Large-Scale Evolution Strategy Based on Search Direction Adaptation

Xiaoyu He, Yuren Zhou, Zefeng Chen, Jun Zhang, *Fellow, IEEE*, and Wei-Neng Chen, *Senior Member, IEEE*

***Abstract*—The covariance matrix adaptation evolution strategy (CMA-ES) is a powerful evolutionary algorithm for single-objective real-valued optimization. However, the time and space complexity may preclude its use in high-dimensional decision space. Recent studies suggest that putting sparse or low-rank constraints on the structure of the covariance matrix can improve the efficiency of CMA-ES in handling large-scale problems. Following this idea, this paper proposes a search direction adaptation evolution strategy (SDA-ES) which achieves linear time and space complexity. SDA-ES models the covariance matrix with an identity matrix and multiple search directions, and uses a heuristic to update the search directions in a way similar to the principal component analysis. We also generalize the traditional 1/5th success rule to adapt the mutation strength which exhibits the derandomization property. Numerical comparisons with nine state-of-the-art algorithms are carried out on 31 test problems. The experimental results have shown that SDA-ES is invariant under search-space rotational transformations, and is scalable with respect to the number of variables. It also achieves competitive performance on generic black-box problems, demonstrating its effectiveness in keeping a good tradeoff between solution quality and computational efficiency.**

***Index Terms*—Evolution strategy, large-scale optimization, search direction adaptation.**

I. Introduction

THE COVARIANCE matrix adaptation evolution strategy (CMA-ES) is one of the state-of-the-art evolutionary algorithms (EAs) for single-objective real-valued optimization [1], [2]. CMA-ES learns all pairwise dependencies between decision variables by adapting a search distribution to the objective landscape. Through sampling new solutions from this distribution, its performance is invariant against any affine transformation of the search space. It also adopts a special scheme called cumulative step-size adaptation (CSA) to adapt the mutation strength automatically. Besides attributing to the rank-based selection operator, CMA-ES exhibits the invariance to order-preserving transformations of the objective function values. These properties make CMA-ES very popular in many practical applications [3]–[5].

Despite its huge popularity, CMA-ES suffers from performance degradation when tackling problems that have a large number of decision variables. These problems, referred to as the large-scale optimization problems (LSOPs), can be formulated as

$$\min_{\boldsymbol{x}\in\Omega} \quad f(\boldsymbol{x})$$

where $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)^T$ is the decision vector, $\Omega \subset R^n$ is the decision space, f is the objective function, and n is the dimension of Ω in large-scale settings (e.g., $n > 200$) [6]. The challenges of LSOPs stem from the fact, usually known as the curse of dimensionality, that the search space increases exponentially with the number of dimensions.

The limitations of CMA-ES in handling LSOPs come from its time and space complexity. The learning procedure of CMA-ES is conducted on a covariance matrix with $[(n(n-1))/2]$ degrees of freedom, thereby taking $O(n^2)$ space and $O(n^2)$ time per generation. Sampling new solutions requires extra $O(n^3)$ computations to decompose the matrix, though the asymptotic time complexity can be theoretically reduced to $O(n^2)$ by postponing the decomposition. In addition, CMA-ES relies on the spectral decomposition when handling numerical errors and ill-conditioning, which is usually seen as computationally inefficient compared with other decomposition techniques. These limitations may preclude the use of CMA-ES for large-scale optimization.

Several approaches have been proposed to address the limitations of CMA-ES. They can be roughly categorized into the following groups.

A. Partitioning the Decision Space

This approach divides an original LSOP into multiple subproblems by grouping the variables. All subproblems are solved by CMA-ES separably, and then recombined in a cooperative coevolution (CC) framework [7]. CC-CMA-ES [8] and CC-GDG-CMA-ES [9] are representatives of this approach.

Manuscript received April 3, 2019; revised July 3, 2019; accepted July 9, 2019. Date of publication July 30, 2019; date of current version February 17, 2021. This work was supported by the National Natural Science Foundation of China under Grant 61773410 and Grant 61673403. This paper was recommended by Associate Editor P. N. Suganthan. *(Corresponding author: Yuren Zhou.)*

X. He, Y. Zhou, and Z. Chen are with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China, and also with the Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, Sun Yat-sen University, Guangzhou 510006, China (e-mail: hxyokokok@foxmail.com; zhouyuren@mail.sysu.edu.cn; chzfeng@mail2.sysu.edu.cn).

J. Zhang is with the University of Victoria, Melbourne, VIC 8001 Australia (e-mail: junzhang@ieee.org).

W.-N. Chen is with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China (e-mail: cschenwn@scut.edu.cn).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author.

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

Digital Object Identifier 10.1109/TCYB.2019.2928563

2168-2267 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

Note that the innovation of this approach is mostly the space partitioning schemes rather than the CMA-ES itself, so this paper will focus on the next several approaches.

B. Restricting the Covariance Matrix

This approach imposes sparse constraints on the structure of the covariance matrix. One simplest way is to restrict the covariance matrix to be diagonal in order to learn the variable scalings along each axes separably. This idea has been utilized in the separable CMA-ES (sep-CMA) [10] and the separable natural evolution strategy (SNES) [11]. Another implementation is to model the covariance matrix with a diagonal matrix combined with a rank-1 perturbation. Such perturbation is usually constructed with a principal search direction, aiming to capture some but not all variable dependencies. The main vector adaptation evolution strategy (MVA-ES) [12], rank-one natural evolution strategy (R1-NES) [13], and VD-CMA [14] all adopt this method. These variants are useful especially when the problem is separable or has weak variable correlations.

C. Using Multiple Direction Vectors

This approach can be seen as an extension to the one described above. But we consider it alone since the structure of its covariance matrix is much more flexible. Its core idea is to maintain a set of m direction vectors, where $m \ll n$ is a small positive integer. These vectors indicate some promising search directions, and can be used to reconstruct the covariance matrix whenever necessary. By tuning m, a trade-off between solution quality and computational efficiency can be achieved. Remarkable examples include the limited memory CMA-ES (L-CMA-ES) [15], VkD-CMA [16], the rank-m ES (RmES) [17], and another limited memory variants called LM-CMA [18]. These variants generally have competitive performance but only require a limited amount of computational burden.

D. Simplifying the Covariance Matrix Adaptation

The adaptation in the standard CMA-ES can be simplified under mild assumption. The simplified version, called matrix adaptation evolution strategy (MA-ES), adapts the Cholesky factor of the covariance matrix and, thus, avoids the time-consuming matrix factorization without significantly worsening the performance [19]. The limited-memory MA-ES (LM-MA) further extends this variant with the idea of restricting the matrix structure [20]. It reconstructs the Cholesky factor with a set of m direction vectors at each generation such that the adaptation achieves linear time and space complexity.

E. Modifying the Mutation Strength Adaptation

This approach adapts the mutation strength without making assumptions about the types of search distributions. This is particularly useful when working with the large-scale CMA-ES variants introduced above, since their distributions are usually degenerate, constrained, or even non-Gaussian. Its basic idea is to keep the success of the sampling operations reaching a predefined level. The median success rule (MSR) defines success as the median objective values getting smaller [21]. Alternatively, the population success rule (PSR) measures success with the rank sum of the current solutions from two successive generations [18]. This idea is further extended in the rank success rule (RSR) where the weighted sum is used to favor the top-ranked solutions [17].

Although using multiple direction vectors have been empirically demonstrated to be quite efficient, how to adapt these vectors is still an open question. For example, in L-CMA-ES, the direction vectors are the first m principal components of the covariance matrix. They are calculated using a thin singular-value decomposition [22]. Its time complexity is $O(m^2n)$ and, hence, it may be still nonscalable when a large m is required. VkD-CMA, LM-CMA, and RmES all have $O(mn)$ time and space complexity. However, VkD-CMA involves the natural gradient calculations [23] while LM-CMA reconstructs two Cholesky factors per generation. Therefore, they seem to be even more complicated than the original CMA-ES. RmES radically simplifies LM-CMA by directly modeling the covariance matrix with these vectors. But it introduces a new question that the suitable number of used direction vectors become problem dependent.

This paper presents a search direction adaptation evolution strategy (SDA-ES), a large-scale extension of CMA-ES. Detailed properties of this approach are summarized as follows.

1) SDA-ES models the covariance matrix with m search directions and an identity matrix. Sampling new solutions are simple as no matrix factorization techniques are required.
2) The search directions are adapted iteratively in a manner which simulates the principal component analysis (PCA) [24].
3) The mutation strength is adapted using a simplified Mann–Whitney U test [25]. This is a generalized 1/5th success rule [26] which does not rely on the Gaussian distribution.
4) SDA-ES only requires $O(mn)$ space. It has a time complexity of $O(mn)$ per function evaluation.

In the remainder of this paper, we first discuss the background and motivation of this paper in Section II. Section III describes the detailed implementations of SDA-ES. Thereafter, we present the simulation results on two benchmark suites in Section IV. Finally, Section V concludes this paper and gives some remarks for the future studies.

II. Background and Motivation

This section briefly describes the CMA-ES algorithm. Other than restating all detailed implementations, we put more emphasis on the modeling of covariance matrix. We also provide a heuristic to reduce the time and space complexity of the PCA for the covariance matrix estimation which motivates this paper.

A. CMA-ES

For a given objective function, CMA-ES assumes the existence of an optimal covariance matrix which removes the

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

variable correlations on its landscape. Particularly, on convex-quadratic objective functions whose variable correlations are all linear, the correlations can be completely removed. Thus, CMA-ES "sees" such functions as spherical ones, and solves them efficiently. This strategy also works for generic black-box objective functions, as their local landscape can be approximated by convex-quadratic functions.

Since the optimal covariance matrix is usually unknown, CMA-ES maintains a multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{m}^{(g)}, \sigma^{(g)}\boldsymbol{C}^{(g)})$ at its gth generation and adapts it gradually, where $\boldsymbol{m}^{(g)}$ is the mean vector, $\boldsymbol{C}^{(g)}$ is the covariance matrix, and $\sigma^{(g)}$ is the mutation strength. This distribution is used to guide the search at the gth generation by sampling new candidate solutions from it. It is updated per generation after all candidate solutions are evaluated.

The covariance matrix adaptation, the key procedure of the distribution adaptation, contains two parts: 1) the rank-1 update [1] and 2) the rank-μ update [2]. The rank-μ update makes use of the maximum-likelihood estimator of the covariance matrix of current population in order to reduce the adaptation time. Note that it is designed to work with a large population size, which is not the case in large-scale optimization. Thus, we only consider the rank-1 update, which can be formulated as follows:

$$\begin{cases} \boldsymbol{C}^{(0)} = \boldsymbol{I}_n \\ \boldsymbol{C}^{(g+1)} = (1-c)\boldsymbol{C}^{(g)} + c\boldsymbol{p}^{(g)}\left(\boldsymbol{p}^{(g)}\right)^T \end{cases} \tag{1}$$

where $\boldsymbol{I}_n$ is the n-dimensional identity matrix, $c \in [0, 1]$ is the learning rate, and $\boldsymbol{p}^{(g)} \in R^n$ is a random vector. The vector $\boldsymbol{p}^{(g)}$, referred to as the evolution path in CMA-ES, is obtained by cumulating successful mutation steps of the mean vector. The rank-1 update repeatedly explores the variable dependencies of the evolution path and, thus, increases the probability of reproducing the successful steps of the population mean. When g is sufficiently large, the obtained covariance matrix is adapted to the function landscape automatically.

B. Covariance Matrix Approximation Based on Principal Component Analysis

The rank-1 update described in (1) takes $O(n^2)$ space and $O(n^2)$ time, and is practically nonscalable in high-dimensional decision space. It is interesting to investigate whether the adaptation can be carried out in a low-dimensional space with the help of dimension reduction techniques such as PCA [24]. This forms the main motivation of this paper.

PCA is a classic statistical technique for linear dimension reduction. It uses an orthogonal transformation to decorrelate a set of possibly correlated variables and produces the basis of correlation eigenvectors. PCA has many different definitions in the literature. The most popular one is to define PCA as the orthogonal projection of the data (e.g., the solutions in an EA) onto a low-dimensional linear space such that the variance of the projected data is maximized. In the following text, we reformulate this definition as a sequence of maximization problems, each of which produces a vector subject to certain orthogonal constraints. Our idea is to use a subset of such vectors to reconstruct the covariance matrix and, therefore, the distribution adaptation can be implemented by updating these vectors. Instead of using the standard PCA, we will show that by discarding some of the orthogonal constraints, a heuristic can be used to approximate these vectors within linear time and space complexity.

1) Reformulating the Covariance Matrix With PCA: Let $z \in R^n$ be a random vector sampled from a Gaussian distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{C})$. Let $\boldsymbol{C} = \sum_{i=1}^{n} \delta_i \boldsymbol{u}_i \boldsymbol{u}_i^T$ be the spectral decomposition of $\boldsymbol{C}$, where $\delta_1 \geq \cdots \geq \delta_n$ are the ordered eigenvalues, and $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$ are the associated eigenvectors. Our task is to find the first m eigenvectors to approximate $\boldsymbol{C}$, where $m \ll n$ is a small integer.

The first eigenvector $\boldsymbol{u}_1$ is the unit vector determining the linear subspace such that the variance of the random vectors projected onto this subspace is maximized

$$\boldsymbol{u}_1 = \arg\max_{\boldsymbol{v}^T\boldsymbol{v}=1} \boldsymbol{v}^T\boldsymbol{C}\boldsymbol{v}. \tag{2}$$

With the identity $C = E[zz^T]$, (2) can be formulated as

$$\boldsymbol{u}_1 = \arg\max_{\boldsymbol{v}^T\boldsymbol{v}=1} \boldsymbol{v}^T E\left[zz^T\right]\boldsymbol{v} = \arg\max_{\boldsymbol{v}^T\boldsymbol{v}=1} E\left[\left(\boldsymbol{v}^T z\right)^2\right] \tag{3}$$

where E denotes the expectation.

Then, the second eigenvector $\boldsymbol{u}_2$ is the one maximizing the projected variance and being orthogonal to $\boldsymbol{u}_1$

$$\boldsymbol{u}_2 = \arg\max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_1=0}} \boldsymbol{v}^T\boldsymbol{C}\boldsymbol{v} = \arg\max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_1=0}} E\left[\left(\boldsymbol{v}^T z\right)^2\right]. \tag{4}$$

Let $z_1 = z$ and $z_2 = z - [(z_1^T\boldsymbol{u}_1)/(\boldsymbol{u}_1^T\boldsymbol{u}_1)]\boldsymbol{u}_1$. By utilizing the orthogonal constraint $\boldsymbol{v}^T\boldsymbol{u}_1 = 0$, (4) can be expressed as

$$\begin{aligned} \boldsymbol{u}_2 &= \arg\max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_1=0}} E\left[\left(\boldsymbol{v}^T\left(z_2 + \frac{z_1^T\boldsymbol{u}_1}{\boldsymbol{u}_1^T\boldsymbol{u}_1}\boldsymbol{u}_1\right)\right)^2\right] \\ &= \arg\max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_1=0}} E\left[\left(\boldsymbol{v}^T z_2\right)^2\right]. \end{aligned} \tag{5}$$

Similarly, additional eigenvectors can be calculated by choosing each new vector to be that which maximizes the projected variance amongst all possible directions orthogonal to those already considered. Specifically, let $z_{j+1} = z - \sum_{i=1}^{j}[(z_i^T\boldsymbol{u}_i)/(\boldsymbol{u}_i^T\boldsymbol{u}_i)]\boldsymbol{u}_i$, $\boldsymbol{u}_{j+1}$ can be formulated as

$$\begin{aligned} \boldsymbol{u}_{j+1} &= \arg\max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_i=0 \\ 1\leq i\leq j}} \boldsymbol{v}^T\boldsymbol{C}\boldsymbol{v} = \arg\max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_i=0 \\ 1\leq i\leq j}} E\left[\left(\boldsymbol{v}^T z\right)^2\right] \\ &= \arg\max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_i=0 \\ 1\leq i\leq j}} E\left[\left(\boldsymbol{v}^T\left(z_{j+1} + \sum_{i=1}^{j}\frac{z_i^T\boldsymbol{u}_i}{\boldsymbol{u}_i^T\boldsymbol{u}_i}\boldsymbol{u}_i\right)\right)^2\right]. \end{aligned} \tag{6}$$

And finally, we have

$$\boldsymbol{u}_{j+1} = \arg\max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_i=0 \\ 1\leq i\leq j}} E\left[\left(\boldsymbol{v}^T z_{j+1}\right)^2\right]. \tag{7}$$

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

2) Heuristic to Simplify the PCA Calculation: Equation (7) contains j orthogonal constraints and, hence, calculating m eigenvectors requires $O(m^2n)$ operations. In order to apply (7) in adapting large-scale covariance matrix, we wish to perform it in linear time. This can be done by discarding $j-1$ orthogonal constraints $\{\boldsymbol{v}^T\boldsymbol{u}_i = 0 | i \in \{1, \ldots, j-1\}\}$. That is, we approximate $\boldsymbol{u}_{j+1}$ with the following relaxed optimization task:

$$\boldsymbol{u}_{j+1} \approx \arg \max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_j=0}} E\left[\left(\boldsymbol{v}^T\boldsymbol{z}_{j+1}\right)^2\right]. \tag{8}$$

Calculating the expectation in (8) is difficult. But it can be "discretized" by using a greedy heuristic

$$\boldsymbol{u}_{j+1}^{(g)} \approx \sum_{i=1}^{g} c_{j+1}^{(i)} \arg \max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_j=0}} \left(\boldsymbol{v}^T\boldsymbol{z}_{j+1}^{(i)}\right)^2 \tag{9}$$

where g denotes the index of generation and $c_{j+1}^{(i)}$ is a weight factor. Equation (9) can be easily merged into an ES algorithm in a way that $\boldsymbol{z}_{j+1}^{(i)}$ is collected from the population at the ith generation.

We also observe that since $\boldsymbol{z}_{j+1} = \boldsymbol{z} - \sum_{i=1}^{j}[(\boldsymbol{z}_i^T\boldsymbol{u}_i)/(\boldsymbol{u}_i^T\boldsymbol{u}_i)]\boldsymbol{u}_i = \boldsymbol{z}_j - [(\boldsymbol{z}_j^T\boldsymbol{u}_j)/(\boldsymbol{u}_j^T\boldsymbol{u}_j)]\boldsymbol{u}_j$, $\boldsymbol{z}_{j+1}$ is orthogonal to $\boldsymbol{u}_j$. This further removes the last orthogonal constraint in (9) and we have

$$\arg \max_{\substack{\boldsymbol{v}^T\boldsymbol{v}=1 \\ \boldsymbol{v}^T\boldsymbol{u}_j=0}} \left(\boldsymbol{v}^T\boldsymbol{z}_{j+1}^{(i)}\right)^2 = \frac{\boldsymbol{z}_{j+1}^{(i)}}{\|\boldsymbol{z}_{j+1}^{(i)}\|_2}. \tag{10}$$

Equation (10) indicates that the direction of $\boldsymbol{u}_{j+1}$ can be approximated by the direction of $\boldsymbol{z}_{j+1}$. Note that when reconstructing the covariance matrix with these approximated eigenvectors, not only their directions but also the length should be taken into consideration. To this end, we omit the normalization in (10), and $\boldsymbol{u}_{j+1}^{(g)}$ can be estimated as

$$\boldsymbol{u}_{j+1}^{(g)} \approx \sum_{i=1}^{g} c_{j+1}^{(i)}\boldsymbol{z}_{j+1}^{(i)} \tag{11}$$

where $j \in \{1, \ldots, m-1\}$ and $\{c_{j+1}^{(1)}, \ldots, c_{j+1}^{(g)}\}$ is a set of weights. The weights should be carefully chosen to make the estimation unbiased under certain condition. Moreover, when incorporated into an ES, (11) as well as the $\boldsymbol{z}_{j+1}^{(i)}$ calculation can be simplified by using recursive updating. This will be detailed in the next section.

Due to the summation operator in (11), the approximated eigenvectors are usually not pairwise orthogonal. Pairwise independence are not guaranteed either, because only a few orthogonal constraints are considered. However, since $\boldsymbol{z}_j$ is orthogonal to $\boldsymbol{u}_{j-1}$, $\boldsymbol{u}_{j+1}$ tends to be independent of $\boldsymbol{u}_j$. Thus, our heuristic turns the "pairwise independence" into "adjacent independence." To differentiate these estimated vectors from the real eigenvectors, we call them the "search directions."

III. Proposed SDA-ES

The proposed SDA-ES adopts the widely used (μ, λ)-ES framework such that λ candidate solutions are generated at each generation and the μ best ones are selected to update the search distribution.

A. Distribution Modeling and Sampling

SDA-ES maintains the following multivariate Gaussian distribution at its gth generation:

$$\mathcal{N}\left(\boldsymbol{m}^{(g)}, \left(\sigma^{(g)}\right)^2\boldsymbol{C}^{(g)}\right) \tag{12}$$

where

$\boldsymbol{m}^{(g)} \in R^n$ mean vector;
$\boldsymbol{C}^{(g)} \in R^{n\times n}$ covariance matrix;
$\sigma^{(g)} \in R$ mutation strength.

1) Covariance Matrix Modeling: SDA-ES differs from other CMA variants in that the covariance matrix takes the following form:

$$\boldsymbol{C}^{(g)} = (1 - c_{\text{cov}})\boldsymbol{I}_n + c_{\text{cov}} \sum_{i=1}^{m} \boldsymbol{q}_i^{(g)}\left(\boldsymbol{q}_i^{(g)}\right)^T \tag{13}$$

where

$c_{\text{cov}} \in (0, 1)$ weight factor;
$m \in Z_+$ number of used search directions;
$\boldsymbol{q}_i^{(g)} \in R^n$ ith search direction, where $i \in \{1, \ldots, m\}$.

Equation (13) approximates the covariance matrix with $\sum_{i=1}^{m}\boldsymbol{q}_i^{(g)}(\boldsymbol{q}_i^{(g)})^T$ whose rank is not larger than m in the case of $m \ll n$. To ensure the positive definiteness, it is linearly combined with an identity matrix under the control of a parameter c_{cov}. Such combination makes (13) have the same form of the well-known shrinkage estimator [27]. Shrinkage here improves the naive estimator $\sum_{i=1}^{m}\boldsymbol{q}_i^{(g)}(\boldsymbol{q}_i^{(g)})^T$ and produces a well-conditioned estimator $\boldsymbol{C}^{(g)}$.

2) Sampling: Sampling new solutions from the distribution $\mathcal{N}(\boldsymbol{m}^{(g)}, (\sigma^{(g)})^2\boldsymbol{C}^{(g)})$ requires no special matrix factorization techniques. Concretely, a new solution can be generated as

$$\boldsymbol{x}^{(g)} = \boldsymbol{m}^{(g)} + \sigma^{(g)}\left(\sqrt{1 - c_{\text{cov}}}\boldsymbol{z}_1 + \sqrt{c_{\text{cov}}}\boldsymbol{Q}^{(g)}\boldsymbol{z}_2\right) \tag{14}$$

where $\boldsymbol{z}_1 \in R^n$ is a random vector sampled from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n)$; $\boldsymbol{z}_2 \in R^m$ is a random vector sampled from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_m)$; and $\boldsymbol{Q}^{(g)} \in R^{n\times m}$ is a matrix whose ith column is $\boldsymbol{q}_i^{(g)}$, that is, $\boldsymbol{Q}^{(g)} = (\boldsymbol{q}_1^{(g)}, \ldots, \boldsymbol{q}_m^{(g)})$.

Equation (14) indicates that a solution sampled from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{C}^{(g)})$ can be seen as an isotropic Gaussian random vector perturbed by another nonisotropic one. Such a perturbation is defined in an m-dimensional space and then transformed to R^n using the linear transformation defined by $\boldsymbol{Q}^{(g)}$.

B. Search Direction Adaptation

SDA-ES adapts the search directions in order to capture the variable dependencies of the function landscape. This procedure is conducted by exploring the linear correlations among the variables of the mutation step of the mean vector. It contains three basic steps.

1) Moving the Mean: At the gth generation, μ solutions are recombined to form the mean vector

$$\boldsymbol{m}^{(g+1)} = \sum_{j=1}^{\mu} \omega_j \boldsymbol{x}_{j:\lambda}^{(g)} \tag{15}$$

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

where $\boldsymbol{x}_{j:\lambda}^{(g)}$ denotes the jth best population member and $f(\boldsymbol{x}_{1:\lambda}^{(g)}) \leq \cdots \leq f(\boldsymbol{x}_{\lambda:\lambda}^{(g)})$; and $\omega_1, \ldots, \omega_\mu$ are the recombination weights subject to $\sum_{j=1}^{\mu} \omega_j = 1$, and $\omega_1 \geq \cdots \geq \omega_\mu > 0$.

2) Calculating the Successful Mutation Step: SDA-ES learns the variable dependencies from the successful mutation step of the mean vector. This successful mutation step is given by

$$z^{(g)} = \sqrt{\mu_{\text{eff}}} \frac{\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}} \tag{16}$$

where $\mu_{\text{eff}} = 1/\sum_{j=1}^{\mu} \omega_j^2$.

It can be verified easily that under random selection, $z^{(g)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}^{(g)})$. In other words, $z^{(g)}$ is a well-defined statistic in estimating the covariance matrix, which explains why $z^{(g)}$ can be utilized in the adaptation procedure.

3) Adaptation: Given the fact $z^{(g)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}^{(g)})$ under random selection, we can use $z^{(g)}$ to update the search directions with the heuristic described in Section II-B.

a) Updating $\boldsymbol{q}_1^{(g)}$: The heuristic in (11) does not work for the first search direction. So we specify an initial step as follows.

Let $\boldsymbol{z}_1^{(g)} = z^{(g)}$. The first search direction $\boldsymbol{q}_1^{(g)}$ is updated as

$$\boldsymbol{q}_1^{(g+1)} = (1 - c_c)\boldsymbol{q}_1^{(g)} + \sqrt{c_c(2 - c_c)}\boldsymbol{z}_1^{(g)} \tag{17}$$

where $c_c \in (0, 1)$ is the learning rate for the search directions. Equation (17) is identical to the cumulation of the evolution path in [1].

b) Updating $\boldsymbol{q}_2^{(g)}, \ldots, \boldsymbol{q}_m^{(g)}$: After updating the jth search direction $\boldsymbol{q}_j^{(g)}$, where $j \in \{1, \ldots, m-1\}$, we first calculate the residual between $\boldsymbol{z}_j^{(g)}$ and its projection onto the new search direction $\boldsymbol{q}_j^{(g+1)}$. It is given by

$$\boldsymbol{z}_j^{(g)} - t \cdot \boldsymbol{q}_j^{(g+1)} \tag{18}$$

where

$$t = \frac{\left(\boldsymbol{z}_j^{(g)}\right)^T \boldsymbol{q}_j^{(g+1)}}{\left(\boldsymbol{q}_j^{(g+1)}\right)^T \boldsymbol{q}_j^{(g+1)}} \tag{19}$$

is the relative length of the projection.

Then, the residual is rescaled as

$$\boldsymbol{z}_{j+1}^{(g)} = \frac{1}{\sqrt{1 + t^2}}\left(\boldsymbol{z}_j^{(g)} - t \cdot \boldsymbol{q}_j^{(g+1)}\right). \tag{20}$$

The scaling coefficient $[1/(\sqrt{1 + t^2})]$ is to normalize the residual such that under random selection, $\boldsymbol{z}_{j+1}^{(g)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}^{(g)})$ holds when $\boldsymbol{z}_j^{(g)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}^{(g)})$ and $\boldsymbol{q}_j^{(g+1)} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C}^{(g)})$. In other words, $\boldsymbol{z}_{j+1}^{(g)}(\boldsymbol{z}_{j+1}^{(g)})^T$ is also an unbiased estimator of $\boldsymbol{C}^{(g)}$. Utilizing this property, the $(j+1)$th search direction can be updated as follows:

$$\boldsymbol{q}_{j+1}^{(g+1)} = (1 - c_c)\boldsymbol{q}_{j+1}^{(g)} + \sqrt{c_c(2 - c_c)}\boldsymbol{z}_{j+1}^{(g)}. \tag{21}$$

Fig. 1 provides an example of updating two search directions in the 2-D decision space. Suppose that the initial distribution is isotropic as shown in Fig. 1(a). The arrowed vectors depict the two search directions $\boldsymbol{q}_1^{(0)}$ and $\boldsymbol{q}_2^{(0)}$ and the dotted line depicts the contour of the probability density. In Fig. 1(b), a vector $\boldsymbol{z}_1^{(0)}$ is plotted to depict the successful mutation step of the population mean. It is recombined with the first search direction $\boldsymbol{q}_1^{(0)}$ using (17), yielding a new direction $\boldsymbol{q}_1^{(1)}$. This increases the probability of reproducing the successful mutation step. We use the same idea to update the other search direction $\boldsymbol{q}_2^{(0)}$. However, reusing $\boldsymbol{z}_1^{(0)}$ directly is inappropriate as the two search directions may become similar, leading to an ill-conditioned covariance matrix in the long run. Therefore, we use its component perpendicular to $\boldsymbol{q}_1^{(1)}$. This component, given by $\boldsymbol{z}_1^{(0)} - [((\boldsymbol{z}_1^{(0)})^T\boldsymbol{q}^{(1)})/((\boldsymbol{q}^{(1)})^T\boldsymbol{q}^{(1)})]\boldsymbol{q}^{(1)}$, is rescaled to obtain a new vector $\boldsymbol{z}_2^{(0)}$ shown in Fig. 1(c). Then, in Fig. 1(d), $\boldsymbol{z}_2^{(1)}$ is used to update $\boldsymbol{q}_2^{(0)}$. The new search directions are finally combined to form a new covariance matrix as shown in Fig. 1(e). The distribution contour becomes an ellipsoid which is elongated into the direction of z vector and, therefore, implies a promising search tendency. It is also observed that the second search direction moves away from $\boldsymbol{q}_1^{(1)}$ because it is recombined with a vector perpendicular to $\boldsymbol{q}_1^{(1)}$. This preserves the interdependence between the two

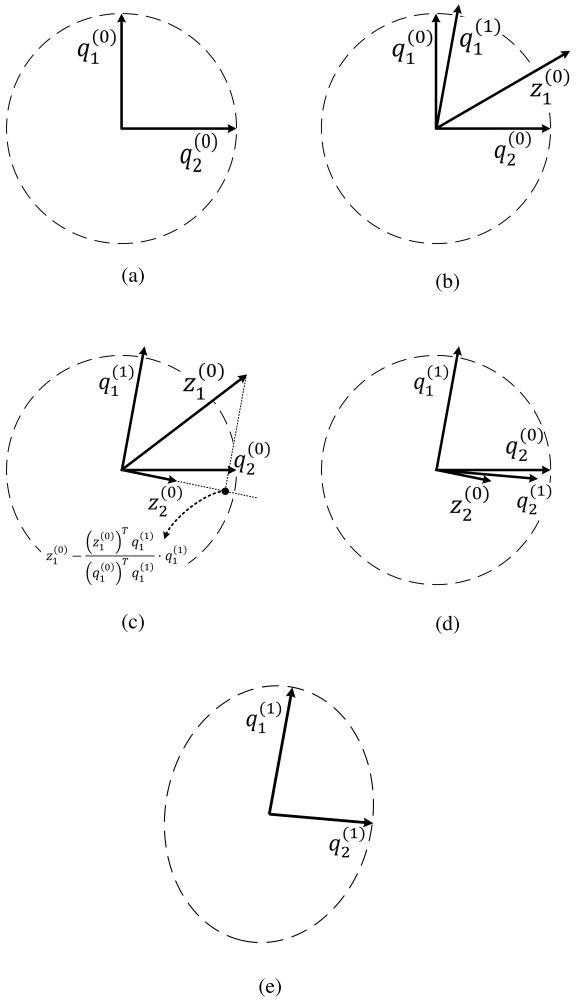


Fig. 1. 2-D example of the search direction adaptation. (a) Initial distribution. (b) Update $\boldsymbol{q}_1^{(0)}$ with $z_1^{(0)}$. (c) Calculate $z_2^{(0)}$. (d) Update $\boldsymbol{q}_2^{(0)}$ with $z_2^{(0)}$. (e) New distribution.

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

search directions which is helpful in large-scale covariance matrix adaptation.

4) Differences From MA-ES and LM-MA: The proposed method and those in MA-ES and LM-MA are all designed to simplify the original CMA-ES, so we would like to make some remarks about their differences.

1) MA-ES is not designed for large-scale optimization, and still maintains a full $n \times n$ matrix. Contrarily, this paper aims to scale up CMA-ES, and so the proposed method only stores a set of vectors (i.e., the search directions).
2) LM-MA and SDA-ES are different in the way of modeling the covariance matrix. SDA-ES reconstructs the covariance matrix while LM-MA reconstructs the Cholesky factor of the covariance matrix. Nevertheless, neither of them require extra computations for matrix decomposition.
3) LM-MA and SDA-ES are different in maintaining the vectors for (covariance) matrix reconstruction. LM-MA updates the used vectors with the weighted summations of vectors that are drawn from an isotropic Gaussian distribution. Whereas in SDA-ES, the search directions are the accumulations of the scaled residuals between the successful mutation steps and previous search directions.

C. Mutation Strength Adaptation

CSA in the original CMA-ES heavily relies on the sample distribution and a proper coordinate system transformation into isotropic coordinates [21]. Due to the incapacity of the sparse covariance matrix to capture all variable correlations, CSA is not suitable for SDA-ES.

Inspired by the comparison-based adaptation schemes, such as MSR [21], PSR [18], and RSR [17], we propose a generalized 1/5th success rule to adapt the mutation strength. The 1/5th success rule is originally developed for (1+1)-ES where "success" means accepting an offspring to be the new parent. It controls the acceptance probability in order to achieve the optimal convergence rate on the sphere function [26], [28]. In the nonelitist multirecombinant SDA-ES, we can generalize the 1/5th success rule by defining success as the rank sum of the objective function values getting smaller (better). It is based on the following two facts.

1) The previous and current populations have the same distribution under random selection. Hence, after combining these two populations and sorting all solutions, the sum of ranks from the previous population is not significantly larger than that from the current population.
2) A small mutation strength is likely to increase the probability of reproducing high-quality solutions.

The first fact can be used directly as the one-tailed null hypothesis of the well-known Mann–Whitney U test [25]. In this way, the success probability can be defined as the probability of rejecting the null hypothesis. We can calculate the success probability using the Mann–Whitney U test, and adapt the mutation strength such that the success probability reaches a predefined target. The procedure works as follows.

1) Calculating the Statistic U: We first combine objective function values from the previous and current populations into a mixed set: $F^{(\text{mix})} = F^{(g)} \cup F^{(g-1)}$. Then, we sort all the values in $F^{(\text{mix})}$ in ascending order. Let R_1 be the sum of the ranks from the previous population. A statistic, called U, is calculated as

$$U = R_1 - \frac{\lambda(\lambda+1)}{2}. \tag{22}$$

Under the null hypothesis of the Mann–Whitney U test, the statistic U has a distribution whose mean is $\lambda^2/2$ and standard deviation is $\sqrt{\lambda^2(2\lambda+1)/12}$.

2) Cumulating the Statistic Z: With the default parameter settings (discussed later), the sample size is large enough (e.g., $|F^{(\text{mix})}| = 2\lambda > 20$) to approximate U with a normal distribution. In other words, we have $U \sim \mathcal{N}(\lambda^2/2, \lambda^2(2\lambda+1)/12)$. Then, a statistic Z is obtained as

$$Z^{(g+1)} = (1-c_s)Z^{(g)} + \sqrt{c_s(2-c_s)}\frac{U-\lambda^2/2}{\sqrt{\lambda^2(2\lambda+1)/12}} \tag{23}$$

where $c_s \in (0, 1)$ is the learning rate for the mutation strength. The use of c_s is to average Z over generations to improve the robustness. The normalization in the last term in (23) is to eliminate the effect of the population size λ since $[(U - \lambda^2/2)/(\sqrt{\lambda^2(2\lambda+1)/12})] \sim \mathcal{N}(0, 1)$.

One distinct feature of the proposed adaptation scheme is that the Z calculation exhibits the so-called derandomization property: it is independent of μ, λ, and other distribution parameters to be adapted. In addition, $Z^{(g+1)} \sim \mathcal{N}(0, 1)$ holds under random selection and when $Z^{(g)} \sim \mathcal{N}(0, 1)$.

Since $Z^{(g+1)}$ is obtained by cumulating the normalized sum of the ranks, it reflects the variation tendency of the rank sum of the objective function values over the generations. In particular, $Z^{(g+1)} > 0$ indicates that the rank sum is getting smaller, and we consider the sampling operation at the gth generation is successful.

3) Applying the 1/5th Success Rule: After obtaining $Z^{(g+1)}$, we adapt the mutation strength as

$$\sigma^{(g+1)} = \sigma^{(g)} exp\left(\frac{1}{d_\sigma}\left(\frac{\Phi(Z^{(g+1)})}{1-p^*} - 1\right)\right) \tag{24}$$

where

$p^* \in (0, 1)$	predefined constant;
d_σ	damping factor controlling the changing rate of the mutation strength;
Φ	cumulative distribution function of the standard univariate normal distribution.

$\Phi(Z^{(g+1)})$ is the obtained success probability, and $1 - \Phi(Z^{(g+1)})$ is the right-tailed p-value, that is, the significance level of the hypothesis that $Z^{(g+1)} > 0$. Therefore, $\Phi(Z^{(g+1)})$ measures how successful the sampling operation at the gth generation is. The adaptation works as follows: when $\Phi(Z^{(g+1)})$ is smaller than a predefined value $1 - p^*$, we decrease the mutation strength to encourage exploitation. The algorithm then behaves like a local search and the success probability will increase naturally. Otherwise, we increase the mutation strength such that the solutions move to unexplored region and the success probability will gradually decrease. This procedure is repeated at each generation such that the success of the sampling operation eventually maintains at a p^* significance level.

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

4) Comparisons With Existing Schemes for Mutation Strength Adaptation: The similarities and differences between the proposed method and existing ones, including MSR, PSR, and RSR, are listed as follows.

1) The Mann–Whitney U test used in the generalized 1/5th success rule degenerates to a median test if the objective function values are drawn independently from two populations that have the same shape. This happens under random selection, and in this case, the proposed method is similar to MSR which captures the evolutionary tendency of the population medians. But they are not necessarily the same since the proposed method also detects the change in the spread of the objective function values even when the medians are equal. Note that there exist extensive studies concerning the difference between the Mann–Whitney U test and the test for medians, and interested readers are referred to [29].
2) PSR, RSR, and our proposed method all rely on the rank sums, but are different in measuring the success. PSR and RSR use the change of the rank sums to measure the success. Instead, our method first transforms the rank sums to success probabilities and, then, utilizes these probabilities to quantify the success. This also makes the considered methods differ in the process of normalization and accumulation.

D. Detailed Implementation

The pseudocode of SDA-ES is given in Algorithm 1. The algorithm maintains an $m \times n$ matrix $\boldsymbol{Q}^{(g)}$ at the generation g. The ith column of $\boldsymbol{Q}^{(g)}$, denoted by $\boldsymbol{Q}^{(g)}_{:,i}$, stores the ith search direction $\boldsymbol{q}^{(g)}_i$. Before starting the main loop, the matrix $\boldsymbol{Q}^{(g)}$ is initialized with columns drawn independent identically distributed from an isotropic Gaussian $10^{-10}\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n)$ in lines 2–4. The constant 10^{-10} is to make the lengths of search directions sufficiently small while the use of random variables is to prevent the potential "Division by Zero" error (occurred in line 16 at the first generation).

At the beginning of the gth generation, a set of λ solutions is sampled using the method described in Section III-A. These solutions are evaluated and the best μ ones are chosen to update the search directions as described in Section III-B. Specifically, the new mean $\boldsymbol{m}^{(g+1)}$ is first formed in line 12, and a vector $\boldsymbol{z}$ is obtained to measure the (scaled) successful mutation step of the population mean in line 13. Then, the search directions are adapted by updating the matrix $\boldsymbol{Q}^{(g)}$ column by column in lines 14–18. More precisely, we first update a search direction with the vector $\boldsymbol{z}$, second calculate the relative projection length of $\boldsymbol{z}$ onto the updated search direction, and finally, replace $\boldsymbol{z}$ with its scaled projection residual. This process is repeated until all the columns are updated.

Lines 19–23 perform the adaptation of the mutation strength. In lines 19 and 20, the objective function values from the previous and current populations are merged and ranked in order to obtain the rank-sum value R_1. Two statistics U and $Z^{(g+1)}$ are calculated based on R_1 in lines 21 and 22. Finally, in line 23, the mutation strength is updated based on the comparison result between $\Phi(Z^{(g+1)})$ and its target value $1 - p^*$.

Algorithm 1 SDA-ES

Input: $\boldsymbol{m}^{(0)}$: mean vector; $\sigma^{(0)}$: mutation strength;

1: $g = 0$
2: **for** $i = 1$ to m **do**
3: $\quad \boldsymbol{Q}^{(0)}_{:,i} \sim 10^{-10}\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n)$
4: **end for**
5: **while** the termination criterion is not met **do**
6: $\quad$ **for** $i = 1$ to λ **do**
7: $\quad\quad z_1 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n)$
8: $\quad\quad z_2 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_m)$
9: $\quad\quad \boldsymbol{x}^{(g)}_i = \boldsymbol{m}^{(g)} + \sigma^{(g)}(\sqrt{1 - c_{\text{cov}}}z_1 + \sqrt{c_{\text{cov}}}\boldsymbol{Q}^{(g)}z_2)$
10: $\quad$ **end for**
11: $\quad$ Evaluate and sort the solutions
12: $\quad \boldsymbol{m}^{(g+1)} = \sum_{j=1}^{\mu} \omega_j \boldsymbol{x}^{(g)}_{j:\lambda}$
13: $\quad z = \sqrt{\mu_{\text{eff}}}\frac{\boldsymbol{m}^{(g+1)} - \boldsymbol{m}^{(g)}}{\sigma^{(g)}}$
14: $\quad$ **for** $i = 1$ to m **do**
15: $\quad\quad \boldsymbol{Q}^{(g+1)}_{:,i} = (1 - c_c)\boldsymbol{Q}^{(g)}_{:,i} + \sqrt{c_c(2 - c_c)}z$
16: $\quad\quad t = \frac{z^T\boldsymbol{Q}^{(g+1)}_{:,i}}{(\boldsymbol{Q}^{(g+1)}_{:,i})^T\boldsymbol{Q}^{(g+1)}_{:,i}}$
17: $\quad\quad z = \frac{1}{\sqrt{1+t^2}}(z - t\boldsymbol{Q}^{(g+1)}_{:,i})$
18: $\quad$ **end for**
19: $\quad F^{(g)} = \{f(\boldsymbol{x}^{(g)}_1), \cdots, f(\boldsymbol{x}^{(g)}_\lambda)\}$
20: $\quad R_1 =$ sum of ranks of $F^{(g-1)}$ in $F^{(g)} \cup F^{(g-1)}$
21: $\quad U = R_1 - \frac{\lambda(\lambda+1)}{2}$
22: $\quad Z^{(g+1)} = (1 - c_s)Z^{(g)} + \sqrt{c_s(2 - c_s)}\frac{U - \lambda^2/2}{\sqrt{\lambda^2(2\lambda+1)/12}}$
23: $\quad \sigma^{(g+1)} = \sigma^{(g)}exp\Big(\frac{1}{d_\sigma}\Big(\frac{\Phi(Z^{(g+1)})}{1-p^*} - 1\Big)\Big)$
24: $\quad g = g + 1$
25: **end while**
26: **return**

E. Complexity

At each generation, sampling a new solution requires $O(mn)$ operations for the matrix-vector multiplication in line 9. Updating the matrix $\boldsymbol{Q}^{(g)}$ in lines 14–18 requires $O(mn)$ operations. Lines 11 and 20 perform two sorting procedures and require $O(\lambda\log\lambda)$ operations. All the other lines can be executed within $O(n)$ operations. Therefore, the time complexity of SDA-ES is $O(\lambda mn)$ per generation or $O(mn)$ per function evaluation.

SDA-ES maintains a population of λ solutions and stores m search directions in an $n \times m$ matrix. Thus, SDA-ES has $O((\lambda + m)n)$ space complexity.

F. Parameters

All parameters used in SDA-ES are summarized in Table I. Among them, λ, μ, $\omega_1, \ldots, \omega_\mu$, and μ_{eff} are those appeared in the original CMA-ES, and they are set based on the recommendation in [2]. The other parameters are discussed as follows.

1) c_{cov} controls the influence of the search directions on the estimated covariance matrix. As pointed out in [17], this kind of parameter essentially determines the changing rate of the covariance matrix and, hence, can be set based on some theoretical findings on this subject. It is

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

TABLE I
PARAMETERS FOR SDA-ES

$\lambda = 4 + \lfloor 3 \ln n \rfloor$, $\mu = \lfloor \frac{\lambda}{2} \rfloor$, $m = 10$, $c_{cov} = \frac{0.4}{\sqrt{n}}$, $c_c = \frac{0.25}{\sqrt{n}}$, $\omega_i = \frac{\ln(\mu+1) - \ln i}{\mu \ln(\mu+1) - \sum_{j=1}^{\mu} \ln j}, i \in \{1, \cdots, \mu\}$, $\mu_{eff} = \frac{1}{\sum_{i=1}^{\mu} \omega_i^2}$, $c_s = 0.3$, $d_\sigma = 1$, $p^* = 0.05$.

found by [30] that the covariance matrix converges to the inverse of the Hessian at the rate of $O(1/\sqrt{n})$ under the information geometric optimization framework, while by [31], that an isotropic ES achieves $O(n)$ expected time for a fixed relative improvement on a spherical function. Hence, a changing rate between $O(1/n)$ and $O(1/\sqrt{n})$ is reasonable. We use $c_{\text{cov}} = 0.4/\sqrt{n}$ in this paper.

2) c_c is the learning rate of the search directions. For a given search direction, about 37% of its information is older than $1/c_c$ generations. Hence, setting it to be inverse proportional to the degrees of freedom in the z vectors seems to be a reasonable choice. Note that $z_1^{(g)}$ in (17) has $O(n)$ degrees of freedom but the subsequent ones may have less degrees of freedom due to the orthogonal constraints. Consequently, we set it to the order of $O(1/\sqrt{n})$. In this paper, $c_c = 0.25/\sqrt{n}$ is used.
3) c_s and d_σ both control the changing rate of the mutation strength. d_σ directly determines the change magnitude of $\ln \sigma^{(g)}$ while c_s has an indirect effect by determining the accumulating rate of the statistic $Z^{(g)}$. Since the proposed adaptation scheme is independent of the population size and the number of parameters to be adapted, setting them to $O(1)$ is reasonable. In this paper, we recommend the settings $c_s = 0.3$ and $d_\sigma = 1$ which have been used in related works [17], [18], [21].
4) p^* is the target significance level for the success of the sampling operation. It plays a similar role like the acceptance probability in the original 1/5th success rule. The likelihood of accepting the null hypothesis, that is, the previous population has a smaller rank-sum value, increases with the decreasing p^*. We set p^* to 0.05 as it is a popular setting in generic statistical tests.
5) m is the number of used search directions. Generally, a larger m value is expected to lead a better performance, but also increases the computational burden. We set m to 10 as it is sufficient for the majority of the test instances as demonstrated by the numerical study.

IV. COMPARATIVE STUDIES

In this section, SDA-ES is compared with nine state-of-the-art algorithms to demonstrate its effectiveness in handling LSOPs.

A. Experimental Settings

1) Test Problems: The performance of SDA-ES is evaluated on two sets of test problems summarized in Table II. All problems have the global minimum value 0.

a) Set 1—basic test problems: The first set consists of 11 classic problems widely used in the EA community.

TABLE II
TEST PROBLEMS

Set 1: Basic Test Problems	
Name	Function
Sphere	$f_{\text{Sphere}}(\boldsymbol{x}) = \sum_{i=1}^{n} x_i^2$
Ellipsoid	$f_{\text{Elli}}(\boldsymbol{x}) = \sum_{i=1}^{n} 10^{6\frac{i-1}{n-1}} x_i^2$
Rosenbrock	$f_{\text{Rosen}}(\boldsymbol{x}) = \sum_{i=1}^{n-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$
Discus	$f_{\text{Discus}}(\boldsymbol{x}) = 10^6 x_1^2 + \sum_{i=2}^{n} x_i^2$
Cigar	$f_{\text{Cigar}}(\boldsymbol{x}) = x_1^2 + 10^6 \sum_{i=2}^{n} x_i^2$
Different Powers	$f_{\text{DiffPow}} = \sum_{i=1}^{n} \lvert x_i \rvert^{2+4\frac{i-1}{n-1}}$
Rotated Ellipsoid	$f_{\text{RotElli}}(\boldsymbol{x}) = f_{\text{Elli}}(\boldsymbol{Rx})$
Rotated Rosenbrock	$f_{\text{RotRosen}}(\boldsymbol{x}) = f_{\text{Rosen}}(\boldsymbol{Rx})$
Rotated Discus	$f_{\text{RotDiscus}}(\boldsymbol{x}) = f_{\text{Discus}}(\boldsymbol{Rx})$
Rotated Cigar	$f_{\text{RotCigar}}(\boldsymbol{x}) = f_{\text{Cigar}}(\boldsymbol{Rx})$
Rotated Different Powers	$f_{\text{RotDiffPow}}(\boldsymbol{x}) = f_{\text{DiffPow}}(\boldsymbol{Rx})$
Set 2: CEC'2010 LSGO Problems	
Function	Description
f_1 to f_3	Fully separable
f_4 to f_8	Single-group 50-non-separable
f_9 to f_{13}	10-group 50-non-separable
f_{14} to f_{18}	20-group 50-non-separable
f_{19} to f_{20}	Fully non-separable

* $\boldsymbol{R}$ is a full rank rotation matrix generated from standard normally distributed entries by Gram-Schmidt orthogonalization.

f_{Sphere}, the simplest and fully separable one, serves as a base for performance analysis. f_{Elli}, f_{Rosen}, f_{Discus}, f_{Cigar}, and f_{DiffPow} are separable or only have weak variable correlations. They are chosen to cover a variety of difficulties, such as ill-conditioning, nonlinear scaling, and flat region. Based on them, we further construct five nonseparable problems, namely, f_{RotElli}, f_{RotRosen}, $f_{\text{RotDiscus}}$, f_{RotCigar}, and $f_{\text{RotDiffPow}}$. This is done by applying an orthogonal transformation $\boldsymbol{R}$ on the function landscape before evaluating the function values [32]. The dimensions of nonrotated problems are set to 1000, 2500, 5000, 7500, and 10 000, respectively. However, each evaluation in the rotated case takes $O(n^2)$ time due to the matrix-vector multiplication in rotating the decision vectors, which is intractable in large-scale settings. Thus, for these rotated problems, we only consider the 1000-D instances. This setting is not likely to have significant influences on the results of this paper, as we will first demonstrate that the proposed algorithm is invariant against the rotational transformation.

b) Set 2—CEC'2010 benchmark problems: The second set is the CEC'2010 benchmark suite containing 20 problems. One distinct feature of this set is that the problem nonseparability can be explicitly controlled. Technically, we randomly classify the decision variables into several groups, each of which contains 50 variables. A predefined number of such groups are selected to go through a rotational transformation such that the correlations only exist among variables from the same group. In this paper, f_1–f_3 are fully separable, and f_{19}–f_{20} are fully nonseparable. For f_4–f_8, f_9–f_{13}, and f_{14}–f_{18}, the numbers of rotated groups are set to 1, 10, and 20, respectively. More details about their formulations and characteristics can be found in [33]. The dimensions of the problems in this set are fixed to 1000.

2) Algorithms for Comparison: CMA-ES and four large-scale variants, including sep-CMA [10], LM-MA [20],

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

LM-CMA [18], and RmES [17] are used in the empirical study. Another four non-CMA-ES-based algorithms, including the CC-based differential evolution (DECC-G) [34], the local search chain-based memetic algorithm (MA-SW) [35], the multiple offspring sampling (MOS) [36], and the random projection-based estimation of distribution algorithm (RP-EDA) [37] are also considered.

CMA-ES and its four variants are compared with SDA-ES on the first set of problems in order to show the rotational invariance and scalability, which are desirable features in designing new ESs. For all these algorithms, the mean vector is uniformly sampled in the range $[-5, 5]^n$ and the mutation strength is initialized to 3. All other parameters are set according to the suggestions from their original papers.

The four non-CMA-ES-based algorithms, as well as the original CMA-ES, are used in the comparison conducted on the second problem set. Among them, DECC-G is one of the pioneering works in large-scale optimization, MA-SW wins the CEC'2010 large-scale competition, while MOS becomes the competition winner from 2013 to 2017. RP-EDA is a recently proposed Gaussian EDA which shows excellent performance on the CEC'2010 benchmark suite. Unlike CMA-ES and its variants, RP-EDA employs an ensemble of distributions estimated in random subspaces. For these algorithms, related parameters are set based on the guidelines of the CEC'2010 competition as well as their original papers.

3) Termination Criteria: We use two termination criteria in the experiments on the basic test problems. The primary one is to stop the algorithm when the best objective value found is below 10^{-8}, which facilitates the study of the algorithms' convergence ability. We also employ the maximum number of function evaluations (denoted by mFEs) as a secondary criterion. This is useful in preventing taking too much time due to the stagnation. We set mFEs to 1×10^8 for $n = 1000$ and 5×10^8 for $n > 1000$.

For the experiments conducted on the second problem set, we terminate all algorithms when the consumed number of function evaluations exceeds 3×10^6, which is the default setting for the CEC'2010 competition.

4) Statistical Method: On each instance in the first test set, all competitors are independently run 20 times. The results obtained from the median run are reported. By median run, we mean the one obtaining the median rank value after sorting the results in lexicographical order such that: 1) the algorithms producing smaller objective values obtain better rank values and 2) if two algorithms yield the same results, the one using less function evaluations obtain better rank values. In the second test set, SDA-ES, DECC-G, and RP-EDA are independently run 25 times according to the guidelines of the CEC competition. For MA-SW and CMA-ES, the results in [17] are used directly. The results of MOS are from [38].

B. Effectiveness of the Generalized 1/5th Success Rule

The mutation strength adaptation is crucial for ESs since it sustains a large diversity or entertains fast convergence to the desired optimum [39]–[41]. Hence, before testing the overall performance of the ESs, we first investigate their effectiveness in adapting the mutation strength.

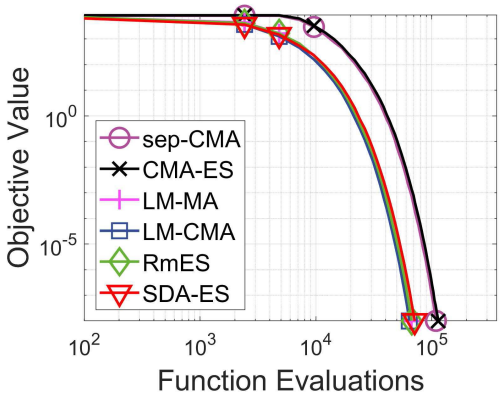


Fig. 2. Median results on the 1000-D Sphere problem, shown by evolutionary trajectories.

We test SDA-ES and other ES-based algorithms on the 1000-D Sphere function f_{Sphere} as shown in Fig. 2. It is known that this function can be efficiently solved using a standard (μ, λ)-ES with optimal mutation strength [42]. Hence, no adaptation for the covariance matrix is required, and the performance is mainly influenced by the mutation strength adaptation. Therefore, this experiment essentially reveals the effectiveness of the generalized 1/5th success rule.

sep-CMA and CMA-ES converge the slowest among all competitors. They have almost the same convergence curves as they both use CSA to control the mutation strength. This observation seems to be inconsistent with the theoretical finding that CSA seeks to optimize the mutation strength [31]. This is not surprising as the related parameters of CSA are not well tuned for large-scale optimization. LM-MA also uses CSA but with a larger adapting rate, and so it performs much better.

On the other hand, SDA-ES converges faster than CMA-ES and sep-CMA. It indicates that the generalized 1/5th success rule has certain advantages over CSA with default parameters. LM-CMA and RmES, which do not rely on CSA, also perform well. Considering their used adaptation schemes, we may conclude that their good performance may be attributed to the biased change of the mutation strength under random selection. However, SDA-ES differs from LM-CMA and RmES in that it quantifies such bias by using an unbiased statistic $Z^{(g)}$.

C. Invariance Under Search Space Rotational Transformations

We investigate the invariance of SDA-ES under rotational transformations on the 1000-D basic test problems. Invariance under rotational transformations refers to the property of a given algorithm that its performance on $f(\boldsymbol{x})$ and $f(\boldsymbol{Rx})$ is identical for any full-rank orthogonal matrix $\boldsymbol{R}$. Usually, the lack of the rotational invariance implies that an algorithm may perform better for some $\boldsymbol{R}$ than for others.

Fig. 3 plots the convergence curves on the nonrotated and rotated problems. In general, CMA-ES exhibits the invariance

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

against rotational transformations as its performance is almost the same on the rotated problems and the corresponding nonrotated problems. On the contrary, its separable version sep-CMA is quite sensitive to the search space rotations, as we can find a significant performance degradation in tackling with rotated problems. For SDA-ES, its covariance matrix is an identity matrix combined with a matrix whose rank is not larger than m, and so it cannot capture all pairwise variable correlations determined by the rotation matrix $\boldsymbol{R}$. However, we observe that the differences in performance on nonrotated and rotated problems are marginal, demonstrating the rotational invariance of SDA-ES.

On the Ellipsoid problem shown in Fig. 3(a) and (b), SDA-ES outperforms sep-CMA, CMA-ES, and RmES, and is competitive with LM-MA, showing the capacity of capturing the landscape characteristics with a low-rank model. This result is interesting, as the landscape of f_{RotElli} is far different from a low-rank model. SDA-ES is also competitive with LM-CMA at the early and middle stage. LM-CMA surpasses SDA-ES at the late stage, and its advantage may come from the cumulation technique for both the evolution paths and the covariance matrix. Another observation is that SDA-ES, LM-CMA, RmES, and LM-MA all surpass CMA-ES despite that none of them learn all variable correlations. One possible reason is the larger learning rate (e.g., c_c in SDA-ES) used in the four variants. Indeed, realizing fast adaptation may be essential since the computational resource is quite limited in the context of large-scale optimization.

The Discus problem has a global structure similar to the Sphere problem. However, it contains local irregularity in that one single direction (e.g., x_1 in the nonrotated case) is a thousand times more sensitive than all others. Since most ESs are initialized with an isotropic distribution, learning the relative scaling between such sensitive direction and others become challenging even on the nonrotated problems. As shown in Fig. 3(e) and (f), sep-CMA fails to convergence in the presence of rotation. On the nonrotated problem, it requires roughly 5×10^4 function evaluations to learn the appropriate scalings before performing an efficient convergence. CMA-ES consumes even more function evaluations due to its large amount of parameters to be adapted. LM-MA, LM-CMA, RmES, and SDA-ES perform better than CMA-ES and are invariant against rotations.

The Different Powers problem is similar to the Ellipsoid problem except that its variable correlations are nonlinear. The closer to the optimum, the more difficult an algorithm gets to approach it further. Therefore, this problem mainly tests the local exploitation ability. As shown in Fig. 3(i) and (j), SDA-ES is able to achieve competitive performance. This may be ascribed to that the vicinity of local optimum on this problem can be approximated by a sphere. In this case, the used identity matrix dominates the search, tending to produce high-quality solutions.

The Rosenbrock and Cigar problems share a common trait that they can be well approximated with a low-rank model: the landscape can be seen as a long valley embedded in a flat or spherical region. By using a sparse covariance matrix, SDA-ES and RmES are better than or as good as other competitors as shown in Fig. 3(c), (d), (g), and (h).

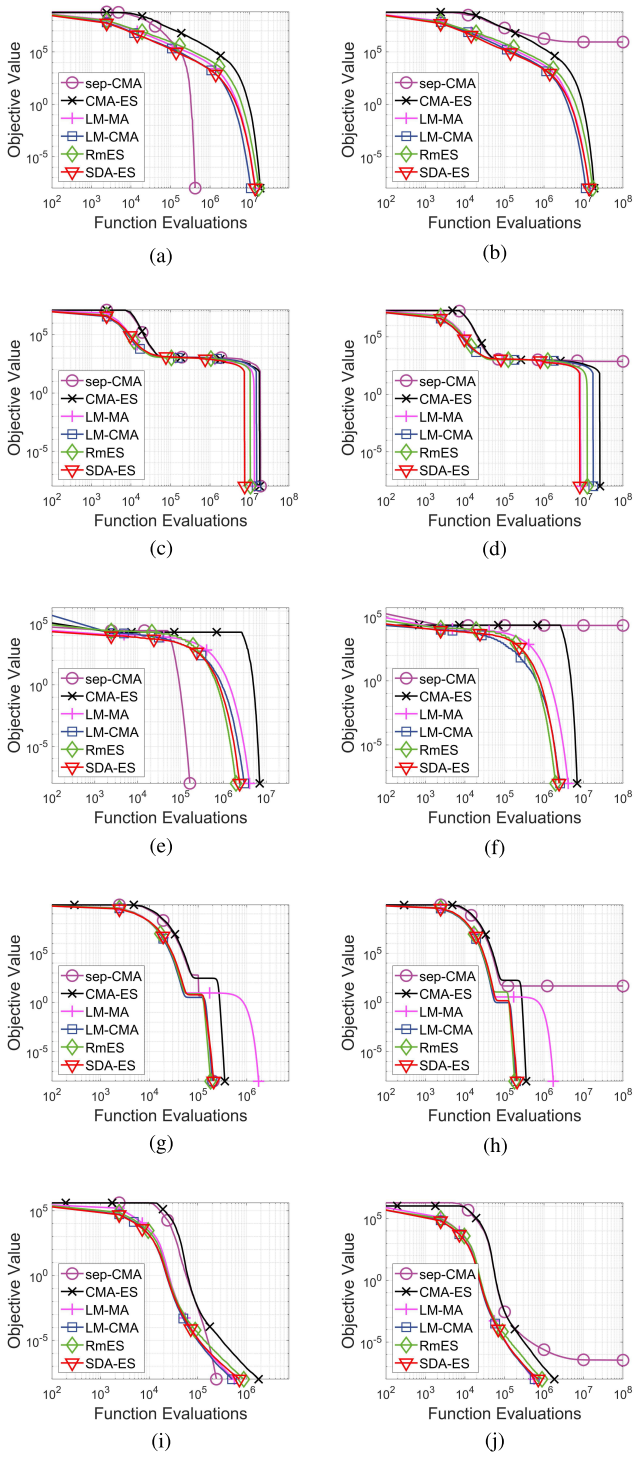


Fig. 3. Median results on the 1000-D basic test problems with and without rotation, shown by evolutionary trajectories. (a) f_{Elli}. (b) f_{RotElli}. (c) f_{Rosen}. (d) f_{RotRosen}. (e) f_{Discus}. (f) $f_{\text{RotDiscus}}$. (g) f_{Cigar}. (h) f_{RotCigar}. (i) f_{DiffPow}. (j) $f_{\text{RotDiffPow}}$.

D. Scalability

We test the scalability of SDA-ES in terms of the number of decision variables. LM-MA, LM-CMA, and RmES are

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

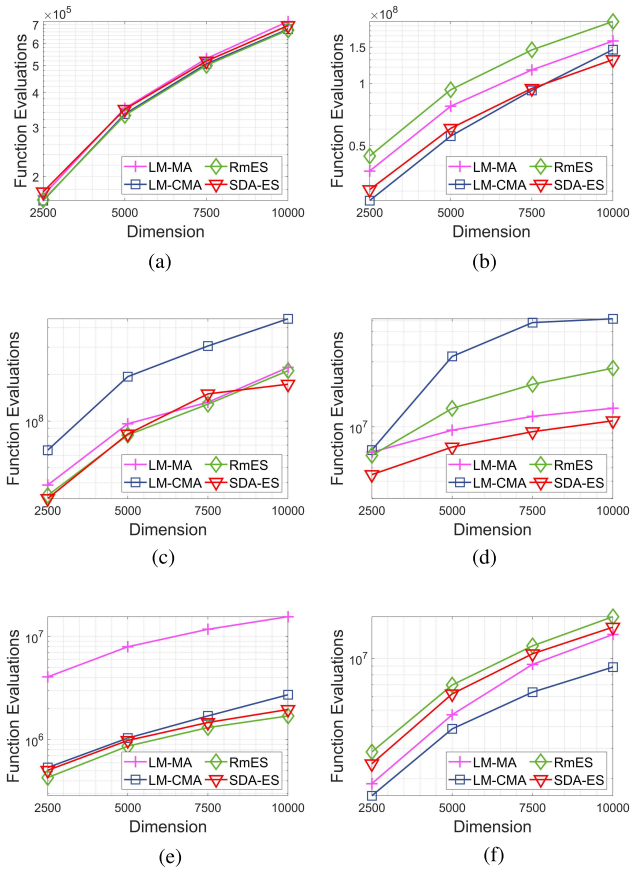


Fig. 4. Median results on the 2500-, 5000-, 7500-, and 10 000-D nonrotated problems. The curves present the number of function evaluations required to reach the accuracy 10^{-8}. (a) f_{Sphere}. (b) f_{Elli}. (c) f_{Rosen}. (d) f_{Discus}. (e) f_{Cigar}. (f) f_{DiffPow}.

selected as they all exhibit rotational invariance. The nonrotated problems in the first problem set are chosen, and their dimensions are set to 2500, 5000, 7500, and 10 000, respectively. The median function evaluations required to reach the accuracy 10^{-8} are given in Fig. 4.

On the Ellipsoid problem, SDA-ES possesses clear advantage over RmES and LM-MA on all instances, and outperforms LM-CMA when $n = 10\,000$. This shows the ability of SDA-ES to capture the global quadratic structure in high-dimensional space. SDA-ES is also robust in handling local irregularity, and insensitive to the initialization even when the initial distribution is inconsistent with the landscape characteristics. This can be verified on the Discus problem where SDA-ES converges the fastest on all instances, whereas the performance of other algorithms deteriorate significantly when the problem dimension increases.

SDA-ES sustains the capability of handling narrow and flat region when the dimension increases. On the Cigar problem, SDA-ES is slightly slower than RmES, but they both converge faster than LM-CMA and LM-MA do. The Rosenbrock problem is much harder than the Cigar problem. It is inherently nonseparable and its long valley cannot be rendered by a single linear transformation. Moreover, its valley becomes longer and narrower as the dimension increases. On this problem, SDA-ES surpasses LM-CMA on all dimensional test instances, and

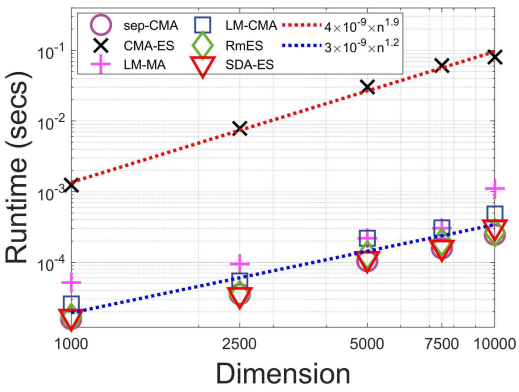


Fig. 5. CPU time per function evaluation on the f_{Elli} problem. The processing time of the function evaluations is excluded. The dotted lines are obtained using the least-square regression with the horizontal and vertical axes both in the logarithmic scale.

is better than or competitive with RmES and LM-MA on the 2500-, 5000-, and 10 000-D instances.

SDA-ES is better than RmES but is surpassed by LM-CMA and LM-MA on the Different Powers problem. But the difference in performance is minor even on the 10 000-D test instance, and all algorithms can converge within less than 2×10^7 function evaluations. Similar results can be observed on the Sphere problem where all competitors behave similarly. Considering that these two problems mainly challenge the convergence in the vicinity of local optima, we may conclude that SDA-ES is competitive with the state-of-the-art variants in the local exploitation ability.

E. Runtime and Memory Usage

We first validate the time complexity of SDA-ES analyzed in Section III-E. To this end, we compare the runtimes of SDA-ES and the other competitors required on the nonrotated Ellipsoid problem. The simulation is conducted on a PC with a 3.30-GHz Intel Core i5-4590 CPU. For a certain algorithm, its runtime refers to the CPU time divided by the number of used function evaluations, with the CPU time of the function evaluations excluded from the measurement. Therefore, the presented result quantifies the internal cost of the algorithm. Fig. 5 presents the results in a log–log plot. CMA-ES is obviously the slowest while the other variants are much faster. Specifically, SDA-ES, sep-CMA, and RmES are the fastest performers, being about 500 times faster than CMA-ES on the 10 000-D test instance. LM-CMA and LM-MA are slightly slower than SDA-ES, but still significantly faster than CMA-ES.

With the time results obtained above, we can determine the time complexity for these algorithms. The method is from [43], in which Bošković *et al.* used the regression models to capture the asymptotic performance for algorithms. Concretely, we plot two regression lines in Fig. 5 to show how fast the runtime scales for different algorithms. The regression lines are based on the least-square fit in a logarithmic scale. The red dotted line, displaying the polynomial $4 \times 10^{-9} \times n^{1.9}$, is derived from the runtime of CMA-ES. It indicates that CMA-ES scales

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

TABLE III
MEMORY USAGE

Algorithms	Number of Floating-point Variables
sep-CMA	$(\lambda + 4)n$
CMA-ES	$2n^2 + (\lambda + 4)n$
LM-MA	$(2\lambda + m + 3)n$
LM-CMA	$(\lambda + 2m + 2)n$
RmES	$(\lambda + m + 2)n$
SDA-ES	$(\lambda + m + 2)n$

approximately quadratically, being consistent with the fact that CMA-ES has to adapt a full covariance matrix. On the other hand, the polynomial derived from the runtimes of the rest algorithms (depicted as a blue dotted line) is $3 \times 10^{-9} \times n^{1.2}$. Hence, we can conclude that SDA-ES and the other large-scale variants scale approximately linearly.

We further examine the memory usage in terms of the number of floating-point variables. Our analysis is based on typical implementations of the algorithms and omit the terms that are independent of n. All six algorithms explicitly store λ solutions and one mean vector. sep-CMA, CMA-ES, and LM-MA store two evolution paths, LM-CMA and Rm-ES store one evolution path, while SDA-ES stores a temporary vector $\boldsymbol{z}$ (in line 13 of Algorithm 1). Apart from those, sep-CMA only requires one more vector to express the diagonal matrix, so it is the cheapest. CMA-ES, on the contrary, is the most expensive one, since it stores one covariance matrix and all its n eigenpairs. LM-CMA, LM-MA, RmES, and SDA-ES all store a set of m vectors to represent promising search directions. In addition, LM-CMA stores m more vectors to express the inverse of the covariance matrix, while LM-MA maintains an extra population of λ solutions. The details of memory usage have been summarized in Table III. Basically, all large-scale variants are well scalable in high-dimensional space.

F. Parameter Sensitiveness

The parameter m is the number of used search directions, and determines the low-rank structure of the covariance matrix. To investigate its effects on the performance of SDA-ES, we conduct experiments on the 1000-D nonrotated problems with m chosen from $\{1, 5, 10, \lambda, \lfloor\sqrt{n}\rfloor\}$.

Fig. 6 plots the median results in terms of the number of function evaluations required to reach the accuracy 10^{-8}. We first find that all algorithm instances behave similarly on the Sphere problem. It means employing more search directions than are necessary will not deteriorate the performance. Similar results can be observed on f_{Rosen} where the descend directions on the objective landscape can be efficiently approximated with single one search direction. This phenomenon may be attributed to that the unnecessary search directions are rapidly shortened in the projection-based adaptation.

On the other problems, using only one search direction is not enough as the corresponding algorithm instance converges much slower. In particular, it leads to a failure of convergence on f_{Cigar} within a reasonable computational budget. On the contrary, setting m to a large value significantly improves the convergence ability. This can be explained as a larger m admits

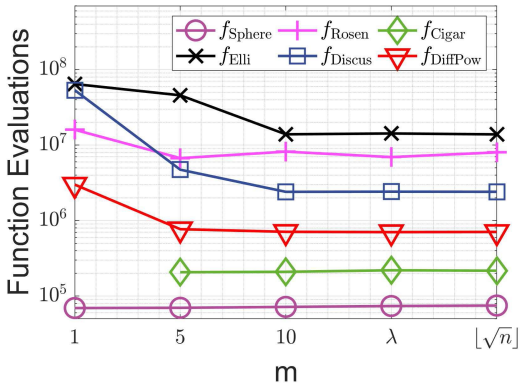


Fig. 6. Number of function evaluations required to reach the accuracy 10^{-8} on the 1000-D nonrotated problems with different settings of m.

capturing more promising tendencies. However, the advantage of using more search directions becomes marginal when $m \geq 10$. Considering that the time and space complexity scales linearly with m, setting m to 10 is a reasonable choice.

G. Performance on CEC'2010 Problems

This section provides the comparison results between SDA-ES and other five state-of-the-art algorithms on the CEC'2010 large-scale benchmark suite. Compared with the first problem set used in the above experiments, this set is more challenging as multimodal problems are involved. It is known that ESs are originally designed for unimodal functions, and are considered as local optimizers rather than global optimizers [44]. Hence, it may be inefficient to directly apply SDA-ES on this set due to its incapacity to escape the local optima. Fortunately, this weakness can be overcome by increasing the population size [45] and restarting the optimizer when a stagnation is detected [46], [47]. Therefore, we randomly restart SDA-ES when the range of the best objective values over the last n generations is below 10^{-8}. Each time an independent restart is launched, we increase the population size by a factor of 2. This strategy encourages SDA-ES to learn the global topology and omit the local characteristics; therefore, improving the performance on multimodal problems.

The median, mean, and standard deviation of the obtained objective values on the CEC'2010 problems are summarized in Table IV. On the whole, SDA-ES obtains the best results on 8 out of 20 problems and is top ranked according to the Friedman test [48]. It also significantly surpasses RP-EDA, DECC-G, and CMA-ES according to the Bonferroni's *post hoc* test [49]. The good performance of SDA-ES comes from its distribution adaptation in handling variable correlations, and the restart strategy in escaping local optima.

On fully separable problems f_1–f_3, SDA-ES is surpassed by MA-SW and MOS. The reason is that both MA-SW and MOS incorporate local search schemes to search along each variable separably or explore only a subset of variables. On the contrary, SDA-ES does not try to detect such separable structure and, therefore, does not perform well on these problems. CMA-ES performs even worse. This may be attributed to its

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
MEDIAN, MEAN, AND STANDARD DEVIATION OF THE OBJECTIVE VALUES OBTAINED ON THE CEC'2010 PROBLEMS. THE BEST AND THE SECOND BEST RESULTS FOR EACH PROBLEM, IN TERMS OF THE MEDIAN OBJECTIVE VALUE, ARE SHOWN WITH DARK AND LIGHT GRAY BACKGROUND, RESPECTIVELY

		RP-EDA	DECC-G	CMA-ES	MA-SW	MOS	SDA-ES
f_1	Median	5.09E+08	3.65E-07	8.49E+06	1.50E-14	0.00E+00	7.77E+03
	Mean	5.06E+08	3.62E-07	8.60E+06	2.10E-14	0.00E+00	7.71E+03
	Std Dev	2.54E+07	1.34E-07	8.01E+05	1.99E-14	0.00E+00	2.07E+03
f_2	Median	8.99E+02	1.32E+03	5.20E+03	7.90E+02	0.00E+00	1.07E+03
	Mean	9.08E+02	1.32E+03	5.21E+03	8.10E+02	0.00E+00	1.12E+03
	Std Dev	6.82E+01	3.45E+01	2.20E+02	5.88E+01	0.00E+00	1.81E+02
f_3	Median	1.53E+00	1.13E+00	2.17E+01	6.11E-13	1.67E-12	1.16E+00
	Mean	1.53E+00	1.13E+00	2.17E+01	7.28E-13	1.65E-12	1.11E+00
	Std Dev	1.86E-01	1.11E-01	1.06E-02	3.40E-13	6.44E-14	2.91E-01
f_4	Median	1.62E+12	2.56E+13	5.22E+13	3.54E+11	1.63E+10	4.17E+09
	Mean	1.57E+12	2.66E+13	5.90E+13	3.53E+11	1.70E+10	4.21E+09
	Std Dev	2.89E+11	8.76E+12	1.98E+13	3.12E+10	6.39E+09	3.31E+08
f_5	Median	1.15E+07	2.49E+08	6.44E+07	2.31E+08	1.08E+08	6.77E+07
	Mean	1.22E+07	2.63E+08	6.37E+07	1.68E+08	1.07E+08	6.84E+07
	Std Dev	3.86E+06	6.60E+07	1.31E+07	1.04E+08	2.49E+07	2.63E+07
f_6	Median	8.24E+01	4.85E+06	2.17E+01	1.60E+00	9.28E-08	2.12E+01
	Mean	8.26E+01	5.12E+06	2.58E+06	8.14E+04	1.11E-07	2.12E+01
	Std Dev	2.14E+00	1.05E+06	7.13E+06	2.84E+05	5.88E-08	4.27E-02
f_7	Median	4.23E+06	7.19E+08	1.41E+09	9.04E+01	0.00E+00	1.15E+05
	Mean	4.21E+06	9.61E+08	1.35E+09	1.03E+02	0.00E+00	1.20E+05
	Std Dev	1.56E+05	6.24E+08	3.29E+08	8.70E+01	0.00E+00	4.19E+04
f_8	Median	4.33E+07	8.82E+07	6.47E+08	3.43E+06	1.38E-07	1.50E+06
	Mean	5.86E+07	7.25E+07	1.08E+09	1.41E+07	1.40E+00	1.86E+06
	Std Dev	3.19E+07	3.19E+07	1.35E+09	3.68E+07	7.01E+00	1.41E+06
f_9	Median	5.30E+08	4.32E+08	9.55E+06	1.40E+07	3.47E+06	9.41E+03
	Mean	5.31E+08	4.26E+08	9.59E+06	1.41E+07	3.59E+06	9.36E+03
	Std Dev	3.17E+07	4.84E+07	1.07E+06	1.15E+06	4.89E+05	1.72E+03
f_{10}	Median	9.08E+02	1.01E+04	5.17E+03	2.07E+03	3.82E+03	1.12E+03
	Mean	9.15E+02	1.02E+04	5.20E+03	2.07E+03	3.81E+03	1.14E+03
	Std Dev	6.99E+01	3.13E+02	2.83E+02	1.44E+02	1.62E+02	1.69E+02
f_{11}	Median	5.64E+01	2.54E+01	2.38E+02	3.75E+01	1.91E+02	4.91E+01
	Mean	5.77E+01	2.54E+01	2.38E+02	3.80E+01	1.91E+02	5.11E+01
	Std Dev	6.20E+00	1.48E+00	1.97E-01	7.35E+00	4.01E-01	1.57E+01
f_{12}	Median	2.44E+05	9.92E+04	0.00E+00	3.50E-06	0.00E+00	0.00E+00
	Mean	2.45E+05	9.76E+04	0.00E+00	3.62E-06	0.00E+00	0.00E+00
	Std Dev	1.95E+04	1.10E+04	0.00E+00	5.92E-07	0.00E+00	0.00E+00
f_{13}	Median	1.31E+06	3.56E+03	7.85E+04	1.07E+03	5.69E+02	1.20E+01
	Mean	1.30E+06	6.17E+03	9.15E+04	1.25E+03	8.23E+02	2.14E+01
	Std Dev	6.01E+04	6.29E+03	6.77E+04	5.72E+02	6.77E+02	2.07E+01
f_{14}	Median	6.79E+08	9.98E+08	1.07E+07	3.09E+07	9.77E+06	9.77E+03
	Mean	6.76E+08	1.00E+09	1.06E+07	3.11E+07	9.69E+06	9.80E+03
	Std Dev	4.28E+07	6.23E+07	1.11E+06	1.93E+06	6.71E+05	1.69E+03
f_{15}	Median	9.30E+02	1.18E+04	5.16E+03	2.72E+03	7.45E+03	1.11E+03
	Mean	9.26E+02	1.20E+04	5.12E+03	2.74E+03	7.44E+03	1.15E+03
	Std Dev	5.68E+01	7.06E+02	1.98E+02	1.22E+02	1.90E+02	1.51E+02
f_{16}	Median	1.48E+02	7.32E+01	4.33E+02	9.44E+01	3.87E+02	8.76E+01
	Mean	1.54E+02	7.21E+01	4.33E+02	9.98E+01	3.79E+02	9.12E+01
	Std Dev	1.59E+01	4.74E+00	2.83E-01	1.40E+01	1.83E+01	1.55E+01
f_{17}	Median	5.29E+05	3.09E+05	0.00E+00	1.26E+00	2.67E-07	0.00E+00
	Mean	5.36E+05	3.09E+05	0.00E+00	1.24E+00	2.73E-07	0.00E+00
	Std Dev	2.50E+04	2.28E+04	0.00E+00	1.25E-01	7.67E-08	0.00E+00
f_{18}	Median	5.80E+04	3.03E+04	2.51E+03	1.19E+03	1.55E+03	2.39E+01
	Mean	6.00E+04	3.23E+04	3.36E+03	1.30E+03	1.77E+03	3.96E+01
	Std Dev	1.14E+04	1.47E+04	1.81E+03	4.36E+02	9.57E+02	3.72E+01
f_{19}	Median	2.10E+06	1.13E+06	2.81E+06	2.85E+05	2.87E+04	4.32E-05
	Mean	2.10E+06	1.13E+06	2.87E+06	2.85E+05	2.92E+04	4.31E-05
	Std Dev	1.11E+05	5.54E+04	6.61E+05	1.78E+04	2.29E+03	8.50E-06
f_{20}	Median	2.06E+03	4.22E+03	8.29E+02	1.06E+03	1.81E+02	4.35E+02
	Mean	2.06E+03	4.45E+03	8.54E+02	1.07E+03	2.93E+02	4.62E+02
	Std Dev	1.59E+02	6.77E+02	6.71E+01	7.29E+01	3.99E+02	1.75E+02
	No. of Best	3	2	2	1	7	8
	Avg Rank	4.35	4.6	4.47	2.95	2.45	2.17
	p Value	0.0034	0.0006	0.0014	1	1	

[1] "Avg Rank" denotes the averaged ranking results among all algorithms according to the Friedman test.
[2] The output of the Friedman test is further adjusted using the Bonferroni correction. The corresponding results, in terms of the "p-Value", are presented in the last row.

low adaptation rate, which has been observed in the above experiments.

Another observation is that SDA-ES outperforms DECC-G on two nonseparable multimodal problems f_2 and f_3 which contain a huge number of local optima. This is impressive, as the differential evolution (DE) [50] in DECC-G is usually considered as having better global exploration ability than ES [51], [52]. This indicates that the restart strategy does improve the performance on multimodal problems.

On f_{19} and f_{20}, SDA-ES produces, respectively, the best and the second best results, showing its advantage in handling fully nonseparable problems. For example, it is the only one that is able to produce near-optimal solutions on f_{19}. We also observe that its performance increases when the number of independent variables decreases. For example, SDA-ES obtains only one best result on single-group 50-nonseparable problems, but performs the best on almost all 10- and 20-group nonseparable problems. Besides, on the 20-group or fully nonseparable problems where no variable is independent, the algorithm performance seems to be mainly determined by the ability of handling variable correlations, while the impact of multimodality becomes inessential. Due to its capacity in handling variable dependencies, SDA-ES shows outstanding performance on these problems.

V. CONCLUSION

This paper presented an SDA-ES for large-scale optimization. Compared with the original CMA-ES, the covariance matrix in SDA-ES is modeled with an identity matrix and a set of search directions. The search directions are updated by simulating the PCA but in a computationally cheaper way. We also provide a generalized 1/5th success rule to control the mutation strength based on the Mann–Whitney U test and the idea of derandomization. SDA-ES has linear time and space complexity. The experiments carried out on 31 test problems demonstrate that SDA-ES is competitive with the state-of-the-art algorithms in handling LSOPs.

In the future, we will investigate how to incorporate the search direction adaptation into the covariance matrix cumulation as it will naturally increase the adaptation speed. Using the search directions to reconstruct the Cholesky factor rather than building the entire covariance matrix would also be interesting. Extending current work to solve constrained problems is also one of the further studies.

REFERENCES

[1] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evol. Comput.*, vol. 9, no. 2, pp. 159–195, Jun. 2001.
[2] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evol. Comput.*, vol. 11, no. 1, pp. 1–18, Mar. 2003.
[3] N. Hansen, A. Niederberger, L. Guzzella, and P. Koumoutsakos, "A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion," *IEEE Trans. Evol. Comput.*, vol. 13, no. 1, pp. 180–197, Feb. 2009.
[4] H.-G. Beyer and S. Finck, "On the design of constraint covariance matrix self-adaptation evolution strategies including a cardinality constraint," *IEEE Trans. Evol. Comput.*, vol. 16, no. 4, pp. 578–596, Aug. 2012.
[5] T. Suttorp, N. Hansen, and C. Igel, "Efficient covariance matrix update for variable metric evolution strategies," *Mach. Learn.*, vol. 75, no. 2, pp. 167–197, May 2009.

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

[6] S. Mahdavi, M. E. Shiri, and S. Rahnamayan, "Metaheuristics in large-scale global continues optimization: A survey," *Inf. Sci.*, vol. 295, pp. 407–428, Feb. 2015.

[7] M. A. Potter and K. A. Jong, "A cooperative coevolutionary approach to function optimization," in *Proc. Parallel Problem Solving Nat. (PPSN III)*, vol. 866. Jerusalem, Israel, 1994, pp. 249–257.

[8] J. Liu and K. Tang, "Scaling up covariance matrix adaptation evolution strategy using cooperative coevolution," in *Intelligent Data Engineering and Automated Learning—IDEAL 2013*, vol. 8206. Heidelberg, Germany: Springer, 2013, pp. 350–357.

[9] Y. Mei, M. N. Omidvar, X. Li, and X. Yao, "A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization," *ACM Trans. Math. Softw.*, vol. 42, no. 2, pp. 1–24, Jun. 2016.

[10] R. Ros and N. Hansen, "A simple modification in CMA-ES achieving linear time and space complexity," in *Proc. 10th Int. Conf. Parallel Problem Solving Nat. PPSN X*, 2008, pp. 296–305.

[11] T. Schaul, T. Glasmachers, and J. Schmidhuber, "High dimensions and heavy tails for natural evolution strategies," in *Proc. 13th Annu. Conf. Genet. Evol. Comput. (GECCO)*, Dublin, Ireland, Jul. 2011, p. 845.

[12] J. Poland and A. Zell, "Main vector adaptation: A CMA variant with linear time and space complexity," in *Proc. 3rd Annu. Conf. Genet. Evol. Comput. (GECCO)*, San Francisco, CA, USA, 2001, pp. 1050–1055.

[13] Y. Sun, T. Schaul, F. Gomez, and J. Schmidhuber, "A linear time natural evolution strategy for non-separable functions," in *Proc. 15th Annu. Conf. Companion Genet. Evol. Comput. (GECCO)*, Amsterdam, The Netherlands, Jul. 2013, p. 61.

[14] Y. Akimoto, A. Auger, and N. Hansen, "Comparison-based natural gradient optimization in high dimension," in *Proc. Annu. Conf. Genet. Evol. Comput. (GECCO)*, Vancouver, BC, Canada, Jul. 2014, pp. 373–380.

[15] J. N. Knight and M. Lunacek, "Reducing the space-time complexity of the CMA-ES," in *Proc. 9th Annu. Conf. Genet. Evol. Comput. (GECCO)*, London, U.K., Jul. 2007, p. 658.

[16] Y. Akimoto and N. Hansen, "Projection-based restricted covariance matrix adaptation for high dimension," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, Denver, CO, USA, Jul. 2016, pp. 197–204.

[17] Z. Li and Q. Zhang, "A simple yet efficient evolution strategy for large scale black-box optimization," *IEEE Trans. Evol. Comput.*, vol. 22, no. 5, pp. 637–646, Oct. 2018.

[18] I. Loshchilov, "LM-CMA: An alternative to L-BFGS for large-scale black box optimization," *Evol. Comput.*, vol. 25, no. 1, pp. 143–171, Mar. 2017.

[19] H.-G. Beyer and B. Sendhoff, "Simplify your covariance matrix adaptation evolution strategy," *IEEE Trans. Evol. Comput.*, vol. 21, no. 5, pp. 746–759, Oct. 2017.

[20] I. Loshchilov, T. Glasmachers, and H.-G. Beyer, "Large scale black-box optimization by limited-memory matrix adaptation," *IEEE Trans. Evol. Comput.*, vol. 23, no. 2, pp. 353–358, Apr. 2019.

[21] O. A. Elhara, A. Auger, and N. Hansen, "A median success rule for non-elitist evolution strategies: Study of feasibility," in *Proc. 15th Annu. Conf. Genet. Evol. Comput. (GECCO)*, Amsterdam, The Netherlands, Jul. 2013, p. 415.

[22] M. Brand, "Fast low-rank modifications of the thin singular value decomposition," *Linear Algebra Appl.*, vol. 415, no. 1, pp. 20–30, May 2006.

[23] S.-I. Amari, "Natural gradient works efficiently in learning," *Neural Comput.*, vol. 10, no. 2, pp. 251–276, Feb. 1998.

[24] B. Moore, "Principal component analysis in linear systems: Controllability, observability, and model reduction," *IEEE Trans. Autom. Control*, vol. 26, no. 1, pp. 17–32, Feb. 1981.

[25] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Ann. Math. Stat.*, vol. 18, no. 1, pp. 50–60, Mar. 1947.

[26] N. Hansen, D. V. Arnold, and A. Auger, "Evolution strategies," in *Springer Handbook of Computational Intelligence*, J. Kacprzyk and W. Pedrycz, Eds. Heidelberg, Germany: Springer, 2015, pp. 871–898.

[27] O. Ledoit and M. N. Wolf, "Honey, I shrunk the sample covariance matrix," *SSRN Electron. J.*, vol. 30, no. 4, pp. 110–119, Jun. 2003.

[28] H.-G. Beyer, *The Theory of Evolution Strategies*. Berlin, Germany: Springer, 2001.

[29] A. Hart, "Mann–Whitney test is not just a test of medians: Differences in spread can be important," *BMJ*, vol. 323, no. 7309, pp. 391–393, Aug. 2001.

[30] H.-G. Beyer, "Convergence analysis of evolutionary algorithms that are based on the paradigm of information geometry," *Evol. Comput.*, vol. 22, no. 4, pp. 679–709, Dec. 2014.

[31] D. Arnold and H.-G. Beyer, "Performance analysis of evolutionary optimization with cumulative step length adaptation," *IEEE Trans. Autom. Control*, vol. 49, no. 4, pp. 617–622, Apr. 2004.

[32] R. Salomon, "Evolutionary algorithms and gradient search: Similarities and differences," *IEEE Trans. Evol. Comput.*, vol. 2, no. 2, pp. 45–55, Jul. 1998.

[33] K. Tang, L. Xiaodong, P. N. Suganthan, Y. Zhenyu, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on large-scale global optimization," Nat. Inspired Comput. Appl. Lab., Univ. Sci. Technol. of China, Hefei, China, Rep., 2009.

[34] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Inf. Sci.*, vol. 178, no. 15, pp. 2985–2999, Aug. 2008.

[35] D. Molina, M. Lozano, and F. Herrera, "MA-SW-Chains: Memetic algorithm based on local search chains for large scale continuous global optimization," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, Jul. 2010, pp. 1–8.

[36] A. LaTorre, S. Muelas, and J.-M. Pena, "Large scale global optimization: Experimental results with MOS-based hybrid algorithms," in *Proc. Congr. Evol. Comput.*, Cancun, Mexico, Jun. 2013, pp. 2742–2749.

[37] A. Kabán, J. Bootkrajang, and R. J. Durrant, "Toward large-scale continuous EDA: A random matrix theory perspective," *Evol. Comput.*, vol. 24, no. 2, pp. 255–291, Jun. 2016.

[38] A. LaTorre, S. Muelas, and J.-M. Peña, "A comprehensive comparison of large scale global optimizers," *Inf. Sci.*, vol. 316, pp. 517–549, Sep. 2015.

[39] N. Hansen, A. Atamna, and A. Auger, "How to assess step-size adaptation mechanisms in randomised search," in *Proc. Parallel Problem Solving Nat. (PPSN XIII)*, Ljubljana, Slovenia, 2014, pp. 60–69.

[40] H.-G. Beyer and M. Hellwig, "The dynamics of cumulative step size adaptation on the ellipsoid model," *Evol. Comput.*, vol. 24, no. 1, pp. 25–57, Mar. 2016.

[41] A. Ostermeier, A. Gawelczyk, and N. Hansen, "Step-size adaptation based on non-local use of selection information," in *Proc. Parallel Problem Solving Nat. (PPSN III)*, vol. 866. Jerusalem, Israel, 1994, pp. 189–198.

[42] D. V. Arnold, "Weighted multirecombination evolution strategies," *Theor. Comput. Sci.*, vol. 361, no. 1, pp. 18–37, Aug. 2006.

[43] B. Bošković, F. Brglez, and J. Brest, "Low-autocorrelation binary sequences: On improved merit factors and runtime predictions to achieve them," *Appl. Soft Comput.*, vol. 56, pp. 262–285, Jul. 2017.

[44] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Proc. IEEE Int. Conf. Evol. Comput.*, May 1996, pp. 312–317.

[45] N. Hansen and S. Kern, "Evaluating the CMA evolution strategy on multimodal test functions," in *Proc. Parallel Problem Solving Nat. (PPSN VIII)*, Birmingham, U.K., 2004, pp. 282–291.

[46] A. Auger and N. Hansen, "A restart CMA evolution strategy with increasing population size," in *Proc. IEEE Congr. Evol. Comput.*, vol. 2, 2005, pp. 1769–1776.

[47] I. Loshchilov, M. Schoenauer, and M. Sebag, "Alternative restart strategies for CMA-ES," in *Proc. Parallel Problem Solving Nat. (PPSN XII)*, vol. 7492. Taormina, Italy, 2012, pp. 296–305.

[48] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *J. Amer. Stat. Assoc.*, vol. 32, no. 200, pp. 675–701, Dec. 1937.

[49] O. J. Dunn, "Multiple comparisons among means," *J. Amer. Stat. Assoc.*, vol. 56, no. 293, pp. 52–64, Mar. 1961.

[50] K. Price, R. Storn, and J. Lampinen, *Differential Evolution A Practical Approach to Global Optimization*, G. Rozenberg, T. Bäck, A. Eiben, J. Kok, and H. Spaink, Eds. Heidelberg, Germany: Springer-Verlag, 2005.

[51] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[52] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution—An updated survey," *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.

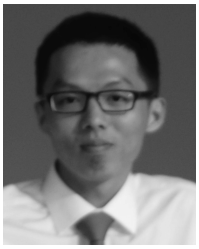
Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.

Xiaoyu He received the B.Eng. degree in computer science and technology from Beijing Electronic Science and Technology Institute, Beijing, China, in 2010, the M.Sc. degree in public administration from the South China University of Technology, Guangzhou, China, in 2016, and the Ph.D. degree in computer science from Sun Yat-sen University, Guangzhou, in 2019.

His current research interests include evolutionary computation and data mining.

Yuren Zhou received the B.Sc. degree in mathematics from Peking University, Beijing, China, in 1988, and the M.Sc. degree in mathematics and the Ph.D. degree in computer science from Wuhan University, Wuhan, China, in 1991 and 2003, respectively.

He is currently a Professor with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China. His current research interests include design and analysis of algorithms, evolutionary computation, and social networks.



Zefeng Chen received the B.Sc. degree in information and computational science from Sun Yat-sen University, Guangzhou, China, in 2013, the M.Sc. degree in computer science and technology from the South China University of Technology, Guangzhou, in 2016, and the Ph.D. degree in computer science from Sun Yat-sen University in 2019.

His current research interests include evolutionary computation, multiobjective optimization, data mining, and machine learning.

Jun Zhang (M'02–SM'08–F'17) received the Ph.D. degree from the City University of Hong Kong, Hong Kong, in 2002.

He is currently a Professor with the University of Victoria, Melbourne, VIC, Australia. His current research interests include computational intelligence, cloud computing, data mining, and power-electronic circuits. He has published over 200 technical papers in the above areas.

Prof. Zhang was a recipient of the China National Funds for Distinguished Young Scientists from the National Natural Science Foundation of China in 2011 and the First-Grade Award in Natural Science Research from the Ministry of Education, China, in 2009. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON CYBERNETICS, and the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS.

Wei-Neng Chen (S'07–M'12–SM'17) received the bachelor's and Ph.D. degrees from Sun Yat-sen University, Guangzhou, China, in 2006 and 2012, respectively.

Since 2016, he has been a Full Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou. He has coauthored over 100 international journal and conference papers, including over 30 papers published in the IEEE TRANSACTIONS journals. His current research interests include computational intelligence, swarm intelligence, network science, and their applications.

Dr. Chen was a recipient of the IEEE Computational Intelligence Society (CIS) Outstanding Dissertation Award in 2016 and the National Science Fund for Excellent Young Scholars in 2016. He is currently the Vice-Chair of the IEEE Guangzhou Section. He is also a Committee Member of the IEEE CIS Emerging Topics Task Force. He serves as an Associate Editor for the IEEE TRANSACTIONS ON NETWORKS AND LEARNING SYSTEMS and *Complex & Intelligent Systems*.

Authorized licensed use limited to: Hanyang University. Downloaded on November 06,2023 at 06:53:04 UTC from IEEE Xplore. Restrictions apply.