Scaling Up Estimation of Distribution Algorithms for Continuous Optimization

Weishan Dong, Tianshi Chen, Peter Tiňo, and Xin Yao, *Fellow, IEEE*

***Abstract*—Since estimation of distribution algorithms (EDAs) were proposed, many attempts have been made to improve EDAs' performance in the context of global optimization. So far, the studies or applications of multivariate probabilistic model-based EDAs in continuous domain are still mostly restricted to low-dimensional problems. Traditional EDAs have difficulties in solving higher dimensional problems because of the curse of dimensionality and rapidly increasing computational costs. However, scaling up continuous EDAs for large-scale optimization is still necessary, which is supported by the distinctive feature of EDAs: because a probabilistic model is explicitly estimated, from the learned model one can discover useful properties of the problem. Besides obtaining a good solution, understanding of the problem structure can be of great benefit, especially for black box optimization. We propose a novel EDA framework with model complexity control (EDA-MCC) to scale up continuous EDAs. By employing weakly dependent variable identification and subspace modeling, EDA-MCC shows significantly better performance than traditional EDAs on high-dimensional problems. Moreover, the computational cost and the requirement of large population sizes can be reduced in EDA-MCC. In addition to being able to find a good solution, EDA-MCC can also provide useful problem structure characterizations. EDA-MCC is the first successful instance of multivariate model-based EDAs that can be effectively applied to a general class of up to 500-D problems. It also outperforms some newly developed algorithms designed specifically for large-scale optimization. In order to understand the strengths and weaknesses of EDA-MCC, we have carried out extensive computational studies. Our results have revealed when EDA-MCC is likely to outperform others and on what kind of benchmark functions.**

***Index Terms*—Estimation of distribution algorithm, large-scale optimization, model complexity control.**

Manuscript received February 26, 2011; revised January 8, 2013; accepted February 8, 2013. Date of publication February 14, 2013; date of current version November 26, 2013. This work was supported in part by EPSRC under Grant EP/J017515/1 to X. Yao, by the National Natural Science Foundation of China under Grant 61100163 to T. Chen, and by the China Scholarship Council under a Scholarship to W. Dong to support his visit to the University of Birmingham, where part of this work was done. The work of X. Yao was also supported by the Royal Society Wolfson Research Merit Award. The work of P. Tiňo was supported by a BBSRC Grant BB/H012508/1.

W. Dong was with the Key Laboratory for Complex Systems and Intelligence Science, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China. He is now with IBM Research—China, Beijing 100193, China (e-mail: weishan.dong@gmail.com).

T. Chen is with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: chentianshi@ict.ac.cn).

P. Tiňo and X. Yao are with the Centre of Excellence for Research in Computational Intelligence and Applications, School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K. (e-mail: p.tino@cs.bham.ac.uk; x.yao@cs.bham.ac.uk).

Digital Object Identifier 10.1109/TEVC.2013.2247404

I. Introduction

ESTIMATION of distribution algorithms (EDAs) [1], [2] have been intensively studied in the context of global optimization. Compared with traditional evolutionary algorithms (EAs) such as genetic algorithms (GAs) [3], there is neither crossover nor mutation operator in EDA. Instead, EDA explicitly builds a probabilistic model of promising solutions in a search space. Then, new solutions are sampled from the model that presents extracted global statistical information from the search space. EDA uses the model as guidance of reproduction to find better solutions. Actually, any EA has an underlying probabilistic model explaining its reproduction behaviors. But in traditional EAs, the underlying model is usually implicitly expressed through evolutionary operators. Once the model is explicitly presented, the algorithm can then be classified as an instance of EDA. EDAs were proposed originally for combinatorial optimization. Research on EDAs has been extended from discrete domain to continuous optimization, and much progress has been made. In this paper, we focus on EDAs in a single objective continuous optimization domain.

Many studies on EDA have been done in the last decade. In general, so far there are two major branches of continuous EDAs. One is based on Gaussian distribution model, which is the most widely used and intensively studied [2], [4]–[11]. The other major branch is based on histogram models [6], [12]–[19]. However, most of the existing studies have the common problem that the performance of EDA is only validated on relatively low-dimensional problems (often much smaller than hundreds of variables). The performance of EDA on higher dimensional problems (e.g., 500-D) is rarely studied. On the other hand, large-scale optimization using other EAs has already become a hot topic in recent years [20]–[23].

As we can see in the following sections, the reason for this is not that researchers simply ignored EDA, but that continuous EDAs have difficulties in high-dimensional search space. Due to relying on learning a model from samples, EDAs heavily suffer from the well-known curse of dimensionality [24]. If considering multidependencies of variables to solve nonseparable problems more effectively, traditional EDAs' fast increasing computational costs also make them impractical to real-world applications. In this paper, we propose a novel EDA framework with model complexity control (MCC), named EDA-MCC, to scale up EDA for continuous optimization. By employing weakly dependent variable identification (WI)

and subspace modeling (SM) in EDA-MCC, we can explicitly control the model complexity to establish a tradeoff between: 1) the performance that may benefit from a complex model and 2) the computational and population complexities that grow rapidly with the problem size and the model complexity. By doing so, EDA-MCC can suffer less from the curse of dimensionality. Experimental comparisons on 13 well-known benchmark functions validate the effectiveness and efficiency of EDA-MCC. We find that EDA-MCC have significant advantages over traditional EDAs when solving large-scale nonseparable problems with few local optima (up to 500-D in experiments) in terms of solution quality and computational cost. The significant difference between EDA-MCC and traditional EDAs with model complexity penalization is also discussed. According to the No Free Lunch Theorem [25], the limitations of EDA-MCC are also analyzed.

If traditional EDAs are not appropriate for large-scale optimization, why do we still strive to scale it up? Our motivation is based on a distinctive advantage of applying EDA compared with other EA: users can discover useful properties of the problem from the learned probabilistic model. Since the model is explicitly built in EDA, it is feasible to observe the learned model structure and its parameters to understand some natures of the problem. For simple univariate (marginal distribution) model-based EDAs, because the interdependencies among variables are completely ignored, it is almost impossible to extract information representing the interdependency of variables or other structural information. On the other hand, multivariate model-based EDAs have such potentials. In EDA-MCC, multidependency is maintained to retain the potentials, while with the degree of model complexity explicitly controlled. To the best of our knowledge, EDA-MCC is the first attempt at scaling up multivariate model-based EDA for large-scale continuous optimization (up to 500-D problems).

The remainder of this paper is organized as follows. In Section II, the difficulties of traditional EDAs on high-dimensional problems are analyzed, especially for Gaussian-based EDAs. In Section III, WI and SM for EDA-MCC are presented in the context of Gaussian model. The difference between EDA-MCC and previous EDAs with model complexity penalization is also discussed. Experimental studies on 50–500-D problems are given in Section IV. In Section V, the dependence of EDA-MCC on its WI and SM parameters is investigated. The scalability of EDA-MCC is studied in Section VI. In Section VII, random partitioning-based SM is compared with a clustering-based SM, the advantage of random partitioning in high-dimensional search space is verified. The problem structure characterization capability of EDA-MCC is demonstrated in Section VIII. In Section IX, the interactions between WI and SM are analyzed. Conclusions are drawn in Section X, and future work is discussed.

II. Difficulties of EDAs on High-Dimensional Problems

A. Related Work

A typical EDA flow is shown in Fig. 1. Each individual in the population presents a solution. One iteration of the loop refers to one generation of evolution.

EDA

Initialize a population $\mathcal{P}$ by generating M individuals randomly.
Repeat until a stopping criterion is met.
1) Select $m \leq M$ individuals from $\mathcal{P}$.
2) $f(\vec{x}) \leftarrow$ Estimate a probability density function from the selected individuals.
3) $\mathcal{P}' \leftarrow$ Sample a number of individuals from $f(\vec{x})$.
4) Combine $\mathcal{P}$ and $\mathcal{P}'$ to create the new $\mathcal{P}$.

Fig. 1. Typical EDA flow.

The primary difference between different EDAs is the probabilistic model adopted. When adopting a Gaussian distribution model, the $f(\vec{x})$ in Fig. 1 has the form of a normal density that is defined by a mean vector $\vec{\mu}$ and a covariance matrix $\mathbf{\Sigma}$. The earliest proposed Gaussian-based EDAs are based on simple univariate Gaussian, such as UMDA$_c^G$ [2] and PBIL$_c$ [4]. In these EDAs, all variables are regarded independent of each other. The simplicity of such models makes them easy to implement and the algorithms are characterized by a low level of computational complexity. Also due to the simplicity, they may have difficulties in solving problems whose variables have strong interdependencies. To remedy this, several EDAs based on multivariate Gaussian have been proposed, such as EMNA$_{\text{global}}$ [2], normal IDEA [5], [6], and EGNA [2], [7]. EMNA$_{\text{global}}$ adopts a conventional maximum likelihood estimated multivariate Gaussian distribution defined by $\vec{\mu}$ and $\mathbf{\Sigma}$. In normal IDEA and EGNA, after obtaining the maximum likelihood estimation (MLE) of $\vec{\mu}$ and $\mathbf{\Sigma}$, a graphical factorization, that is, a Bayesian factorization (i.e., a Gaussian network) is constructed, usually by greedy search. Constructing graphical factorization introduces additional computational complexity along with MLE, but the computational time in solution sampling can be reduced. On the other hand, if we want to sample new solutions from a conventional multivariate Gaussian distribution as in EMNA$_{\text{global}}$, decomposing $\mathbf{\Sigma}$ is a must [26]. Since these EDAs are essentially based on the same multivariate Gaussian distribution, their performances are similar—at least no significant superiority of one over another has been reported so far.[1] Later, some extensions of these EDAs have been proposed to improve their poor explorative ability, such as EEDA [8], CT-AVS-IDEA [9], and SDR-AVS-IDEA [10]. These EDAs scale $\mathbf{\Sigma}$ according to some criterions after MLE. A comparative study of different covariance matrix scaling strategies can be found in [11]. Besides these single Gaussian-based EDAs, EDAs adopting Gaussian mixture distribution [27]–[33] have been proposed for solving multimodal and hard deceptive problems. Hybrid optimization algorithms based on Gaussian EDAs have also been proposed [34], [35].

Interestingly, previous studies have shown that although Gaussian models cannot always offer an accurate estimation of

[1] Some comparisons between EMNA$_{\text{global}}$ and EGNA can be found in [2]. However, few comparisons involving normal IDEA have been made.

the true distribution of promising solutions, they can nevertheless provide useful information for guiding the global search on many unimodal and some, but not all, multimodal problems. So far no satisfactory explanation of this phenomenon has been presented in the literature. It will be interesting in the future to study when a multimodal problem is easy or hard for a given single Gaussian-based EDA, e.g., by using recently proposed analytical approaches [36]–[39]. However, except for univariate Gaussian-based EDAs, most (if not all) existing studies of multivariate Gaussian-based EDAs are restricted to low-dimensional problems.

Continuous EDAs using histogram models include several EDAs based on univariate histogram [6], [12], [13], [15], [18] and some based on multivariate histogram [14], [16], [17], [19]. Histogram models are more flexible than Gaussian models because of the convenience to describe arbitrary multimodality. However, if considering multiple variable dependencies such as full interdependency, the required number of bins can increase exponentially with problem size [40], which makes multivariate histogram models hard to be applied to large-scale nonseparable problems in practice. Although some efforts have been made to improve the scalability of multivariate histogram model-based EDAs [14], [16], most existing results of these EDAs are also restricted to low-dimensional problems ($\leq$50-D, even lower than multivariate Gaussian-based EDAs).

To the best of our knowledge, there have been only a few attempts of studying continuous EDA on large-scale ($\geq$ 500-D) problems, including: 1) a univariate model based EDA, LSEDA-gl, proposed by Wang and Li [41]; 2) application of UMDA_c^G and EGNA as logistic regression regularizers on a "large k (genes), small N (samples)" microarray classification problem, proposed by Bielza *et al.* [42]; 3) study of parallel implementation of EGNA_{EE} on sphere function, proposed by Mendiburu *et al.* [43]; and 4) studies of a Gaussian EDA, namely AMaLGaM, on up to 1000-D problems done by Bosman [44]. However, these attempts have their limitations. LSEDA-gl is a univariate EDA where a mixed Gaussian and Lévy distribution is adopted. As discussed, it lacks the capability of modeling multidependencies. In [42], a multivariate EDA was utilized as a parameter optimizer of a logistic regression model with (order of) 500 parameters, trained via constrained maximum likelihood. The parameters were constrained to certain intervals, effectively regularizing the model. However, the general performance of the multivariate EDA on broader types of high-dimensional problems is still unknown. In [43], the study focuses on the parallel multivariate EDA's performance in terms of speed up of execution time but not on solution quality, and only one test function is involved in experiment. In [44], variants of AMaLGaM (with or without memory) using univariate Gaussian, Bayesian factorized (multivariate) Gaussian, and multivariate Gaussian with full covariance matrix were tested on problems up to 1000-D, 400-D, and 200-D, respectively. As can be seen, multivariate models' higher complexities reduce the size of applicable problems. In this paper, from a totally different perspective from [44], we propose a novel scalable multivariate EDA framework that is simpler in design yet capable of solving even larger problems. An open and important question is: Can we expect promising performance and moderate computational cost of multivariate EDAs on larger scale problems?

B. Curse of Dimensionality

Since EDAs completely rely on probabilistic models built from finite data samples, they must suffer from the well-known curse of dimensionality [24]. The more flexible and complex the model is, the more data it requires to yield a reliable estimation and to sustain enough good performance. According to the curse of dimensionality theory, the amount of data to sustain a given spatial density increases exponentially with the dimensionality of the search space. This will adversely impact any method based on spatial density, unless the data follows certain simple distributions. Obviously the latter condition is not always satisfied in practice. The population size of EDA has to grow quickly as the problem size grows to sustain good performance. Since EDA tries to learn some global statistical information from m sampled data (i.e., individuals selected from the population of M individuals, see Fig. 1), m has to be sufficiently large, which also requires a large population size M when some level of selection pressure needs to be maintained. Of course, the demand of the increasing population size can be of different levels when models have different levels of complexity. For simple univariate EDAs, when solving an n-dimensional problem, it estimates n 1-D distributions independently. When population size M is large enough for estimating these n distributions and finding good enough solution, M does not necessarily grow as n grows. However, for multivariate models, the more degrees of freedom make them usually require larger population sizes, which can be validated from our experiments. When the problem size is large, EDAs with complex multivariate models can become inapplicable because the large population size may consume considerable computational resources (see Section II-C). There is an urgent need for techniques that can reduce the required computational resources without affecting (too much) the precisions of learning a probabilistic model.

Since previous results (e.g., [6]) have shown that: 1) Gaussian models suffer less from the curse of dimensionality than histogram models, which is reasonable because Gaussian models usually have much fewer degrees of freedom, and 2) single Gaussian models have fewer degrees of freedom than Gaussian mixture models, in the following sections, we focus on using single multivariate Gaussian models to scale up EDA. Univariate Gaussian models are also involved in analysis and experiments. However, it should be noticed that our conclusions can be generalized and are not restricted only to Gaussian models. Although previous research has shown that single Gaussian model-based EDAs can perform well on many unimodal and multimodal problems, they still have known limitations other than the effect of the curse of dimensionality. Specifically, Gaussian EDAs using MLE are supposed to have poor explorative ability. Theoretical analysis of UMDA_c^G [45], [46] proved that the maximal distance that the mean of the population can move across the search space is bounded, and the algorithm is guaranteed to converge since the population

TABLE I
SUMMARY OF ONE-GENERATION COMPUTATIONAL COMPLEXITY

	UMDA_c^G	$\text{EMNA}_{\text{global}}$
Model estimation	$O(nm)$	$O(n^2m)$
Solution sampling	$O(nM)$	$O(n^2M)$

variance converges to zero. Although theoretical analysis have not been developed, similar results of multivariate Gaussian-based EDAs using MLE were also observed in experimental studies [9], [11], [28], [47]. Therefore, several Gaussian-based EDAs with covariance matrix scaling [8]–[10] were proposed. But the effectiveness of these techniques in very high-dimensional search space still lacks validation.

C. Computational Cost

Besides the curse of dimensionality, computational cost of an EDA (especially multivariate EDA) can also restrict its application to large-scale problems. In an EDA, if excluding fitness evaluation, the model estimation and subsequent solution sampling determine its overall computational complexity, which also depends on the model complexity. In general, univariate EDAs have lower level of computational complexity than multivariate EDAs. Empirical studies can show that, even for problems whose fitness function evaluation is not too time-consuming, multivariate EDAs' overall runtime can become unacceptable in practice. Here we consider the computational complexity brought by the model within one generation. For two representative EDAs of different model complexities: a univariate Gaussian EDA, UMDA_c^G [2], and a multivariate Gaussian EDA, $\text{EMNA}_{\text{global}}$ [2], analytical computational complexities in terms of data access are given as below. Suppose the current model is estimated from the selected individuals of the last generation. M denotes the population size, and m denotes the number of selected individuals, $m = \tau M$, usually $0.3 \leq \tau \leq 0.5$ [2], [28]. The computational complexities of UMDA_c^G and $\text{EMNA}_{\text{global}}$ are shown in Table I. For detailed computation please see Appendix A.

UMDA_c^G and any other univariate Gaussian EDAs share the same model structure and only differ in the way the model parameters are updated. These EDAs share mostly the same level of computational complexity. However, different multivariate Gaussian EDAs have different computational complexities. As mentioned above, $\text{EMNA}_{\text{global}}$ estimates model via MLE and sampling solutions via decomposition of covariance matrix, while normal IDEA and EGNA build a graphical factorization after MLE, and then fit the parameters of the factorization and sample solutions by traversing the graph. The MLE in all three is exactly the same, and thus, they share a same computational complexity in this step. For the latter steps, $\text{EMNA}_{\text{global}}$'s computational complexity is easy to analyze since decomposing a covariance matrix constantly costs cubic time with problem size. Whereas the graphical factorization in normal IDEA and EGNA can be obtained by several different structure search algorithms, whose computational complexities depend on the specific algorithms and the data samples. After obtaining the structure, in normal IDEA, the conditional variances of the factorization are computed by the inverse of covariance matrix [5], which costs the same computational complexity as decomposing a covariance matrix. We can infer that normal IDEA's computational complexity is higher than that of $\text{EMNA}_{\text{global}}$. In EGNA, the parameters of Gaussian network are computed in a different way, making the computational cost difficult to establish analytically. Literature on EGNA did not provide analytical computational complexity either. Also, considering the fact that multivariate Gaussian-based EDAs with covariance matrix scaling have additional computations, here we choose $\text{EMNA}_{\text{global}}$ as the representative of all multivariate Gaussian EDAs to analyze the computational complexity. The analysis of $\text{EMNA}_{\text{global}}$ approximately gives a lower bound of all multivariate Gaussian EDAs.

As mentioned above, when a univariate model is sufficient for solving a problem, M and m do not necessarily grow as n grows. Table I shows that, in this case, the overall computational cost of univariate EDAs, such as UMDA_c^G, can grow linearly with n. Although the model's simplicity can restrict its performance, its computational cost grows mildly. On the other hand, the overall computational cost of multivariate Gaussian EDAs, such as $\text{EMNA}_{\text{global}}$, grows much faster. Although [9] reported that a necessary M grows approximately with $\sqrt{n}$ for normal IDEA, in practice it is usually true that $M > m > n$. Overall computational cost of a typical multivariate Gaussian EDA thus grows at least with $O(n^3)$. In Section IV, more illustrative comparisons of CPU time will be made by experimental studies.

III. SCALING UP EDA: EDA-MCC

In short, there are three requirements to be met to scale up multivariate model-based EDA to large-scale problems.

1) Multivariate search needs to be preserved as much as possible.
2) Computational cost must be acceptable and grow mildly.
3) Running with small population sizes is preferred.

It can be seen that the differences in performance and computational complexity between EDAs using univariate Gaussian and multivariate Gaussian models are essentially relevant to the complexity of the Gaussian model. Intuitively, univariate Gaussian has simple structure and lower computational cost, but has difficulty in characterizing complicated interdependencies between variables. Multivariate Gaussian has complex structure and thus higher computational cost, but can effectively model interdependencies between variables. There is a need for an appropriate tradeoff between model complexity and computational cost such that an EDA can have promising performance on nonseparable problems with mild computational costs. We propose to reach such an attractive tradeoff in an EDA by two steps: WI and SM. The resulting EDA framework is called EDA-MCC.

A. Weakly Dependent Variable Identification (WI)

A multivariate Gaussian represents the (linear) interdependencies between variables by their covariances. According to the definition of covariance, we have

$$cov(X_i, X_j) = E((X_i - \mu_i)(X_j - \mu_j)) \tag{1}$$

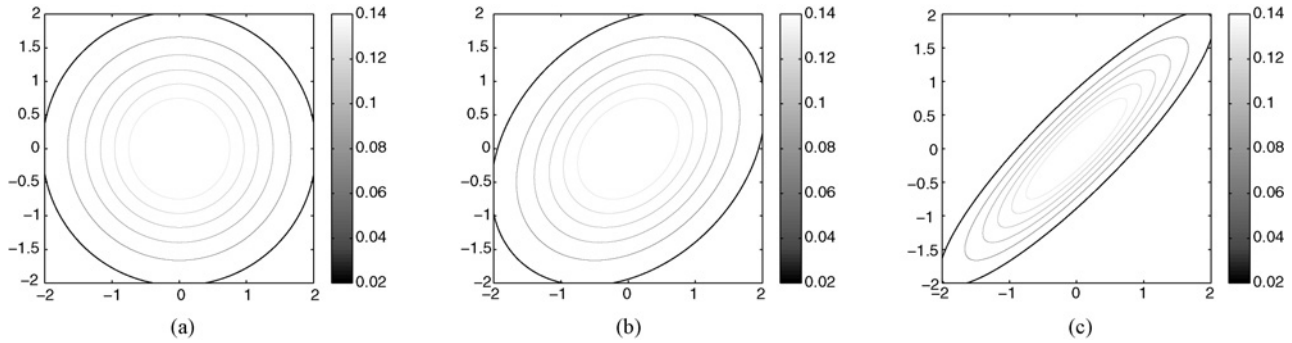

Fig. 2. Demonstrations of 2-D Gaussian distributions with different correlation coefficients. The contours denote the Gaussian densities. In every subfigure, each of the two variables has a standard deviation equal to 1; so, here the correlation coefficient equals the covariance. (a) Correlation = 0. (b) Correlation = 0.3. (c) Correlation = 0.9.

where $cov(X_i, X_j)$ is the covariance between variables X_i and X_j, $i, j = 1, \ldots, n$, E is the expected value operator. We also have

$$corr(X_i, X_j) = \frac{cov(X_i, X_j)}{\sigma_i \sigma_j} \tag{2}$$

where $corr(X_i, X_j)$ is the linear correlation coefficient between X_i and X_j, σ_i and σ_j are the standard deviations of X_i and X_j respectively, $\sigma_i > 0$, $\sigma_j > 0$, $i, j = 1, \ldots, n$. According to the definition, a correlation coefficient cannot exceed 1 in absolute value. Thus, correlation coefficients can also be seen as normalized covariances.

Suppose during the evolutionary process of a multivariate Gaussian EDA, if at some generation, the correlation coefficients are close to zero, which means the observed linear dependencies between variables are weak, then the distribution that the model can learn will be little different from a univariate Gaussian. The algorithm's exhibited behavior in this generation does not differ much from a univariate Gaussian EDA, either. Fig. 2 shows an example of 2-D Gaussian distribution with different correlation coefficients. As can be seen, there is a tradeoff between: 1) the computational complexity and requirement of population size that increase with respect to a more complex model, and 2) the performance that potentially benefits from a complex model. In this case, we find that switching the current model to a univariate Gaussian can greatly reduce the computational complexity and the requirement of population size without significantly affecting the performance. In a way, the algorithm can ignore the weak correlations so that the search effort could be focused on stronger ones. Inspired by this, we first identify those approximately independent (weakly dependent/correlated) variables, and then apply a simple univariate model on them. We call this strategy the WI.

Weakly dependent variables can be identified by first calculating an $n \times n$ global correlation matrix, then picking out variables whose absolute values of correlation coefficients to all the other variables are no larger than a threshold θ ($0 \leq \theta \leq 1$). The set of such weakly dependent variables, denoted by $\mathcal{W}$, is defined as

$$\mathcal{W} = \left\{ X_i \mid |corr(X_i, X_j)| \leq \theta, \forall j = 1, \ldots, n, j \neq i \right\}. \tag{3}$$

WI

1) Calculate an $n \times n$ global correlation matrix $\boldsymbol{C}$ based on m_{corr} individuals. $\boldsymbol{C}_{ij} = corr(X_i, X_j), i, j = 1, \ldots, n$.
2) Use $\boldsymbol{C}$ to construct $\mathcal{W}$ according to (3).
3) Estimate a univariate model for $\mathcal{W}$ based on the m selected individuals.

Fig. 3. Main flow of weakly dependent variable identification (WI).

As can be seen, applying a univariate model on $\mathcal{W}$ implies explicitly removing the dependencies on the variables in $\mathcal{W}$, which can reduce the computational complexity. If a proper θ exists, a good tradeoff between the gain in computational cost and the loss of model precision can be found.

In contrast to weakly dependent, the rest of the variables are regarded as strongly dependent. The set of strongly dependent variables, denoted by $\mathcal{S}$, is defined as

$$\mathcal{S} = \{X_i \mid X_i \notin \mathcal{W}, i = 1, \ldots, n\}. \tag{4}$$

Let $\mathcal{V}$ denote the set of all variables $\mathcal{V} = \{X_i \mid i = 1, \ldots, n.\}$. Obviously, we have $\mathcal{V} = \mathcal{W} \bigcup \mathcal{S}$ and $\emptyset = \mathcal{W} \bigcap \mathcal{S}$.

Note that if we use a global correlation matrix for the purpose of identifying $\mathcal{W}$, we do not need a large number of samples as we do for estimating a reliable global covariance matrix for the purpose of guiding the search, even though computing a correlation matrix is essentially of no difference with computing a covariance matrix. Because the precision of covariance matrix directly impacts the sampling procedure and thus influences the algorithm's behavior, it does require a sufficiently large amount of data with respect to problem size. However, if we just use a correlation matrix for a coarse learning such as identifying weakly dependent variables, its precision does not directly influence the sampling. Later we will see that, with working with SM, a small sample size for WI (100 for 50–500-D problems) can be sufficient, which also helps reduce the computational cost of EDA-MCC.

Let m_{corr} denote the sample size for constructing a global correlation matrix $\boldsymbol{C}$. The main flow of WI is depicted in

SM

1) Construct $\mathcal{S}$ according to (4).
2) Randomly partition $\mathcal{S}$ into $\lceil |\mathcal{S}|/c \rceil$ non-intersected subsets: $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{\lceil |\mathcal{S}|/c \rceil}$, where c is a user specified parameter defining the maximal size of a subset ($1 \leq c \leq n$).
3) Estimate a multivariate model for each subset based on the m selected individuals.

Fig. 4. Main flow of subspace modeling (SM).

Fig. 3. Here, the term weakly dependent/correlated is not a strictly defined term as in the statistics domain. Whether a variable is classified into $\mathcal{W}$ or not is determined by both the correlation matrix at hand and the user specified parameter θ. The correlation matrix reflects the observed information in the search space, while different values of θ can reflect the user's confidence on the univariate model. The larger θ is, the more probable that more variables are optimized by the univariate model. Then, less computational cost and a smaller population size may be required. In this paper, we find a proper value of $\theta = 0.3$ for EDA-MCC (see Sections IV–VI) such that: 1) the model precision is only slightly worse; 2) the computational complexity and the requirement of population size are greatly reduced; and 3) the overall performance of EDA-MCC can be significantly better than the previous EDAs, at least on the 13 problems investigated in experiments.

Note that for non-Gaussian EDAs, weakly dependent may not be identical to weakly correlated. If applying WI to non-Gaussian models, the identification method may need redefinition. One can also imagine other ways of defining weakly/strongly dependent variables. For instance, the variables can be classified as weakly or strongly dependent by considering their correlation with the function to be optimized. The idea of separating weakly dependent variables from strongly dependent ones in this context is interesting and worth further consideration in the future. However, as typically done in EDA implementations, our definition of weak/strong dependency is restricted to variables only (within the context of building a local Gaussian model on the variables) and the model does not reflect any correlation between a variable and the function value.

B. Subspace Modeling (SM)

Suppose we only have a small population size M (and thus m), and $|\mathcal{S}|$ is still too large for m samples to give a reliable estimation for a multivariate Gaussian model. To obtain better overall performance, as a tradeoff, we project the m points to several subspaces of the n-dimensional search space, then build model and sample solutions on subspaces. When it is impractical to further increase m, building subspace models and using their combination to approximate the global estimation can be a good choice. We call it the SM, whose flow is shown in Fig. 4. Each subset of $\mathcal{S}$, i.e., group of variables, corresponds to a subspace. All the m samples are projected to $\lceil |\mathcal{S}|/c \rceil$ subspaces,[2] and we build a multivariate model for each subspace. The capacity c indicates the maximal size of a subspace. It represents to what extent we trust the m samples to give reliable estimation. By dividing the variables into several separated subspaces and projecting the samples to lower dimensional subspaces, the EDA only considers the local dependencies among variables belonging to the same subspace, and the density of samples for each subspace will increase. This technique probably offers a feasible way for alleviating the growth of population size with respect to a growing problem size, which will be validated by our experimental results in later sections.

	X_{k1}	X_{k2}	X_{k3}	X_{k4}	X_{k5}	X_{k6}	X_{k7}	X_{k8}
X_{k1}	1.79	0.92	1.31	0	0	0	0	0
X_{k2}	0.92	2.41	0.59	0	0	0	0	0
X_{k3}	1.31	0.59	3.88	0	0	0	0	0
X_{k4}	0	0	0	1.54	−0.23	0.75	0	0
X_{k5}	0	0	0	−0.23	1.21	−0.84	0	0
X_{k6}	0	0	0	0.75	−0.84	1.82	0	0
X_{k7}	0	0	0	0	0	0	1.95	0.56
X_{k8}	0	0	0	0	0	0	0.56	2.94

Fig. 5. Example of approximated global covariance matrix on $\mathcal{S}$ after performing SM. $\mathcal{S} = \{X_1, \ldots, X_8\}$, c=3. $(X_{k1}, \ldots, X_{k8})$ is a random permutation of $(X_1, \ldots, X_8)$. The three subsets of $\mathcal{S}$ are $\mathcal{S}_1 = \{X_{k1}, X_{k2}, X_{k3}\}$, $\mathcal{S}_2 = \{X_{k4}, X_{k5}, X_{k6}\}$, and $\mathcal{S}_3 = \{X_{k7}, X_{k8}\}$.

After randomly partitioning $\mathcal{S}$, variables of different subsets are regarded independently. When we use a multivariate Gaussian to model each subspace, combination of all subspace Gaussian models can be seen as an approximation to the global Gaussian estimation on $\mathcal{S}$. The global mean vector on $\mathcal{S}$ is still identical to the combination of subspace models, but the global covariance matrix is approximated by a block diagonal matrix whose main diagonal blocks are the subspace covariance matrices. Fig. 5 shows an example. If $|\mathcal{S}| \leq c$, the variables can be kept together within one group. If $|\mathcal{S}| > c$, it means that the size of current $\mathcal{S}$ is beyond the capability of a global multivariate model that m samples can estimate according to the user's experience or preference. Therefore, we have to make a concession by explicitly eliminating some dependencies between variables while keeping the rest. As will be shown later, WI and SM are performed in every generation, thus the random partition is not fixed throughout evolution. Variables from different subsets in current generation have the chance to be grouped in one subset and keep their interactions in the next generations. When sampling a new individual, its variables in $\mathcal{S}$ are sampled from the subspace models they belong to. Then they are concatenated with those sampled variables in $\mathcal{W}$. The evaluation of a newly sampled individual is the same as in traditional EDA.

The random subspace partitioning method proposed here is a simple and the most straightforward one. Experiments will show that although we use only the simplest SM method, it indeed significantly improves EDAs' performance on large-scale problems. Of course, more sophisticated subspace partitioning can be developed. For example, $\mathcal{S}$ can be divided into several clusters of variables according to the correlation coefficients, and each cluster is regarded a subspace. However,

[2]For a real number x, $\lceil x \rceil$ is the smallest integer y, such that $y \geq x$.

it can be imagined that such techniques also heavily suffer from the curse of dimensionality. With a finite sample size, we cannot expect good clustering in very high-dimensional space. Section VII will present comparison between the random subspace partitioning and a clustering-based one. Experiments will show that the simple random partitioning can perform significantly better on large problems.

C. Model Complexity Control: WI + SM

By incorporating WI and SM within the EDA framework, we explicitly control the model complexity: 1) WI reduces the model complexity by approximation of univariate model, and 2) SM further reduces the complexity of the multivariate part of the model by approximation of subspace models. Let $\mathcal{S}_k$ ($1 \leq k \leq \lceil |\mathcal{S}|/c \rceil$) denote a subset of $\mathcal{S}$ and vector $\vec{s_k}$ denote realizations of the variables in $\mathcal{S}_k$. After performing WI and SM, the final joint pdf has the form

$$f(\vec{x}) = \prod_{X_i \in W} g_i(x_i) \prod_{k=1}^{\lceil |\mathcal{S}|/c \rceil} h_k(\vec{s_k}) \tag{5}$$

where $g_i(\cdot)$ is the univariate pdf of variable X_i, and $h_k(\cdot)$ is the multivariate pdf of variables in $\mathcal{S}_k$. For instance, we can assign $g_i(\cdot)$ to a univariate Gaussian as (6) and assign $h_k(\cdot)$ to a multivariate Gaussian as (8).

Based on WI + SM, the main flow of the proposed EDA framework, namely EDA with model complexity control (EDA-MCC), is given in Fig. 6. The WI and SM steps in Fig. 6 are essentially the same as Figs. 3 and 4. As discussed above, for the purpose of coarse learning, m_{corr} does not need to be as large as m. So we sample m_{corr} individuals out of the m selected individuals to calculate correlation matrix $\boldsymbol{C}$. Because duplicate samples cannot contribute to correlation estimation, sampling without replacement is adopted. Experiments in Sections IV–VI will show that a small $m_{\text{corr}} = 100$ can work fine for problem sizes up to 500-D.

The comparison of computational complexity of EDA-MCC, UMDA$_c^G$, and EMNA$_{\text{global}}$ is shown in Table II. For details of computation please refer to Appendix B. Because $m_{\text{corr}} \leq m$ and $c \leq n$, in a same number of generations, EDA-MCC's computational complexity is always between the complexities of a typical univariate Gaussian EDA and a typical multivariate one. Besides, if EDA-MCC requires smaller m and M, the computational cost can be further reduced. Specifically, in experiments, we will apply a UMDA$_c^G$ model as (6) for variables in $\mathcal{W}$, and an EEDA model mentioned in Section II for each subset of $\mathcal{S}$. EEDA [8] is a multivariate Gaussian EDA using covariance matrix scaling. After performing MLE, EEDA scales the covariance matrix by resetting its minimum eigenvalue to its maximum eigenvalue. EEDA regards the direction of the eigenvector with the minimum eigenvalue as an approximation to the fitness function's gradient. Previous studies [11], [35] have shown that by enlarging the variance along this direction, EEDA can have better explorative ability than EMNA$_{\text{global}}$ and require smaller population sizes. Since the covariance matrix scaling can be done in $O(n)$ [11], EEDA has roughly the same computational complexity as EMNA$_{\text{global}}$ when using the same parameters. Therefore, the computational complexity analysis of EDA-MCC in Table II still holds.

EDA-MCC

Initialize a population $\mathcal{P}$ by generating M individuals randomly.

Repeat until a stopping criterion is met.

1) Select $m \leq M$ individuals from $\mathcal{P}$.
2) Randomly sample $m_{corr} \leq m$ individuals from the m selected individuals without replacement.
3) Build a model using WI+SM, as (5):
   a) WI:
      i) Calculate the correlation matrix $\boldsymbol{C}$ based on the m_{corr} sampled individuals, $\boldsymbol{C}_{ij} = corr(X_i, X_j), i, j = 1, \ldots, n$, as (2).
      ii) Construct $\mathcal{W}$ based on $\boldsymbol{C}$, as (3).
      iii) $\forall X_i \in \mathcal{W}$, estimate a univariate model $g_i(\cdot)$ based on the m selected individuals.
   b) SM:
      i) Construct $\mathcal{S}$, as (4).
      ii) Randomly partition $\mathcal{S}$ into $\lceil |\mathcal{S}|/c \rceil$ non-intersected subsets: $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_{\lceil |\mathcal{S}|/c \rceil}$, $1 \leq c \leq n$.
      iii) Estimate a multivariate model $h_k(\cdot)$ for each subset $\mathcal{S}_k$ based on the m selected individuals, $k = 1, \ldots, \lceil |\mathcal{S}|/c \rceil$.
4) $\mathcal{P}' \leftarrow$ Sample new individuals: Sample from $g_i(\cdot)$ and $h_k(\cdot)$ independently, then combine sampled variables into one reproduced individual.
5) Combine $\mathcal{P}$ and $\mathcal{P}'$ to create the new $\mathcal{P}$.

Fig. 6. Main flow of EDA-MCC.

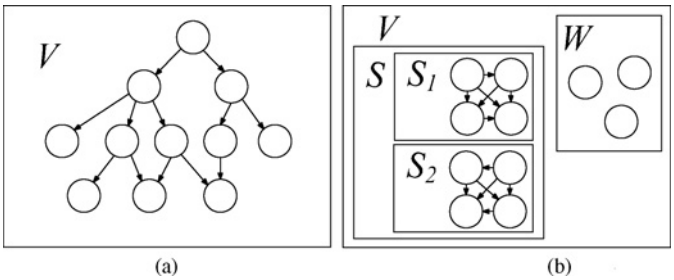

Fig. 7. Demonstration of model structures after applying traditional approaches and WI + SM, respectively. Each circle represents a variable and directed edges represent the dependency. (a) Previous approaches. (b) WI + SM.

D. Difference Between EDA-MCC and EDAs with Model Complexity Penalization

Several other approaches for controlling/penalizing the model complexity in EDAs have also been proposed. For instance, EGNA$_{EE}$ [2] uses the edge exclusion test to control the structure complexity of a Gaussian network, or uses the BGe (Bayesian Gaussian equivalence) metric and local search to

TABLE II
COMPARISON OF ONE-GENERATION COMPUTATIONAL COMPLEXITY

	UMDA_c^G	$\text{EMNA}_{\text{global}}$	EDA-MCC
Model estimation	$O(nm)$	$O(n^2m)$	$[O(n^2m_{\text{corr}}) + O(nm),\ O(n^2m_{\text{corr}}) + O(cnm)]$
Solution sampling	$O(nM)$	$O(n^2M)$	$[O(nM),\ O(cnM)]$

learn the structure. Normal IDEA [28] uses the BIC (Bayesian Information Criterion) metric to penalize the complexity of a normal pdf factorization. Real-coded Bayesian Optimization Algorithm (rBOA) [30]–[32] also employs Bayesian factorization with BIC metric and greedy search, but in contrast to EGNA and normal IDEA, it fits Gaussian mixture model instead of a single Gaussian. There are significant differences between EDA-MCC and these approaches.

1) Fig. 7 shows typical model structures after applying previous approaches' model estimation and WI + SM. Compared with WI + SM, previous approaches can be seen as implicitly controlling the model complexity. Therefore, model estimation in these approaches often leads to a large connected graph, although some dependencies between variables are removed. This means that the variables are still modeled by a big multivariate model.[3] In contrast, WI + SM explicitly partitions the variables into several separated groups with a size limit (parameter c). Then a number of small models are applied to $\mathcal{W}$ and subsets of $\mathcal{S}$. The failure of a big model on large problems can be seen from the fact that few results of previous algorithms on problems having hundreds of variables are reported. A possible explanation is that it is due to the curse of dimensionality and the computational complexity issues. As n grows, a big model's performance quickly deteriorates and its computational cost also rapidly increases.
2) Previous approaches are mostly trying to precisely learn a complex global structure from data, which is in fact impractical in high-dimensional space. They also involve complicated computation that makes the computational complexity of EDAs become even higher. On the other hand, if WI + SM is used, the global structure is just roughly learned. Since it is too hard to perform good global learning in high-dimensional space, WI + SM tries to perform good learning in divided subspaces to give a better approximated global estimation. Fortunately, the controlling parameters θ and c both have explicit physical implications that can be interpreted and set easily (Section V will give more discussions on these parameters and empirical guidelines of setting them). WI and SM do not introduce additional time-consuming computation into EDA. They can even help reduce the overall computational complexity. But we can also imagine that if the global structure can be successfully learned under some conditions, WI + SM will not outperform traditional approaches.
3) Compared with previous approaches, WI + SM is more flexible in terms of introducing different search strategies into EDAs. For example, probabilistic models other than Gaussian can also be applied to $\mathcal{W}$ and $\mathcal{S}$. Applying different models on different subsets of $\mathcal{S}$ for diversity consideration is also feasible. This allows the easy development of new EDAs and hybrid algorithms. But in this paper we only discuss Gaussian models.

[3]In rBOA, the global pdf is factorized as a product of linear combinations of subproblems, which are still essentially fragments of a Gaussian network, and thus can be interconnected via overlapping variables. Even if they are the maximal compound subproblems [31] that are totally separated from each other, there is no explicit control on the size of a subproblem. It may also result in big multivariate subproblems.

IV. EXPERIMENTAL STUDIES

A. Experimental Setup

1) *Involved Algorithms:* Four algorithms are involved in experimental comparisons: UMDA_c^G [2], $\text{EMNA}_{\text{global}}$ [2], EEDA [8], and EDA-MCC. As extensions of the analyses on computational complexity, we select UMDA_c^G as a representative of univariate Gaussian EDAs, and $\text{EMNA}_{\text{global}}$ as a representative of multivariate Gaussian EDAs. Both algorithms are based on MLE. Since many theoretical studies, experimental comparisons and real-world applications of these two EDAs have been made [2], [7], [8], [11], [15]–[19], [29], [34], [35], [41]–[43], [45], [46], [48]–[51], comparing them makes sense. EEDA is included as a representative of multivariate Gaussian EDAs using covariance matrix scaling. It can be seen as an extension of $\text{EMNA}_{\text{global}}$, making it easy to implement based on $\text{EMNA}_{\text{global}}$. In EDA-MCC, we apply a UMDA_c^G model for variables in $\mathcal{W}$, and an EEDA model for each subset of $\mathcal{S}$. Such an implementation can yield fair comparisons with UMDA_c^G, $\text{EMNA}_{\text{global}}$, and EEDA. To fairly compare CPU time cost, all algorithms are implemented with C++ using a same template design and they share the same basic data structures and numerical computation library. They only differ on model estimation and solution sampling modules.

2) *Test Functions:* Test functions are listed in Table III. They are selected from classical benchmarks in [7], [52] and CEC2005 special session [53]. All 13 functions are minimization problems. For details of the CEC2005 functions, including the shifted global optima and the transformation matrices, etc., please refer to [53]. These functions contain several comparison pairs, from which we can see whether an algorithm is sensitive to the shifted or rotated function landscape. The 13 functions can also be classified into three groups:

1) separable unimodal problems: F_1 and F_2;
2) nonseparable problems with only a few ($\leq$ 2) local optima: F_3–F_{10};
3) multimodal problems with many local optima: F_{11}–F_{13}.

3) *Common Parameter Settings:* For traditional EDAs such as UMDA_c^G, $\text{EMNA}_{\text{global}}$, and EEDA, besides τ representing

TABLE III
TEST FUNCTIONS USED IN EXPERIMENTS

	Description	**Expression**	**Domain**
F_1	sphere (f_1 in [52])	$F(\vec{x}) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^n$
F_2	shifted sphere (F_1 in [53])	$F(\vec{x}) = \sum_{i=1}^{n} z_i^2 + f_{bias_1}, \quad \vec{z} = \vec{x} - \vec{o}$	$[-100, 100]^n$
F_3	Schwefel's problem 2.21 (f_4 in [52])	$F(\vec{x}) = \max_i\{\|x_i\|, 1 \le i \le n\}$	$[-100, 100]^n$
F_4	shifted F_3	$F(\vec{x}) = \max_i\{\|z_i\|, 1 \le i \le n\}, \quad \vec{z} = \vec{x} - \vec{o}$	$[-100, 100]^n$
F_5	Schwefel (F_2 in [7])	$F(\vec{x}) = \sum_{i=1}^{n}[(x_1 - x_i^2)^2 + (x_i - 1)^2]$	$[-10, 10]^n$
F_6	shifted F_5	$F(\vec{x}) = \sum_{i=1}^{n}[(z_1 - z_i^2)^2 + (z_i - 1)^2], \quad \vec{z} = \vec{x} - \vec{o} + \vec{1}$	$[-10, 10]^n$
F_7	Rosenbrock (f_5 in [52])	$F(\vec{x}) = \sum_{i=1}^{n-1}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-100, 100]^n$
F_8	shifted Rosenbrock (F_6 in [53])	$F(\vec{x}) = \sum_{i=1}^{n-1}[100(z_{i+1} - z_i^2)^2 + (z_i - 1)^2] + f_{bias_6}, \quad \vec{z} = \vec{x} - \vec{o} + \vec{1}$	$[-100, 100]^n$
F_9*	shifted rotated high conditioned elliptic (F_3 in [53])	$F(\vec{x}) = \sum_{i=1}^{n}(10^6)^{\frac{i-1}{n-1}} z_i^2 + f_{bias_3}$ $\vec{z} = (\vec{x} - \vec{o}) \cdot \boldsymbol{M}$	$[-100, 100]^n$
F_{10}	Schwefel 2.6 with global optimum on bounds (F_5 in [53])	$F(\vec{x}) = max\{\|\boldsymbol{A}_i\vec{x} - \boldsymbol{B}_i\|\} + f_{bias_5}$ $i = 1, \ldots, n.$	$[-100, 100]^n$
F_{11}	Rastrigin (f_9 in [52])	$F(\vec{x}) = \sum_{i=1}^{n}[x_i^2 - 10cos(2\pi x_i) + 10]$	$[-5, 5]^n$
F_{12}*	shifted rotated Rastrigin (F_{10} in [53])	$F(\vec{x}) = \sum_{i=1}^{n}[z_i^2 - 10cos(2\pi z_i) + 10] + f_{bias_{10}}, \quad \vec{z} = (\vec{x} - \vec{o}) \cdot \boldsymbol{M}$	$[-5, 5]^n$
F_{13}	shifted expanded Griewank plus Rosenbrock (F_{13} in [53])	See [53], page 16.	$[-3, 1]^n$

* Note that the transformation matrix $\boldsymbol{M}$ in F_9 and F_{12} is not the population size M.
The domains of function F_7 and F_{11} are changed from original definitions in [52] to make them consistent with the domains of F_8 and F_{12}, respectively. F_4 and F_6 are shifted version of F_3 and F_5, respectively. The shifted global optima are generated following the same way of [53].

the selection pressure, the only parameter is the population size M. Given a fixed maximal number of fitness evaluations (max. #eval) as in many real-world applications, a larger M may offer better learning, but also reduces the maximal number of generations in the meantime, and vice versa for smaller M. People are aware of the tradeoff between population size and number of generations, and understand that the balance between the two factors, which may even vary from problem to problem, has significant influence on the performance of EDAs. As in most (if not all) studies on EDAs, our investigation does not emphasize the setting of population size. Instead, for each EDA, we always apply four settings of population size, $M \in \{200, 500, 1000, 2000\}$, aimed at releasing promising performance of the algorithms as much as possible on every problem. In experiments, for each algorithm, given a problem with a specific problem size n, we compare the average best solutions obtained among the four population sizes, and choose the best result. The population size leads to the best result is also recorded for comparison. All algorithms use $\tau = 0.5$ in all tests (thus $m = M/2$). The initial populations are always generated uniformly within the search space. Elitist approach is adopted for all algorithms, i.e., only one best individual is survived into the next generation, together with $(M - 1)$ newly sampled individuals they constitute a new generation. All these settings are widely used when studying these EDAs in previous publications. For each problem, we test problem sizes $n \in \{50, 100\}$. The max. #eval is set according to [53], i.e., max. #eval = $10000 \times n$. Algorithms are terminated only when their #eval exceed this limit. The results are averaged over 25 independent runs. All experiments are done on a P4 2.40 GHz computer with 512 MB RAM.

4) *Parameters of EDA-MCC:* Through all experiments of EDA-MCC, we set $m_{\text{corr}} = 100$, $\theta = 0.3$ for WI. We regard $m_{\text{corr}} = 100$ points as enough to calculate the correlation coefficients between any pair of variables (a pair of variables implies a 2-D space). We set $\theta = 0.3$ here because it is a popular threshold to define weakly correlated in the context of statistics. In our experience, we have also observed that WI can be sensitive to the value of θ. For example, a small value of $\theta = 0.15$ may result in an empty $\mathcal{W}$, i.e., all of the variables are regarded as strongly correlated with each other, which makes WI a null operation. Large $\theta = 0.6$ may lead to $\mathcal{W} = \mathcal{V}$, i.e., EDA-MCC degrades itself into a UMDA$_c^G$and discards all the dependencies among variables. To release the power of EDA-MCC most, there must be an optimal θ given a problem and other parameters. Different problems and other parameters may also lead to different optimal value of θ. As mentioned above, θ can reflect the user's confidence on univariate model. To have reasonable analysis on the effects of WI, we set a constant and moderate value of $\theta = 0.3$ for all tests. Here our aim is to demonstrate that EDA can benefit from WI, whereas which value of θ benefits EDA most on a specific problem can be an independent issue. For SM, we set $c = 20$. In practice, the settings of c can be determined by m according to user's preference and experience. In normal cases, if a larger m can be applied, c can also be set larger, and vice versa. When m is large enough to give a reliable estimation on the entire n-dimensional space, we can set $c = n$, which implies that we fully trust the global estimation rather than approximating it by combination of subspace models. But meanwhile, we should also afford the computational complexity. On the other hand, a smaller c can significantly reduce the computational complexity. Users can weigh the pros and cons and then set c. Parameters m_{corr}, θ, and c all have explicit physical implications. Their values are either bounded or can be determined with the guidance of other predetermined parameters or user's preference. It should be straightforward to set these parameters when applying EDA-MCC to a new problem. In Section V, the influence of different θ and c will be investigated. Empirical guidelines of setting EDA-MCC parameters will also be given.

TABLE IV
SOLUTION QUALITY COMPARISON

Prob.	n	UMDA$_c^G$	EMNA$_{\text{global}}$	EEDA	EDA-MCC
F_1	50	**0 ± 0**	1.3e-11 ± 6.3e-11§	**0 ± 0**	**0 ± 0**
	100	**0 ± 0**	1.4e+01 ± 5.6e+00§	**0 ± 0**	**0 ± 0**
F_2	50	**0 ± 0**	4.5e+04 ± 2.2e+03§	**0 ± 0**	**0 ± 0**
	100	**0 ± 0**	1.4e+05 ± 4.0e+03§	5.3e-10 ± 1.4e-09§	**0 ± 0**
F_3	50	2.6e-04 ± 1.5e-05§	1.2e-01 ± 1.2e-01§	1.8e-08 ± 2.4e-09§	**0 ± 0**
	100	2.6e-02 ± 8.3e-02§	3.3e+00 ± 7.0e-01§	1.5e-03 ± 8.5e-04§	**0 ± 0**
F_4	50	3.4e+01 ± 2.5e+00§	4.1e+01 ± 2.6e+00§	1.4e-05 ± 6.8e-05§	**0 ± 0**
	100	4.7e+01 ± 3.1e+00§	5.8e+01 ± 2.7e+00§	8.1e+00 ± 1.4e+00§	**0 ± 0**
F_5	50	1.5e+01 ± 4.1e+00§	1.5e+02 ± 1.4e+01§	2.4e-02 ± 3.7e-03§	**0 ± 0**
	100	1.3e+02 ± 2.7e+01§	6.7e+02 ± 7.5e+01§	3.8e-01 ± 4.7e-02§	**0 ± 0**
F_6	50	1.4e+01 ± 5.2e+00§	6.6e+03 ± 9.4e+02§	1.0e-01 ± 1.2e-02§	**0 ± 0**
	100	1.8e+02 ± 2.6e+01§	2.2e+04 ± 2.1e+03§	7.2e+00 ± 7.9e-01§	**0 ± 0**
F_7	50	4.8e+01 ± 3.4e-02§	5.7e+01 ± 5.9e+00§	5.0e+01 ± 9.2e+00†	**4.7e+01 ± 2.1e-01**
	100	9.7e+01 ± 6.4e-02§	2.7e+03 ± 1.5e+03§	9.7e+01 ± 3.7e-01§	**9.6e+01 ± 7.5e-02**
F_8	50	4.1e+02 ± 9.1e+02§	4.0e+09 ± 7.5e+08§	5.2e+02 ± 1.0e+03§	**4.8e+01 ± 1.5e-01**
	100	9.3e+02 ± 3.1e+03§	1.8e+10 ± 1.9e+09§	4.4e+04 ± 4.4e+04§	**9.6e+01 ± 1.3e-01**
F_9	50	4.3e+07 ± 4.1e+06§	1.8e+09 ± 2.4e+08§	4.1e+06 ± 1.4e+06	**3.6e+06 ± 1.5e+06**
	100	4.3e+07 ± 3.1e+06§	4.9e+08 ± 9.7e+07§	2.2e+07 ± 3.7e+06§	**9.6e+06 ± 2.5e+06**
F_{10}	50	4.9e+03 ± 1.8e+02§	2.9e+04 ± 1.4e+03§	**2.0e+03 ± 2.0e+02**§	3.1e+03 ± 3.4e+02
	100	5.9e+03 ± 4.3e+02§	7.8e+04 ± 2.1e+03§	4.4e+03 ± 6.0e+02§	**1.9e+03 ± 3.6e+02**
F_{11}	50	**0 ± 0**§	7.7e+00 ± 5.0e+00§	3.1e+02 ± 1.3e+01§	2.9e+02 ± 1.4e+01
	100	**0 ± 0**§	1.4e+02 ± 2.4e+01§	7.3e+02 ± 1.5e+01§	7.5e+02 ± 1.6e+01
F_{12}	50	**2.1e+00 ± 9.5e-01**§	3.2e+02 ± 2.1e+01§	3.1e+02 ± 1.7e+01†	3.0e+02 ± 1.46e+01
	100	**8.6e+00 ± 2.1e+00**§	9.0e+02 ± 2.9e+01§	7.3e+02 ± 2.5e+01	7.4e+02 ± 2.35e+01
F_{13}	50	**7.8e+00 ± 8.3e-01**§	9.9e+01 ± 2.4e+01§	2.7e+01 ± 1.1e+00*	2.6e+01 ± 9.2e-01
	100	**1.5e+01 ± 2.0e+00**§	1.2e+03 ± 1.9e+02§	3.8e+01 ± 2.6e+01§	6.5e+01 ± 1.6e+00

* The value of Asymp. Sig. (two-tailed) < 0.05 when compared with the results of EDA-MCC.
† The value of Asymp. Sig. (two-tailed) < 0.01 when compared with the results of EDA-MCC.
§ The value of Asymp. Sig. (two-tailed) < 0.001 when compared with the results of EDA-MCC.
The results are divided into three groups according to the problem properties. The means and standard deviations of $F(\vec{x}) - F(\vec{x}*)$ for 25 runs are reported. If the value is below 1e-12, we regard it as zero. The best result (with the minimal mean value) is bolded in each row. Results of EDA-MCC are compared with others algorithms' by nonparametric Mann–Whitney U test. The significance level is shown by markers (*, †, and §). No marker implies no significant difference.

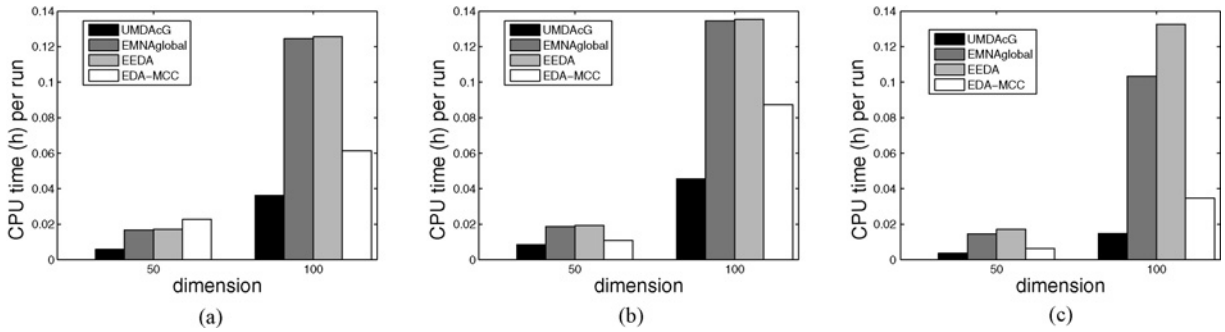

Fig. 8. Comparison of average CPU time on F_2, F_8, and F_{11}. (a) F_2: shifted sphere. (b) F_8: shifted Rosenbrock. (c) F_{11}: Rastrigin.

B. Experimental Results

We record the difference between the best fitness that an algorithm can find and the known global optimum, i.e., $F(\vec{x}) - F(\vec{x}*)$. The value is always non-negative for minimization problems. The smaller it is, the better an algorithm performs. The means and standard deviations of $F(\vec{x}) - F(\vec{x}*)$ for each algorithm in each test are summarized in Table IV. If $F(\vec{x}) - F(\vec{x}*) < 1e{-}12$, then we regard it as zero, i.e., the global optimum is reached. If multiple results among the four-population-size tests reach the optimum, we report the one exhibiting the fastest convergence. Table V shows the corresponding population sizes of the algorithms. Because the CPU time comparisons on different problems are similar, we only show the CPU time comparisons on selected functions F_2, F_8 and F_{11} in Fig. 8.

C. Discussion and Analysis

1) *Separable Unimodal Problems:* The separable and unimodal structures of F_1 and F_2 can facilitate univariate model-based EDAs in solving the problems although this is not always the case. Our experiments show that, in our case, UMDA$_c^G$ and EDA-MCC perform well. However, EMNA$_{\text{global}}$, which relies on global multivariate estimation, exhibits significant performance degradation as n grows. EEDA also performs

TABLE V
POPULATION SIZE COMPARISON

Prob.	n	UMDA$_c^G$	EMNA$_{\text{global}}$	EEDA	EDA-MCC
F_1	50	500	2000	1000	**200**
	100	500	2000	2000	**200**
F_2	50	500	2000	1000	**200**
	100	**1000**	2000	2000	**1000**
F_3	50	2000	2000	1000	**200**
	100	2000	2000	2000	**200**
F_4	50	2000	2000	1000	**200**
	100	2000	2000	2000	**200**
F_5	50	2000	2000	**200**	**200**
	100	2000	2000	**200**	**200**
F_6	50	2000	2000	1000	**200**
	100	2000	2000	2000	**200**
F_7	50	1000	2000	2000	**500**
	100	1000	2000	2000	**500**
F_8	50	2000	2000	**1000**	2000
	100	2000	2000	2000	**500**
F_9	50	2000	2000	500	**200**
	100	2000	2000	1000	**200**
F_{10}	50	2000	2000	1000	**200**
	100	2000	2000	2000	**200**
F_{11}	50	1000	2000	**200**	2000
	100	2000	2000	**200**	2000
F_{12}	50	2000	2000	**1000**	2000
	100	2000	2000	**500**	2000
F_{13}	50	500	2000	**200**	500
	100	500	2000	**200**	1000

Population sizes used by the algorithms to generate the results in Table IV are shown. On each problem, the smallest population size is marked in bold.

TABLE VI
COMPARISON BETWEEN EEDA AND EDA-MCC ON 50-D−200-D F_{10}

n	EEDA	EDA-MCC
50	**2.0e+03 ± 2.0e+02** (1000)	3.1e+03 ± 3.4e+02 (200)
100	4.4e+03 ± 6.0e+02 (2000)	**1.9e+03 ± 3.6e+02** (200)
150	1.7e+04 ± 1.2e+03 (2000)	**3.1e+03 ± 4.0e+02** (500)
200	2.9e+04 ± 2.0e+03 (2000)	**4.3e+03 ± 7.7e+02** (500)

Population sizes used are shown in brackets. In each row, the significantly better result (determined by nonparametric Mann–Whitney U test) is shown in bold. For all results of EEDA, the value of asymp. sig. (two-tailed) < 0.001 when compared with the results of EDA-MCC.

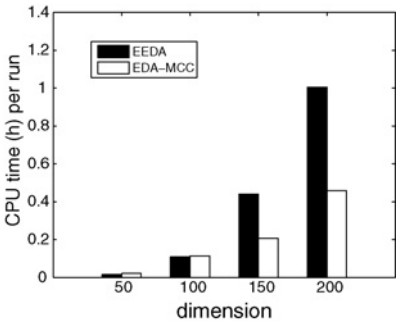

Fig. 9. Average CPU time of EEDA and EDA-MCC on F_{10}.

well due to its better explorative ability than EMNA$_{\text{global}}$, but not as good as UMDA$_c^G$ and EDA-MCC on 100-D F_2. Overall, EDA-MCC performs the best among all the multivariate EDAs with statistical significance, and it performs as well as UMDA$_c^G$. Also note that EMNA$_{\text{global}}$ and EEDA can perform worse when the optimum is shifted away (in F_2) from the center of search space (in F_1).

Although the CPU time of an algorithm may depend on population sizes and thus different number of generations, it reflects the computational time needed to exert an algorithm's best performance. We can find that UMDA$_c^G$ costs the least CPU time whereas EMNA$_{\text{global}}$ and EEDA cost the most. EDA-MCC's CPU time grows faster than UMDA$_c^G$ but slower than EMNA$_{\text{global}}$ and EEDA. Since F_1 and F_2 are easy for UMDA$_c^G$ model, the required population size indeed grows mildly. However, the population sizes of EMNA$_{\text{global}}$ and EEDA keep at high levels. EDA-MCC's requirement of large population size is significantly relaxed due to WI + SM. Meanwhile, EDA-MCC shows significantly better performance.

2) *Nonseparable Problems with Only A Few Local Optima:* This group of functions are either unimodal or only have two local optima, which implies the problems have clear inner structures. The nonseparable properties pose significant difficulties for UMDA$_c^G$. We can see that UMDA$_c^G$ fails to perform the best on any test. On the other hand, EDA-MCC performs the best on all tests except 50-D F_{10}. EMNA$_{\text{global}}$ performs the worst and EEDA performs generally between UMDA$_c^G$ and EDA-MCC. Note that F_4, F_6, and F_8 are shifted versions of F_3, F_5, and F_7, respectively. On the unshifted versions, although UMDA$_c^G$ and EEDA perform significantly worse than EDA-MCC, the solutions they found are not too bad. However, once the global optima are shifted away, their performance become much worse. EMNA$_{\text{global}}$ has a similar issue and it always performs the worst. Among all, only EDA-MCC shows robust performance to the shifts of the global optima.

The CPU time costs are similar to that of previous group of functions. EDA-MCC's CPU time grows much slower than EMNA$_{\text{global}}$ and EEDA. Although UMDA$_c^G$ costs the least CPU time, it always performs worse than EDA-MCC. EDA-MCC also needs the smallest population sizes in most cases, except on 50-D F_8. As we can see on F_{12} in the next group that, the best population size of EDA-MCC and EEDA can sometimes fluctuate as n grows. This can be explained as that since they have better explorative ability, they can benefit from either a) large population sizes, or b) large budget of number of generations (by applying a small population size). However, for UMDA$_c^G$ and EMNA$_{\text{global}}$ that fully relies on MLE, the population sizes usually keep increasing as n grows.

In this group, F_7–F_{10} are relatively hard problems that no algorithm achieves satisfying solutions. But to the best of our knowledge as well as we can see in the following 500-D tests that no known algorithm can find good solutions on these problems, and EDA-MCC is in fact the best so far in general. Among these problems, F_{10}'s global optimum is on the bounds of the domain, which requires explorative ability the most among all 13 problems. On 50-D F_{10}, EEDA performs the best since it has a global guidance of the gradient and a relatively good estimation can be obtained. On the other hand, because EDA-MCC partitions the search space, search along the approximated global gradient is not

so effective as EEDA. But as the problem size grows to 100-D, EDA-MCC significantly outperforms EEDA. This confirms the effectiveness of using subspace models to approximate the global estimation. In higher dimensional space where a precise global estimation is hard to obtain, approximating the global estimation by combination of subspace models can achieve better performance for EDA. To further verify the effectiveness of the combination of subspace models, we extend our experiments on F_{10} to 150-D and 200-D to compare EEDA and EDA-MCC. Experimental settings are the same as previous ones. The comparison is shown in Table VI and Fig. 9. We can see that if n grows even larger, combination of subspace models can be significantly better than a poor global model. EDA-MCC not only finds significantly better solutions, but also scales to larger problems better, i.e., with a much slower increase in CPU time cost.

On this group of functions, UMDA$_c^G$ cannot perform as well as EDA-MCC, but its computational cost is always much lower. One may wonder whether a bigger CPU time budget for UMDA$_c^G$ would lead to superior performances over EDA-MCC. In Fig. 10 we plot the averaged evolutionary curves of 25 runs for all algorithms in 100-D tests to give an answer. We can see that the evolutionary curves of UMDA$_c^G$ all quickly become flat as the algorithm proceeds. This implies the fact that even given more CPU time, UMDA$_c^G$ cannot find better solution but converges to a suboptimal one.

Another possible reason why UMDA$_c^G$ does not perform well is that the population sizes applied are still not large enough. Therefore, we further test even larger population sizes $M \in \{4000, 8000, 16\,000\}$ (and still $m = M/2$) for UMDA$_c^G$ on 100-D functions in this group. Results on representative functions are summarized in Table VII. We can see that larger population sizes do not help UMDA$_c^G$ obtain better results. To be specific, only on F_8 the result using $M = 4000$ becomes a little better, but still much worse than EDA-MCC. On other functions, large population sizes perform even worse. This implies that the failure of UMDA$_c^G$ on these functions is primarily due to its model simplicity, either larger population sizes or longer CPU time budget may not lead to better performance.

In a word, on this group of nonseparable functions, EDA-MCC performs significantly the best. UMDA$_c^G$ fails on all problems because of its model simplicity. EMNA$_{\text{global}}$ and EEDA cannot perform well in high-dimensional tests.

3) *Multimodal Problems with Many Local Optima:* These functions all have a huge number of local optima, which results in highly complicated function landscape and makes the problems hard to solve. Using the same sample size, the estimated multivariate model cannot be as reliable as on the previous group of problems. The results coincide with this intuition. Although F_{11} is separable, results show that it is not easy to solve for multivariate Gaussian EDAs. A previous study [11] has shown that if a small population size is applied, EMNA$_{\text{global}}$ and EEDA cannot perform well on this problem, and EEDA may even perform worse than EMNA$_{\text{global}}$. The huge number of local optima can mislead the multivariate search and the covariance matrix scaling. UMDA$_c^G$ performs the best and EMNA$_{\text{global}}$ the second on this function. Both EEDA and EDA-MCC adopting covariance matrix scaling fail to reach the optimum. Applying a rotation to F_{11} makes F_{12} nonseparable. Even the global optimum of F_{12} is shifted, compared with the results on F_{11} (see Table IV), surprisingly UMDA$_c^G$ still outperforms the others, whereas EMNA$_{\text{global}}$ becomes much worse. EEDA and EDA-MCC approximately hold the solution quality. Intuitively, nonseparable problem is hard for UMDA$_c^G$. However, the results reveal that high-dimensional F_{12} is even much harder for multivariate Gaussian model. On expanded multimodal function F_{13}, UMDA$_c^G$ again performs the best. It seems that the complicated problem structure of this group of functions poses similar difficulties to EDA-MCC, and simple algorithms such as UMDA$_c^G$ can be good enough on these problems. CPU time comparisons on these problems are similar to previous ones that EDA-MCC's CPU time is always between UMDA$_c^G$ and EMNA$_{\text{global}}$. Since EDA-MCC based on WI + SM cannot perform well, its required population size also becomes large.

4) *The Failure of EDA-MCC and the Success of UMDA$_c^G$ on F_{11}–F_{13}:* To further analyze the failure of EDA-MCC and the success of UMDA$_c^G$ on F_{11}–F_{13} (three problems sharing the common property of having a huge number of local optima), additional experiments are presented here. Generally speaking, the experiments concern two characteristics of EDAs that may be closely related to the performance on these problems. Our goal is to find the intrinsic reasons that prevent EDA-MCC from performing well.

The first characteristic we take into account is the model complexity of EDA. On a specific problem, a multivariate Gaussian EDA does not necessarily outperform a univariate Gaussian EDA. The failures of several multivariate Gaussian EDAs and the success of univariate Gaussian EDA (UMDA$_c^G$) on F_{11}, F_{12}, and F_{13} probably imply that using high dependency degree (i.e., high model complexity) for these problems is no longer effective. If such an intuition can be validated, then the failures of EDA-MCC are very likely to attribute to the failures of high dependency degree, not the novel WI + SM techniques adopted by EDA-MCC. Therefore, we test explicitly controlling the dependency degree by changing c, i.e., from original settings $c = 20$ to $c = 2$. Note that if $c = 1$, EDA-MCC will perform exactly the same as UMDA$_c^G$, and $c = 2$ restricts the multivariate dependencies to the minimal degree that at most dependencies of two variables are considered. We also add 10-D tests to see what happens in low dimension. Note that for 10-D tests, $c = 20$ is essentially identical to $c = 10$ since all variables can be included.

Another characteristic that may influence the performance of an EDA is the base multivariate model, which also indicates the method of estimating the probabilistic model. UMDA$_c^G$ adopts MLE, and EMNA$_{\text{global}}$ model is more similar to the UMDA$_c^G$ model than the others because of also using MLE. UMDA$_c^G$'s promising performance on the three problems may indicate that MLE is more efficient than covariance matrix scaling on these problems. Therefore, we replace the EEDA model with the EMNA$_{\text{global}}$ model in EDA-MCC to test the effect of base model. By crossing over the settings of base multivariate model and c, we have four candidates to be compared with UMDA$_c^G$: 1) EDA-MCC with EEDA model,

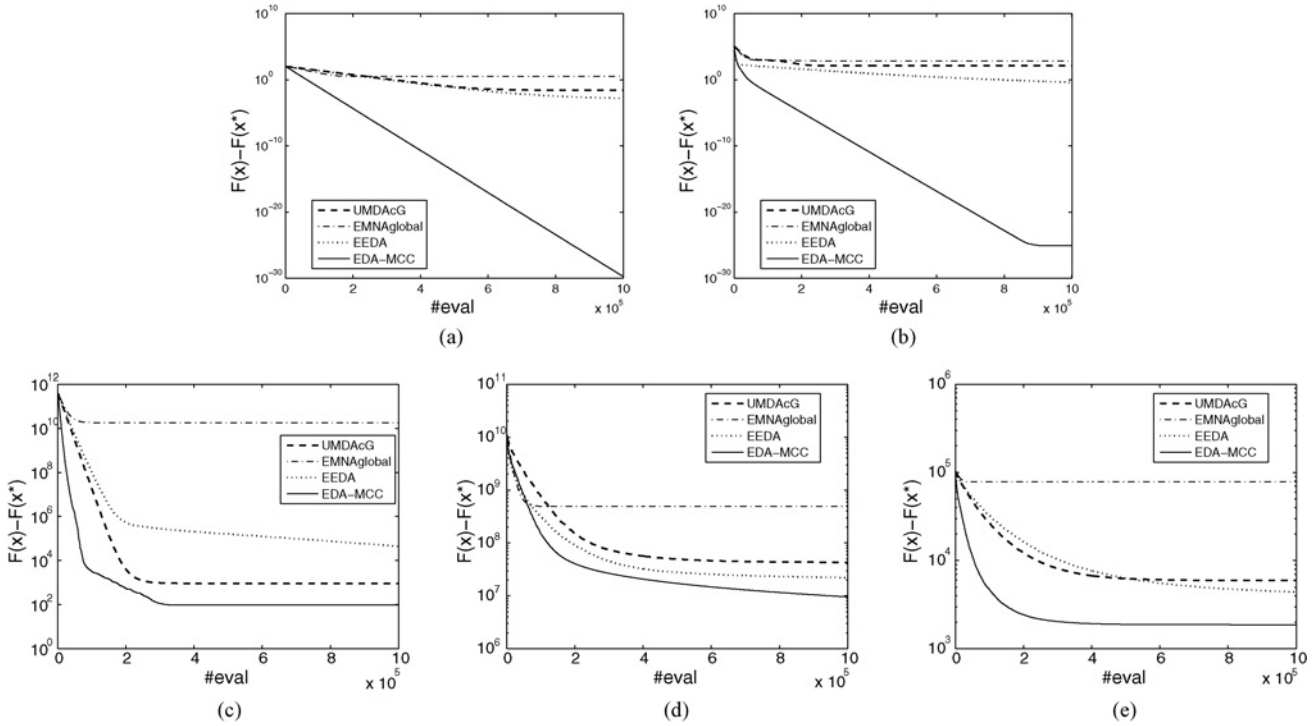

Fig. 10. Evolutionary curves on 100-D F_3, F_5, F_8, F_9, and F_{10}. Curves of F_4, F_6, F_7 are similar to that of F_3, F_5, F_8, respectively, and thus are omitted. (a) F_3. (b) F_5. (c) F_8. (d) F_9. (e) F_{10}.

TABLE VII

RESULTS OF UMDA$_c^G$ USING LARGE POPULATION SIZES ON 100-D F_3, F_5, F_8, F_9, AND F_{10}

Prob.	**EDA-MCC**	**UMDA$_c^G$, M = 2000**	**UMDA$_c^G$, M = 4000**	**UMDA$_c^G$, M = 8000**	**UMDA$_c^G$, M = 16000**
F_3	0 ± 0	2.6e-02 ± 8.3e-02	6.7e-02 ± 2.7e-03	2.6e+00 ± 8.7e-02	1.6e+01 ± 3.6e-01
F_5	0 ± 0	1.3e+02 ± 2.7e+01	1.3e+02 ± 1.7e+01	1.3e+02 ± 1.4e+01	7.4e+02 ± 3.3e+01
F_8	9.6e+01 ± 1.3e-01	9.3e+02 ± 3.1e+03	1.2e+02 ± 4.7e+01	2.4e+02 ± 4.4e+01	9.6e+05 ± 9.2e+04
F_9	9.6e+06 ± 2.5e+06	4.3e+07 ± 3.1e+06	4.9e+07 ± 2.7e+06	9.5e+07 ± 3.5e+06	4.2e+08 ± 3.3e+07
F_{10}	1.9e+03 ± 3.6e+02	5.9e+03 ± 4.3e+02	6.0e+03 ± 2.8e+02	9.1e+03 ± 2.0e+02	2.0e+04 ± 5.2e+02

Results of EDA-MCC and UMDA$_c^G$ using M = 2000 are also directly included from Table IV. On each problem, the value of asymp. sig. (two-tailed) < 0.001 when any UMDA$_c^G$ result is compared with EDA-MCC result using nonparametric Mann–Whitney U test.

TABLE VIII

COMPARISON OF DIFFERENT BASE MULTIVARIATE MODELS AND DIFFERENT SUBSPACE SIZES

Prob.	n	UMDA$_c^G$	EDA-MCC with EEDA model c = 20	EDA-MCC with EEDA model c = 2	EDA-MCC with EMNA$_{\text{global}}$ model c = 20	EDA-MCC with EMNA$_{\text{global}}$ model c = 2
F_{11}	10	**0 ± 0**	**0 ± 0**	**0 ± 0**	**0 ± 0**	**0 ± 0**
	50	**0 ± 0**	2.88e+02±1.36e+01§	2.96e+02±1.13e+01§	6.31e-08±1.52e-07§	4.81e-08±5.93e-08§
	100	**0 ± 0**	7.49e+02±1.61e+01§	7.96e+02±2.33e+01§	**0 ± 0**	1.52e-04±7.62e-04
F_{12}	10	5.83e-02±2.91e-01	**8.46e-04±2.86e-03**	1.68e-01±3.70e-01	1.59e-01±3.73e-01	1.33e-01±3.68e-01
	50	**2.08e+00±9.49e-01**	2.96e+02±1.46e+01§	2.97e+02±1.50e+01§	7.30e+00±2.47e+00§	8.70e+00±3.58e+00§
	100	**8.57e+00±2.07e+00**	7.41e+02±2.35e+01§	8.01e+02±1.61e+01§	2.66e+01±7.51e+00§	2.54e+01±3.96e+00§
F_{13}	10	1.33e+00±2.13e-01	**1.31e+00±2.57e-01**	1.33e+00±3.09e-01	1.45e+00±3.91e-01	1.46e+00±3.39e-01
	50	**7.77e+00±8.34e-01**	2.64e+01±9.20e-01§	2.59e+01±1.05e+00§	8.13e+00±1.37e+00	8.16e+00±1.58e+00
	100	**1.52e+01±1.98e+00**	6.53e+01±1.64e+00§	6.82e+01±2.09e+00§	1.63e+01±1.97e+00	1.66e+01±1.54e+00*

* The value of Asymp. Sig. (two-tailed) < 0.05 when compared with the results of UMDA$_c^G$.
† The value of Asymp. Sig. (two-tailed) < 0.01 when compared with the results of UMDA$_c^G$.
§ The value of Asymp. Sig. (two-tailed) < 0.001 when compared with the results of UMDA$_c^G$.

The best results for each row are shown in bold. Results of UMDA$_c^G$ are compared with results of each of the other four implementations of EDA-MCC by nonparametric Mann–Whitney U test. The significance level is shown by markers (*, †, and §). No marker implies no significant difference.

$c = 20$; 2) EDA-MCC with EEDA model, $c = 2$; 3) EDA-MCC with $\text{EMNA}_{\text{global}}$ model, $c = 20$; and 4) EDA-MCC with $\text{EMNA}_{\text{global}}$ model, $c = 2$. Still, for each implementation, four population sizes are applied in each test. The best result among the four population size tests is reported. The comparisons are summarized in Table VIII.

We can find that in 10-D tests, there is no significant difference among candidate algorithms. EDA-MCC can be as good as UMDA_c^G. In 50-D and 100-D tests, different degrees of multidependencies does not help EDA-MCC achieve as good performance as UMDA_c^G, no matter that the base model is EEDA model or EMNA model. This implies that on these problems, if the computational resources (max. #eval) are limited, utilizing multidependencies among variables may not be an effective strategy. To be specific, as long as considering the multidependencies, even only with the minimal degree ($c = 2$), the search is misled by the complex function landscape. As n grows, this effect becomes more serious. Nevertheless, changing from EEDA model to $\text{EMNA}_{\text{global}}$ model does help to find better solutions, although the results are not always as good as UMDA_c^G. It implies that, when n is large, the "radical" covariance matrix scaling can be easily misled by the complex function landscapes. On the other hand, the more "conservative" MLE performs better. Covariance matrix scaling is more effective only when n is small. Of course, discussions here are restricted to our predefined population sizes and the max. #eval. Since EDA-MCC can perform as good as UMDA_c^G on low-dimensional 10-D tests, we guess that with extremely large population size and sufficiently large budget of max. #eval, EDA-MCC has the potential to come up with or even outperform UMDA_c^G. But considering the fast increasing number of local optima and the fast increasing complexity of the function landscape as n grows, EDA-MCC's requirement of population size and #eval to outperform UMDA_c^G will also increase tremendously. This can also be explained by the effect of the curse of dimensionality. Therefore, when facing problems with many local optima, it may be computationally too expensive to apply a multivariate search and expect good performance. In this case, a cheap and simple univariate algorithm such as UMDA_c^G can be a better choice given limited computational resources.

D. Summary So Far

It is discovered by the above experiments that compared with traditional EDAs, EDA-MCC shows remarkable effectiveness and efficiency on high-dimensional nonseparable problems with only a few local optima. On simple separable problems, EDA-MCC is comparable with UMDA_c^G. But on problems with too many local optima, it does not work as well as simple UMDA_c^G. In any case, EDA-MCC offers a partial solution to the three requirements raised at the beginning of Section III.

1) The multivariate Gaussian-based search is preserved in EDA-MCC, which leads to promising performance on large-scale nonseparable problems.
2) Computational cost of EDA-MCC is usually lower than traditional multivariate Gaussian EDAs; its CPU time cost also grows much slower as problem size grows.
3) EDA-MCC can work with small population sizes for large-scale optimizations.

Conditions under which EDA-MCC may succeed or fail can also be summarized.

1) In low-dimensional search space with sufficient data, where the global estimation is sufficiently precise, EDA-MCC may not be better than traditional EDAs.
2) In high-dimensional search space with sparse data only, where the global estimation is far from precise, EDA-MCC can be more effective. However, if the function landscape has a huge number of local optima as in F_{11}–F_{13}, EDA-MCC as well as traditional multivariate Gaussian EDAs will fail. In this case, simple univariate Gaussian EDAs can be more effective and efficient.
3) The success of EDA-MCC does not mean that it can escape from the curse of dimensionality. EDA-MCC just suffers less from it by explicitly controlling the model complexity. If using a fixed finite population size, EDA-MCC and any other EDAs relying on learning will inevitably fail in extremely high-dimensional search space.

We also note that although EDA-MCC can have better performance than traditional EDAs, in some cases (e.g., on problems F_9 and F_{10}), none of the candidates perform well enough to find a high quality solution. More effective and efficient search strategies for large-scale optimization are still to be developed.

E. Experimental Results on 500-D Functions

Now, we further enlarge the problem size of F_1–F_{13} to 500-D, and compare EDA-MCC with traditional EDAs and several optimization algorithms specifically designed for large-scale optimization. The involved traditional EDAs include UMDA_c^G and MIMIC_c^G [2]. MIMIC_c^G is also a Gaussian EDA, whose model complexity is between UMDA_c^G and those multivariate Gaussian EDAs. The variable dependency in MIMIC_c^G is a chain-shaped structure with bivariate conditional Gaussian densities. Multivariate Gaussian EDAs such as $\text{EMNA}_{\text{global}}$, EEDA, and EGNA, are not included because their CPU time on any of the 500-D functions is too long to be acceptable.

Recently, Yang *et al.* [54] proposed a cooperative coevolution framework for large-scale optimization, and an algorithm named DECC-G, which uses differential evolution (DE) as the base algorithm in the framework, was proposed. DECC-G also adopts variable partitioning strategy, but within the cooperative coevolution framework, when DECC-G is activating the variables of one group, all the other variables are fixed. The evaluation of currently activated variables are calculated in the context of fixing other variables. In EDA-MCC, though variables are also grouped into several subsets, their optimizations are simultaneous and synchronized. EDA-MCC is not an instance of cooperative coevolution. In [54], DECC-G has been compared with three other algorithms, SaNSDE, FEPCC, and DECC-O, on several 500-D and 1000-D functions, and it shows outstanding performance compared

TABLE IX
COMPARISONS OF 500-D TESTS

Prob.	SaNSDE	FEPCC	DECC-O	DECC-G	UMDA$_c^G$	MIMIC$_c^G$	EDA-MCC	sep-CMA-ES
F_1	2.41e-11	4.90e-08	**0**	**0**	**0 ± 0**	**0 ± 0**	**0 ± 0**	**0 ± 0**
F_2	2.61e-11	–	1.04e-12	**0**	**0 ± 0**	2.56e+02 ± 2.2e+02§	**0 ± 0**	**0 ± 0**
F_3	4.07e+01	9.00e-05	6.01e+01	**4.58e-05**	1.35e+01 ± 2.9e+00§	4.40e-01 ± 1.4e-01§	2.79e-01 ± 2.3e-02	1.40e+02 ± 1.4e+01§
F_4	8.29e+01	–	1.05e+02	7.00e+01	6.92e+01 ± 4.2e+00§	7.93e+01 ± 4.8e-01§	**3.27e-01 ± 3.7e-02**	1.41e+02 ± 1.2e+01§
F_5	9.30e-07	–	1.37e+02	6.66e-08	2.60e+03 ± 2.8e+02§	2.03e+02 ± 2.1e+01§	**0 ± 0**	**0 ± 0**
F_6	1.02e-06	–	1.44e+02	9.59e-08	6.61e+03 ± 8.7e+02§	1.07e+03 ± 2.6e+01§	**0 ± 0**	**0 ± 0**
F_7	1.33e+03	–	6.64e+02	4.92e+02	4.96e+02 ± 1.4e+01	4.93e+02 ± 8.6e-02	6.42e+02 ± 4.1e+02	**2.91e+02 ± 2.6e+01**§
F_8	2.71e+03	–	1.71e+03	1.56e+03	3.44e+04 ± 9.8e+04§	3.75e+08 ± 8.5e+07§	6.77e+02 ± 6.3e+02	**2.87e+02 ± 2.9e+01**§
F_9	6.88e+08	–	4.78e+08	3.06e+08	4.72e+08 ± 1.6e+07§	4.44e+08 ± 7.1e+06§	8.03e+07 ± 1.1e+07	**7.98e+07 ± 1.7e+07**
F_{10}	4.96e+05	–	2.40e+05	1.15e+05	3.48e+04 ± 8.4e+02§	1.03e+05 ± 7.8e+02§	**2.09e+04 ± 1.3e+03**	1.20e+05 ± 9.4e+03§
F_{11}	2.84e+02	1.43e-01	1.76e+01	**0**	2.27e+00 ± 1.2e+00§	4.80e+03 ± 4.0e+01§	5.24e+03 ± 3.9e+01	2.14e+03 ± 9.9e+01§
F_{12}	6.97e+03	–	1.50e+04	5.33e+03	**7.55e+01 ± 6.5e+00**§	5.03e+03 ± 4.7e+01§	5.25e+03 ± 4.2e+01	2.28e+03 ± 1.8e+02§
F_{13}	2.53e+02	–	**2.81e+01**	2.09e+02	7.90e+01 ± 3.1e+00§	4.73e+02 ± 4.7e+00§	4.52e+02 ± 5.0e+00	1.03e+02 ± 7.1e+00§

§ The value of Asymp. Sig. (two-tailed) < 0.001 when compared with the results of EDA-MCC.

For each problem, the best result is bolded. Since results of SaNSDE, FEPCC, DECC-O, and DECC-G in [54] only contain mean performance, we are not able to give standard deviations. Results of EDA-MCC are compared with results of UMDA$_c^G$, MIMIC$_c^G$, and sep-CMA-ES, respectively, by nonparametric Mann–Whitney U test. The significance level is indicated by marker §. Some results of FEPCC are not reported in [54], and are thus left blank. Two-tailed Friedman test shows that all algorithms (except FEPCC whose data is not available) are not equivalent at the significance level of 0.05. Post-hoc Nemenyi tests demonstrate that EDA-MCC outperforms SaNSDE, DECC-O, and MIMIC$_c^G$ at the significance level of 0.05 [55]. Moreover, according to one-tailed Wilcoxon signed ranks tests, EDA-MCC outperforms UMDA$_c^G$ at the significance level of 0.15. At the same significance level, EDA-MCC does not significantly outperform DECC-G and sep-CMA-ES.

with other algorithms. Here, we compare EDA-MCC with the results reported in [54].[4]

Another algorithm, sep-CMA-ES, recently proposed by Ros and Hansen [56], is also included in the comparison. Because the original CMA-ES is incapable of handling problems with more than several hundred dimensions [57], sep-CMA-ES was developed only using a diagonal covariance matrix in a Gaussian model while keeping the original covariance matrix adaptation. Several recent studies (e.g., [56] and [57]) investigated its performance on problems larger than 500-D. Although sep-CMA-ES uses a diagonal covariance matrix as well as UMDA$_c^G$, their model estimations are far different. One major difference is that sep-CMA-ES relies on cumulation of the information gathered in the evolution path to model the covariance matrix, which is mainly heuristic-based, and thus only requires a very small population size. In contrast, a typical EDA, such as UMDA$_c^G$, estimates the covariance matrix only by samples in current generation with MLE, which is learning-based; thus, it usually requires a much larger population size than sep-CMA-ES does. As will be seen later in experiments, this could lead to significantly different performance. We use recommended parameters of sep-CMA-ES [56] to conduct the comparison, with population size $\lambda = 4 + \lfloor 3\ln(n) \rfloor$ (i.e., 22 when $n = 500$), selected size $\mu = \lfloor \frac{\lambda}{2} \rfloor$, initial standard deviation (step size σ) identical to one third of the search interval, and initial search point the center of the search space. The implementation of sep-CMA-ES is derived from a C version of CMA-ES.[5]

Following [54], we set the max. #eval to 2.5e+06. The population size of DECC-G is 100 and its subcomponent dimension is 100 for all tests. For the parameters of SaNSDE, FEPCC, and DECC-O please refer to [54]. For UMDA$_c^G$ and MIMIC$_c^G$, population size $M = 2000$ and selected size $m = 1000$ are adopted. In EDA-MCC, still we use UMDA$_c^G$ model for $\mathcal{W}$ and EEDA model for subsets of $\mathcal{S}$. We set population size $M = 200$, selected size $m = 100$, $m_{\text{corr}} = 100$, $\theta = 0.3$, and $c = 100$ for all tests. If $M = 200$ is too small for solving a problem, we consequently test $M = 500$ and $M = 1000$ to see whether better performance can be obtained while keeping the selection pressure. We highly trust the small population sizes that for $c = 100$ dimensional subspace, we still have confidence in the subspace models. The result is that EDA-MCC needs $M = 1000$ on F_3, F_4, and F_{10}, and only $M = 200$ on all other functions. Other experimental settings are the same as previous ones. Detailed comparisons are summarized in Table IX.

On the simplest separable F_1 and F_2, EDA-MCC, UMDA$_c^G$, DECC-O, DECC-G, and sep-CMA-ES perform very well. On the second group of nonseparable functions F_3–F_{10}, EDA-MCC and sep-CMA-ES show the most stable good performance. Interestingly, although sep-CMA-ES only adopts diagonal covariance matrix, it performs generally well on these nonseparable functions, which was also reported in [57]. But only on two Ronsenbrock functions (F_7 and F_8) it significantly outperforms EDA-MCC, whereas EDA-MCC significantly outperforms sep-CMA-ES on F_3, F_4, and F_{10}. Both EDA-MCC and sep-CMA-ES reach the global optimum on F_5 and F_6. On F_9, although sep-CMA-ES has a little better average performance, there is no significant difference with EDA-MCC's. If we compare DECC-G with EDA-MCC, only on F_3 and F_7, DECC-G performs better than EDA-MCC. But DECC-G is rather sensitive to the shifted global optimum. On the shifted functions F_4 and F_8, EDA-MCC performs well holding almost the same performance, whereas DECC-G becomes much worse. Similar situations happen on F_{11} and its shifted rotated version F_{12}. EDA-MCC is not sensitive to the shifted and rotated function landscape as DECC-G is.

[4]Results on F_4–F_6 are not available in [54]. These results are obtained by running the source code provided by the authors of [54].

[5]http://www.lri.fr/~hansen/cmaes_c.tar.

TABLE X
PERFORMANCE COMPARISONS OF DIFFERENT θ AND c ON 100-D F_2. RESULTS ARE AVERAGED OVER 25 RUNS

	$c = 5$	$c = 10$	$c = 20$	$c = 30$	$c = 40$	$c = 50$
$\theta = 0.2$	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0
$\theta = 0.25$	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0
$\theta = 0.3$	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0
$\theta = 0.35$	0 ± 0	1.96e-01±9.82e-01	0 ± 0	0 ± 0	7.2e-02±3.6e-01	0 ± 0
$\theta = 0.4$	8.2e+00±3.5e+01	1.8e+00±9.0e+00	9.8e-02±3.7e-01	2.8e-03±1.4e-02	1.8e-05±8.9e-05	1.1e+00±4.6e+00

TABLE XI
PERFORMANCE COMPARISONS OF DIFFERENT θ AND c ON 100-D F_8. RESULTS ARE AVERAGED OVER 25 RUNS

	$c = 5$	$c = 10$	$c = 20$	$c = 30$	$c = 40$	$c = 50$
$\theta = 0.2$	4.4e+06±2.1e+07	9.5e+01±2.9e-01	2.3e+02±6.9e+02	9.6e+01±1.1e-01	9.6e+01±2.1e-01	9.6e+01±3.9e-01
$\theta = 0.25$	1.1e+02±8.0e+01	9.5e+01±2.0e-01	9.6e+01±1.4e-01	1.3e+02±1.6e+02	9.6e+01±9.0e-02	9.6e+01±5.0e-01
$\theta = 0.3$	9.9e+01±1.2e+01	9.9e+01±1.3e+01	9.6e+01±1.3e-01	9.7e+01±1.2e-01	9.7e+01±2.1e-01	9.7e+01±3.9e-01
$\theta = 0.35$	2.1e+04±7.3e+04	2.2e+02±2.4e+02	7.9e+02±2.4e+03	9.5e+03±2.6e+04	7.7e+03±3.3e+04	1.2e+03±3.2e+03
$\theta = 0.4$	6.3e+06±1.4e+07	1.3e+06±1.6e+06	1.2e+06±2.3e+06	1.4e+06±4.0e+06	2.5e+06±6.0e+06	1.1e+06±2.3e+06

For the last group of functions (F_{11}–F_{13}), having a huge number of local optima, as analyzed above, UMDA$_c^G$ shows clear advantage in general. On F_{13}, DECC-O and UMDA$_c^G$ performs much better than the others. This is consistent with previous observations. Because DECC-O optimizes the function of one variable at a time within the cooperative coevolution framework, its behaviors are similar to some extent, to UMDA$_c^G$. Therefore, they should be more effective on these problems. The exception that DECC-O fails on F_{12} can be explained by its sensitiveness to the shifted global optimum. As for sep-CMA-ES, although it also uses univariate model, its performance on F_{11}–F_{13} is far worse than UMDA$_c^G$. This seems to be due to the heuristics by which the covariance matrix is estimated in sep-CMA-ES. The results show that the standard "conservative" MLE adopted in UMDA$_c^G$ can be more effective than the heuristics adopted in sep-CMA-ES on large-scale problems with many local optima.

We also find that MIMIC$_c^G$ fails to perform the best on any problem. Due to more suffering from the curse of dimensionality, it is neither as effective as UMDA$_c^G$ on problems that simple univariate model can already handle, nor as good as EDA-MCC on nonseparable problems with clear structure. The results again validate our analysis on the difficulties of traditional EDAs in high-dimensional search spaces.

Generally speaking, EDA-MCC, with a relatively small population size, shows robust performance on these 500-D problems, especially on nonseparable problems with only a few local optima. It performs statistically better than SaNSDE, DECC-O, UMDA$_c^G$, and MIMIC$_c^G$. Although DECC-G also performs generally well, its sensitiveness to shifted global optimum is evidently a disadvantage. Sep-CMA-ES also performs generally well, notably on nonseparable problems (F_5–F_8), which is interesting considering the univariate nature of the Gaussian model. This could be a topic worthy of study in future work. In a word, we can say that EDA-MCC is the first successful application of multivariate EDA on a general class (13 in total) of 500-D problems since continuous EDAs have been proposed. Moreover, compared with other EAs, EDA-MCC and UMDA$_c^G$ show their significant superiority on 8 out of the 13 functions, implying the advantages of using probabilistic models and statistical learning for optimization. Also note that we did not tune the parameters of EDA-MCC further on specific problems. Its potential performance can be even better on real-world large-scale problems.

V. INFLUENCE OF PARAMETERS θ AND c

In this section, the dependence of EDA-MCC on the newly introduced parameters θ and c are investigated. Guidelines of setting these parameters are also given.

A. Influence Tests

A separable function F_2 and a nonseparable function F_8 are selected from the 13 test functions (Table III) as demonstration. Different combinations of θ and c are tested on the two functions with problem size $n = 100$, where $\theta \in \{0.2, 0.25, 0.3, 0.35, 0.4\}$ and $c \in \{5, 10, 20, 30, 40, 50\}$. The population size and selected size are adopted from previous experiments of EDA-MCC and kept fixed, i.e., $M = 1000, m = 500$ for F_2, and $M = 500, m = 250$ for F_8. The performance comparison of combinations of θ and c are summarized in Tables X and XI.

From the results we can see that on separable F_2, as long as $\theta \leq 0.3$, different c does not change the performance. But when $\theta > 0.3$, the performance becomes a little unstable. Note that because current implementation of EDA-MCC uses EEDA model on subsets of $\mathcal{S}$, even when adopting a large θ, as long as $\mathcal{S}$ is not empty, EDA-MCC performs differently from UMDA$_c^G$. When variable dependencies are overly eliminated by large θ, according to the definition of covariance matrix scaling, the performance can become unstable since the gradient is likely to be poorly approximated. But generally speaking, on separable problems different θ and c do not have much impact on EDA-MCC's performance.

On nonseparable F_8, only when $\theta \leq 0.3$, different c does not much change the best performance thus far, except when combining with a very small c. Large θ (> 0.3) can make $\mathcal{S}$ easily become empty, which is hazardous to solving nonseparable problems. Large c is not harmful for solving

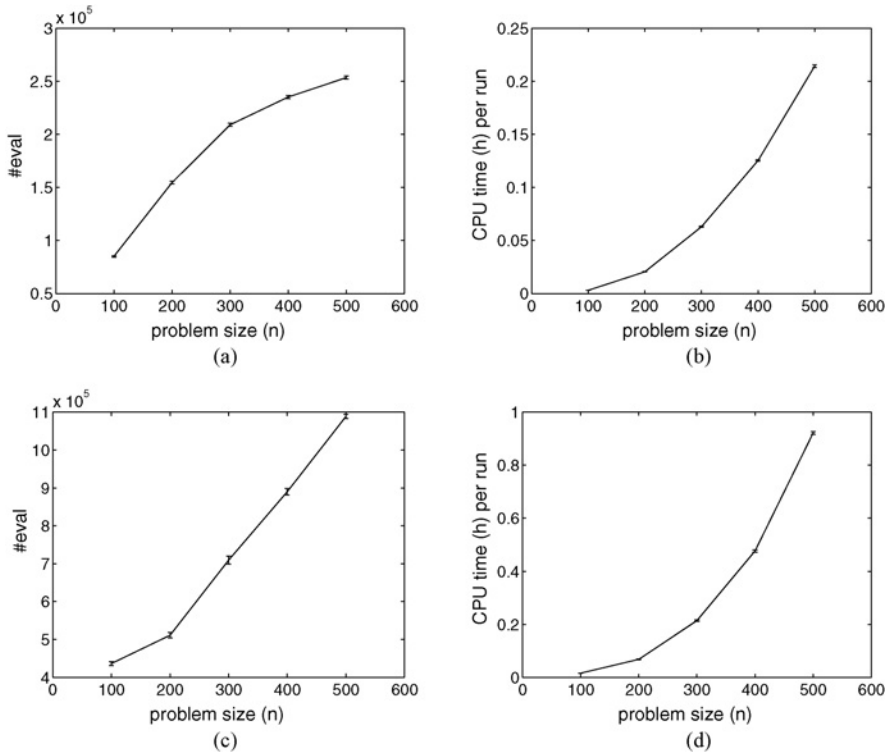

Fig. 11. Scalability results of EDA-MCC on F_1 and F_5. (a) #eval on F_1. (b) CPU time on F_1. (c) #eval on F_5. (d) CPU time on F_5.

nonseparable problems, although it may cost longer CPU time as analyzed before. However, too small c has similar effect of large θ that the dependencies between variables are overly eliminated. Since the partition of $\mathcal{S}$ is random, considering the nonseparability, it further makes covariance matrix scaling fail together with a small θ. We can conclude that too large θ is hazardous for nonseparable problems. Also, a too small c is not recommended either because it has similar effect to a large θ.

Generally, setting θ around 0.3 is good for these problems. With such a θ, the value of c does not impact overall performance much when population size is sufficiently large, but may lead to different CPU time cost according to Table II.

B. Guidelines of Setting θ and c

Intensive experiments in Section IV have suggested that, in most cases, a population size no larger than 2000 (and often, only 200) is sufficient for EDA-MCC to obtain satisfactory results on problems no larger than 500-D. Besides, a constant selection pressure $\tau = 0.5$ and a constant $m_{\text{corr}} = 100$ also seems enough for dealing with these 50–500-D problems. With such settings, θ around 0.3 will be good in most cases. For the value of c, considering: 1) the CPU time cost is very often necessary to care about in lots of real-world applications, and 2) a too large c close to n (especially when n is very large) also requires a sufficiently large population size to have reliable subspace model estimation and thus increases the computational cost, we suggest to set c a linear fraction of the problem size n, e.g., $c = n/5$. Note that as shown in the above influence tests, when the population size is sufficiently large, different c impacts little on performance but may result in different CPU time efficiency for problem solving. In the next section, we will demonstrate the scalability of EDA-MCC under these parameter setting guidelines.

VI. Scalability of EDA-MCC

In this section, we study the scalability of EDA-MCC in terms of CPU time cost and number of function evaluations (#eval) needed to reach the global optimum. On different problems, the scalability of EDA-MCC may be different. Here, two test functions, separable F_1 and nonseparable F_5, on which EDA-MCC can find the global optimum with acceptable time, are selected for empirical studies on problem sizes $n \in \{100, 200, 300, 400, 500\}$. The algorithm terminates only when the global optimum is reached. We also use the four population size settings as in Section IV, and select the result with the least #eval for plotting. Interestingly, EDA-MCC with population size $M = 200$, is always the fastest in reaching the optimum in these tests. And, results with the least #eval also always cost the least CPU time among the four population size tests. We set $\tau = 0.5$, $m_{\text{corr}} = 100$, $\theta = 0.3$, and $c = n/5$ as Section V suggests. All results are averaged over 25 independent runs and obtained on a computer with Intel Core2 2.66 GHz CPU and 3GB RAM. Fig. 11 depicts the #eval and the CPU time (mean and error bars) needed in solving the two problems.

Under the above parameter settings, we find that the #eval needed to find the global optimum grows mildly for the two benchmarks. On simple separable F_1, the growing speed even decreases as n grows. On nonseparable F_5, it grows approx-

SM-GC

1) Construct $\mathcal{S}$ according to (4).
2) Partition $\mathcal{S}$ into non-intersected subsets $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_k, 1 \leq k \leq n$:
   a) $i \leftarrow 1$.
   b) **Repeat** until $\mathcal{S} = \emptyset$.
      i) Find two variables $X_1, X_2 \in \mathcal{S}$ maximizing $|corr(X_1, X_2)| > \theta$. Exit current loop if not found.
      ii) Create $\mathcal{S}_i \leftarrow \{X_1, X_2\}$. $\mathcal{S} \leftarrow \mathcal{S} \setminus \mathcal{S}_i$.
      iii) **Repeat** while $|\mathcal{S}_i| < c$, where c defines the maximal size of a subset ($2 \leq c \leq n$).
         A) Find a variable $X \in \mathcal{S}$ maximizing $|corr(X, Y)| > \theta$, where $\forall Y \in \mathcal{S}_i$. Exit current loop if not found.
         B) $\mathcal{S}_i \leftarrow \mathcal{S}_i \bigcup \{X\}$. $\mathcal{S} \leftarrow \mathcal{S} \setminus \{X\}$.
      iv) $i \leftarrow i + 1$.
   c) If $\mathcal{S} \neq \emptyset$, estimate a univariate model for the rest variables in $\mathcal{S}$.
3) Estimate a multivariate model for each subset based on the m selected individuals.

Fig. 12. Subspace modeling by greedy clustering (SM-GC). Note that the partition step is changed from original SM and minimal value of c is changed to 2 since there is no need to cluster if $c = 1$. Parameter θ here is the same as defined in (3).

imately lineally as n. On both problems, the CPU time costs needed also grow mildly, that is, approximately quadratically. The small intervals between the error bars in Fig. 11 also imply the stable performance of EDA-MCC. In a word, EDA-MCC shows good scalability on the two problems investigated, at least for problem sizes $\leq$ 500-D. Since the experiments in Section IV have demonstrated that EDA-MCC can have better performance and need less CPU time than traditional multivariate Gaussian EDAs on large-scale problems, we can expect EDA-MCC to have better scalability than traditional multivariate Gaussian EDAs in general.

VII. Subspace Modeling By Clustering Variables

In EDA-MCC, we randomly partition $\mathcal{S}$ into subspaces in SM. One may ask whether a more sophisticated way of partitioning $\mathcal{S}$ can be applied, e.g., partitioning subspaces by clustering the variables in $\mathcal{S}$ based on the strength of the interdependencies. Intuitively, such a method should work well when sample size is large enough compared with the problem size n. But as n grows very large (e.g., $n = 500$) and only a small sample size is available (e.g., population size $M = 200$ and selected size $m = 100$), its performance may not be as good as random partition since any learning method, including unsupervised clustering, will be greatly affected by the curse of dimensionality. In this section, we replace the SM in EDA-MCC with a greedy clustering method named SM-GC (Subspace Modeling by Greedy Clustering), and compare it with original EDA-MCC. The new resulting algorithm is called EDA-MCC-GC.

The details of SM-GC are shown in Fig. 12. SM-GC partitions subspaces in the following steps. First, a pair of variables, whose absolute correlation is the largest among the ones above θ, is picked up from $\mathcal{S}$ as an initial cluster. This implies the pair of variables are the most strongly dependent among all. Then, a variable outside the cluster is selected and added to the cluster, on the condition that its correlation to the existing variables in the cluster is the strongest. The operation iterates until the cluster reaches the maximal size c or no strongly dependent variable can be found from the perspective of the cluster. A cluster refers to a partitioned subspace. Then, the dependencies between the clusters and the rest of the variables in $\mathcal{S}$ are eliminated. An outer loop keeps generating new subspaces in a greedy manner until all variables in $\mathcal{S}$ have been partitioned or when there is no strongly dependent variables left. If after clustering, $\mathcal{S}$ is still nonempty, a univariate model is applied to the rest of the variables since they are now regarded weakly dependent by the algorithm.

We compare EDA-MCC-GC with EDA-MCC on three representative problems: F_2, F_8, and F_{11} with $n \in \{50, 100\}$. Population sizes, parameters θ and c of EDA-MCC-GC, are set the same as used in EDA-MCC in previous 50-D and 500-D experiments. Results and parameters used are summarized in Table XII. We can find that on 50-D tests, there is no significant difference between EDA-MCC-GC and EDA-MCC. However, on 500-D tests where a small sample size is applied, EDA-MCC performs significantly better. This verifies our previous discussion that when applied to large-scale problems with a small sample size, partitioning subspaces based on clustering might not be as effective as random partition. Although the illustrative experiments cannot exclude the possibility that some delicate clustering approach might outperform random partition on specific large-scale problems, a clustering approach often requires relatively higher computational cost. In contrast, random partition is simple and efficient, which can be considered a default component of EDA-MCC.

VIII. Characterization of Problem Properties By EDA-MCC

As our motivation of scaling up EDA, we regard a major advantage of using EDA other than traditional EA is that we can obtain some feedback on the problem properties through observing the probabilistic model learned. We believe that the learned model structure should reflect some underlying properties of the problem. In addition to finding a solution, EDA has its unique capability in this aspect. However, such an advantage of EDA has not been deeply investigated. In a recent paper [58], discrete EDA model has been used to represent interactions between the protein conformations by probability models. But still, rare study has been done on continuous EDA models to characterize the structure of optimization problems.

In EDA-MCC, we can do so by visually analyzing the model structure obtained from WI + SM. When running EDA-MCC in previous experiments, we also record the results of WI

TABLE XII

COMPARISONS BETWEEN EDA-MCC-GC AND EDA-MCC ON 50-D AND 500-D F_2, F_8, AND F_{11}

Prob.	n	EDA-MCC-GC	EDA-MCC	Parameters
F_2	50	**0 ± 0**	**0 ± 0**	$M = 200, m = 100, m_{\text{corr}} = 100, \theta = 0.3, c = 20$
	500	1.32e+05 ± 2.73e+05§	**0 ± 0**	$M = 200, m = 100, m_{\text{corr}} = 100, \theta = 0.3, c = 100$
F_8	50	4.78e+01 ± 2.34e-01	**4.77e+01 ± 1.52e-01**	$M = 2000, m = 1000, m_{\text{corr}} = 100, \theta = 0.3, c = 20$
	500	6.32e+11 ± 1.29e+12§	**6.77e+02 ± 6.28e+02**	$M = 200, m = 100, m_{\text{corr}} = 100, \theta = 0.3, c = 100$
F_{11}	50	3.00e+02 ± 1.45e+01	**2.88e+02 ± 1.36e+01**	$M = 2000, m = 1000, m_{\text{corr}} = 100, \theta = 0.3, c = 20$
	500	6.25e+03 ± 1.01e+03§	**5.24e+03 ± 3.86e+01**	$M = 200, m = 100, m_{\text{corr}} = 100, \theta = 0.3, c = 100$

§ The value of Asymp. Sig. (two-tailed) < 0.001 when compared with the results of EDA-MCC.

For each test, the best result is bolded. Results of EDA-MCC are directly from Tables IV and IX. Results of EDA-MCC are compared with that of EDA-MCC-GC by nonparametric Mann–Whitney U test.

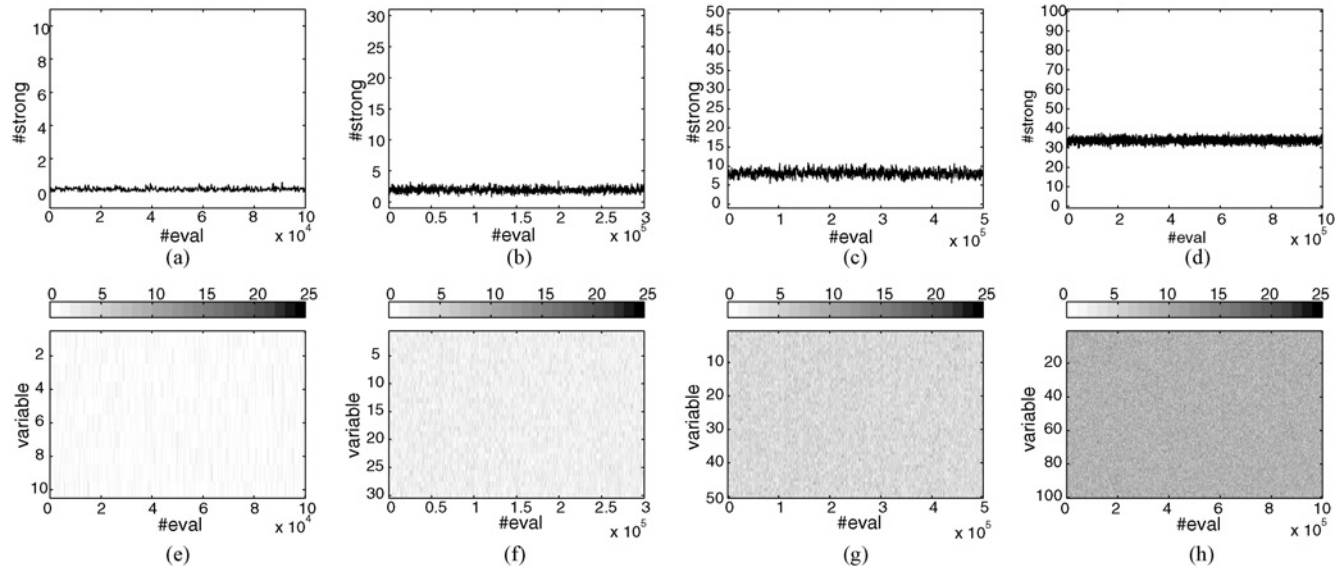

Fig. 13. WI results on F_1: sphere. The darker the element of $\boldsymbol{Q}$ is, the more times a variable is partitioned into $\mathcal{S}$ at the specific #eval during the 25 runs. (a) 10-D average #strong. (b) 30-D average #strong. (c) 50-D average #strong. (d) 100-D average #strong. (e) 10-D $\boldsymbol{Q}$. (f) 30-D $\boldsymbol{Q}$. (g) 50-D $\boldsymbol{Q}$. (h) 100-D $\boldsymbol{Q}$.

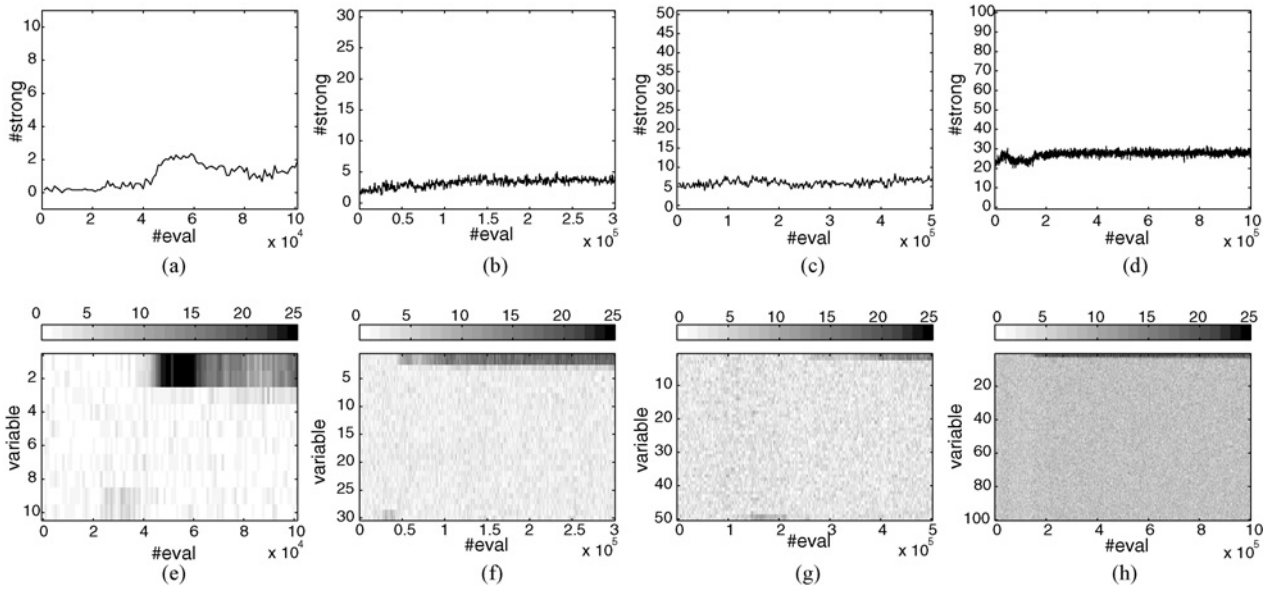

Fig. 14. WI results on F_8: shifted Rosenbrock. The darker the element of $\boldsymbol{Q}$ is, the more times a variable is partitioned into $\mathcal{S}$ at the specific #eval during the 25 runs. (a) 10-D average #strong. (b) 30-D average #strong. (c) 50-D average #strong. (d) 100-D average #strong. (e) 10-D $\boldsymbol{Q}$. (f) 30-D $\boldsymbol{Q}$. (g) 50-D $\boldsymbol{Q}$. (h) 100-D $\boldsymbol{Q}$.

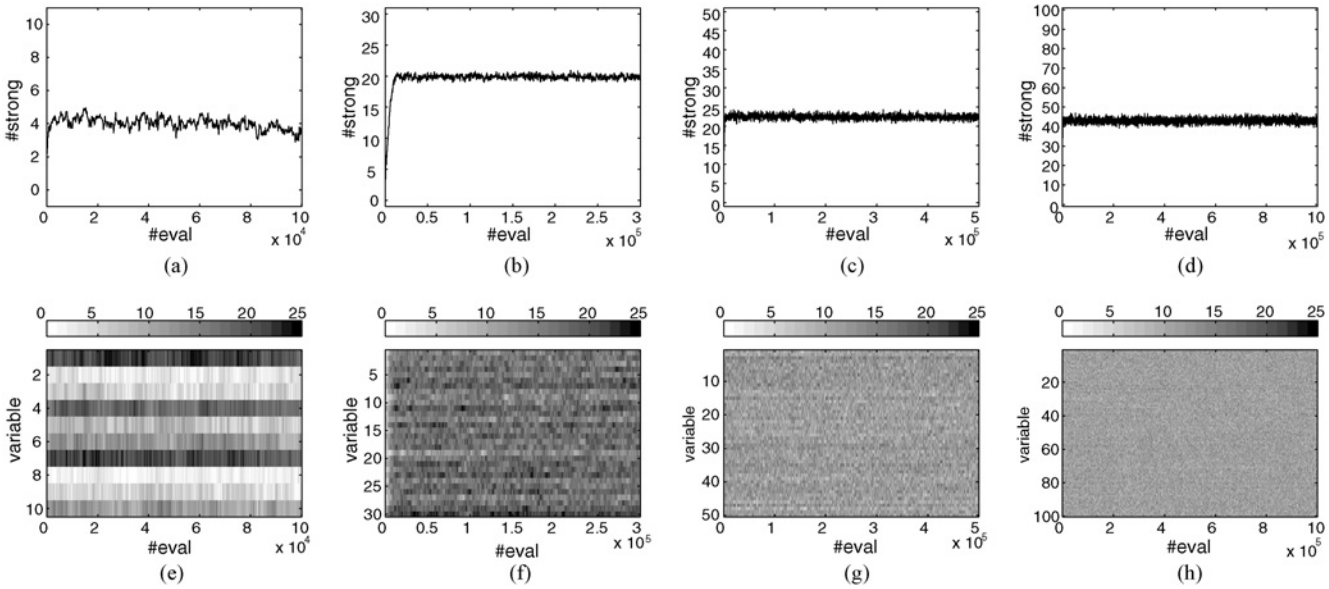

Fig. 15. WI results on F_9: shifted rotated high conditioned elliptic. The darker the element of $\boldsymbol{Q}$ is, the more times a variable is partitioned into $\mathcal{S}$ at the specific #eval during the 25 runs. (a) 10-D average #strong. (b) 30-D average #strong. (c) 50-D average #strong. (d) 100-D average #strong. (e) 10-D $\boldsymbol{Q}$. (f) 30-D $\boldsymbol{Q}$. (g) 50-D $\boldsymbol{Q}$. (h) 100-D $\boldsymbol{Q}$.

procedure in every generation. By analyzing these records, we can give in-depth analysis on the problem properties characterization capability of EDA-MCC. We record the number of strongly dependent variables (#strong), i.e., $|\mathcal{S}|$, and the elements in $\mathcal{S}$. The curves of the average #strong of multiple (25 in all previous experiments) runs during evolution thus can be plotted. Which variables are partitioned into $\mathcal{S}$ can also be plotted by a matrix $\boldsymbol{Q}$. Each row of $\boldsymbol{Q}$ corresponds to a variable. Each column corresponds to one generation. Its element $\boldsymbol{Q}_{ij}$ on the ith row and the jth column, ranging from 0 to 25, indicates how many runs (out of the 25 runs) partitioned variable x_i into $\mathcal{S}$ at generation j. Because visually examining matrix $\boldsymbol{Q}$ with 50 or 100 rows is relatively hard for human eyes, we conduct additional 10-D and 30-D experiments for EDA-MCC. Results of 500-D experiments are even harder to read so we omit them here. The 10-D and 30-D tests are based on the same settings as previous 50-D and 100-D experiments in Section IV. Since an $n \in \{10, 30\}$ is relatively small, it is easier to visually examine the results and summarize the changing trends as n grows. For the purpose of comparing average #strong and matrix $\boldsymbol{Q}$ in the same figure more clearly, we transform the column of $\boldsymbol{Q}$, which indicates the number of generations into the number of evaluations (#eval) in all the following figures. The horizontal axis of average #strong graphs is also converted to #eval. Due to the page length limit, here we only report the results on F_1, F_8, F_9, and F_{12}. Although the results seem to be the solo effect of WI, actually SM plays an important role in working with WI. The interactions between WI and SM will be analyzed in Section IX.

From Fig. 13 we can see that on separable F_1, #strong remains at a low level. But as n grows, the level of #strong also becomes higher. It can be interpreted as the effects of data sparsity in higher dimensional space. Using fixed θ through all experiments, the size of $\mathcal{W}$ can reduce as the search space enlarges (thus #strong can increase), because EDA-MCC may capture some correlations that do not actually exist. The relatively low level of #strong is consistent with the separability of the problem. Furthermore, the gray levels of matrices $\boldsymbol{Q}$ are nearly uniform, indicating that all the variables in $\mathcal{S}$ are observed to play identical roles for optimizing. It is also consistent with the function's expression.

Fig. 14 shows that EDA-MCC correctly recognizes the problem structures of shifted Rosenbrock F_8. The variable dependency of the problem is a chain-like structure. The first variable determines the second, the second determines the third, and so on. We can see that WI first identifies the last pair of variables, then it quickly realizes that the first pair of variables are the most important. The structural information of the problem is clearly and precisely identified.

Experiments have shown that EDA-MCC significantly outperforms others on shifted rotated high conditioned elliptic F_9. Fig. 15 shows that WI always helps EDA-MCC recognize the problem structure. The WI results clearly show that some variables are constantly identified as strongly dependent during evolution (the dark rows of $\boldsymbol{Q}$). Furthermore, by checking the expression of F_9 (see Table III), we can see that the coefficient $\sum_{i=1}^{n}(10^6)^{\frac{i-1}{n-1}}$ before z_i^2 increases exponentially with i given fixed n. Thus, among the transformed variables $z_i (1 \le i \le n)$, z_n mostly impacts the function. F_9 can also be written as

$$\begin{aligned} F(\vec{x}) &= \sum_{i=1}^{n}\Big(\sqrt{(10^6)^{\frac{i-1}{n-1}}} \cdot z_i\Big)^2 + f_{bias_3} \\ &= \sum_{i=1}^{n}\Big(\sqrt{(10^6)^{\frac{i-1}{n-1}}} \cdot \sum_{j=1}^{n}(x_j - o_j)\boldsymbol{M}_{ji}\Big)^2 + f_{bias_3} \\ &= \sum_{i=1}^{n}\Big(\sum_{j=1}^{n}(x_j - o_j)\boldsymbol{M}_{ji}\sqrt{(10^6)^{\frac{i-1}{n-1}}}\Big)^2 + f_{bias_3} \\ &= \sum_{i=1}^{n}\Big(\sum_{j=1}^{n}(x_j - o_j)\boldsymbol{R}_{ji}\Big)^2 + f_{bias_3} \end{aligned}$$

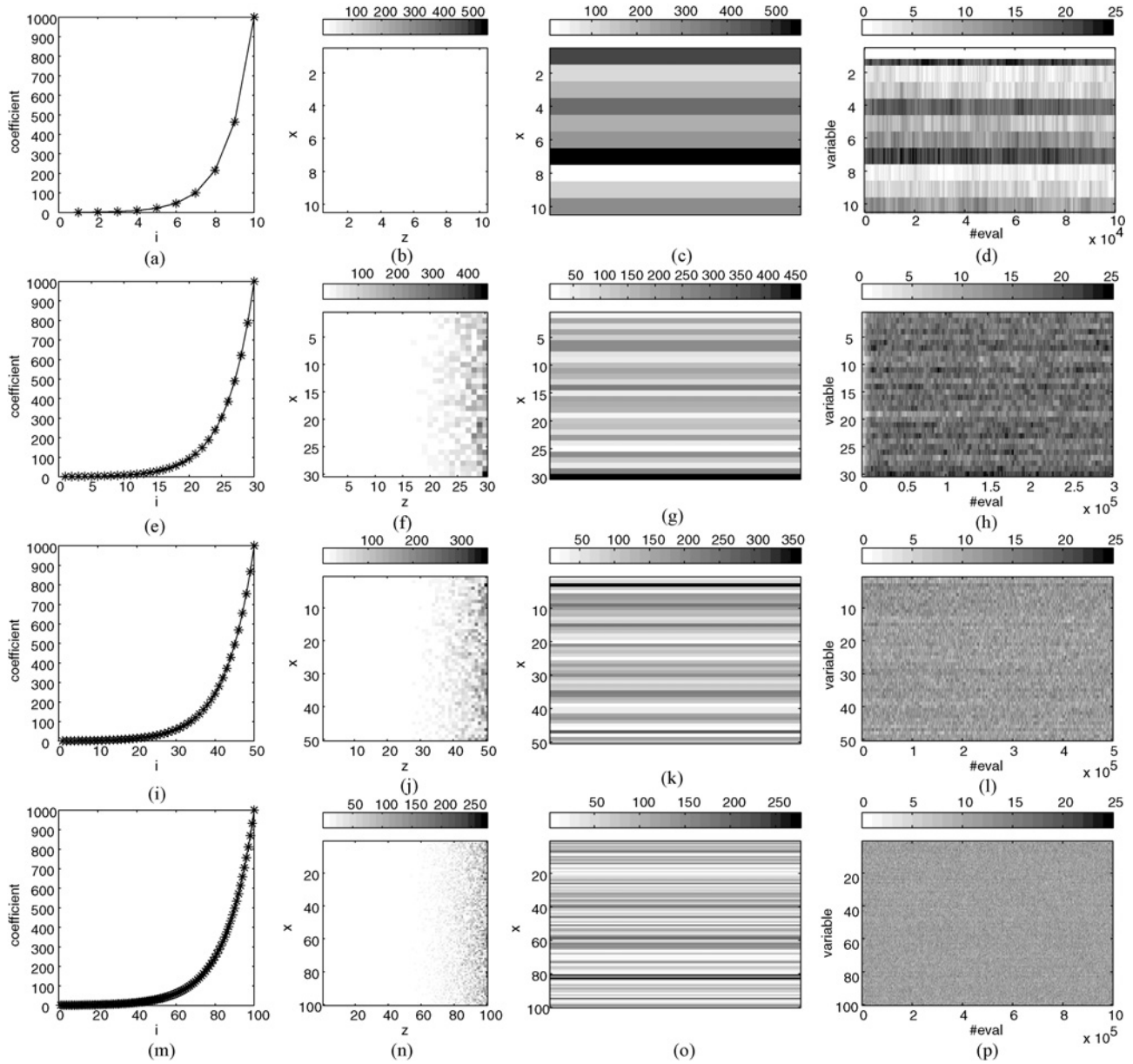

Fig. 16. Explanations of WI results on F_9. The coefficients of z_i are shown in the first column. Second column demonstrates $Abs(\boldsymbol{R})$. Third column shows the nth column of $Abs(\boldsymbol{R})$, denoted as $Abs(\boldsymbol{R})(:, n)$. The experimental $\boldsymbol{Q}$ results are shown in the last column, which are directly adopted from Fig. 15. We can see that the graphs in the last two columns are similar, especially for lower dimensional tests. (a) 10-D coefficients of z_i. (b) 10-D $Abs(R)$. (c) 10-D $Abs(R)(:, 10)$. (d) 10-D $\boldsymbol{Q}$ in experiment. (e) 30-D coefficients of z_i. (f) 30-D $Abs(R)$. (g) 30-D $Abs(R)(:, 30)$. (h) 30-D $\boldsymbol{Q}$ in experiment. (i) 50-D coefficients of z_i. (j) 50-D $Abs(R)$. (k) 50-D $Abs(R)(:, 50)$. (l) 50-D $\boldsymbol{Q}$ in experiment. (m) 100-D coefficients of z_i. (n) 100-D $Abs(R)$. (o) 100-D $Abs(R)(:, 100)$. (p) 100-D $\boldsymbol{Q}$ in experiment.

where $\boldsymbol{R}_{ji} = \boldsymbol{M}_{ji} \cdot \sqrt{(10^6)^{\frac{i-1}{n-1}}}$, $1 \leq i, j \leq n$. $\boldsymbol{M}_{ji}$ is the element of $\boldsymbol{M}$ (value can be found in [53]). Matrix $\boldsymbol{R}$ partly represents to what extent the original variables $\vec{x}$ impact the function value. Roughly speaking, $\boldsymbol{R}_{ji}$ indicates the effect of x_j onto z_i and thus onto $F(\vec{x})$. Because F_7 is nonlinear, it is hard to analyze the exact impact of each variable. But since z_n mainly impacts the function, we can instead analyze the nth column of $\boldsymbol{R}$ that can partly indicate the impact of $\vec{x}$ onto z_n and thus onto $F(\vec{x})$ to give a rough analysis. We plot the curves of coefficient $\sqrt{(10^6)^{\frac{i-1}{n-1}}}$ as subfigures in the first column of Fig. 16. The subfigures in the second column show the absolute value of matrix $\boldsymbol{R}$, $Abs(\boldsymbol{R})$. We use absolute value because both positive or negative coefficients of a variable can influence the function value. The subfigures in the third column show the nth column of $Abs(\boldsymbol{R})$, which is denoted as $Abs(\boldsymbol{R})(:, n)$. To compare them with the experimental results $\boldsymbol{Q}$ shown in the last column of Fig. 16, we stretch the widths to make them same size. Here, $\boldsymbol{Q}$ are directly from Fig. 15. We can see that when n is large, the domination of z_n becomes weak because the coefficients of z_{n-1}, z_{n-2}, etc., approach the coefficient of z_n. Therefore, the difference between the rough analysis and the experimental results also becomes larger. However, for all four tests, we can always find the evidence that WI

TABLE XIII
COMPARISON AMONG ''WI + SM,'' ''SM ONLY,'' AND ''WI ONLY'' ON 100-D PROBLEMS

Prob.	WI + SM	SM only	WI only
F_2	**0±0**	**0±0**	**0±0**
F_8	**9.65e+01±1.3e-01**	1.00e+02±2.3e+01	4.51e+03±2.1e+04
F_9	**9.59e+06±2.5e+06**	9.01e+09±1.1e+09§	3.33e+07±6.7e+06§
F_{10}	**1.87e+03±3.6e+02**	8.15e+04±3.9e+03§	2.39e+04±2.3e+03§
F_{11}	7.49e+02±1.6e+01	7.82e+02±1.7e+01§	**7.36e+02±1.1e+01**§
F_{13}	6.53e+01±1.6e+00	6.97e+01±1.8e+00§	**6.51e+01±1.1e+00**

§ The value of Asymp. Sig. (two-tailed) < 0.001 when compared with the results of "WI + SM."

Results are averaged over 25 runs. For each test, the best result is bolded. Results of "WI + SM" are compared with results of "SM Only" and "WI Only," respectively, by nonparametric Mann–Whitney U test.

successfully recognizes the problem structure. Those variables most impacting optimization are correctly identified as dark rows in $\boldsymbol{Q}$.[6]

Fig. 17 shows the WI results on shifted rotated Rastrigin F_{12}. Results here also help explain why UMDA$_c^G$ performs well on this problem while EDA-MCC fails. By examining the WI results on Rastrigin F_{11} (not shown here), we find that the results are very similar to Fig. 17. Since F_{11} is separable, the results are reasonable. As analyzed above, due to the inefficiency of covariance matrix scaling on this function with a huge number of local optima, EDA-MCC cannot perform well. However, on nonseparable F_{12}, WI still fails to recognize the problem structure because the sample size (selected size) is far less enough considering the huge number of local optima. From the information that WI can gather, F_{12} just looks like a separable problem and no useful interdependencies are learned from the samples. As a result, EDA-MCC cannot perform well.

EDA-MCC's remarkable capability of characterizing the problem properties are clearly shown in this section. Although in some cases, EDA-MCC cannot find better solutions than candidate algorithms, its capability of describing the problems' underlying structural information is remarkable throughout the experiments. We regard it as the most valuable aspect of EDA-MCC. However, for F_{11}–F_{13}, which has a huge number of local optima, EDA-MCC still has limitation. It should also be noticed that in current implementation of EDA-MCC, we have not tried every possible univariate model on $\mathcal{W}$ and multivariate model on $\mathcal{S}$ other than the two Gaussian models employed. Therefore, even if EDA-MCC correctly characterizes the problem properties, such information may not be fully exploited due to the limitation of Gaussian models. This can possibly explain why in some cases EDA-MCC cannot outperform other algorithms, even with correct problem structure characterization. We have to admit that our results are still restricted within the capability of Gaussian models.

One thing that needs to be addressed is that when solving a real-world problem in practice, a user may not want or be able to run EDA-MCC for multiple times to obtain the problem's structural information. In this case, a recommended way is to allow EDA-MCC for restarts, and aggregate the information collected over multiple trials to generate the $\boldsymbol{Q}$ matrix.

[6]It is recommended to refer to the high resolution version of these figures.

IX. ROLES OF WI AND SM, AND THEIR INTERACTIONS

In this section, we analyze the roles of WI and SM, and their interactions in EDA-MCC. Besides the implementation of EDA-MCC using WI + SM, we also implement an "SM only" version and a "WI only" version. We compare them with EDA-MCC on 100-D of the 13 test functions to analyze their respective roles. But to save space, we only report comparisons on selected functions F_2, F_8–F_{11}, and F_{13} here. The parameters of "SM only" and "WI only" are exactly the same as the respective settings of SM and WI in previous EDA-MCC experiments. For each test, the population sizes of all three versions are set to the same as the selected best results of EDA-MCC.

The solution results are shown in Table XIII. We can see that when WI + SM performs the best, it usually finds order-of-magnitude better solutions than "SM only" and "WI only." Because "SM only" applies several multivariate models on all variables, the ways dealing with those actually weakly dependent variables are not so efficient. Therefore, it fails to perform the best on any function except the simplest F_2. On the other hand, "WI only" can perform slightly better than WI + SM on F_{11} and F_{13} and the same as WI + SM on F_2, but much worse on the others. The average CPU time costs are illustrated in Fig. 18. Although "SM only" cannot find solutions of comparable quality, its CPU time cost is usually acceptable or comparable with WI + SM, whereas "WI only" can cost much more CPU time. Generally speaking, WI + SM shows much more robust performance and moderate computational time cost than "SM only" and "WI only." It is also interesting that "WI only" can perform slightly better than WI + SM on F_{11} and F_{13}. This implies that SM does not contribute a bit on these functions. It is consistent with our previous conclusions in Section IV-C4 that subspace partitioning with changing c does not help solve these functions. Without SM, "WI only" can even perform a little better. But when SM is necessary, e.g., on F_8–F_{10}, "WI only" will fail.

To investigate the interaction between WI and SM in terms of EDA-MCC's capability of characterizing problem structure, we plot the WI results (#strong and $\boldsymbol{Q}$ matrix) of "WI only" on F_8 and F_{11} in Fig. 19. WI results of "WI only" on other functions are similar to either of the two. We can see that on problems with strong variable interdependencies like F_8, without SM, the precision of global multivariate model on $\mathcal{S}$ quickly deteriorates as the search proceeds. It affects not only the solution quality but also the WI procedure. Based on samples drawn from the imprecise global model, WI also becomes so useless that eventually all variables are partitioned into $\mathcal{S}$. It also results in high computational costs in modeling and sampling. On the other hand, when SM does not help as on F_{11}, "WI only" can still characterize the problem structure properly and finds solutions with the same or better quality.

We can conclude that SM helps maintain the global precision of the model (though it is approximated with subspace models), and thus helps WI more effectively recognize the problem structure. On the other hand, WI helps to apply suitable modeling and search strategies on weakly dependent and strongly dependent variables respectively, so that EDA-

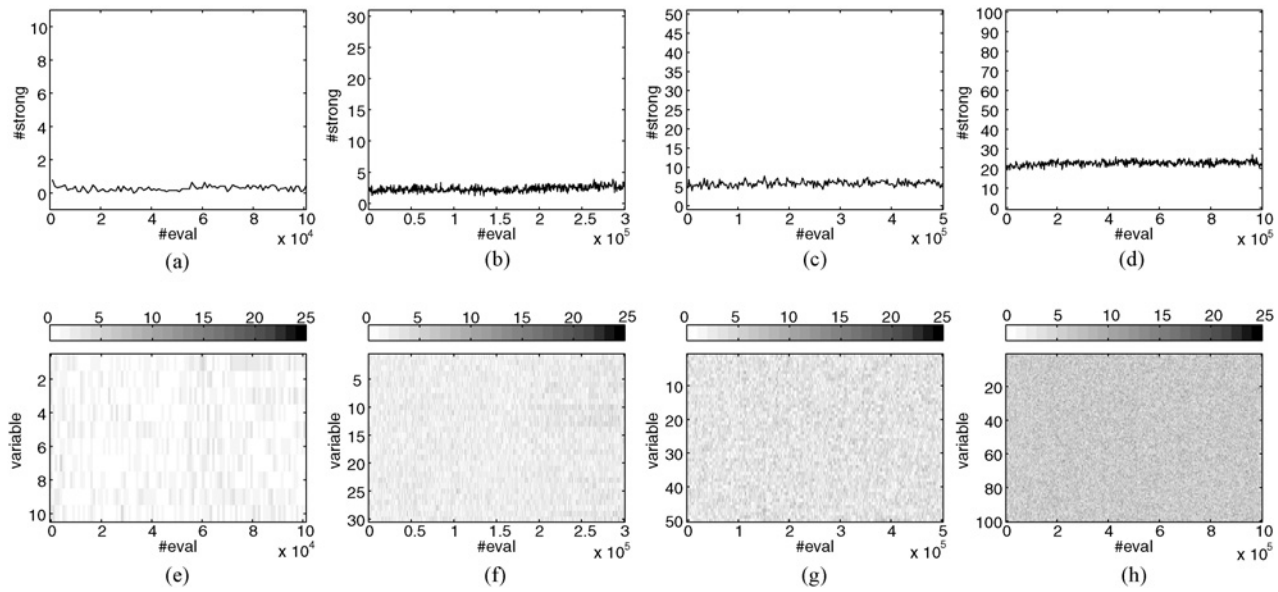

Fig. 17. WI results on F_{12}: shifted rotated Rastrigin. The darker the element of $\boldsymbol{Q}$ is, the more times a variable is partitioned into $\mathcal{S}$ at the specific #eval during the 25 runs. (a) 10-D average #strong. (b) 30-D average #strong. (c) 50-D average #strong. (d) 100-D average #strong. (e) 10-D $\boldsymbol{Q}$. (f) 30-D $\boldsymbol{Q}$. (g) 50-D $\boldsymbol{Q}$. (h) 100-D $\boldsymbol{Q}$.

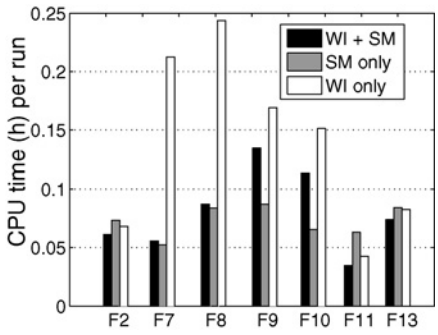

Fig. 18. Comparison of CPU time of "WI + SM," "SM only," and "WI only."

MCC can find good solutions effectively. In a word, the success of EDA-MCC, in terms of both the problem structure characterization capability, and the robust performance on large-scale optimization problems, are due to the interaction between WI and SM.

X. Conclusion and Future Work

In this paper, we first analyze the difficulties of continuous EDAs in high-dimensional search space. Due to the curse of dimensionality, given a finite population size, the performance of traditional EDAs deteriorates quickly as the problem size grows large. Their computational cost also increases fast when adopting a multivariate model for nonseparable problems. To improve the performance and reduce the computational cost for large-scale optimization, a novel multivariate EDA with model complexity control (EDA-MCC) is proposed. By employing WI and SM techniques, EDA-MCC shows significantly better performance than traditional EDAs on large-scale nonseparable problems (up to 500-D) with only a few local optima. The computational complexity and the requirement of large population sizes can be significantly reduced in EDA-MCC. Besides, EDA-MCC exhibits good scalability, and more importantly, the remarkable problem property characterization capability. When solving a problem, EDA-MCC will not only find a solution, but also give users feedback on the problem's structure. Such a capability can be far more valuable than just obtaining a solution. It is especially useful when facing a black box optimization problem. Based on the extracted problem structural information, more efficient algorithms can be designed specifically to give better solutions. The limitations of EDA-MCC are also analyzed. First, in low-dimensional search space where available population size is usually sufficiently large to offer a good global model estimation, EDA-MCC may not be as effective as traditional

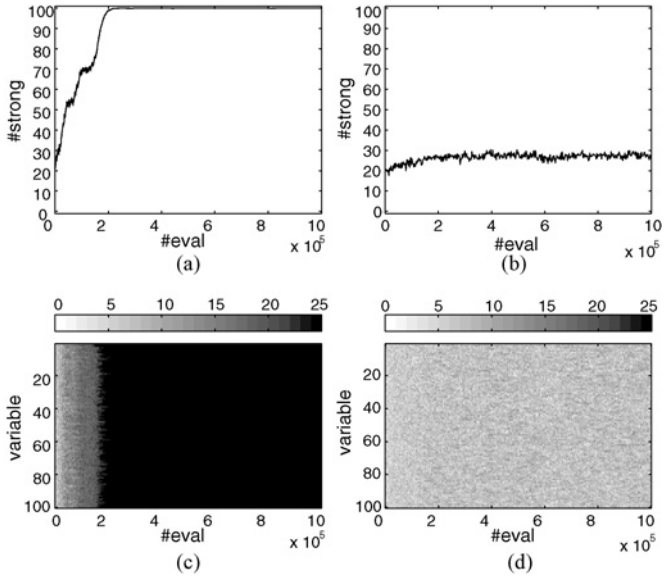

Fig. 19. Results of WI in "WI only" on F_8 and F_{11}. (a) F_8: average #strong. (b) F_{11}: average #strong. (c) F_8: $\boldsymbol{Q}$. (d) F_{11}: $\boldsymbol{Q}$.

EDAs. The advantage of EDA-MCC over traditional EDAs appears in high-dimensional space where a given population size fails to give a reliable global model estimation. Second, when facing large-scale nonseparable problems that have a huge number of local optima, EDA-MCC may not be as effective or efficient as a simple univariate Gaussian EDA. We should note that current discussions and implementation on EDA-MCC are still restricted to Gaussian models. Different base univariate and multivariate models other than Gaussian are still to be tested and analyzed. Moreover, smarter self-adaptive setting of θ and c is still an interesting issue that is left for our future work.

APPENDIX A
COMPUTATIONAL COMPLEXITY OF UMDA_c^G AND $\text{EMNA}_{\text{global}}$

We consider the one-generation computational complexity here. Suppose the current model is built from the selected individuals of the last generation. Vector $\vec{X}$ denotes an individual and X_i denotes the ith variable of $\vec{X}$. The problem is n-dimensional. M denotes the population size and m denotes the number of selected individuals. Without loss of generality, we assume $|\mathcal{P}'| = |\mathcal{P}| = M$.

1) *$UMDA_c^G$:* Let μ_i and σ_i^2 denote the mean and the variance of X_i, respectively ($i = 1, \ldots, n$). The joint pdf is

$$f(\vec{x}) = \prod_{i=1}^{n} f_{\mathcal{N}}(x_i; \mu_i, \sigma_i^2) = \prod_{i=1}^{n} \frac{1}{\sigma_i \sqrt{2\pi}} e^{-\frac{(x_i-\mu_i)^2}{2\sigma_i^2}}. \quad (6)$$

- Model estimation.
  Estimate (μ_i, σ_i^2) for X_i ($i = 1, \ldots, n$):
  1) Traverse m selected individuals to estimate $\mu_1, \ldots, \mu_n$: $O(nm)$.
  2) Traverse m selected individuals to estimate $\sigma_1^2, \ldots, \sigma_n^2$: $O(nm)$.

  Overall complexity: $O(nm)$.
- Sampling new solutions.
  For X_i, we need to generate a standard normal random number ζ, then do

$$x_i \leftarrow \mu_i + \zeta \cdot \sigma_i. \quad (7)$$

  Since such an operation is fast, we suppose sampling one variable costs $O(1)$, thus $O(n)$ is needed for n variables. Repeating M times to create $\mathcal{P}'$ costs $O(nM)$.
  Overall complexity: $O(nM)$.

2) *$EMNA_{\text{global}}$:* Let $\vec{\mu}$ and $\mathbf{\Sigma}$ denote the n-dimensional mean vector and the $n \times n$ covariance matrix, respectively. The joint pdf is

$$f(\vec{x}) = f_{\mathcal{N}}(\vec{x}; \vec{\mu}, \mathbf{\Sigma}) = \frac{1}{(2\pi)^{\frac{N}{2}} |\mathbf{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{x}-\vec{\mu})^T \mathbf{\Sigma}^{-1} (\vec{x}-\vec{\mu})}. \quad (8)$$

- Model estimation.
  1) Traverse m selected individuals to estimate $\vec{\mu}$: $O(nm)$.
  2) Traverse m selected individuals to estimate $\mathbf{\Sigma}$: $O(n^2 m)$.

  Overall complexity: $O(n^2 m)$.
- Sampling new solutions.
  1) Before first time sampling, we need $O(n^3)$ to decompose $\mathbf{\Sigma}$ such that $\mathbf{\Sigma} = \boldsymbol{H}\boldsymbol{H}^T$ [26].
  2) To sample a new solution, we need to generate a standard normal random vector $\vec{\zeta}$, then do

$$\vec{x} \leftarrow \vec{\mu} + \vec{\zeta} \cdot \boldsymbol{H}. \quad (9)$$

  Primary cost here is the $O(n^2)$ matrix multiplications. Repeating M times to create $\mathcal{P}'$ costs $O(n^2 M)$.

  Note that for $\text{EMNA}_{\text{global}}$, usually $M > n$ in practice, which means the population size is usually larger than the problem size. Therefore, the overall complexity of sampling is dominated by $O(n^2 M)$ in second step.
  Overall complexity: $O(n^2 M)$.

APPENDIX B
COMPUTATIONAL COMPLEXITY OF EDA-MCC

Computation here using the same premises in Appendix A. We give the one-generation computational complexity of EDA-MCC. Here all $g_i(\cdot)$ are univariate Gaussian models, and all $h_k(\cdot)$ are multivariate Gaussian models.

- Model estimation.
  1) Sampling m_{corr} individuals from m selected individuals: $O(m_{\text{corr}})$.
  2) Traverse m_{corr} sampled individuals to calculate the global correlation matrix $\boldsymbol{C}$: $O(n^2 m_{\text{corr}})$.
  3) Traverse $\boldsymbol{C}$ to construct $\mathcal{W}$: $O(n^2)$.
  4) Building $g_i(\cdot)$ and $h_k(\cdot)$.
     Consider two extreme situations:
     - When $\mathcal{W} = \mathcal{V}$, all n variables are identified as "weakly dependent":
       a) Building $g_i(\cdot)$, $i = 1, \ldots, n$:
          Same order as UMDA_c^G model estimation, $O(nm)$.
       b) No need to build $h_k(\cdot)$.
     - When $\mathcal{W} = \emptyset$, all n variables are identified as "strongly dependent":
       a) No need to build $g_i(\cdot)$.
       b) Building $h_k(\cdot)$, $k = 1, \ldots, \lceil n/c \rceil$:
          Same order as building a c dimensional $\text{EMNA}_{\text{global}}$ model $\lceil n/c \rceil$ times, $O(c^2 m \cdot n/c) = O(cnm)$.

  Thus, the overall complexity is between

$$O(n^2 m_{\text{corr}}) + O(nm) \quad (10)$$

  and

$$O(n^2 m_{\text{corr}}) + O(cnm). \quad (11)$$

  Also note that $1 \ll m_{\text{corr}} \leq m$, $1 \leq c \leq n$.
- Sampling solutions.
  Consider two extreme situations:
  - When $\mathcal{W} = \mathcal{V}$, all n variables are sampled from $g_i(\cdot)$, $i = 1, \ldots, n$:

1) Sampling from $g_i(\cdot), i = 1, \ldots, n$:
Same order as UMDA$_c^G$ solution sampling, $O(nM)$.
2) No need to sample from $h_k(\cdot)$.

– When $\mathcal{W} = \emptyset$, all n variables are sampled from $h_k(\cdot), k = 1, \ldots, \lceil n/c \rceil$:
1) No need to sample from $g_i(\cdot)$.
2) Sampling from $h_k(\cdot), k = 1, \ldots, \lceil n/c \rceil$:
Same order as sampling from a c dimensional EMNA$_{\text{global}}$ model $\lceil n/c \rceil$ times, $O(c^2 M \cdot n/c) = O(cnM)$.

Thus, the overall complexity is between

$$O(nM) \tag{12}$$

and

$$O(cnM) \ . \tag{13}$$

ACKNOWLEDGMENT

The authors are grateful to Dr. Zhenyu Yang, Dr. Yang Yu, Dr. Tapabrata Ray, and Dr. Lu Wang for their insightful comments that helped improve this paper, and to Dr. Alexander Mendiburu who kindly provided the source codes of MIMIC$_c^G$ and EGNA.

REFERENCES

[1] H. Mühlenbein and G. Paaß, "From recombination of genes to the estimation of distributions. I. Binary parameters," in *Proc. Parallel Problem Solving Nature IV*, 1996, pp. 178–187.
[2] P. Larrañaga and J. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Boston, MA, USA: Kluwer Academic, 2002.
[3] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA, USA: Addison-Wesley, 1989.
[4] M. Sebag and A. Ducoulombier, "Extending population-based incremental learning to continuous search spaces," in *Proc. Parallel Problem Solving Nature V*, 1998, pp. 418–427.
[5] P. Bosman and D. Thierens, "Expanding from discrete to continuous estimation of distribution algorithms: The IDEA," in *Proc. Parallel Problem Solving Nature VI*, 2000, pp. 767–776.
[6] P. Bosman and D. Thierens, "Continuous iterated density estimation evolutionary algorithms within the IDEA framework," in *Proc. Optimization Building Using Probabilistic Models OBUPM Workshop GECCO*, 2000, pp. 197–200.
[7] P. Larrañaga, R. Etxeberria, J. Lozano, and J. Peña, "Optimization in continuous domains by learning and simulation of Gaussian networks," in *Proc. GECCO*, 2000, pp. 201–204.
[8] M. Wagner, A. Auger, and M. Schoenauer, "EEDA: A new robust estimation of distribution algorithm," Rapport de Recherche (Res. Rep.) RR-5190, INRIA, 2004.
[9] J. Grahl, P. Bosman, and F. Rothlauf, "The correlation-triggered adaptive variance scaling IDEA," in *Proc. GECCO*, 2006, pp. 397–404.
[10] P. Bosman, J. Grahl, and F. Rothlauf, "SDR: A better trigger for adaptive variance scaling in normal EDAs," in *Proc. GECCO*, 2007, pp. 492–499.
[11] W. Dong and X. Yao, "Unified eigen analysis on multivariate Gaussian based estimation of distribution algorithms," *Inform. Sci.*, vol. 178, no. 15, pp. 3000–3023, 2008.
[12] S. Tsutsui, M. Pelikan, and D. Goldberg, "Evolutionary algorithm using marginal histogram models in continuous domain," in *Proc. Optimization Building Using Probabilistic Models OBUPM Workshop GECCO*, 2001, pp. 230–233.
[13] 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. 2003, pp. 443–450.
[14] P. Pošík, "Distribution tree-building real-valued evolutionary algorithm," *Parallel Problem Solving Nature VIII*, 2004, pp. 372–381.
[15] N. Ding, S. Zhou, and Z. Sun, "Optimizing continuous problems using estimation of distribution algorithm based on histogram model," in *Proc. 6th Conf. Simulated Evol. Learning*, 2006, pp. 545–552.
[16] N. Ding, J. Xu, S. Zhou, and Z. Sun, "Reducing computational complexity of estimating multivariate histogram-based probabilistic model," in *Proc. IEEE Congr. Evol. Comput.*, 2007, pp. 111–118.
[17] N. Ding and S. Zhou, "Linkages detection in histogram-based estimation of distribution algorithm," in *Linkage in Evolution Computation*, Ying-Ping Chen and Meng-Hiot Lim, Eds. Berlin/Heidelberg, Germany: Springer-Verlag, 2008, pp. 25–40.
[18] N. Ding, S. Zhou, and Z. Sun, "Histogram-based estimation of distribution algorithm: A competent method for continuous optimization," *J. Comput. Sci. Technol.*, vol. 23, no. 1, pp. 35–43, 2008.
[19] N. Ding, S. Zhou, H. Zhang, and Z. Sun, "Marginal probability distribution estimation in characteristic space of covariance-matrix," in *Proc. IEEE Congr. Evol. Comput.*, 2008, pp. 1589–1595.
[20] K. Tang, X. Yao, P. Suganthan, C. MacNish, Y. P. Chen, C. M. Chen, and Z. Yang. (2007). "Benchmark functions for the CEC'2008 special session and competition on large scale global optimization," Tech. Rep. [Online]. Available: http://nical.ustc.edu.cn/cec08ss.php
[21] K. Tang, X. Li, P. Suganthan, Z. Yang, and T. Weise. (2009). "Benchmark functions for the CEC'2010 special session and competition on large scale global optimization," Tech. Rep. [Online]. Available: http://nical.ustc.edu.cn/cec10ss.php
[22] "Workshop for evolutionary algorithms and other metaheuristics for continuous optimization problems: A scalability test," in *Proc. 9th Int. Conf. Intell. Syst. Des. Appl.*, 2009 [Online]. Available: http://sci2s.ugr.es/programacion/workshop/Scalability.html
[23] "Special issue on 'Scalability of evolutionary algorithms and other metaheuristics for large scale continuous optimization problems,'" *Soft Comput.*, vol. 15, no. 11, Nov. 2011.
[24] J. Friedman, "An overview of predictive learning and function approximation," *NATO ASI Series Comput. Syst. Sci.*, vol. 136, pp. 1–61, 1994.
[25] D. Wolpert and W. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.
[26] L. Devroye, *Non-Uniform Random Variate Generation*. New York, NY, USA: Springer-Verlag, 1986.
[27] M. Gallagher, M. Frean, and T. Downs, "Real-valued evolutionary optimization using a flexible probability density estimator," in *Proc. GECCO*, 1999, pp. 840–846.
[28] P. Bosman and D. Thierens, "Advancing continuous IDEAs with mixture distributions and factorization selection metrics," in *Proc. Optimization Building Using Probabilistic Models OBUPM Workshop GECCO*, 2001, pp. 208–212.
[29] 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.
[30] C. Ahn, R. Ramakrishna, and D. Goldberg, "Real-coded Bayesian optimization algorithm: Bringing the strength of BOA into the continuous world," in *Proc. GECCO*, 2004, pp. 840–851.
[31] C. Ahn, "Real-coded Bayesian optimization algorithm," in *Advances in Evolutionary Algorithms: Theory, Design and Practice*. Berlin, Germany: Springer, 2006, pp. 85–124.
[32] C. Ahn and R. Ramakrishna, "On the scalability of real-coded Bayesian optimization algorithm," *IEEE Trans. Evol. Comput.*, vol. 12, no. 3, pp. 307–322, Jun. 2008.
[33] M. Li, D. Goldberg, K. Sastry, and T. Yu, "Real-coded ECGA for solving decomposable real-valued optimization problems," in *Linkage in Evolutionary Computation*, Ying-Ping Chen and Meng-Hiot Lim, Eds. Berlin/Heidelberg, Germany: Springer-Verlag, 2008, 61–86.
[34] J. Sun, Q. Zhang, and E. Tsang, "DE/EDA: A new evolutionary algorithm for global optimization," *Inform. Sci.*, vol. 169, nos. 3–4, pp. 249–262, 2005.
[35] W. Dong and X. Yao, "NichingEDA: Utilizing the diversity inside a population of EDAs for continuous optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2008, pp. 1260–1267.
[36] T. Chen, K. Tang, G. Chen, and X. Yao, "On the analysis of average time complexity of estimation of distribution algorithms," in *Proc. IEEE Congr. Evolutionary Comput.*, Sep. 2007, pp. 453–460.
[37] T. Chen, P. K. Lehre, K. Tang, and X. Yao, "When is an estimation of distribution algorithm better than an evolutionary algorithm?" in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 1470–1477.
[38] T. Chen, K. Tang, G. Chen, and X. Yao, "Rigorous time complexity analysis of univariate marginal distribution algorithm with margins," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 2157–2164.

[39] T. Chen, K. Tang, G. Chen, and X. Yao, "Analysis of computational time of simple estimation of distribution algorithms," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 1–22, Feb. 2010.
[40] 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, and E. Cantu-Paz, Eds. Berlin/Heidelberg, Germany: Springer-Verlag, 2006, pp. 91–120.
[41] Y. Wang and B. Li, "A restart univariate estimation of distribution algorithm: Sampling under mixed Gaussian and Lévy probability distribution," in *Proc. IEEE Congr. Evolutionary Comput.*, Jun. 2008, pp. 3917–3924.
[42] C. Bielza, V. Robles, and P. Larrañaga, "Estimation of distribution algorithms as logistic regression regularizers of microarray classifiers," *Methods Inform. Med.*, vol. 48, no. 3, pp. 236–241, 2009.
[43] A. Mendiburu, J. Lozano, and J. Miguel-Alonso, "Parallel implementation of EDAs based on probabilistic graphical models," *IEEE Trans. Evol. Comput.*, vol. 9, no. 4, pp. 406–423, Aug. 2005.
[44] P. Bosman, "On empirical memory design, faster selection of Bayesian factorizations and parameter-free Gaussian EDAs," in *Proc. GECCO*, ACM, 2009, pp. 389–396.
[45] C. González, J. Lozano, and P. Larrañaga, "Mathematical modelling of UMDAc algorithm with tournament selection: Behaviour on linear and quadratic functions," *Int. J. Approximate Reasoning*, vol. 31, no. 3, pp. 313–340, 2002.
[46] J. Grahl, S. Minner, and F. Rothlauf, "Behaviour of UMDAc with truncation selection on monotonous functions," in *Proc. IEEE Congr. Evol. Comput.*, Sep. 2005, pp. 2553–2559.
[47] B. Yuan and M. Gallagher, "On the importance of diversity maintenance in estimation of distribution algorithms," in *Proc. GECCO*, 2005, pp. 719–726.
[48] Z. Zhang, W. Dong, K. Huang, and T. Tan, "EDA approach for model based localization and recognition of vehicles," in *Proc. 7th Int. Workshop Visual Surveillance, IEEE Conf. Comput. Vision Pattern Recognit.*, Jun. 2007, pp. 1–8.
[49] P. Pošík and V. Franc, "Estimation of fitness landscape contours in EAs," in *Proc. GECCO*, 2007, pp. 562–569.
[50] W. Dong and X. Yao, "Covariance matrix repairing in Gaussian based EDAs," in *Proc. 2007 IEEE Congr. Evol. Comput.*, Sep. 2007, pp. 415–422.
[51] A. Auger and N. Hansen, "Evolution strategies and related estimation of distribution algorithms," in *Proc. GECCO*, 2008, pp. 2727–2740.
[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] P. Suganthan, N. Hansen, J. Liang, K. Deb, Y. Chen, A. Auger, and S. Tiwari. (2005). "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Tech. Rep. [Online]. Available: http://www.ntu.edu.sg/home/EPNSugan
[54] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Inform. Sci.*, vol. 178, no. 15, pp. 2985–2999, 2008.
[55] J. Demˇsar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learning Res.*, vol. 7, pp. 1–30, 2006.
[56] R. Ros and N. Hansen, "A simple modification in CMA-ES achieving linear time and space complexity," in *Proc. Parallel Problem Solving Nature X*, 2008, pp. 296–305.
[57] M. Omidvar and X. Li, "A comparative study of CMA-ES on large scale global optimisation," in *Proc. Advances Artif. Intell.*, 2011, pp. 303–312.
[58] R. Santana, P. Larrañaga, and J. Lozano, "Protein folding in simplified models with estimation of distribution algorithms," *IEEE Trans. Evol. Comput.*, vol. 12, no. 4, pp. 418–438, Aug. 2008.

Weishan Dong received the B.E. degree in computer science and technology from the University of Science and Technology of China, Hefei, China, in 2004, and the Ph.D. degree in pattern recognition and intelligent system from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2009. He also studied as a joint Ph.D. student at the School of Computer Science, University of Birmingham, Birmingham, U.K., from 2008 to 2009.

He is currently a Research Staff member at IBM Research—China, Beijing, China. His current research interests include data mining, evolutionary computation, and computer vision, as well as their industrial applications, such as asset management systems, crime analytics solutions, and business intelligence software.

Dr. Dong was a recipient of the Best Conference Paper Award at IEEE SOLI 2011 and the IBM Research Accomplishment of Asset Optimization for Smarter Cities in 2012.

Tianshi Chen received the B.S. degree in mathematics from the Special Class for the Gifted Young, University of Science and Technology of China (USTC), Hefei, China, in 2005, and the Ph.D. degree in computer science from the Department of Computer Science and Technology, USTC, in 2010.

He is currently an Assistant Professor at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His current research interests include evolutionary computation, applications of computational intelligence techniques in computer architecture research, and parallel computing.

Dr. Chen was a recipient of the China Computer Federation Distinguished Doctoral Dissertation Award in 2011 and the Chinese Academy of Sciences Distinguished Doctoral Dissertation Award in 2011 for his Ph.D. work on computational complexity analysis of evolutionary algorithms.

Peter Tiňo received the M.Sc. degree from the Slovak University of Technology and the Ph.D. degree from the Slovak Academy of Sciences.

He was a Fulbright fellow at the NEC Research Institute, Princeton, NJ, USA and a Post-Doctoral fellow at the Austrian Research Institute for AI, Vienna, Austria, and at Aston University, U.K. Since 2003, he has been with the Centre of Excellence for Research in Computational Intelligence and Applications School of Computer Science, University of Birmingham, Birmingham, U.K., where he is currently a Reader in complex and adaptive systems. His current research interests include dynamical systems, machine learning, probabilistic modeling of structured data, evolutionary computation, and fractal analysis.

Dr. Tiňo was a recipient of the Fulbright Fellowship in 1994, the U.K.–Hong-Kong Fellowship for Excellence in 2008, three Outstanding Paper of the Year Awards for the IEEE TRANSACTIONS ON NEURAL NETWORKS in 1998, 2011, the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION in 2010, and the Best Paper Award at ICANN 2002. He is on the Editorial Board of several journals.

Xin Yao (F'03) is currently a Chair (Professor) of computer science and the Director of the Centre of Excellence for Research in Computational Intelligence and Applications, School of Computer Science, University of Birmingham, Birmingham, U.K. He has more than 400 refereed publications in international journals and conferences. His current research interests include evolutionary computation and ensemble learning.

Prof. Yao is a Distinguished Lecturer of the IEEE Computational Intelligence Society. He was the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION from 2003 to 2008. He has been invited to give more than 70 keynote/plenary speeches at international conferences. His work won the 2001 IEEE Donald G. Fink Prize Paper Award, the 2010 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award, the 2010 BT Gordon Radley Award for Best Author of Innovation (Finalist), the 2011 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award, and many other Best Paper Awards at conferences. He was a recipient of the prestigious Royal Society Wolfson Research Merit Award in 2012 and the 2013 IEEE CIS Evolutionary Computation Pioneer Award.