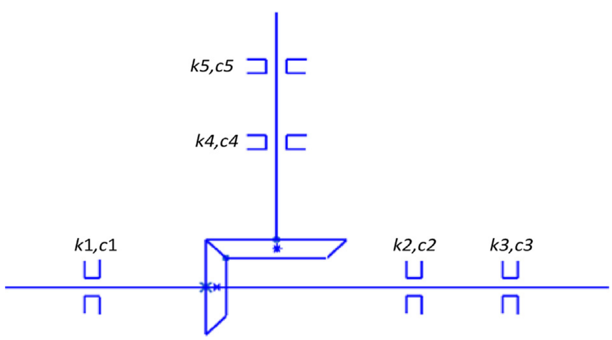


Fig. 24. Axis structure diagram of propeller.

Although there are several elements in Eq. (18), when working with computationally expensive problems, the most significant part contributing to the total computational effort is F. When F is significantly large, the other elements in Eq. (18) can be negligible. To validate this viewpoint, the Rosenbrock function is taken for an example and it is assumed that one exact function evaluation needs 5 min (300 s). The time costs of S-JADE for the Rosenbrock function with different dimensions are reported in Table 7. It can be seen that the rate of time cost for exact function evaluations $F/(F + T_{comp})$ is higher than 97% for all tested Rosenbrock functions. This shows that F is the main contribution to the total computational cost when it is significantly large.

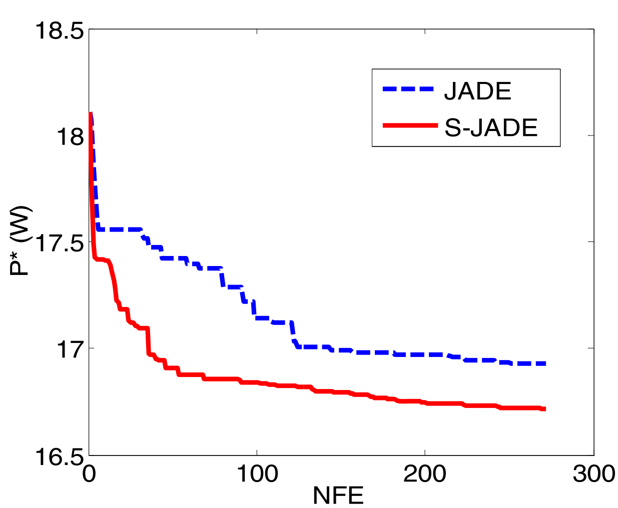


Fig. 25. Convergence curves of JADE and S-JADE for the propeller design problem.

4.5. Engineering applications

4.5.1. Optimal design of a stepped cantilever beam

The engineering application design problem is cited from Cheng et al. [22]. It involves minimizing the tip deflection (δ) of a stepped cantilever beam subject to constraints, as shown in Fig. 21. Because the problem is very complex, there is still no known analytical optimum.

For testing the proposed algorithm, a cantilever beam with $d = 10$ steps is chosen. It bears a $P = 50$ kN force on the tip. $E = 200$ GPa and $\sigma_{allow} = 350$ MPa are selected as the properties for the material used. For each beam step, there are three variables for the optimization: width (b_i), height (h_i), and length (l_i) of the beam step. Therefore, 30 input variables exist in this minimization problem and are arrayed in the following order: $X = [b_1, h_1, l_1, b_2, h_2, l_2, \cdots, b_{10}, h_{10}, l_{10}]$ The optimizing problem is described as follows:

$$Min \quad \delta = \frac{P}{3E}\sum_{i=1}^{d}\left[\frac{12}{b_i h_i^2}\left(\left(\sum_{j=i}^{d} l_j\right)^3 - \left(\sum_{j=i+1}^{d} l_j\right)^3\right)\right] \tag{19}$$

Subject to

$$\frac{6P\sum_{j=i}^{d} l_j}{b_i h_i^2} \le \sigma_{allow}, \quad i = 1, 2, \cdots 10.$$

$$\frac{h_i}{b_i} \le AR, \quad i = 1, 2, \cdots 10.$$

$$\sum_{i=1}^{d} b_i h_i l_j \le V_{max}$$

$$\sum_{i=1}^{d} l_j \ge L_{min}$$

$$b_i \in [0.01, 0.05m], \quad h_i \in [0.3, 0.65m], \quad l_j \in [0.5, 1m], \quad i = 1, 2, \cdots 10.$$

where σ_{allow} is the bending stress constraint of all step beams, $AR = 25$ is the aspect ratios constraint of all cross sections of the step beams, $V_{max} = 1.2$ is volume constraint of the materials and $L_{min} = 5$ is the length constraint of the step beams.

To obtain a robust optimizing result, the problem is tested by the S-

JADE algorithm for 25 times. The initial population points include the points designed by Latin hypercube sampling method and the center point of the design space. To test the effects of N and L, the parameters N and L in S-JADE are set using different values including $N = L = 90$, $N = L = 60$, $N = L = 30$ and $N = 30$, $L = 10$. In handling the constraints of this problem, a simple judgment criterion is used. It defines that, if the candidate offspring points outbreak the constraints, they will not be updated and evaluated by the objective function. The corresponding offspring points will not be updated and they will be set the same as their parent points. Besides, the global surrogate in S-JADE only searches in the local neighbor region belonging to the current best population point. The optimizing results are listed in Table 8. To make a comparison, the results of the trust region based mode pursuing sampling (TRMPS2) cited from a previous study [22] are also listed in Table 8. The TRMPS2 is also proposed to solve high dimensional expensive optimization problems. In addition, some popular metaheuristic algorithms including Artificial Bee Colony (ABC) [72], Teaching-Learning-Based Optimization (TLBO) [73] and Moth-Flame Optimization (MFO) [74] are tested for comparison. To make the comparison fair, the same constraint handling strategy as S-JADE is used for these algorithms. The comparison results in Table 8 show that the proposed S-JADE algorithm is the most efficient because it can converge to a better optimum with the fastest speed when compared with other optimizing algorithms. Moreover, the smaller values of N and L can help to speed up convergence of S-JADE according to testing results plotted in Fig. 22. The optimal design found by the proposed algorithm is $X_{opt} = [0.050000, 0.650000, 0.500000, 0.050000, 0.650000, 0.500000,$ $0.050000, 0.650000, 0.500000, 0.050000, 0.650000, 0.500000,$ $0.050000, 0.650000, 0.500000, 0.050000, 0.650000, 0.500000,$ $0.050000, 0.650000, 0.500000, 0.050000, 0.650000, 0.600001, 0.050000,$ $0.650000, 0.500000, 0.050000, 0.650000, 0.500000]$, $\delta_{opt} = 0.0062793$.

4.5.2. Optimal design of bearings in an all-direction propeller

The all-direction propeller has been widely used in drilling platforms and ships that require high positioning accuracy. In the design of a propeller, how to enhance the vibration resisting property is an important topic to improve the service performance of the propeller. Power flow is a dynamic index used to reflect the vibration effect of the propeller. With the aim of obtaining an optimal design solution with good dynamic performance, it is utilized as an optimization objective which should be minimized. But the calculation of the propeller power flow should be done by computational expensive finite element analysis simulations, which makes the acquisition of the power flow a black-box problem. Therefore, to improve the design efficiency, surrogates can be used to approximate the relationship between the power flow of the propeller and several key structural parameters including the stiffness and damping coefficients of the bearings in the propeller drive system. To show the optimization problem better, the different models of the propeller are shown in Fig. 23. The finite element model (FEM) is used for power flow simulation.

The optimizing problem can be summarized as:

$$\begin{aligned}
&find\ \ K = [k_1, k_2, k_3, k_4, k_5, c_1, c_2, c_3, c_4, c_5]^T \\
&\min\ \ P^* = \sqrt{\sum_{j=1}^{Z} (P_j)^2} \\
&P_j = \omega / 2\pi \int_0^{\frac{\omega}{2\pi}} \mathrm{Re}(f_j)\mathrm{Re}(v_j)\mathrm{d}t\, s.t.\quad k_i^L \le k_i \le k_i^U, c_i^L \le c_i \le c_i^U, \quad i = 1,2,3,4,5
\end{aligned} \tag{20}$$

where k_i and c_i are the stiffness and damping coefficients of different bearings in propeller as Fig. 24 shows, k_i^L and k_i^U are the range of k_i and c_i^L and c_i^U are the range of c_i. P_j is the power flow at the selected evaluation point j in the propeller, and Z is the sum of evaluation points distributed in different bearings, which is set as 10 in this study. The power flow is the sum of product of force and speed in a cycle as Eq. (20) defines. More details about power flow calculation and application for vibration control can be read elsewhere [75]. The initial values and the range of design variables are given as: $[k_1, k_2, k_3, k_4, k_5, c_1, c_2, c_3, c_4, c_5,]_{\text{initial}} = [3, 4, 3, 3, 4, 1.5, 2, 1.5, 1.5, 1.5, 2]$. The corresponding initial power flow of the propeller is $P^* = 22.58\text{W}$. $k_1, k_3, k_4 \in [2, 4] \times 10^7\text{N/m}, k_2, k_5 \in [2, 6] \times 10^7\text{N/m}$, $c_1, c_3, c_4 \in [1, 2] \times 10^7\text{Ns/m}, c_2, c_5 \in [1, 3] \times 10^7\text{Ns/m}$. The axis structure diagram of the propeller is shown in Fig. 24.

For this optimization problem, the JADE and S-JADE algorithms are used. Because this design problem involves expensive power flow simulation, the optimization process is stopped when the number of all simulation points reaches 300 and the experiment of JADE and S-JADE will be only carried out for five times to obtain a robust result. The convergence curves of different algorithms for the propeller design problem are depicted in Fig. 25. It can be seen that S-JADE can make the propeller's power flow decrease at a fast speed and to a smaller value when compared with the initial design and the compared JADE algorithm. Besides, considering that the optimizing efficiency could be also influenced by the time costs of JADE and S-JADE, the computational costs of JADE and S-JADE in the optimization process are also listed in Table 9. Running a power flow simulation takes about 5 min. Obtaining 300 simulation points needs approximately 300 × 5 × 60 = 90,000 s. The time cost of the running procedures of JADE algorithm itself is about 0.03 s, which could be negligible. The time of running the procedures of S-JADE algorithm itself is about 480s. Though the procedures of S-JADE could take more time in the optimization process, they can be negligible when compared with the simulation time for exact function evaluation (F) according to Table 9. Therefore, proposed S-JADE algorithm is thought to be more efficient than JADE in the optimal design of different bearings in the all-direction propeller.

5. Conclusions

This paper introduces a new efficient surrogate-guided differential evolution algorithm for optimization of computationally high dimensional expensive problems. The proposed algorithm is based on the basic JADE algorithm. Compared with JADE, it adds to use the optimum information provided by both the global surrogate built in the whole design space and the local surrogates built in the neighbor region around each current population point. In addition, a simple surrogate-guided selection

Table 9
Comparison between initial solution and optimal solutions.

Metrics	Initial design	Optimal design (JADE)				Optimal design (S-JADE)			
		best	mean	worst	std.	best	mean	worst	std.
Power flow (P^*) (W)	22.5842	16.9015	16.9301	16.9588	**0.02**	**16.6516**	**16.7181**	**16.7816**	0.06
Time cost (F) (s)	0	300*5*60 = 90,000							
Time cost (T_{comp}) (s)	0	300*5*60 + 0.03 = 90,000.03				300*5*60 + 540 = 90,540			
Rate (F/T_{comp})	N/A	300*5*60/90,000.03 = 100.00%				300*5*60/90,540 = 99.40%			

strategy is proposed to further improve its optimizing efficiency. To validate the proposed S-JADE algorithm, it is tested by several widely used high dimensional numerical benchmark problems whose dimensions vary from 20 to 200. An overall comparison between the S-JADE and other non-surrogate-assisted and surrogate-assisted metaheuristic algorithms is made. The results show that S-JADE has high optimization efficiency for most of tested problems.

Compared with the basic DE algorithms such as JADE, CoBiDE and MPEDE, S-JADE has vivid performance superiority for all tested problems. For some common functions such as the 20D unimodal Ellipsoid, multimodal Ackley and Griewank functions, S-JADE even can search close to the global optimum within 1000 function evaluations. This proves that the way of combining surrogates in S-JADE has a positive effect on improving the optimizing efficiency of the basic JADE algorithm. The global and local surrogates could provide useful optimum information and an approximate landscape for a problem. They could direct S-JADE to search in a relatively accurate and efficient way. Compared with GS-SOMA, S-JADE can obtain a more accurate optimizing result in a fast convergence speed for most of the tested problems. It especially performs much better than GS-SOMA for some multimodal functions such as the 30D Ackley and Shifted Rotated Rastrigin functions. The reason could be contributed to that the neighbor region partition strategy is used in S-JADE. It can control the diversity of population points reasonably in the S-JADE's optimization process. Compared with GPEME, S-JADE has a faster convergence speed especially for problems whose dimension is more than 30. However, GPEME has a higher optimizing accuracy for some relatively low dimensional problems whose dimension is less than 30 though S-JADE also can search close to the global optimum. The reason could be that GPEME used the DE/best/1 search mechanism, which is biased to local search. In another aspect, the surrogates usually could not be built very accurately, thus making their predicted optimum and response information also not very accurate. The optimization accuracy of S-JADE is thus influenced. Besides, S-JADE has a vivid performance advantage over SA-COSO for almost all the 50 and 100 dimensional problems. However, it performs slightly worse than SA-COSO for the very complicated 50D Shifted Rotated Rastrigin function. The reason could be that SA-COSO searches based on the PSO search mechanism. This could make it suitable for optimization of this function. Finally, two engineering applications including the optimal design of a stepped cantilever beam and the optimal design of bearings in an all-direction propeller are used to validate the high efficiency of S-JADE. To sum up, the proposed surrogate-guided differential evolution algorithm is promising for optimization of the high dimensional expensive problems especially for those problems with dimensions of more than 30.

From the above discussion, some improvement still can be done in the future to improve the optimizing efficiency of S-JADE further for high dimensional expensive problems. Firstly, the proposed way of combining surrogates with JADE could be applied to other efficient DE algorithms or other metaheuristic algorithms. This is because, according this study, the DE variant of MPEDE and the PSO variant of SA-COSO seem to perform better than JADE and S-JADE respectively for some problems. Besides, the optimization accuracy of proposed S-JADE should be improved. This problem could be solved by multiple surrogates. They usually can provide more accurate optimum and response information than the stand-alone surrogate and thus direct S-JADE to search in an accurate way. Moreover, the multi-fidelity surrogate [76] could be considered in the metamodeling of S-JADE because it is usually a useful technique to improve the optimization efficiency for expensive problems. Finally, the proposed algorithm is expected to be applied in more engineering practice in the future study.

Acknowledgment

Financial support from the National Natural Science Foundation of China under Grant No.51805180, No.51435009 and No.51721092, Natural Science Foundation of Hubei Province under Grant No. 2018CFA078, and Program for HUST Academic Frontier Youth Team is gratefully acknowledged.

References

[1] M. Mitchell, An Introduction to Genetic Algorithms, MIT Press, 1998.
[2] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Micro Machine and Human Science, 1995. MHS'95, Proceedings of the Sixth International Symposium on, IEEE, 1995, pp. 39–43. https://doi.org/10.1109/MHS.1995.494215.
[3] R. Storn, K. Price, Differential evolution–a simple and efficient metaheuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (4) (1997) 341–359. https://doi.org/10.1023/A:1008202821328.
[4] Y.W. Leung, Y. Wang, An orthogonal genetic algorithm with quantization for global numerical optimization, IEEE Trans. Evol. Comput. 5 (1) (2001) 41–53. https://doi.org/10.1109/TEVC.2004.82689510.1109/4235.910464.
[5] J.T. Tsai, T.K. Liu, J.H. Chou, Hybrid Taguchi-genetic algorithm for global numerical optimization, IEEE Trans. Evol. Comput. 8 (4) (2004) 365–377. https://doi.org/10.1109/TEVC.2004.826895.
[6] H. Wang, H. Sun, C. Li, S. Rahnamayan, J.S. Pan, Diversity enhanced particle swarm optimization with neighborhood search, Inf. Sci. 223 (2013) 119–135. https://doi.org/10.1016/j.ins.2012.10.012.
[7] J.J. Jamian, M.N. Abdullah, H. Mokhlis, M.W. Mustafa, A.H.A. Bakar, Global particle swarm optimization for high dimension numerical functions analysis, J. Appl. Math. 2014 (2) (2014) 1–14. https://doi.org/10.1155/2014/329193.
[8] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Trans. Evol. Comput. 13 (2) (2009) 398–417. https://doi.org/10.1109/TEVC.2008.927706.
[9] R. Mallipeddi, P.N. Suganthan, Q.K. Pan, M.F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, Appl. Soft Comput. 11 (2) (2011) 1679–1696. https://doi.org/10.1016/j.asoc.2010.04.024.
[10] W. Gong, Z. Cai, D. Liang, Adaptive ranking mutation operator based differential evolution for constrained optimization, IEEE Trans. Cybern. 45 (4) (2015) 716–727. https://doi.org/10.1109/TCYB.2014.2334692.
[11] T.W. Simpson, A.J. Booker, D. Ghosh, A.A. Giunta, P.N. Koch, R.J. Yang, Approximation methods in multidisciplinary analysis and optimization: a panel discussion, Struct. Multidiscip. Optim. 27 (5) (2004) 302–313. https://doi.org/10.1007/s00158-004-0389-9.
[12] X. Cai, H. Qiu, L. Gao, X. Shao, Metamodeling for high dimensional design problems by multi-fidelity simulations, Struct. Multidiscip. Optim. 56 (1) (2017) 151–166. https://doi.org/10.1007/s00158-017-1655-y.
[13] M.F. Hussain, R.R. Barton, S.B. Joshi, Metamodeling: radial basis functions, versus polynomials, Eur. J. Oper. Res. 138 (1) (2002) 142–154. https://doi.org/10.1016/S0377-2217(01)00076-5.
[14] A.J. Smola, B. Schölkopf, A tutorial on support vector regression, Stat. Comput. 14 (3) (2004) 199–222. https://doi.org/10.1023/B:STCO.0000035301.49549.88.
[15] G.G. Wang, Adaptive response surface method using inherited Latin hypercube design points, Trans.-Am. Soc. Mech. Eng., J. Mech. Des. 125 (2) (2003) 210–220. https://doi.org/10.1115/1.1561044.
[16] R. Jin, W. Chen, T.W. Simpson, Comparative studies of metamodeling techniques under multiple modeling criteria, Struct. Multidiscip. Optim. 23 (1) (2001) 1–13. https://doi.org/10.1007/s00158-001-0160-4.
[17] S. Shan, G.G. Wang, Development of adaptive rbf-hdmr model for approximating high dimensional problems, in: ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers, 2009, pp. 727–740. https://doi.org/10.1115/DETC2009-86531.
[18] S. Shan, G.G. Wang, Metamodeling for high dimensional simulation-based design problems, J. Mech. Des. 132 (5) (2010) 051009. https://doi.org/10.1115/1.4001597.
[19] D.R. Jones, M. Schonlau, W.J. Welch, Efficient global optimization of expensive black-box functions, J. Glob. Optim. 13 (4) (1998) 455–492. https://doi.org/10.1023/A:1008306431147.
[20] H.M. Gutmann, A radial basis function method for global optimization, J. Glob. Optim. 19 (3) (2001) 201–227. https://doi.org/10.1023/A:1011255519438.
[21] L. Wang, S. Shan, G.G. Wang, Mode-pursuing sampling method for global optimization on expensive black-box functions, Eng. Optim. 36 (4) (2004) 419–438. https://doi.org/10.1080/03052150410001686486.
[22] G.H. Cheng, A. Younis, K.H. Hajikolaei, G.G. Wang, Trust region based mode pursuing sampling method for global optimization of high dimensional design problems, J. Mech. Des. 137 (2) (2015) 021407. https://doi.org/10.1115/1.4029219.
[23] H. Wang, Y. Jin, J. Doherty, Committee-based active learning for surrogate-assisted particle swarm optimization of expensive problems, IEEE Trans. Cybern. 47 (9) (2017) 2664–2677. https://doi.org/10.1109/TCYB.2017.2710978.
[24] Y. Jin, Surrogate-assisted evolutionary computation: recent advances and future challenges, Swarm Evol. Comput. 1 (2) (2011) 61–70. https://doi.org/10.1016/j.swevo.2011.05.001.
[25] S. Das, S.S. Mullick, P.N. Suganthan, Recent advances in differential evolution–an updated survey, Swarm Evol. Comput. 27 (2016) 1–30. https://doi.org/10.1016/j.swevo.2016.01.004.
[26] M.T. Emmerich, K.C. Giannakoglou, B. Naujoks, Single-and multiobjective evolutionary optimization assisted by Gaussian random field metamodels, IEEE Trans. Evol. Comput. 10 (4) (2006) 421–439. https://doi.org/10.1109/TEVC.2005.859463.

[27] Q. Zhang, W. Liu, E. Tsang, B. Virginas, Expensive multiobjective optimization by MOEA/D with Gaussian process model, IEEE Trans. Evol. Comput. 14 (3) (2010) 456–474. https://doi.org/10.1109/TEVC.2009.2033671.

[28] Z. Zhou, Y.S. Ong, P.B. Nair, A.J. Keane, K.Y. Lum, Combining global and local surrogate models to accelerate evolutionary optimization, IEEE Trans. Syst. Man Cybern. -Syst., Part C (Applications and Reviews) 37 (1) (2007) 66–76. https://doi.org/10.1109/TSMCC.2005.855506.

[29] B. Liu, Q. Zhang, G.G. Gielen, A Gaussian process surrogate model assisted evolutionary algorithm for medium scale expensive optimization problems, IEEE Trans. Evol. Comput. 18 (2) (2014) 180–192. https://doi.org/10.1109/TEVC.2013.2248012.

[30] B. Liu, Q. Chen, Q. Zhang, G. Gielen, V. Grout, Behavioral study of the surrogate model-aware evolutionary search framework, in: Evolutionary Computation (CEC), 2014 IEEE Congress on, IEEE, 2014, pp. 715–722. https://doi.org/10.1109/CEC.2014.6900373.

[31] B. Liu, H. Yang, M.J. Lancaster, Global optimization of microwave filters based on a surrogate model-assisted evolutionary algorithm, IEEE Trans. Microw. Theor. Tech. 65 (6) (2017) 1976–1985. https://doi.org/10.1109/TMTT.2017.2661739.

[32] B. Liu, V. Grout, A. Nikolaeva, Efficient global optimization of actuator based on a surrogate model assisted hybrid algorithm, IEEE Trans. Ind. Electron. 65 (7) (2018) 5712–5721. https://doi.org/10.1109/TIE.2017.2782203.

[33] L.G. Fonseca, A.C. Lemonge, H.J. Barbosa, A study on fitness inheritance for enhanced efficiency in real-coded genetic algorithms, in: Evolutionary Computation (CEC), 2012 IEEE Congress on, IEEE, 2012, pp. 1–8. https://doi.org/10.1109/CEC.2012.6256154.

[34] R.G. Regis, Particle swarm with radial basis function surrogates for expensive black-box optimization, J. Comput. Sci. 5 (1) (2014) 12–23. https://doi.org/10.1016/j.jocs.2013.07.004.

[35] R. Mallipeddi, M. Lee, An evolving surrogate model-based differential evolution algorithm, Appl. Soft Comput. 34 (2015) 770–787. https://doi.org/10.1016/j.asoc.2015.06.010.

[36] W. Gong, A. Zhou, Z. Cai, A multioperator search strategy based on cheap surrogate models for evolutionary optimization, IEEE Trans. Evol. Comput. 19 (5) (2015) 746–758. https://doi.org/10.1109/TEVC.2015.2449293.

[37] L. Vincenzi, P. Gambarelli, A proper infill sampling strategy for improving the speed performance of a surrogate-assisted evolutionary algorithm, Comput. Struct. 178 (2017) 58–70. https://doi.org/10.1016/j.compstruc.2016.10.004.

[38] C. Praveen, R. Duvigneau, Low cost PSO using metamodels and inexact pre-evaluation: application to aerodynamic shape design, Comput. Methods Appl. Mech. Eng. 198 (9–12) (2009) 1087–1096. https://doi.org/10.1016/j.cma.2008.11.019.

[39] C. Sun, Y. Jin, J. Zeng, Y. Yu, A two-layer surrogate-assisted particle swarm optimization algorithm, Soft Comput. 19 (6) (2015) 1461–1475. https://doi.org/10.1007/s00500-014-1283-z.

[40] C. Jin, A.K. Qin, K. Tang, Local ensemble surrogate assisted crowding differential evolution, in: Evolutionary Computation (CEC), 2015 IEEE Congress on, IEEE, 2015, pp. 433–440. https://doi.org/10.1109/CEC.2015.7256922.

[41] S.M. Elsayed, T. Ray, R.A. Sarker, A surrogate-assisted differential evolution algorithm with dynamic parameters selection for solving expensive optimization problems, in: Evolutionary Computation (CEC), 2014 IEEE Congress on, IEEE, 2014, pp. 1062–1068. https://doi.org/10.1109/CEC.2014.6900351.

[42] C. Sun, Y. Jin, R. Cheng, J. Ding, J. Zeng, Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems, IEEE Trans. Evol. Comput. 21 (4) (2017) 644–660. https://doi.org/10.1109/TEVC.2017.2675628.

[43] Y.S. Ong, P.B. Nair, A.J. Keane, Evolutionary optimization of computationally expensive problems via surrogate modeling, AIAA J. 41 (4) (2003) 687–696. https://doi.org/10.2514/2.1999.

[44] M.D. Parno, T. Hemker, K.R. Fowler, Applicability of surrogates to improve efficiency of particle swarm optimization for simulation-based problems, Eng. Optim. 44 (5) (2012) 521–535. https://doi.org/10.1080/0305215X.2011.598521.

[45] Y. Tang, J. Chen, J. Wei, A surrogate-based particle swarm optimization algorithm for solving optimization problems with expensive black box functions, Eng. Optim. 45 (5) (2013) 557–576. https://doi.org/10.1080/0305215X.2012.690759.

[46] A. Safari, K.H. Hajikolaei, H.G. Lemu, G.G. Wang, A high-dimensional model representation guided PSO methodology with application on compressor airfoil shape optimization, in: ASME Turbo Expo 2016: Turbomachinery Technical Conference and Exposition, American Society of Mechanical Engineers, 2016. V02CT45A013-V02CT45A013, https://doi.org/10.1115/GT2016-56741.

[47] I. Tsoukalas, P. Kossieris, A. Efstratiadis, C. Makropoulos, Surrogate-enhanced evolutionary annealing simplex algorithm for effective and efficient optimization of water resources problems on a budget, Environ. Model. Softw 77 (2016) 122–142. https://doi.org/10.1016/j.envsoft.2015.12.008.

[48] H. Yu, Y. Tan, J. Zeng, C. Sun, Y. Jin, Surrogate-assisted hierarchical particle swarm optimization, Inf. Sci. 454 (2018) 59–72. https://doi.org/10.1016/j.ins.2018.04.062.

[49] Z. Zhou, Y.S. Ong, M.H. Nguyen, D. Lim, A study on polynomial regression and Gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm, in: Evolutionary Computation, 2005. The 2005 IEEE Congress on, vol. 3, IEEE, 2005, pp. 2832–2839. https://doi.org/10.1109/CEC.2005.1555050.

[50] Y.S. Ong, P.B. Nair, A.J. Keane, K.W. Wong, Surrogate-assisted Evolutionary Optimization Frameworks for High-Fidelity Engineering Design Problems, Knowledge Incorporation in Evolutionary Computation, Springer, Berlin, Heidelberg, 2005, pp. 307–331, https://doi.org/10.1007/978-3-540-44511-1_15.

[51] D. Lim, Y.S. Ong, Y. Jin, B. Sendhoff, Trusted evolutionary algorithm, in: Evolutionary Computation, 2006, CEC 2006, IEEE Congress on, IEEE, 2006, pp. 149–156. https://doi.org/10.1109/CEC.2006.1688302.

[52] Z. Zhou, Y.S. Ong, M.H. Lim, B.S. Lee, Memetic algorithm using multi-surrogates for computationally expensive optimization problems, Soft Comput. 11 (10) (2007) 957–971. https://doi.org/10.1007/s00500-006-0145-8.

[53] Y.S. Ong, K.Y. Lum, P.B. Nair, Hybrid evolutionary algorithm with Hermite radial basis function interpolants for computationally expensive adjoint solvers, Comput. Optim. Appl. 39 (1) (2008) 97–119. https://doi.org/10.1007/s10589-007-9065-5.

[54] D. Lim, Y. Jin, Y.S. Ong, B. Sendhoff, Generalizing surrogate-assisted evolutionary computation, IEEE Trans. Evol. Comput. 14 (3) (2010) 329–355. https://doi.org/10.1109/TEVC.2009.2027359.

[55] M.N. Le, Y.S. Ong, S. Menzel, Y. Jin, B. Sendhoff, Evolution by adapting surrogates, Evol. Comput. 21 (2) (2013) 313–340. https://doi.org/10.1162/EVCO_a_00079.

[56] L. Shi, K. Rasheed, A survey of fitness approximation methods applied in evolutionary algorithms, in: Computational Intelligence in Expensive Optimization Problems, Springer, Berlin, Heidelberg, 2010, pp. 3–28. https://doi.org/10.1007/978-3-642-10701-6_1.

[57] R.T. Haftka, D. Villanueva, A. Chaudhuri, Parallel surrogate-assisted global optimization with expensive functions–a survey, Struct. Multidiscip. Optim. 54 (1) (2016) 3–13. https://doi.org/10.1007/s00158-016-1432-3.

[58] S. Das, S.S. Mullick, P.N. Suganthan, Recent advances in differential evolution–an updated survey, Swarm Evol. Comput. 27 (2016) 1–30. https://doi.org/10.1016/j.swevo.2016.01.004.

[59] A.A. Mullur, A. Messac, Metamodeling using extended radial basis functions: a comparative approach, Eng. Comput. 21 (3) (2006) 203. https://doi.org/10.1007/s00366-005-0005-7.

[60] A. Díaz-Manríquez, G. Toscano, C.A.C. Coello, Comparison of metamodeling techniques in evolutionary algorithms, Soft Comput. 21 (19) (2017) 5647–5663. https://doi.org/10.1007/s00500-016-2140-z.

[61] K. Holmström, An adaptive radial basis algorithm (ARBF) for expensive black-box global optimization, J. Glob. Optim. 41 (3) (2008) 447–464. https://doi.org/10.1007/s10898-007-9256-8.

[62] M.J. Powell, Radial basis functions in 1990, Adv. Numer. Anal. 2 (1992) 105–210.

[63] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (1) (2011) 4–31. https://doi.org/10.1109/TEVC.2010.2059031.

[64] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958. https://doi.org/10.1109/TEVC.2009.2014613.

[65] S. Kitayama, M. Arakawa, K. Yamazaki, Sequential approximate optimization using radial basis function network for engineering optimization, Optim. Eng. 12 (4) (2011) 535–557. https://doi.org/10.1007/s11081-010-9118-y.

[66] P.N. Suganthan, N. Hansen, J.J. Liang, K. Deb, Y.P. Chen, A. Auger, S. Tiwari, Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization, KanGAL report, 2005, p. 2005005.

[67] J.J. Liang, B.Y. Qu, P.N. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, 2013.

[68] B. Liu, Q. Chen, Q. Zhang, J.J. Liang, Problem Definitions and Evaluation Criteria for Computational Expensive Optimization, Technical Report, 2013.

[69] S. Shan, G.G. Wang, Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions, Struct. Multidiscip. Optim. 41 (2) (2010) 219–241. https://doi.org/10.1007/s00158-009-0420-2.

[70] Y. Wang, H.X. Li, T. Huang, L. Li, Differential evolution based on covariance matrix learning and bimodal distribution parameter setting, Appl. Soft Comput. 18 (2014) 232–247. https://doi.org/10.1016/j.asoc.2014.01.038.

[71] G. Wu, R. Mallipeddi, P.N. Suganthan, R. Wang, H. Chen, Differential evolution with multi-population based ensemble of mutation strategies, Inf. Sci. 329 (2016) 329–345. https://doi.org/10.1016/j.ins.2015.09.009.

[72] D. Karaboga, An Idea Based on Honey Bee Swarm for Numerical Optimization, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department, 2005.

[73] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems, Comput. Aided Des. 43 (3) (2011) 303–315. https://doi.org/10.1016/j.cad.2010.12.015.

[74] S. Mirjalili, Moth-flame optimization algorithm: a novel nature-inspired heuristic paradigm, Knowl. Based Syst. 89 (2015) 228–249. https://doi.org/10.1016/j.knosys.2015.07.006.

[75] Y.P. Xiong, J.T. Xing, W.G. Price, A general linear mathematical model of power flow analysis and control for integrated structure–control systems, J. Sound Vib. 267 (2) (2003) 301–334. https://doi.org/10.1016/S0022-460X(03)00194-9.

[76] X. Cai, H. Qiu, L. Gao, L. Wei, X. Shao, Adaptive radial-basis-function-based multifidelity metamodeling for expensive black-box problems, AIAA J. 55 (7) (2017) 2424–2436. https://doi.org/10.2514/1.J055649.