An Estimation of Distribution Algorithm With Filtering and Learning

Lixin Tang, *Senior Member, IEEE*, Xiangman Song, Jiyin Liu, and Chang Liu, *Member, IEEE*

***Abstract*— Estimation of distribution algorithm (EDA) is an efficient population-based stochastic search technique. Since it was proposed, many attempts have been made to improve its performance in the context of nonlinear continuous optimization. However, the success of EDA depends on the accuracy of modeling, the effectiveness of sampling, and the ability of exploration. An effective EDA often needs to take some measures to adjust the model and to guide sampling. In this article, we propose a novel EDA which applies the idea of Kalman filtering to revise the modeling data and a learning strategy to improve sampling. The filtering scheme modifies the modeling data set using an estimation error matrix based on historic solution data. During the sampling process, the learning strategy determines the region to sample next based on the sampling outcomes so far, instead of completely random sampling. The proposed EDA also employs a multivariate probabilistic model based on copula function and can quickly reach the promising area in which the optimal solution is likely to be located. A collection of general benchmark functions are used to test the performance of the proposed algorithm. Computational experiments show that the EDA is effective.**

***Note to Practitioners*—In many process industries, there exist black-box operation optimization problems and large-scale nonlinear optimization problems with variable coupling. For these problems, it is difficult to establish mechanism models between input and output. However, real-time data can be measured from the system through sensors. We can utilize this process information to optimize the system so as to attain the desired objective. In this article, we propose a novel estimation of distribution algorithm (EDA) which applies a filtering scheme to revise the modeling data and a learning strategy to improve sampling, which can solve the problems with the characteristics of nonlinearity, variable coupling, and large scale. Computational experiments show that the EDA is effective. In the future, the proposed algorithm can be applied to some practical optimization problems such as operation optimization in blast furnace, which is considered as a continuous production process with variable coupling. The algorithm has the potential to help optimizing the process control parameters.**

***Index Terms*— Estimation of distribution algorithm (EDA), filtering, learning sampling, multivariate probabilistic model.**

Manuscript received February 17, 2020; revised May 9, 2020; accepted June 5, 2020. Date of publication December 22, 2020; date of current version July 2, 2021. This article was recommended for publication by Associate Editor K. Liu and Editor S. A. Reveliotis upon evaluation of the reviewers' comments. This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB0901900, in part by the Major International Joint Research Project of the National Natural Science Foundation of China under Grant 71520107004, in part by the Major Program of National Natural Science Foundation of China under Grant 71790614, in part by the Fund for Innovative Research Groups of the National Natural Science Foundation of China under Grant 71621061, and in part by the 111 Project under Grant B16009. *(Corresponding author: Lixin Tang.)*

Lixin Tang is with the Key Laboratory of Data Analytics and Optimization for Smart Industry, Ministry of Education, Northeastern University, Shenyang 110819, China (e-mail: lixintang@mail.neu.edu.cn).

Xiangman Song is with the Liaoning Engineering Laboratory of Operation Analytics and Optimization for Smart Industry, State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang 110819, China (e-mail: sxm123121@163.com).

Jiyin Liu is with the School of Business and Economics, Loughborough University, Loughborough LE11 3TU, U.K. (e-mail: j.y.liu@lboro.ac.uk).

Chang Liu is with the Liaoning Key Laboratory of Manufacturing System and Logistics, Northeastern University, Shenyang 110819, China (e-mail: lc1987328@126.com).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASE.2020.3019694

I. Introduction

INTELLIGENT algorithms have become very popular for solving optimization problems and are studied widely in the practical problems [1]–[4]. Estimation of distribution algorithms (EDAs) are evolutionary algorithms based on estimation and sampling from probabilistic models. The first EDA was developed by Mühlenbein and Paaß [5], and several improved EDAs have been proposed in recent years. Gaussian distribution is commonly used for modeling in continuous EDAs. Univariate Marginal Distribution Algorithm with Gaussian models for continuous domains (UMDA_c^G) [6], the earliest proposed univariate Gaussian-based EDA, is easy to implement due to its ignoring the dependence of all variables. But it is difficult to solve problems in which the variables have strong dependence with each other. Estimation of Multivariate Normal Algorithm ($\mathrm{EMNA}_{\mathrm{global}}$) [6] was proposed to overcome this. It used a conventional maximum likelihood estimated multivariate Gaussian distribution. Later, Eigenspace EDA (EEDA) [7] was proposed to improve the poor explorative capability. Besides, EDAs based on complex Gaussian distribution are proposed to solve multimodal and intractable problems. Some are with Gaussian mixture distribution [8]–[10]. Considering premature convergence of the traditional Gaussian EDAs, EDA^2 was proposed to deal with it [11]. EDAs have also been used in hybrid algorithms. An algorithm combining Differential Evolution (DE) and EDA was proposed for the global continuous optimization problem [12], and it used an offspring generation scheme, which was similar to the DE crossover. For large-scale optimization problems, EDA with Model Complexity Control (EDA-MCC) [13] performed well using a specially designed multivariate model. Gaussian models can furnish useful information to search the optimal solution, but they cannot always provide an accurate distribution of promising solutions.

Increasingly, EDAs based on different models have been studied. With the capacity of describing arbitrary multimodality, the Histogram models in EDAs [14]–[18] are more flexible than Gaussian models. However, the complexity can be increased with problem size on account of multiple variable dependencies [19]. Some EDAs can learn the structure and parameters of the model in each generation. The factorized distribution algorithm (FDA) [20] can extract a factorization structure from *a priori* knowledge of the problem. EDAs based on Bayesian networks [21] using directed acyclic graphical models can update local parameters according to the conditional probability distributions. Based on Markov random fields, Distribution Estimation using Markov random fields (DEUM) [22] presented a fitness modeling approach to estimating the parameters of the model. Clustering techniques [23] and other statistical methods, such as some variants of the expectation–maximization (EM) algorithm [24], have been used to learn mixtures of distributions in EDAs. These EDAs have given greater emphasis to the modeling structure. But the frameworks of these EDAs are all similar and the historic individual information has not been utilized fully.

In recent years, EDAs have been proposed to deal with many practical problems. A Pareto-based EDA was developed to solve a multiobjective flow-shop scheduling problem [25]. Using a mixed probability distribution model and an adaptive scheme, a new EDA was proposed to solve a multipolicy insurance investment problem [26].

In this article, a novel estimation of distribution algorithm with filtering and learning (EDA-FL) is proposed to solve continuous optimization problems more effectively. The contributions of our proposed algorithm are as follows.

1) A new EDA framework incorporating filtering is proposed. The idea of Kalman filtering is adopted to modify some individuals, which in turn affect the model, using an estimation error matrix from historic information.
2) A sampling strategy with learning is proposed. Using this strategy, the decision on where to sample is made according to the statistical information about the objective values of sampled individuals to avoid completely random sampling.

To illustrate the effectiveness, the proposed algorithm is tested through computational experiments on benchmark functions.

The rest of this article is organized as follows. Section II presents the background knowledge, including EDA, copulas, and Kalman filtering. Section III describes the fundamental ideas of the EDA-FL algorithm. Experimental studies are shown in Section IV. Finally, conclusions are given in Section V.

II. Background

A. Estimation of Distribution Algorithm

EDA is an evolutionary algorithm that uses a probability model based on the distribution of good individuals in the current population to generate individuals of the next generation. The probability model also changes as the population evolves in the search process. Unlike in other evolutionary algorithms, in EDA, each individual in the new generation is not generated by mutation and crossover operations on the individuals of the old generation. Instead, the evolution is at the "macro" level. So, EDA emphasizes on the global information in the search process and has a strong exploration capability.

```
Estimation of Distribution Algorithm
1: Set values of parameters; Set generation number $k = 0$;
2: Randomly generate an initial feasible population $\mathbf{Pop}(0)$;
3: While stopping criterion is not satisfied Do
4:     Select the best individuals from population $\mathbf{Pop}(k)$;
5:     Estimate a probability distribution;
6:     Sample a number of individuals from the distribution
7:     Combine the modeling individuals with the sampled
       individuals to form the new population $\mathbf{Pop}(k+1)$;
8:     Increment generation number $k = k + 1$;
9: End while
```

Fig. 1. Pseudo-code of EDA.

EDA works iteratively, as shown in Fig. 1, where **Pop** represents population or a set of individuals. In this article, we use bold capital letters for matrices or populations, e.g., $\mathbf{A}$ and $\mathbf{Pop}^{mod}$. Vectors and individuals are represented by lowercase bold characters, e.g., $\mathbf{u}$ and $\mathbf{x}_{\mathbf{best}}$. The main steps of EDA are described as follows.

1) *Selection:* Selection aims to obtain a set of good individuals from the current population. It is expected that the selected individuals can describe the solution space accurately and comprehensively. The most widely used selection method is the truncation selection.
2) *Modeling:* Using a statistical method, the probabilistic model is constructed based on the relevant statistical information extracted from the selected individuals.
3) *Sampling:* In the sampling step, a set of individuals called sampled individuals is generated from the constructed probabilistic model. The sampling method is dependent on the modeling method used.
4) *Combination:* In every generation, the offspring population is often obtained by combining the modeling individuals and sampled individuals. In addition, individuals generated randomly according to the uniform distribution in the search space can also be added into the offspring population. The commonly used stopping criterion is the number of fitness evaluations (*n-FEs*) reaching a predefined maximum value.

B. Copulas

A concise definition of copulas is given in [27]: a copula is a multivariate probabilistic distribution function $C(\bullet)$ of D-dimensional random vector $\mathbf{u} = (u_1, \ldots, u_D)$ on the unit hypercube $[0, 1]^D$ with some properties. As can easily be seen, the most well-known copula is the independence distribution function

$$C(u_1, \ldots, u_D) = \prod_{i=1}^{D} u_i. \tag{1}$$

There are two important copulas, which are interesting in practical applications, namely, Gaussian-copula and empirical-copula. Their marginal distributions are Gaussian and empirical distributions, respectively. Other copulas include t-copula, Clayton-copula, Gumbel-copula, and Frank-copula [27].

The key theorem was introduced by Nelsen [28] to separate the effect of dependence from the effect of marginal distributions in a copula.

For a D-dimensional random vector $\mathbf{x} = (x_1, x_2, \ldots, x_D)$, its joint distribution function being $F(\mathbf{x})$ with the marginal distributions $u_d = F_d(x_d)$, $d = 1, \ldots, D$, there exists a copula $C(\mathbf{u})$, which is a multivariate distribution function of vector $\mathbf{u} = (u_1, u_2, \ldots, u_D)$ in $[0, 1]^D$. The following equality was set up and the proof was given in [28]:

$$\begin{aligned} C(u_1, \ldots, u_d) &= F(x_1, \ldots, x_d) \\ &= F\left(F_1^{-1}(u_1), \ldots, F_d^{-1}(u_d)\right) \end{aligned} \tag{2}$$

where $F_d^{-1}(u_d) = \sup\{x_d | F_d(x_d) \leq u_d\}$.

With the analysis above, each $\mathbf{x} = (x_1, x_2, \ldots, x_D)$ can be generated in two steps. The first step is to generate a random vector $\mathbf{u} = (u_1, u_2, \ldots, u_D)$ with the copula function. In the second step, vector $\mathbf{x} = (x_1, x_2, \ldots, x_D)$ is generated using the common inverse function of the marginal distribution of each variable, i.e., $x_d = F_d^{-1}(u_d)$, $d = 1, \ldots, D$.

In recent years, many EDAs using copulas as probabilistic models (copula-EDAs) have been proposed. Based on the copula theory, a framework of EDA was proposed in [29] to solve 2-D optimization problems. Archimedean-copula, Gumbel-copula, and Gaussian-copula have all been used in multivariate EDAs [30]–[34], which were tested on problems of low dimensions in small scale experiments. The maximum likelihood estimation was used in a copula-EDA to estimate the parameters of copula [35]. An EDA based on Gaussian copulas was proposed in [36] to solve multiobjective problems and applied to RFID network planning. An EDA using multivariate extension of the Archimedean copula was presented in [37]. A hybrid algorithm was proposed in [38] combining Archimedean copula-based EDA and artificial bee colony algorithm to achieve faster convergence. A new EDA based on parallel copula was also proposed to improve the efficiency [39]. These algorithms were tested on a small number of benchmark functions. To deal with the major drawbacks, including too much emphasis on parameters and premature convergence, a new EDA based on multivariate elliptical copulas was presented, where the parameters were estimated and a population diversity technique was used [40]. EDAs have also been applied to solve problems in practical settings. For example, an EDA was proposed in [41] to solve problems involving various types of constraints, while EDAs were applied in [42] and [43] to solve flow-shop scheduling problems.

C. Kalman Filtering Process

Kalman filtering [44] is a recursive estimator that uses a series of predictions and observations to make estimation of unknown variables.

For a discrete-time linear system, Kalman filtering model is used to predict the true state $\mathbf{x} \in \mathbb{R}^n$ at time k from the state at $(k-1)$ according to a linear stochastic difference equation

$$\mathbf{x}(k) = \mathbf{A}\mathbf{x}(k-1) + \mathbf{B}\theta(k) + \mathbf{w}(k). \tag{3}$$

The Kalman filtering process

```
1: Set initial values $\mathbf{x}(0)$ and $\mathbf{P}(0)$ for state variable and error
   covariance; Set generation number $k = 1$.
2: **While** stopping criterion is not satisfied **Do**
3:     **Predict**:
           Project the state ahead $\mathbf{x}^-(k)$;
           Project the error covariance ahead $\mathbf{P}^-(k)$;
4:     **Observe and Revise**:
           Compute the gain $\mathbf{g}(k)$;
           Update the state $\mathbf{x}(k)$;
           Update the error covariance $\mathbf{P}(k)$;
       $k = k+1$
5: **End while**
```

Fig. 2. Kalman filtering process.

At time k an observation $\mathbf{y} \in \mathbb{R}^m$ of the true state is given by

$$\mathbf{y}(k) = \mathbf{H}\mathbf{x}(k) + \mathbf{v}(k) \tag{4}$$

where $\mathbf{A}$ is the state-transition matrix which is applied to the previous state, $\mathbf{B}$ is the coefficient matrix for the control input $\theta \in \mathbb{R}^l$, $\mathbf{H}$ is known as the observation matrix, $\mathbf{w}(k)$ is the process noise, and $\mathbf{v}(k)$ is observation noise.

The Kalman filtering process can be described in Fig. 2, where $\mathbf{x}^-(k)$ is the predicted state, $\mathbf{P}^-(k)$ is the predicted error covariance, $\mathbf{g}(k)$ is the Kalman gain, $\mathbf{x}(k)$ is the revised state, and $\mathbf{P}(k)$ is the posteriori error covariance. $\mathbf{x}(0)$ and $\mathbf{P}(0)$ are the initial values of the state variable and the error covariance, respectively.

In this article, we construct the new EDA framework through prediction, sampling, and revision operations using the idea of Kalman filtering.

III. Proposed Algorithm

A. New EDA Framework Incorporating Filtering

In essence, an optimization problem can be viewed as a system. The solution process of the optimization problem can then be considered as the process of searching for the ideal state of the system guided by observation, for which Kalman filtering is an effective technique. Muruganantham *et al.* [45] have proposed a new dynamic multiobjective optimization evolutionary algorithms using Kalman filtering, in which a prediction model is designed to learn patterns from past data. In our new EDA, we transform the best individuals in the last population to make predictions of the optimal solution. A set of the best individuals among those in the last population as well as the predicted individuals are then used for modeling. The samples generated can be considered as observed solutions. To improve the accuracy of the solution, having been inspired by the Kalman filtering process, we revise some individuals based on the difference between the predicted and observed data. The revision in turn affects the set of individuals for modeling in the next generation (iteration). Hopefully, this can improve the effectiveness of the model and thus the EDA process. Our new EDA framework with filtering can be outlined as Fig. 3. In the framework, $\mathbf{Pop}(k)$ is the

Algorithm 1. Overall Framework of the New EDA

```
Input: a stopping criterion;
       $D$: dimension of problem; $NP$: size of population;
       $\alpha$: proportion of population used for modeling;
       $k_{max}$: maximum number of iteration;
       $m$: maximum number of iterations without improvement;
       $n\text{-}FEs$: predefined number of fitness evaluations.
Output: the best solution for optimization problem.
    /* Construct initial population $\mathbf{Pop}(0)$. */
1:  $\mathbf{Pop}(0)$ = INITIALIZATION($D$, $NP$);
2:  $k = 1$; // $k$ is the generation counter.
    /* Select a set of modeling individuals, $\mathbf{Pop}^{mod}(1)$, and
    take the best individual $\mathbf{x_{pcbest}}(1)$ in $\mathbf{Pop}^{mod}(1)$. */
3:  $\mathbf{Pop}^{mod}(1)$= SELECT($\alpha$*$NP$, $\mathbf{Pop}(0)$);
    $\mathbf{x_{pcbest}}(1)$ = BEST($\mathbf{Pop}^{mod}(1)$);
4:  $C_1(u)$ = MODEL($\mathbf{Pop}^{mod}(1)$); // modelling.
5:  $\mathbf{Pop}^{sam}(1)$ = SAMPLE((1-$\alpha$)*$NP$, $C_1(u)$); // sample (*observe*).
    /* Form a new population. */
6:  $\mathbf{Pop}(1)$ = COMBINE($NP$, $\mathbf{Pop}^{mod}(1)$, $\mathbf{Pop}^{sam}(1)$);
    /* *Calculate error matrix* $\mathbf{P}(1)$ using $\mathbf{x_{pcbest}}(1)$ and the
    best $D$ individuals in $\mathbf{Pop}^{sam}(1)$. */
7:  $\mathbf{P}(1)$ = CAL($\mathbf{x_{pcbest}}(1)$, $\mathbf{Pop}^{sam}(1)$);
8:  $k = 2$;
    /* Main loop. */
9:  While *the stopping criterion is not satisfied* do
        /* Make $NP$ predictions of the optimal solution. */
10:     $\mathbf{Pop}^{pre}(k)$ = PREDICT($NP$, $\mathbf{Pop}(k\text{-}1)$); // *predict*.
        $\mathbf{x_{pcbest}}(k)$ = BEST($\mathbf{Pop}^{pre}(k) \cup \mathbf{Pop}(k\text{-}1)$);
        /* Select the best $\alpha$*$NP$ individuals from $\mathbf{Pop}^{pre}(k)$,
        and $\mathbf{Pop}(k\text{-}1)$ and $\mathbf{Pop}^{rev}(k\text{-}1)$, use these for modeling.
        */
11:     $\mathbf{Pop}^{mod}(k)$ = SELECT($\alpha$*$NP$, $\mathbf{Pop}(k\text{-}1) \cup \mathbf{Pop}^{rev}(k\text{-}1)$
                  $\cup\, \mathbf{Pop}^{pre}(k)$);
12:     $C_k(u)$ = MODEL($\mathbf{Pop}^{mod}(k)$);
13:     $\mathbf{Pop}^{sam}(k)$ = SAMPLE((1-$\alpha$)*$NP$, $C_k(u)$); //*observe*.
14:     $\mathbf{Pop}(k)$ = COMBINE($NP$, $\mathbf{Pop}^{mod}(k)$, $\mathbf{Pop}^{sam}(k)$);
        /* *Calculate error gain vector.* */
15:     $\mathbf{g}(k)$ = GAIN($\mathbf{P}(k\text{-}1)$, $\mathbf{x_{pcbest}}(k\text{-}1)$, $\mathbf{Pop}^{sam}(k\text{-}1)$);
        /* *Revise some of the individuals.* */
16:     $\mathbf{Pop}^{rev}(k)$ = REVISE($\mathbf{Pop}^{pre}(k)$, $\mathbf{Pop}^{sam}(k)$);
        /* *Calculate the error matrix* $\mathbf{P}(k)$. */
17:     $\mathbf{P}(k)$ = CAL($\mathbf{x_{pcbest}}(k)$, $\mathbf{Pop}^{sam}(k)$);
18:     Set $k = k + 1$.
19: End while
```

Fig. 3. New EDA framework with filtering.

population of iteration k; $\mathbf{Pop}^{mod}(k)$, $\mathbf{Pop}^{sam}(k)$, $\mathbf{Pop}^{pre}(k)$, and $\mathbf{Pop}^{rev}(k)$ are the sets of modeling, sampled, predicted, and revised individuals, respectively. The names of functions are in capitals and their meanings are obvious.

In each iteration of this framework, the basic EDA steps are listed in normal font. In this article, we introduce a learning strategy to improve the sampling step. The details of the strategy will be presented in Section III-B. We incorporate the idea of Kalman filtering in the framework and the steps related to filtering are indicated in italic. The EDA search process is clearly not a linear system, thus the same linear transformations used in Kalman filtering for linear systems cannot be directly applied. By defining new error matrix and gain vector based on individuals and their objective values in the EDA process, we propose to use different forms of transformation (still keeping linear) here to achieve similar purpose to those in linear systems. The filtering steps embedded in the EDA framework include prediction, observation, and updating of the gain vector and error matrix. Details are given below in the description of the relevant steps.

1) Initialization and the First Iteration: The function $\mathbf{Pop}(0) = \text{INITIALIZATION}(D, NP)$ in Step 1 of the framework initializes the algorithm, which is generating an initial population of NP individuals by drawing the D-dimension variable values randomly from uniform distributions over their search ranges. The algorithm then starts iterations.

The first iteration ($k = 1$) is presented separately in the framework (steps 2–7) because some filtering operations need information from previous iteration and so need to be done differently or cannot be done in the first iteration. The main operations in later iterations ($k > 1$) are presented in steps 9–19 of the framework. The first iteration of the new EDA framework includes two filtering operations, observation (in step 5) and error matrix $\mathbf{P}(k)$ calculation (step 7), which enable the other filtering operations in the next iterations.

To calculate $\mathbf{P}(k)$, we first find the best individual $\mathbf{x_{pcbest}}(k)$ in the population using function BEST(•) (steps 3 and 10), and select the best D individuals among the sampled ones. $\mathbf{P}(k)$ is then calculated as a $D \times D$ matrix with each row being the difference between one of these D individuals and the best individual $\mathbf{x_{pcbest}}(k)$

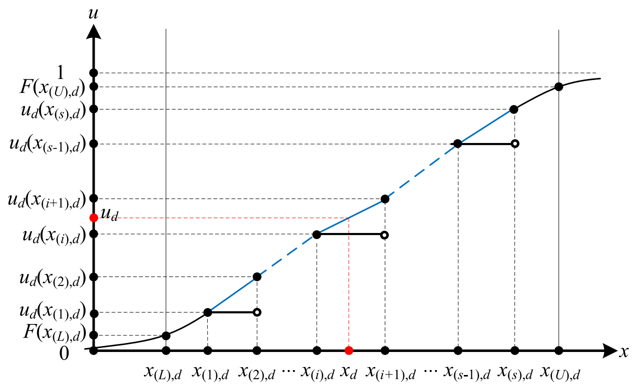
$$\mathbf{P}(k) = \begin{bmatrix} p_{11}(k) & \dots & p_{1D}(k) \\ \dots & \dots & \dots \\ p_{D1}(k) & \dots & p_{DD}(k) \end{bmatrix} = \begin{bmatrix} x_{\text{pcbest},1}(k) - z_{1,1}(k) & \dots & x_{\text{pcbest},D}(k) - z_{1,D}(k) \\ \dots & \dots & \dots \\ x_{\text{pcbest},1}(k) - z_{D,1}(k) & \dots & x_{\text{pcbest},D}(k) - z_{D,D}(k) \end{bmatrix} \tag{5}$$

where $\mathbf{z_i}(k) = [z_{i,1}(k), z_{i,2}(k), \dots, z_{i,D}(k)]$, $i = 1, \dots, D$, are the best D sampled individuals in $\mathbf{Pop}^{sam}(k)$. In this step, the error matrix $\mathbf{P}$ is calculated in $O(D^2)$ time.

2) Prediction (Step 10): The optimal solution is predicted by transforming the best individual in the population $\mathbf{Pop}(k-1)$ plus some noise represented by the difference of two randomly selected individuals in the population. This is expressed in the following formula. Function PREDICT(NP, $\mathbf{Pop}(k-1)$) in Fig. 3 generates NP predicted solutions in this way and stores them in $\mathbf{Pop}^{pre}(k)$

$$\mathbf{x_i^-}(k) = \mathbf{A}\mathbf{x_{best}}(k-1) + \mathbf{\Lambda}[\mathbf{x_a}(k-1) - \mathbf{x_b}(k-1)] \tag{6}$$

where $\mathbf{x_i^-}(k)$, $i = 1, \dots, NP$, is a predicted solution. $\mathbf{A}$ is identity matrix, $\mathbf{x_{best}}(k-1)$ is the best individual, $\mathbf{x_a}(k-1)$ and $\mathbf{x_b}(k-1)$, $a \neq b$, are two random individuals in the last population $\mathbf{Pop}(k-1)$. To guarantee diversity of predictions,



u
1
$F(x_{(U),d})$
$u_d(x_{(s),d})$
$u_d(x_{(s\text{-}1),d})$
$u_d(x_{(i+1),d})$
u_d
$u_d(x_{(i),d})$
$u_d(x_{(2),d})$
$u_d(x_{(1),d})$
$F(x_{(L),d})$
0
x
$x_{(L),d}$ $x_{(1),d}$ $x_{(2),d}$ ⋯ $x_{(i),d}$ x_d $x_{(i+1),d}$ ⋯ $x_{(s\text{-}1),d}$ $x_{(s),d}$ $x_{(U),d}$

Fig. 4. Obtaining x_d from u_d using inverses of the distribution.

$\mathbf{\Lambda}$ is set to be a diagonal matrix. p_0 proportion of the diagonal elements of $\mathbf{\Lambda}$ are randomly generated from the uniform distribution $U[0,1]$, while all the other diagonal elements are the same with a value between 0 and 1. Each of NP predicted solutions needs $O(D)$ steps for generating D elements, so this is done in $O(D{*}NP)$ time.

3) Selection: In steps 3 and 11, the α proportion ($\alpha < 1$) of the best individuals is selected according to their objective function values. Hence, $s = \alpha{*}NP$ denotes the number of the selected individuals for modeling. In step 11 (iteration 2 onward), the best $\alpha{*}NP$ are selected among individuals in the population as well as the revised and predicted individuals.

4) Modeling (Steps 4 and 12): Modeling is designed to establish the probability distribution of optimal solution using the selected set of solutions. For all individuals $\mathbf{x_i} = (x_{i,1}, x_{i,2}, \ldots, x_{i,D})$, $i = 1, \ldots, s$, in $\mathbf{Pop}^{mod}$, we first sort their elements in each dimension d and denote the ith smallest element in dimension d as $x_{(i),d}$, $i = 1, \ldots, s$; $d = 1, \ldots, D$. With this information we then calculate $\mathbf{u} = (u_1, \ldots, u_d, \ldots, u_D)$ as follows. Fig. 4 shows the marginal distribution $F_d(x_d)$ being a mixture of empirical distribution and Gaussian distribution, which is a piecewise function. This model allows sample individuals to be outside the area of modeling individuals, which enhances the ability of exploration

$$\begin{cases} u_d\left(x_{(i),d}\right) = F\left(x_{(1),d}\right) + (i-1) * \left[F\left(x_{(s),d}\right) - F\left(x_{(1),d}\right)\right]/(s-1) \\ \qquad\qquad i = 1, \ldots, s \\ F\left(x_{(\cdot),d}\right) = P\left(x \le x_{(\cdot),d}, x \sim N\left(\bar{x}, \sigma^2\right)\right) \\ \qquad\qquad x_{(\cdot),d} \le x_{(1),d} \text{ or } x_{(\cdot),d} \ge x_{(s),d}. \end{cases} \tag{7}$$

Each selected individual corresponds to a point in u space. Meanwhile, we divide interval [0,1] into K partitions $S_1, \ldots, S_K$, and the length of each partition $\delta = 1/K$. So there are K^D subcubes. Then the copula density function is defined as

$$c(u_1, \ldots, u_D) = N_j/s/\delta^D, \quad j \in \{S_1, \ldots, S_K\}^D \tag{8}$$

where N_j denotes the number of points in certain subcube j. As remarked in [46], this step has a time complexity of O($D*s*\log s$).

5) Sampling (Steps 5 and 13): The distribution model in the last step is used to generate $(1 - \alpha)*\ NP$ individuals. This sampling operation of EDA here also serves as observation operation for filtering. The objective function value of a sampled individual, $f(\mathbf{x})$, can be viewed as a value of the observation variable $\mathbf{y}$ in Kalman filtering. In this article, the method of an interpolation is used as the inverses of the distribution to generate the individual $\mathbf{x} = (x_1, x_2, \ldots, x_D)$. We first generate a vector $\mathbf{u} = (u_1, u_2, \ldots, u_D)$. Here, u_1 is a random variate from the uniform distribution on interval [0,1], while u_d, $d = 2, \ldots, D$, is generated with the conditional distribution function $C_d(u_d|u_1, \ldots, u_{d-1})$ considering the relation $C_d(u_d|u_1, \ldots, u_{d-1}) = u_d^*$, where u_d^*, $d = 2, \ldots, D$, is a random number generated from the uniform distribution on (0,1]. For a generated u_d that falls in the interval between $u_d(x_{(i),d})$ and $u_d(x_{(i+1),d})$, the corresponding x_d is in the interval between $x_{(i),d}$ and $x_{(i+1),d}$, $i = 1, \ldots, s-1$, and can be calculated using interpolation as follows:

$$x_d = \begin{cases} x_{(L),d}, & \text{if } u_d \le F\left(x_{(L),d}\right) \\ x_{(L),d} + \left(x_{(1),d} - x_{(L),d}\right)\left(u_d - F\left(x_{(L),d}\right)\right)/ \\ \quad \left(u_d\left(x_{(1),d}\right) - F\left(x_{(L),d}\right)\right) \\ & \text{if } F\left(x_{(L),d}\right) < u_d \le u_d\left(x_{(1),d}\right) \\ x_{(i),d} + \left(x_{(i+1),d} - x_{(i),d}\right)\left(u_d - u_d\left(x_{(i),d}\right)\right)/ \\ \quad \left[F'/(s-1)\right] \\ & \text{if } u_d\left(x_{(i),d}\right) < u_d \le u_d\left(x_{(i+1),d}\right), \\ & i = 1, \ldots, s-1 \\ x_{(s),d} + \left(x_{(U),d} - x_{(s),d}\right)\left(u_d - u_d\left(x_{(s),d}\right)\right)/ \\ \quad \left(F\left(x_{(U),d}\right) - u_d\left(x_{(s),d}\right)\right) \\ & \text{if } u_d\left(x_{(s),d}\right) < u_d \le F\left(x_{(U),d}\right) \\ x_{(U),d}, & \text{if } F\left(x_{(U),d}\right) < u_d \end{cases}$$

where

$$F' = u_d\left(x_{(s),d}\right) - u_d\left(x_{(1),d}\right). \tag{9}$$

In this step, a random vector $\mathbf{x}$ is generated in $O(D{*}\log K)$ time. The sampled individuals are then combined with the modeling individuals to form a new population (steps 6 and 14).

6) Calculating the Gain Vector (Function GAIN(•) in Step 15): The gain vector $\mathbf{g}(k)$ is a D-dimensional vector with each element being the average error weight of an individual in a specific dimension relative to the error of the objective value. Using $\mathbf{P}(k-1)$, $g_d(k)$, $d = 1, \ldots, D$, is calculated as follows in (10), shown at the bottom of the next page, where $\mathbf{P}(k-1)$ is the error matrix, $f(\mathbf{z_i}(k-1))$ and $f(\mathbf{x_{pcbest}}(k-1))$ are the objective values of $\mathbf{z_i}(k-1)$ and $\mathbf{x_{pcbest}}(k-1)$, respectively. This step has a time complexity of $O(D)$.

7) Revision: Employing the error gain vector, the revised individuals can be obtained in step 16 (function REVISE(•)) using the following formulas. The meaning of the above-defined gain vector makes the revision operation linear

$$\widehat{\mathbf{x}}_\mathbf{i}(k) = \mathbf{x_i}(k) + \mathbf{g}(k)[f(\mathbf{z_{best}}(k)) - f(\mathbf{x_i}(k))] \tag{11}$$

$$\widehat{\mathbf{x}}_\mathbf{pcbest}(k) = \mathbf{x_{pcbest}}(k) + \mathbf{g}(k)\left[f(\mathbf{z_{best}}(k)) - f\left(\mathbf{x_{pcbest}}(k)\right)\right] \tag{12}$$

$$\widehat{\mathbf{z}}_\mathbf{i}(k) = \mathbf{z_i}(k) + \mathbf{g}(k)\left[f\left(\mathbf{x_{pcbest}}(k)\right) - f(\mathbf{z_i}(k))\right] \tag{13}$$

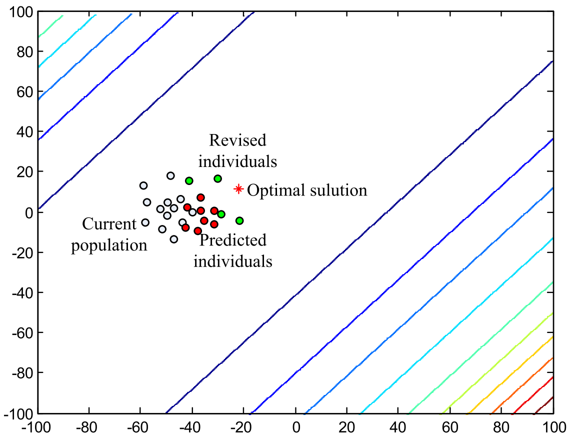


Fig. 5. Influence on modeling individuals by predicting and revising.

where $\mathbf{x_i}(k)$, $i = 1, \ldots, NP_1$, is the best NP_1 individuals among the predicted individuals and those in the population $\mathbf{Pop}(k-1)$. $\mathbf{z_i}(k)$, $i = 1, \ldots, NP_2$, is the best NP_2 individuals of the sampled individuals, $\mathbf{z_{best}}(k)$ is the best sampled individual. $NP_1 = p_1 * \alpha * NP$ and $NP_2 = p_2 * \alpha * NP$ are determined by selecting the values of parameters $p_1(<1)$ and $p_2(<1)$. $\widehat{\mathbf{x}}_\mathbf{i}(k)$, $\widehat{\mathbf{x}}_\mathbf{pcbest}(k)$ and $\widehat{\mathbf{z}}_\mathbf{i}(k)$ are the revised individuals of $\mathbf{x_i}(k)$, $\mathbf{x_{pcbest}}(k)$ and $\mathbf{z_i}(k)$, respectively. $\mathbf{x_{pcbest}}(k)$ is obtained in steps 3 and 10 of Algorithm 1. Formula (12) is a special case of (11). The calculations in (11) and (13) generate the revised individuals in $O(NP_1 * D + NP_2 * D)$ time.

8) Calculating Error Matrix: The error matrix is calculated in step 17 (function CAL(•)), using (5).

Overall, in the new EDA framework, we introduced two distinct operations "Predict" and "Revise" which do not exist in the traditional EDA. The potential advantage of this is illustrated in Fig. 5, where the red asterisk represents the optimal solution. As iteration goes on, the individuals in the population may tend to concentrate in a small area. In this case it is difficult for the traditional EDA to evolve further even though the optimal solution is still far away from this area. The prediction and revision operations in the new EDA generate new solutions. The predicted and revised individuals as illustrated in Fig. 5 not only tend to be closer to the optimal solution but also increase the diversity of the population and so makes it evolve further.

B. Learning Sampling Strategy

The process of EDA takes a long time to reach a good final solution if the sampling is completely random. To search the solution space more efficiently, we use the idea of optimal learning (see [47]) in the sampling process of each generation, i.e., after an initial set of individuals is sampled randomly, each new individual is sampled in a promising region which is determined using the existing individuals. Using learning techniques to improve optimization process becomes popular in recent years. These techniques often consider the special features of the optimization problem being solved. For example, in the process of solving a premarshaling problem in [48], a learning model is used to choose the next move. We use optimal learning to improve the sampling process of EDA as a general optimization method.

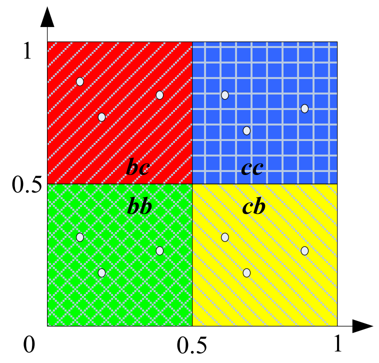


Fig. 6. Four selections of (u_1^*, u_2^*).

In the EDA based on copula function, because of the randomization of u_1 and u_d^*, $d = 2, \ldots, D$, we need sample large number of individuals to get a sufficiently good solution. Trying to improve this, we divide the whole sampling range of each dimension into multiple intervals, from which the vector $\mathbf{u}^* = (u_1^*, u_2^*, \ldots, u_D^*)$ can be generated, where u_1^* is defined as u_1 for convenience. Now we divide the range [0, 1] into M equal intervals. So the length L of each interval equals $1/M$. Then u_d^* can be generated from M alternative intervals $((j-1) \times L,\ j \times L]$, $j = 1, \ldots, M$. Note that the first interval for u_1 also includes 0. Consider a simple example with $D = 2$, $M = 2$, denote $b := (0, 0.5]$ and $c := (0.5, 1]$. So (u_1^*, u_2^*) can be obtained from four alternative regions: bb, bc, cb, and cc, as shown in Fig. 6. We can select each region for (u_1^*, u_2^*) and then sample individuals accordingly. Fig. 7 shows the individuals generated when (u_1^*, u_2^*) are sampled from different regions. The axes in Fig. 7 represent the decision variables for this example, x_1 and x_2. Suppose the objective function is $x_1^2 + x_2^2$, Fig. 7 shows that the individuals generated with (u_1^*, u_2^*) sampled from bc and cc are better than those with (u_1^*, u_2^*) from bb and cb. So it will be beneficial to find a learning strategy such that promising regions of u^* can be identified and individuals are generated accordingly.

Based on the above discussions, a learning sampling strategy is proposed to search for better individuals rapidly. Similar to statistical learning, this learning updates the distribution information of the objective value after taking a sample. It then uses the distribution information to select the best region to sample next. The concept of knowledge gradient (KG) introduced in [49] can be applied to mathematically identify which alternative should be sampled next. This is done by determining the region where sample is likely to improve the current best outcome. If the prior distributions of the objective value in the independent alternatives have the same variance and sample error but different means, then the alternative with

$$g_d(k) = \frac{\sum_{i=1}^{D} \left(P_{i,d}(k-1)\right)}{\sum_{d'=1}^{D} \sum_{i=1}^{D} |P_{i,d'}(k-1)| \sum_{i=1}^{D} \left(f(\mathbf{z_i}(k-1)) - f\left(\mathbf{x_{pcbest}}(k-1)\right)\right)/D} \tag{10}$$

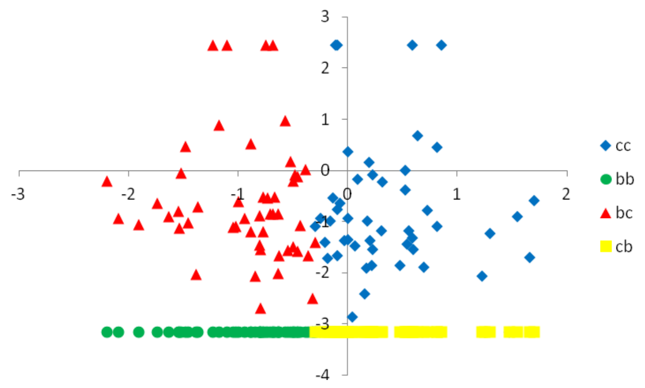


Fig. 7. Sampling individuals (x_1, x_2) in different regions.

Learning sampling strategy

```
Input: $D$; $NP$; $C_k(u)$; $\alpha$;
    $M$: number of equal intervals of range [0, 1];
    $\alpha_1$: sampling proportion after making a decision;
    $n_{max}$: maximum number of sampling decisions;
Output: the sampled population $\mathbf{Pop}^{sam}(k)$.
    /* Sample an initial set of individuals in generation $k$ of Algorithm 1 */
1:  $\mathbf{Pop}^{sam}(k)$ = SAMPLE((1-$\alpha$)*$NP$, $C_k(u)$);
    /* Denote the distribution information ($\mu^n_{u^*_{d,j}}$, $(\sigma^n_{u^*_{d,j}})^2$) of each $u^*_{d,j}$ in the $j^{th}$ interval of the $d^{th}$ dimension as DIST$_{d,j}$ */
2:  Initialize DIST$_{d,j}$ using $\mathbf{Pop}^{sam}(k)$;
3:  $n$ = 1; // $n$ is the iteration counter.
4:  While $n \le n_{max}$ do
5:      Make the $n^{th}$ sampling decision using DIST$_{d,j}$, which has complexity $O(D*M)$.
6:      $\mathbf{Pop}^{D}$ = SAMPLE($\alpha_1$*(1-$\alpha$)*$NP$, $C_k(u)$) using the $n^{th}$ sampling decision. Each individual is generated in O($D$*log$K$) time;
7:      Add $\mathbf{Pop}^{D}$ into $\mathbf{Pop}^{sam}(k)$;
8:      Calculate DIST$_{d,j}$ using $\mathbf{Pop}^{sam}(k)$. This is done in ($D$*$M$*$NP$/2) time.
9:      Set $n$ = $n$ + 1.
10: End while
11: Keep the best (1-$\alpha$)*$NP$ individuals in $\mathbf{Pop}^{sam}(k)$ and delete others.
```

Fig. 8. Pseudo-code of learning sampling strategy.

the best mean should be chosen to sample. But the choice should also depend on the variance and the alternatives with larger variance could be attractive choices as sampling there could get a better objective value. The KG gets its name because in a sense it points in the direction of maximal information by taking an expectation of the difference between the current mean and the mean after choosing to sample a certain alternative.

So the task of the learning sampling is to make a decision, which is to select an interval from M distinct intervals for each u^*_d, $d = 1, \ldots, D$. The learning sampling strategy can be outlined as Fig. 8, where α_1 is the sampling proportion after making a sampling decision; $n_{\max}$ is the maximum number of sampling decisions. The details are described as follows.

The first step of the learning sampling is to sample an initial set of individuals and calculate the prior distribution of objective function value according to the statistical information of the individuals located in the jth interval of the dth dimension. We assume the prior distribution is a normal distribution with mean $\mu^0_{u^*_{d,j}}$ and variance $(\sigma^0_{u^*_{d,j}})^2$ for each alternative $u^*_{d,j}$, $j = 1, \ldots, M$. In the Bayesian view we define $\beta^0_{u^*_{d,j}}$ as the precision of our belief, which is the inverse of the variance in our estimate $\mu^0_{u^*_{d,j}}$ of the true mean.

Starting from this initial information, we make a sequence of subsequent sampling decisions, $(u^*_{d,j})^1, (u^*_{d,j})^2, \ldots$ Consider the sampling decision $(u^*_{d,j})^n$, $n = 1, 2, \ldots$ Assume the sampling error of measuring alternative $u^*_{d,j}$ is $\varepsilon^n_{u^*_{d,j}} \sim N(0, 1/\beta^\varepsilon_{u^*_{d,j}})$, where $\beta^\varepsilon_{u^*_{d,j}}$ is precision of sample, and the resulting sample observation is

$$\hat{y}^n_{u^*_{d,j}} = \mu^{n-1}_{u^*_{d,j}} + \varepsilon^n_{u^*_{d,j}}. \tag{14}$$

Because decisions are made sequentially, so the mean and precision can be updated by $(u^*_{d,j})^1$, $\hat{y}^1_{u^*_{d,j}}, \ldots, (u^*_{d,j})^n$, $\hat{y}^n_{u^*_{d,j}}$. The updating formula is

$$\begin{cases} \mu^n_{u^*_{d,j}} = \dfrac{\beta^{n-1}_{u^*_{d,j}} \mu^{n-1}_{u^*_{d,j}} + \beta^\varepsilon_{u^*_{d,j}} \hat{y}^n_{u^*_{d,j}}}{\beta^{n-1}_{u^*_{d,j}} + \beta^\varepsilon_{u^*_{d,j}}} \\ \beta^n_{u^*_{d,j}} = \beta^{n-1}_{u^*_{d,j}} + \beta^\varepsilon_{u^*_{d,j}}. \end{cases} \tag{15}$$

The state of knowledge before the nth sampling is defined as

$$S^{n-1}_{d,j} = \left(\mu^{n-1}_{u^*_{d,j}}, \beta^{n-1}_{u^*_{d,j}}\right), \quad j = 1, \ldots, M. \tag{16}$$

If we stop measuring now, we would pick the mean of best interval

$$V\left(S^{n-1}_{d,j}\right) = \max_{u^*_{d,j}} \mu^{n-1}_{u^*_{d,j}}, \quad j = 1, \ldots, M. \tag{17}$$

Now if we choose measuring alternative $u^*_{d,j}$ as the decision, $(u^*_{d,j})^n$, the next state will be $S^n_{d,j} = S(S^{n-1}, (u^*_{d,j})^{n-1}, \hat{y}^n_{u^*_{d,j}})$. The best interval would be identified by

$$V\left(S^n_{d,j}\right) = \max_{u^*_{d,j}} \mu^n_{u^*_{d,j}}, \quad j = 1, \ldots, M. \tag{18}$$

We would like to choose $u^*_{d,j}$ to maximize the expected value of (18). We can also think of (18) as choosing $u^*_{d,j}$ to maximize the expected incremental value

$$v^{\mathrm{KG},n}_{u^*_{d,j}} = E\left[V\left(S^n_{d,j}\right)\right] - V\left(S^{n-1}_{d,j}\right). \tag{19}$$

Equation (19) is viewed as the KG.

The interval to be sampled at the nth sampling for each variable j can be determined by

$$\left(u^*_{d,j}\right)^n = \arg\max_j v^{\mathrm{KG},n}_{u^*_{d,j}}. \tag{20}$$

In our learning sampling strategy, we start by computing the change in the variance of our estimate of $\mu^n_{u^*_{d,j}}$ given our state of knowledge $S^n_{d,j}$

$$\begin{aligned} \left(\tilde{\sigma}^n_{u^*_{d,j}}\right)^2 &= \mathrm{Var}\left[\mu^n_{u^*_{d,j}} - \mu^{n-1}_{u^*_{d,j}}\right] \\ &= \left(\sigma^{n-1}_{u^*_{d,j}}\right)^2 - \left(\sigma^n_{u^*_{d,j}}\right)^2 \\ &= \frac{\left(\sigma^{n-1}_{u^*_{d,j}}\right)^2}{1 + \left(1/\beta^\varepsilon_{u^*_{d,j}}\right)^2 \Big/ \left(\sigma^{n-1}_{u^*_{d,j}}\right)^2}. \end{aligned} \tag{21}$$

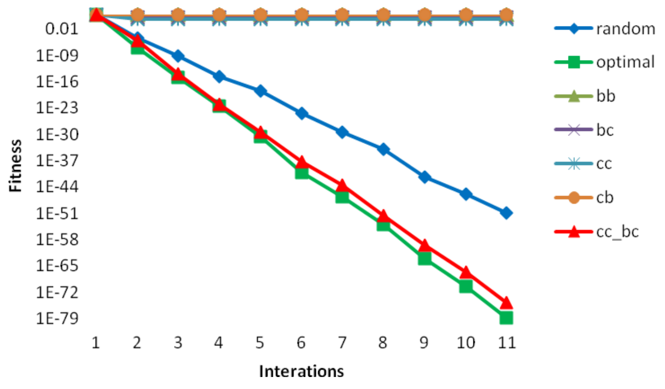


Fig. 9. Process of iterations in different regions on $x_1^2 + x_2^2$.

We then compute normalized influence of decision $(u_{d,j}^*)^n$, calculated by current estimate of the value of $\mu_{u_{d,j}^*}^n$, and the best of the rest, and the number of standard deviations, given by

$$\zeta_{u_{d,j}^*}^n = -\left|\left(\mu_{u_{d,j}^*}^{n-1} - \max_{u_{d,j}^{*\prime} \neq u_{d,j}^*} \mu_{u_{d,j}^{*\prime}}^{n-1}\right) \Big/ \tilde{\sigma}_{u_{d,j}^*}^n\right|. \tag{22}$$

Next, we use a standard formula given by

$$f(\zeta) = \zeta\Phi(\zeta) + \varphi(\zeta) \tag{23}$$

where $\Phi(\zeta)$ and $\varphi(\zeta)$ are the cumulative probability function and the density function, respectively, of the standard normal distribution. Finally, the KG is given by

$$\upsilon_{u_{d,j}^*}^{\mathrm{KG},n} = \left(\tilde{\sigma}_{u_{d,j}^*}^n\right)^2 f\left(\zeta_{u_{d,j}^*}^n\right). \tag{24}$$

Then we can make the sampling decision according to (20).

As Fig. 9 shows, the solution obtained by learning sampling strategy is better than other methods with the same number of iterations.

For the complexity, we remark that the learning sampling strategy has $O(n_{\max}D{\cdot}M{\cdot}NP/2) + O(D{\cdot}\log K\ (1-\alpha)NP) + O(n_{\max}D{\cdot}M)$ steps. Note that M, K, and $n_{\max}$ are small constants. So the time complexity is $O(cD{\cdot}NP)$, here c is a small positive integer.

IV. Experimental Studies

In order to test the performance of the proposed algorithm, a comprehensive series of benchmark functions of 10/30/50/100/500 dimensions are used. The algorithms are developed using C# language on the platform of Microsoft Visual Studio 2005. All experiments are carried out on a Lenovo personal computer with 3.30-GHz Intel CPU and 4-GB RAM.

A. Test Problems

1) CEC 2013 Benchmark Suite: The 28 benchmark functions in the CEC 2013 benchmark suite are utilized to evaluate the algorithm performance, among which F_1 to F_5 are unimodal functions; F_6 to F_{20} are basic multimodal functions; F_{21} to F_{28} are composition functions. The problem definitions and evaluation criteria can be obtained from [50].

TABLE I
Parameters of the Proposed Algorithm

Parameter	Value
NP	400
α	0.6
p_0	0.1
K	2
M	2
α_1	0.1
n_{max}	1
$\beta_{u_{d,j}^*}^{\varepsilon}$	0.1

2) Scaling-Up Test Functions: To further test the performance and the universality of our proposed algorithm, the 13 classical functions are selected from [13] and used. They are presented in Table IV in supplementary material, which can be found in the supplementary file of this article. These are minimization problems and can be classified into three groups as follows. F_{29} and F_{30} are separable unimodal problems; F_{31} to F_{37} belong to nonseparable problems with only a few local optima; F_{38} to F_{41} are multimodal problems with many local optima. F_{30}, F_{32}, F_{34}, F_{36}, and F_{39} are the shifted or rotated function of F_{29}, F_{31}, F_{33}, F_{35}, and F_{38}, respectively, and the values of the shifted global optima and the transformation matrices can be obtained in [51]. More details about these functions can be found in [51] and [52].

B. Comparison Strategies and Metrics

For each algorithm in this article, we use x^* and x' to indicate the known optimum solution and the best solution obtained when the stopping criterion is satisfied, respectively. Therefore, the function error value $(F(x') - F(x^*))$ is employed to evaluate the algorithms' performance, and if the value is below 1e-8, we see it as zero. The mean and variance are calculated through many independent runs for CEC 2013 benchmark functions and the scaling-up test functions. In each run the algorithms are executed for dimension × 10 000 FEs [51] and dimension × 11 000 FEs [13] on the two classes of problems, respectively. This setting is the same for all algorithms tested. The mean error and the corresponding standard deviation are calculated and presented in the numerical results tables, where the best mean error value is shown in bold for each benchmark function. The standard deviation appears after the "±" sign. Because of the space limitation, some tables are placed in the supplementary file of this article.

To get a direct indication of the performance of our proposed algorithm, the number of the best results in each test group are listed at the bottom of the numerical results tables.

To obtain a statistical conclusion, the two-sided Wilcoxon rank sum test [53] is employed to evaluate the performance between EDA-FL and other algorithms. The value of significance level is set to be 0.05. The marks "+," "−," and "=" indicate that the compared algorithm is superior to EDA-FL, EDA-FL performs better, and there is no statistically significant difference between EDA-FL and compared algorithm, respectively.

TABLE II
MEAN AND STANDARD DEVIATIONS OF BENCHMARK TEST FUNCTIONS FOR PARAMETER ANALYSIS

Pro.	D	EDA-F (p_1=0, p_2=0) Mean/Std	EDA-F (best prediction) Mean/Std	EDA-F (p_1=0.05,p_2=0.05) Mean/Std	EDA-F (p_1=0.1, p_2=0) Mean/Std	EDA-F (p_1=0.2, p_2=0.4) Mean/Std
	10	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**
F_1	30	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**
	50	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**
	10	1.14e-01±1.63e-01	3.96e-02±3.70e-02	8.88e-05±1.12e-04	**3.61e-05±5.93e-05**	3.53e+00±9.20e+00
F_2	30	4.14e+00±2.18e+00	**1.47e+00±1.32e+00**	7.72e+00±3.25e+01	1.50e+00±3.06e+00	1.46e+01±3.26e+01
	50	1.18e+01±5.82e+00	9.15e+00±8.50e+00	**2.76e+00±4.55e+00**	6.42e+00±8.29e+00	7.88e+01±1.10e+02
	10	1.42e+00±2.49e-01	**3.47e-02±2.34e-02**	1.54e-01±1.66e-01	1.10e-01±7.80e-02	1.95e-01±1.25e-01
F_{16}	30	2.89e+00±5.98e-01	**3.81e-01±2.00e-01**	5.44e-01±3.00e-01	5.94e-01±2.89e-01	5.78e-01±3.81e-01
	50	3.66e+00±8.42e-01	**6.80e-01±3.77e-01**	8.30e-01±3.41e-01	9.60e-01±4.39e-01	1.02e+00±3.99e-01
	10	**9.29e+00±3.07e+00**	1.03e+01±1.38e-01	9.30e+00±3.08e+00	9.79e+00±2.21e+00	1.03e+01±2.30e+00
F_{17}	30	3.12e+01±4.32e-01	**3.11e+01±3.04e-01**	**3.11e+01±4.13e-01**	3.12e+01±6.61e-01	3.15e+01±4.62e-01
	50	5.23e+01±1.45e+00	**5.21e+01±1.55e+00**	5.26e+01±1.47e+00	5.29e+01±1.76e+00	5.29e+01±7.51e-01
	10	3.93e-01±1.31e-01	4.40e-01±1.31e-01	**3.66e-01±1.30e-01**	4.61e-01±1.14e-01	4.81e-01±1.75e-01
F_{19}	30	**1.09e+00±2.41e-01**	1.13e+00±2.65e-01	1.17e+00±2.58e-01	1.31e+00±2.00e-01	1.48e+00±3.27e-01
	50	2.15e+00±4.69e-01	**1.88e+00±3.68e-01**	2.20e+00±4.98e-01	2.47e+00±5.31e-01	2.60e+00±4.76e-01
	10	**1.91e+02±2.74e+01**	1.96e+02±1.76e+01	1.96e+02±1.81e+01	1.92e+02±2.47e+01	1.96e+02±1.68e+01
F_{26}	30	**2.00e+02±3.10e-07**	**2.00e+02±3.25e-07**	**2.00e+02±5.63e-06**	**2.00e+02±4.57e-06**	**2.00e+02±2.49e-06**
	50	**2.00e+02±1.78e-06**	**2.00e+02±1.30e-06**	**2.00e+02±7.04e-06**	**2.00e+02±1.30e-05**	**2.00e+02±1.56e-05**
	10	2.70e+02±7.33e+01	**2.68e+02 ± 7.48e+01**	2.80e+02±6.16e+01	2.90e+02±4.47e+01	2.80e+02±6.16e+01
F_{28}	30	**3.00e+02±5.29e-13**	**3.00e+02±5.68e-13**	**3.00e+02±4.42e-08**	**3.00e+02±2.19e-08**	**3.00e+02±6.17e-03**
	50	5.50e+02±6.71e+02	5.48e+02±6.64e+02	**4.00e+02±7.47e-11**	5.52e+02±6.81e+02	5.55e+02±6.92e+02
No. of Best		9	14	10	7	6

TABLE III
MEAN AND STANDARD DEVIATIONS OF RESULTS ON 10-D BENCHMARK TEST FUNCTIONS

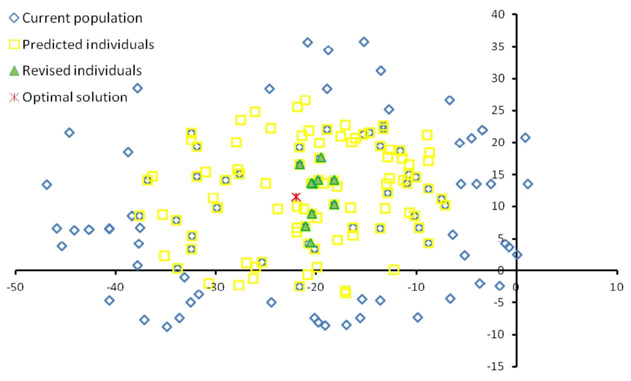
Pro. (10-D)	EDA Mean/Std	EDA-F Mean/Std	EDA-FL Mean/Std
F_1	5.21e-01±8.12e-01	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**
F_2	8.03e+03 ±2.72e+03	3.96e-02±3.70e-02	**3.87e-04±2.82e-04**
F_3	1.27e+08 ±1.65e+08	1.17e+01±1.35e+01	**1.26e+00±2.31e+00**
F_4	7.74e+03 ±2.92e+03	2.51e-02±1.90e-02	**3.24e-04±2.47e-04**
F_5	2.22e+00 ±3.93e+00	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**
F_6	1.84e+01 ±7.93e+00	4.71e+00±5.00e+00	**4.32e+00±4.97e+00**
F_7	1.28e+01 ±5.74e+00	4.90e-02±1.88e-01	**1.78e-03±8.74e-03**
F_8	2.05e+01 ±0.00e+00	**2.03e+01±0.00e+00**	**2.03e+01±0.00e+00**
F_9	3.05e+00 ±1.06e+00	**1.87e+00±1.28e+00**	1.91e+00±1.54e+00
F_{10}	6.05e+00 ±4.86e+00	2.24e-01±1.35e-01	**9.39e-02±7.56e-02**
F_{11}	7.29e-01 ±7.64e-01	**0.00e+00±0.00e+00**	**0.00e+00±0.00e+00**
F_{12}	**1.05e+01 ±6.54e+00**	1.08e+01±6.02e+00	1.08e+01±4.42e+00
F_{13}	**1.35e+01 ±7.56e+00**	1.66e+01±7.64e+00	1.47e+01±7.56e+00
F_{14}	2.73e+00 ±3.05e+00	**8.40e-01±1.41e+00**	1.43e+00±1.88e+00
F_{15}	1.23e+03 ±1.64e+02	**8.48e+02±2.68e+02**	9.28e+02±3.06e+02
F_{16}	1.14e+00 ±1.83e-01	3.47e-02±2.34e-02	**1.42e-02±1.36e-02**
F_{17}	1.11e+01 ±1.65e+00	1.03e+01±2.40e-01	**9.31e+00±3.31e+00**
F_{18}	2.96e+01 ±3.85e+00	2.07e+01±4.87e+00	**1.95e+01±5.56e+00**
F_{19}	5.83e-01 ±1.96e-01	**4.40e-01±1.31e-01**	5.12e-01±1.77e-01
F_{20}	2.89e+00 ±4.80e-01	2.81e+00±6.01e-01	**2.56e+00±4.88e-01**
F_{21}	4.00e+02 ±3.73e-01	4.00e+02±2.28e-13	**3.92e+02±4.00e+01**
F_{22}	**3.68e+01 ±3.30e+01**	4.52e+01±5.44e+01	4.09e+01±4.99e+01
F_{23}	9.81e+02 ±2.65e+02	9.41e+02±2.82e+02	**9.18e+02±3.36e+02**
F_{24}	2.11e+02 ±3.50e+00	2.02e+02±3.65e+00	**2.01e+02±3.42e+00**
F_{25}	2.08e+02 ±4.43e+00	**1.93e+02±2.44e+01**	1.94e+02±2.23e+01
F_{26}	**1.91e+02 ±2.70e+01**	1.96e+02±1.76e+01	1.93e+02± 2.35e+01
F_{27}	3.55e+02 ±3.46e+01	3.00e+02±4.83e-03	**3.00e+02±9.53e-05**
F_{28}	3.23e+02 ±2.41e+01	**2.68e+02±7.48e+01**	2.92e+02±4.00e+01
No. of Best	4	10	18

Moreover, to evaluate the performance of the convergence, the convergence curves about the mean error values of the best solutions for each algorithm are presented.

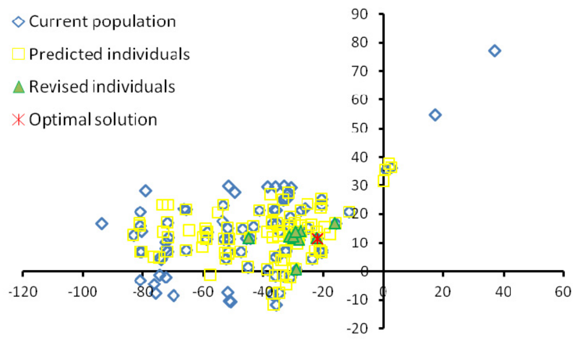
To have a reliable and fair comparison, the experimental results of all competitor algorithms are the reported data from the original articles or the websites of authors.

C. Parameter Analytics

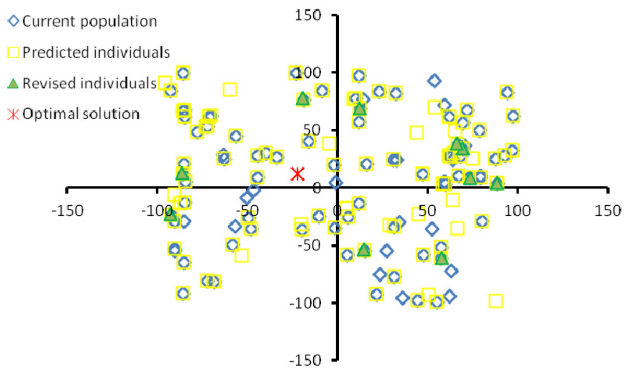
In our proposed algorithm with filtering framework only (EDA-F), the step of prediction has great influence, where $\mathbf{\Lambda}$ is a diagonal matrix with $D \times D$ dimension. The diagonal elements represent the noise level. To increase the randomness of noise, we set p_0 is 0.1. The constant values of the diagonal



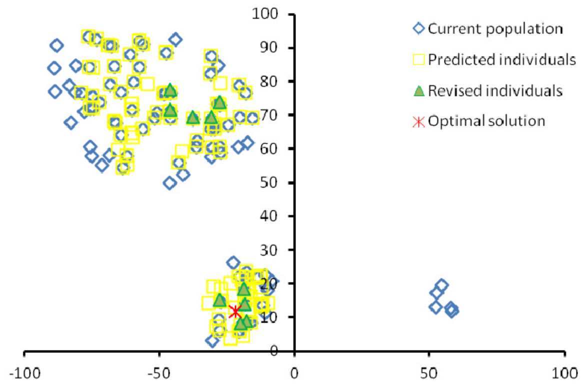
(*a*)



(*b*)

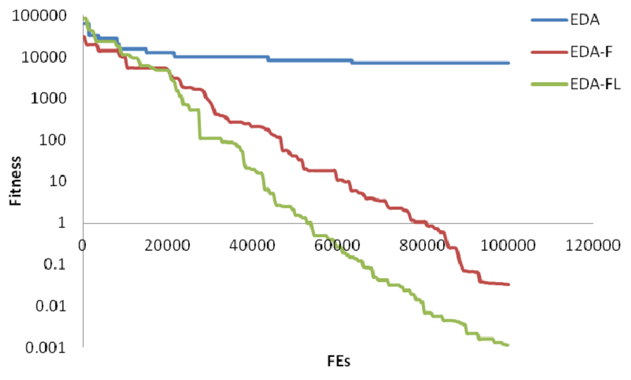


(*c*)

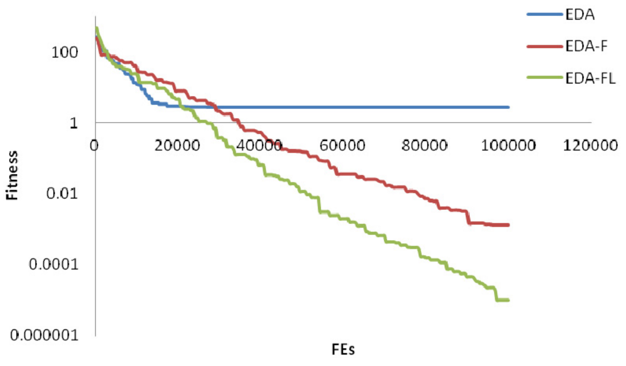


(*d*)

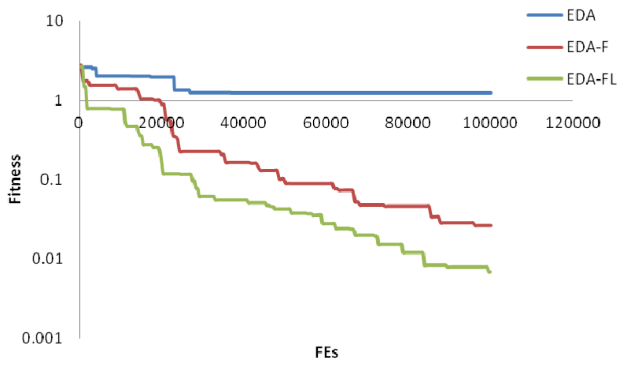
Fig. 10. Influence on modeling individuals by predicting and revising on the function (a) F_2, (b) F_7, (c) F_{16}, and (d) F_{28}.



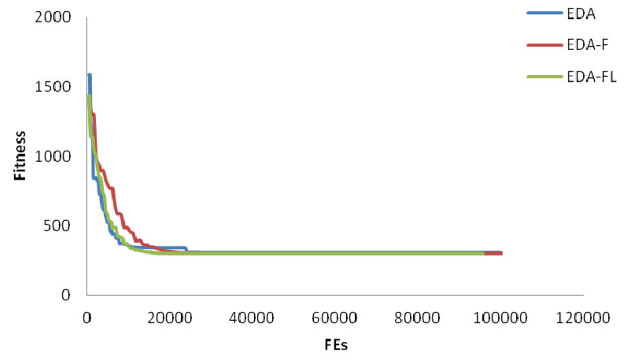
(a)



(b)



(c)



(d)

Fig. 11. Convergence sketches of the function (a) F_2, (b) F_7, (c) F_{16}, and (d) F_{28}.

elements are set as 0.9. Similarly, the step of revision plays an important role in our algorithm. The performance has great differences for different combinations of p_1 and p_2 values. To obtain the best combination, we test some representative functions with different p_1 and p_2. The other parameters for the algorithm are given in Table I. The means and standard

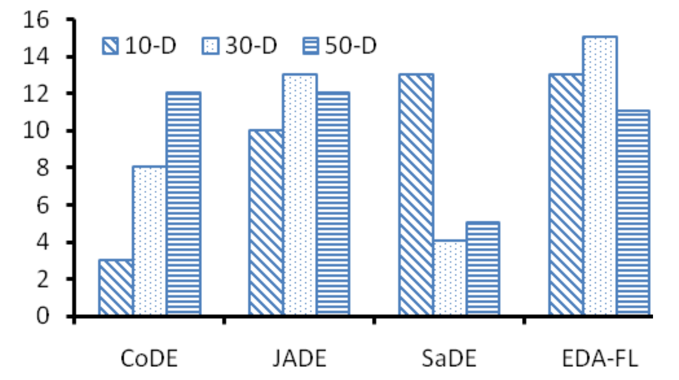


Fig. 12. Number of cases on which each algorithm performs the best in the comparison between EDA-FL and state-of-the-art DE algorithms.

deviations for ten runs are listed in Table II. As Table II shows, when the best prediction individual is revised through (12), the algorithm achieves the best solutions in 14 cases for 10/30/50 dimension. Regarding the total number of the best solutions obtained by the algorithm with other parameter settings, ($p_1 = 0$, $p_2 = 0$), ($p_1 = 0.05$, $p_2 = 0.05$), ($p_1 = 0.1$, $p_2 = 0$), and ($p_1 = 0.2$, $p_2 = 0.4$), are 9, 10, 7, and 6, respectively.

Result Analysis: When $p_1 = 0$ and $p_2 = 0$ indicating that there are no revised individuals going into modeling individuals set, the performance is not good. This proves that the operation "Revise" in filtering is effective to improve the performance. When the values of p_1 and p_2 increase to large values, the performance shows a decreasing trend. The root cause of this is that the revised individuals can increase the diversity of the modeling population. When the individuals are revised too much, the precision of distribution will decrease.

D. Experimental Results

1) Experiment Series 1 (Evaluation on the Strategies): In our proposed algorithm, there are two innovation points: a new framework with filtering and a strategy of learning sampling. To evaluate the performance of them, three versions of the algorithm, EDA, EDA-F, and EDA-FL, are tested on CEC 2013 benchmark functions with 10-D. The parameters are set as in Table I. The means and standard deviations for ten runs are presented in Table III. The best result (with the minimal mean value) is bolded in each row. The results indicate that EDA-F is better than EDA and EDA-FL is superior to EDA-F according to the number of best solutions. To further demonstrate the performance and mechanism of filtering, a class of experiments is executed to display the locations of current population, predicted individuals, revised individuals, and optimal solution in certain iteration. Fig. 10 shows the influence on modeling individuals by predicting and revising on 2-D function F_2, F_7, F_{16}, and F_{28}. The red asterisk denotes the optimal solution. The yellow square box and green triangle indicate the predicted individuals and revised individuals, which tend to be closer to the optimal solution. The convergence sketches on 10-D function F_2, F_7, F_{16}, and F_{28} are presented in Fig. 11, where we can see that EDA-FL has a quick convergence speed.

Result Analysis: In our new EDA, the potential advantage of the two distinct operations "Predict" and "Revise" is illustrated

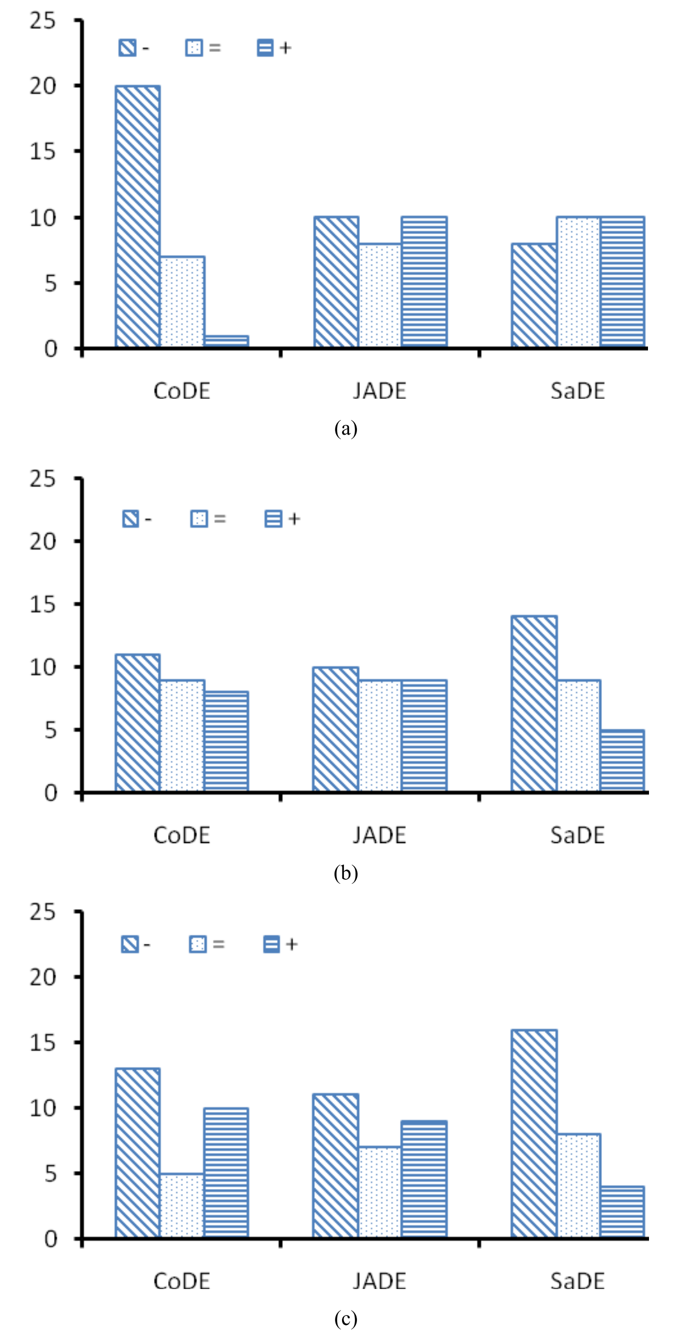


Fig. 13. Statistical significance results of comparison between EDA-FL and other EDAs on (a) 10-D, (b) 30-D, and (c) 50-D problems, respectively.

in Fig. 10. The predicted and revised individuals not only tend to be closer to the optimal solution but also increase the diversity of the population and so make it evolve further. Moreover, when adding the learning sampling strategy, our algorithm converges faster. This point is consistent with the idea of getting a good sample through learning.

2) Experiment Series 2 (Comparison of EDA-FL With Other State-of-the-Art Evolutionary Algorithms for CEC 2013 Benchmark Suite With 10/30/50 Dimension): To embody the competitiveness of our proposed algorithm, in this section, we compare the proposed EDA-FL with three DE algorithms that have outstanding performance, among the state-of-the-art intelligent algorithms, including composite DE (CoDE)

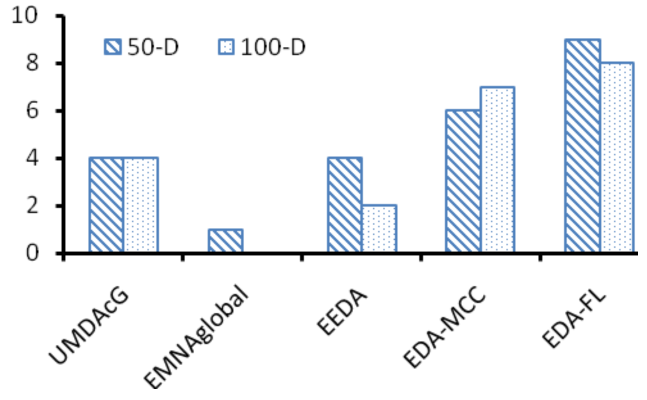


Fig. 14. Number of cases on which each algorithm performs the best in the comparison between EDA-FL and other EDAs.

[54], Adaptive DE with optional external archive algorithm (JADE) [55], and DE algorithm with Strategy Adaptation (SaDE) [56]. The parameters are set as in Table I. The means and standard deviations for ten runs are presented in Tables V–VII, which can be found in the supplementary file of this article. The results of CoDE, JADE, and SaDE are from [53]. Fig. 12 shows the number of cases on which each algorithm performs the best in the comparison between EDA-FL and state-of-the-art DE algorithms. To show the performance more clearly, the statistical significance results are summarized in Fig. 13.

Result Analysis: With great exploitation ability, our proposed algorithm has great advantages on the unimodal function F_1–F_5. Besides, when the dimension increases to 50, the algorithm outperforms all other three algorithms, showing that our proposed algorithm EDA-FL has strong competitiveness.

3) Experiment Series 3 (Comparison With Other EDAs and Further Algorithms for the Scaling-Up Benchmark Functions With 50/100/500 Dimension): Four algorithms are involved in experimental comparisons for 50-D/100-D/500-D: UMDA$_c^G$ [6], EMNA$_{\text{global}}$ [6], EEDA [7], and EDA-MCC [13]. UMDA$_c^G$ is a univariate Gaussian EDA, and EMNA$_{\text{global}}$ is multivariate Gaussian EDA. Because the two EDAs have been studied theoretically and experimentally and applied in real-world, it makes sense comparing them with our proposed algorithm. EDA-MCC has been proposed to solve the large-scale optimization problem, so it is significant to be compared. Two other algorithms, Mutual Information Maximization for Input Clustering to continuous domains where the underlying probability model for every pair of variables is assumed to be a bivariate Gaussian (MIMIC$_c^G$) and separable Covariance Matrix Adaptation Evolution Strategy (sep-CMA-ES), are included in the comparison for 500-D, because they have good performance for high dimension problems [13]. The means and standard deviations for ten runs are presented in Tables VIII–XI, which can be found in the supplementary file of this article. The best result (with the minimal mean value) is bolded in each row. Fig. 14 shows the number of cases on which each algorithm performs the best in the comparison between EDA-FL and other EDAs. And the statistical significance results are summarized in Fig. 15.

Result Analysis: With great exploitation ability, our proposed algorithm has great advantages on the unimodal function

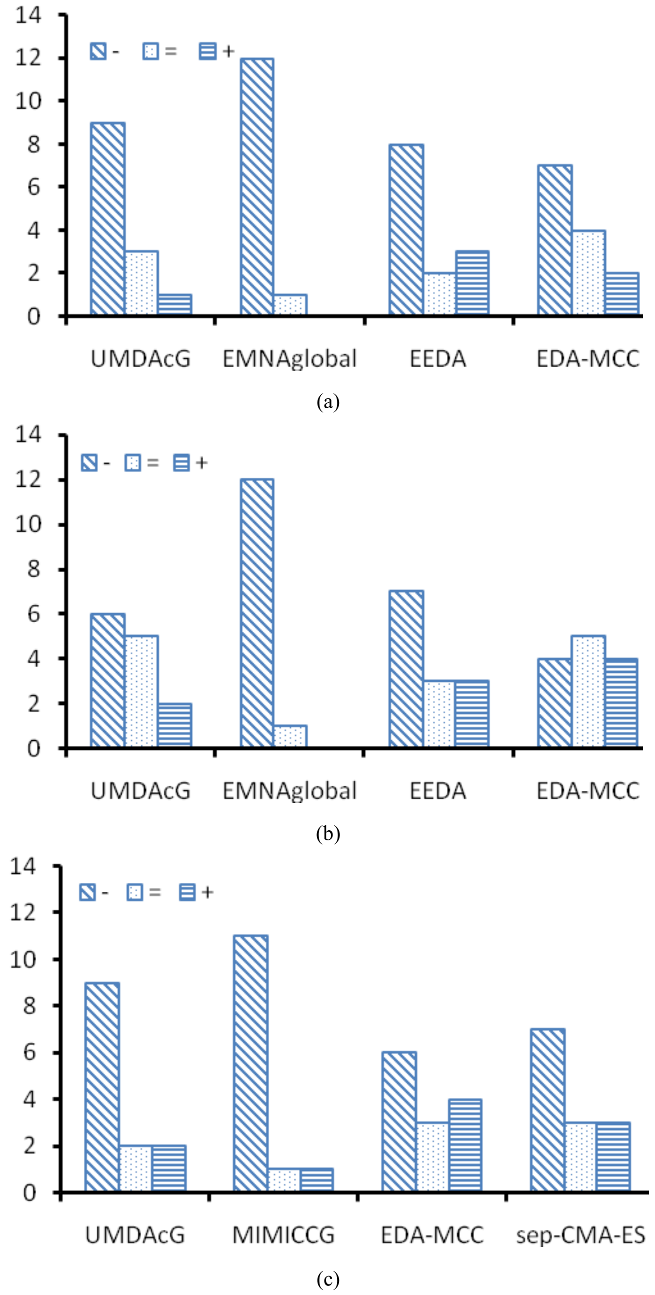


Fig. 15. Statistical significance results of comparison between EDA-FL and other EDAs on (a) 50-D, (b) 100-D, and (c) 500-D problems, respectively.

F_1–F_5. Again, the algorithm gives clearly better results than the other algorithms in high dimension (500-D). The results indicate that our proposed algorithm is effective.

V. Conclusion

In this article, we have proposed a new EDA-FL to solve nonlinear continuous optimization problem. First, we propose a new EDA framework incorporating filtering where the idea of Kalman filtering is adopted to modify some individuals, which in turn affect the model, using an estimation error matrix from historic information. Then, a sampling strategy with learning is proposed. Using this strategy, the decision on where to sample is made according to the statistical information about the objective values of sampled individuals to avoid completely random sampling. Computational experiments on a collection of general benchmark functions with different dimensions demonstrate that the EDA-FL is effective.

Given the promising results of introducing the filtering and learning mechanisms in EDA, further research could be done to improve the filtering operations and the learning strategy. Another direction could be to apply the new EDA to problems in practice and problems with discrete variables.

References

[1] L. X. Tang, C. Liu, J. Y. Liu, and X. P. Wang, "An estimation of distribution algorithm with resampling and local improvement for an operation optimization problem in steelmaking process," *IEEE Trans. Syst., Man., Cybern. Syst.*, to be published, doi: 10.1109/TSMC.2019.2962880.

[2] C. Liu, L. Tang, J. Liu, and Z. Tang, "A dynamic analytics method based on multistage modeling for a BOF steelmaking process," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1097–1109, Jul. 2019.

[3] C. Liu, L. Tang, and J. Liu, "A stacked autoencoder with sparse Bayesian regression for end-point prediction problems in steelmaking process," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 2, pp. 550–561, Apr. 2020.

[4] L. X. Tang and Y. Meng, "Data analytics and optimization for smart industry," *Frontier Eng. Manage.*, 2020, doi: 10.1007/s42524-020-0126-0.

[5] H. Mühlenbein and G. Paaß, "From recombination of genes to the estimation of distributions I. Binary parameters," *Parallel Problem Solving From Nature—PPSN IV* (Lecture Notes in Computer Science), vol. 1411. Berlin, Germany: Springer-Verlag, 1996, pp. 178–187.

[6] P. Larrñaga and J. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Boston, MA, USA: Kluwer Academic, 2002.

[7] M. Wagner, A. Auger, and M. Schoenauer, "EEDA: A new robust estimation of distribution algorithm," INRIA, Rapport de Recherche, Rocquencourt, France, Tech. Rep. RR-5190, 2004.

[8] M. Gallagher, M. Frean, and T. Downs, "Real-valued evolutionary optimization using a flexible probability density estimator," in *Proc. GECCO*, 1999, pp. 840–846.

[9] P. Bosman and D. Thierens, "Advancing continuous IDEAs with mixture distributions and factorization selection metrics," in *Proc. Optim. Building Using Probabilistic Models OBUPM Workshop (GECCO)*, 2001, pp. 208–212.

[10] Q. Lu and X. Yao, "Clustering and learning Gaussian distribution for continuous optimization," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 35, no. 2, pp. 195–204, May 2005.

[11] Y. Liang, Z. Ren, X. Yao, Z. Feng, A. Chen, and W. Guo, "Enhancing Gaussian estimation of distribution algorithm by exploiting evolution direction with archive," *IEEE Trans. Cybern.*, vol. 50, no. 1, pp. 140–152, Jan. 2020.

[12] J. Sun, Q. Zhang, and E. Tsang, "DE/EDA: A new evolutionary algorithm for global optimization," *Inf. Sci.*, vol. 169, nos. 3–4, pp. 249–262, Feb. 2005.

[13] W. Dong, T. Chen, P. Tino, and X. Yao, "Scaling up estimation of distribution algorithms for continuous optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 6, pp. 797–822, Dec. 2013.

[14] S. Tsutsui, M. Pelikan, and D. Goldberg, "Evolutionary algorithm using marginal histogram models in continuous domain," in *Proc. Optim. Building Using Probabilistic Models OBUPM Workshop (GECCO)*, 2001, pp. 230–233.

[15] B. Yuan and M. Gallagher, "Playing in continuous spaces: Some analysis and extension of population-based incremental learning," in *Proc. IEEE Congr. Evol. Comput.*, vol. 1, Dec. 2003, pp. 443–450.

[16] N. Ding, S. Zhou, and Z. Sun, "Optimizing continuous problems using estimation of distribution algorithm based on histogram model," in *Proc. 6th Conf. Simulation Evol. Learn.*, 2006, pp. 545–552.

[17] N. Ding, S. Zhou, H. Zhang, and Z. Sun, "Marginal probability distribution algorithm: A competent method for continuous optimization," *J. Comput. Sci. Technol.*, vol. 23, no. 1, pp. 35–43, 2008.

[18] A. Zhou, J. Y. Sun, and Q. F. Zhang, "An estimation of distribution algorithm with cheap and expensive local search methods," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 807–822, Dec. 2015.

[19] P. Bosman and D. Thierens, "Numerical optimization with real-valued estimation-of-distribution algorithms," in *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*, M. Pelikan, K. Sastry, E. Cantu-Paz, Eds. Berlin, Germany: Springer-Verlag, 2006, pp. 91–120.

[20] H. Mühlenbein, T. Mahnig, and A. Ochoa, "Schemata, distributions and graphical models in evolutionary optimization," *J. Heuristics*, vol. 5, no. 2, pp. 213–247, 1999.

[21] R. Etxeberria and P. Larrañaga, "Global optimization using Bayesian networks," in *Proc. 2nd Symp. Artif. Intell. (CIMAF)*, 1999, pp. 332–399.

[22] S. Shakya and J. McCall, "Optimization by estimation of distribution with DEUM framework based on Markov random fields," *Int. J. Auto. Comput.*, vol. 4, no. 3, pp. 262–272, 2007.

[23] M. Pelikan and D. E. Goldberg, "Genetic algorithms, clustering, and the breaking of symmetry," in *Proc. 6th Int. Conf. Parallel Problem Solving Nature-PPSN VI*, 2007, pp. 385–394.

[24] R. Santana, P. Larrañaga, and J. A. Lozano, "Mixtures of Kikuchi approximations," in *Proc. 17th Eur. Conf. Mach. Learn. (ECML)*, (Lecture Notes in Artificial Intelligence), vol. 4212, 2006, pp. 365–376.

[25] W. Shao, D. Pi, and Z. Shao, "A Pareto-based estimation of distribution algorithm for solving multiobjective distributed no-wait flow-shop scheduling problem with sequence-dependent setup time," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1344–1360, Jul. 2019.

[26] W. Shi, W.-N. Chen, Y. Lin, T. Gu, S. Kwong, and J. Zhang, "An adaptive estimation of distribution algorithm for multipolicy insurance investment planning," *IEEE Trans. Evol. Comput.*, vol. 23, no. 1, pp. 1–14, Feb. 2019.

[27] D. Pfeifer and J. Nešlehová, "Modeling dependence in finance and insurance: The copula approach," *Blätter DGVFM*, vol. 26, no. 4, p. 805, 2004.

[28] R. B. Nelsen, "An introduction to copulas," in *Lecture Notes Statistics*, vol. 139. New York, NY, USA: Springer, 1999.

[29] L. F. Wang, J. C. Zeng, and Y. Hong, "Estimation of distribution algorithm based on copula theory," in *Proc. IEEE Cong. Evol. Comput.*, 2009, pp. 1057–1063.

[30] Y. Gao, "Multivariate estimation of distribution algorithm with laplace transform Archimedean copula," in *Proc. Int. Conf. Inf. Eng. Comput. Sci.*, Dec. 2009, pp. 1–5.

[31] L. F. Wang, X. D. Guo, J. C. Zeng, and Y. Hong, "Using Gumbel copula and empirical marginal distribution in estimation of distribution algorithm," in *Proc. 3rd Int. Workshop Adv. Comput. Intell.*, 2010, pp. 583–587.

[32] A. Cuesta-Infante, R. Santana, J. I. Hidalgo, C. Bielza, and P. Larrañaga, "Bivariate empirical and n-variate Archimedean copulas in estimation of distribution algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2010, pp. 1–8.

[33] B. L. Ye, H. Gao, X. P. Wang, and J. C. Zeng, "Estimation of distribution algorithm based on nested Archimedean copulas constructed with Lévy subordinators," in *Proc. IEEE 11th Int. Conf. Comput.-Aided Ind. Design Concept. Design*, Nov. 2010, pp. 1586–1590.

[34] Y. Gao, X. Hu, and H. Liu, "Estimation of distribution algorithm based on multivariate Gaussian copulas," in *Proc. IEEE Int. Conf. Prog. Informat. Comput.*, Dec. 2010, pp. 254–257.

[35] X. Guo, L. Wang, J. Zeng, and X. Zhang, "Copula estimation of distribution algorithm with PMLE," in *Proc. 7th Int. Conf. Natural Comput.*, Jul. 2011, pp. 1077–1081.

[36] Y. Gao, L. Peng, F. Li, M. Liu, and X. Hu, "Pareto-based multi-objective estimation of distribution algorithm with Gaussian copulas and application in RFID network planning," in *Proc. IEEE 5th Int. Conf. Adv. Comput. Intell. (ICACI)*, Oct. 2012, pp. 370–373.

[37] H. D. de Mello, Jr., A. V. A. da Cruz, and M. M. B. R. Vellasco, "Estimation of distribution algorithm based on a multivariate extension of the Archimedean copula," in *Proc. BRICS Cong. Comput. Intell. 11th Brazilian Cong. Comput. Intell.*, 2013, pp. 75–80.

[38] H. D. Xu, M. Y. Jiang, and K. Xu, "Archimedean copula estimation of distribution algorithm based on artificial bee colony algorithm," *J. Syst. Eng. Electron.*, vol. 26, no. 2, pp. 388–396, 2015.

[39] M. Hyrš and J. Schwarz, "Advanced parallel copula based EDA," in *Proc. IEEE Symp. Series Comput. Intell.*, 2016, pp. 1–8.

[40] H. D. de Mello, Jr., L. Martí, A. V. A. da Cruz, and M. M. B. R. Vellasco, "Evolutionary algorithms and elliptical copulas applied to continuous optimization problems," *Inf. Sci.*, vol. 369, pp. 419–440, Nov. 2016.

[41] S. Gao and C. W. de Silva, "Estimation distribution algorithms on constrained optimization problems," *Appl. Math. Comput.*, vol. 339, pp. 323–345, Dec. 2018.

[42] B. Qian, Z. C. Li, and R. Hu, "A copula-based hybrid estimation of distribution algorithm form-machine reentrant permutation flow-shop scheduling problem," *Appl. Soft Comput.*, vol. 61, pp. 921–934, Dec. 2017.

[43] C. Liu, H. Chen, R. Xu, and Y. Wang, "Minimizing the resource consumption of heterogeneous batch-processing machines using a copula-based estimation of distribution algorithm," *Appl. Soft Comput.*, vol. 73, pp. 283–305, Dec. 2018.

[44] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, Mar. 1960.

[45] A. Muruganantham, K. C. Tan, and P. Vadakkepat, "Evolutionary dynamic multiobjective optimization via Kalman filter prediction," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2862–2873, Dec. 2016.

[46] J. C. Strelen and F. Nassaj, "Analysis and generation of random vectors with copulas," in *Proc. Winter Simulation Conf.*, Dec. 2007, pp. 488–496.

[47] W. B. Powell and P. Frazier, "Optimal learning," in *Proc. Tuts. Oper. Res.*, 2008, pp. 213–246.

[48] P. Ge, Y. Meng, J. Liu, L. Tang, and R. Zhao, "Logistics optimisation of slab pre-marshalling problem in steel industry," *Int. J. Prod. Res.*, vol. 58, no. 13, pp. 4050–4070, 2019, doi: 10.1080/00207543.2019.1641238.

[49] S. S. Gupta and K. J. Miescke, "Bayesian look ahead one-stage sampling allocations for selection of the best population," *J. Stat. Planning Inference*, vol. 54, no. 2, pp. 229–244, Sep. 1996.

[50] J. J. Liang, B. Y. Qu, P. N. Suganthan, and G. A. Hernández-Díaz. (2013). *Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization*. [Online]. Available: http://www.ntu.edu.sg/home/epnsugan/

[51] P. Suganthan *et al.*, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Kanpur Genet. Algorithms Lab., IIT Kanpur, Kanpur, India, Tech. Rep. 2005005, 2005. [Online]. Available: http://www.ntu.edu.sg/home/EPNSugan/

[52] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[53] L. Tang, Y. Dong, and J. Liu, "Differential evolution with an individual-dependent mechanism," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 560–574, Aug. 2015.

[54] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 55–66, Feb. 2011.

[55] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Enhancing differential evolution utilizing proximity-based mutation operators," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 99–119, Feb. 2011.

[56] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.

Lixin Tang (Senior Member, IEEE) received the B.Eng. degree in industrial automation, the M.Eng. degree in systems engineering, and the Ph.D. degree in control theory and application from Northeastern University, Shenyang, China, in 1988, 1991, 1996, respectively.

He is currently a Fellow of the Chinese Academy of Engineering, Beijing, China; the Director of Key Laboratory of Data Analytics and Optimization for Smart Industry, Ministry of Education, China; and the Head of Centre for Artificial Intelligence and Data Science, Northeastern University. His research articles have appeared in academic journals, such as *Operations Research*, *INFORMS Journal on Computing*, *IIE Transactions*, IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON POWER SYSTEMS, and IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY. His research interests cover industrial big data science, data analytics and machine learning, reinforcement learning and dynamic optimization, computational intelligent optimization, plant-wide production and logistics planning, production and logistics batching and scheduling and engineering applications in manufacturing (steel, petroleum-chemical, nonferrous), energy, resources industry, and logistics systems.

Dr. Tang serves as an Associate Editor for *IISE Transactions*, IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON CYBERNETICS, *Journal of Scheduling*, *International Journal of Production Research*, *Journal of the Operational Research Society*, on the Editorial Board for *Annals of Operations Research*, and an Area Editor for the *Asia-Pacific Journal of Operational Research*.

Xiangman Song is currently pursuing the Ph.D. degree in logistics optimization and control with the Institute of Industrial and Systems Engineering, Northeastern University, Shenyang, China.

His research interests include operations analytics and optimization for smart industry, optimal and predictive control, data analytics and machine learning, and computational intelligent optimization.

Jiyin Liu received the B.Eng. degree in industrial automation and the M.Eng. degree in systems engineering from Northeastern University, Shenyang, China, in 1982 and 1985, respectively, and the Ph.D. degree in manufacturing engineering and operations management from the University of Nottingham, Nottingham, U.K., in 1993.

He is currently a Professor of Operations Management with the School of Business and Economics, Loughborough University, Loughborough, U.K. He is also a Cheung Kong Scholars Visiting Chair Professor with the Institute of Industrial and Systems Engineering, Northeastern University. He has authored articles in journals such as the *European Journal of Operational Research*, IEEE Transactions, IIE Transactions, the *International Journal of Production Research*, *Naval Research Logistics*, *Operations Research*, and *Transportation Research*. His research interests are in operations planning and scheduling problems in production, logistics, and supply chains, and mathematical modeling, optimization, and heuristic methods.

Chang Liu (Member, IEEE) is currently pursuing the Ph.D. degree in systems engineering with the Institute of Industrial and Systems Engineering, Northeastern University, Shenyang, China.

He has authored several articles in journals such as the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, and *Neurocomputing*. His research interests include data analytics and optimization for smart industry, model predictive control, machine learning, computational intelligent optimization, and engineering applications in various industrial processes.