MMES: Mixture Model based Evolution Strategy for Large-Scale Optimization

Xiaoyu He, Zibin Zheng, Yuren Zhou

Abstract—This work provides an efficient sampling method for the covariance matrix adaptation evolution strategy (CMA-ES) in large-scale settings. In contrast to the Gaussian sampling in CMA-ES, the proposed method generates mutation vectors from a mixture model, which facilitates exploiting the rich variable correlations of the problem landscape within a limited time budget. We analyze the probability distribution of this mixture model and show that it approximates the Gaussian distribution of CMA-ES with a controllable accuracy. We use this sampling method, coupled with a novel method for mutation strength adaptation, to formulate the mixture model based evolution strategy (MMES) — a CMA-ES variant for large-scale optimization. The numerical simulations show that, while significantly reducing the time complexity of CMA-ES, MMES preserves the rotational invariance, is scalable to high dimensional problems, and is competitive against the state-of-the-arts in performing global optimization.

Index Terms—large-scale optimization, covariance matrix adaptation, mutation strength adaptation, evolution strategy, mixture model.

I. INTRODUCTION

Evolution strategies (ESs) [1] are a class of powerful evolutionary algorithms for black-box real-valued optimization. ESs sample candidate solutions from a multivariate Gaussian distribution and adapt the distribution to increase the likelihood of reproducing high-quality solutions. The covariance matrix adaptation ES (CMA-ES) [2], [3], a modern ES implementation that resembles second-order methods, adapts the covariance matrix to the shape of the function landscape by learning the linear correlations among variables. In the ideal situation when all variable correlations are learned, the CMA-ES on any convex quadratic function behaves like a standard ES on a spheric function, which exhibits a linear convergence rate [4]. CMA-ES also has the invariance against the rotational transformations on the decision space and the order-preserving transformations on the objective function. These properties make CMA-ES a popular black-box solver in many real-world applications such as artificial intelligence [5], [6], engineering design [7], [8], and automatic control [9], [10].

The standard CMA-ES explicitly store all variable correlations in a covariance matrix of size $n \times n$, leading to a time complexity of $O(n^2)$ per generation and a space complexity of $O(n^2)$, where $n$ is the number of variables. Thus, CMA-ES is more time-/space-consuming than other typical evolutionary algorithms, which is usually regarded as the main limitation in large-scale optimization tasks. One pioneering work to address this issue is the separable CMA-ES (sep-CMA) [11], sep-CMA discards all off-diagonal entries of the covariance matrix and reduces the time and space complexity to $O(n)$. However, this makes sep-CMA incapable of handling variable correlations and causes significant performance degradation on non-separable functions. Later studies generally focus on seeking a trade-off between keeping only diagonal entries and keeping the full matrix.

One popular method for balancing the performance and efficiency of CMA-ES in large-scale optimization is based on the covariance matrix reconstruction techniques. The basic idea is to model the covariance matrix with a set of $m$ vectors, which is much computationally cheaper than explicitly maintaining a full covariance matrix, provided that $m \ll n$. The direction vectors are designed to represent promising search directions, so the algorithms learn only a few variable correlations that contribute substantially to the optimization progress. Algorithms based on this method [12]–[16] generally have competitive performance and preserve the rotational invariance, but only have a limited amount of computational burden. There also exist works [17], [18] that hybridize this idea with the diagonal-learning strategy, aiming to further exploit the separability on relatively simple problems. Incorporating CMA-ES with the cooperative co-evolution (CC) framework [19] is another alternative for reducing the algorithm complexity. The CC based variants [20]–[23] implicitly maintain a set of sub-blocks of the covariance matrix and can perform well on additive separable problems. However, they receive all deficiencies of the CC framework and, for example, cannot solve fully non-separable functions [24].

Covariance matrix modeling techniques have also been studied in the context of estimation of distribution algorithms. In [25], [26], the high-dimensional covariance matrix is defined as an ensemble of low-dimensional covariance matrices that are estimated from randomly projected solutions, while in [27] it is reconstructed from the principal components of high-quality population members. These algorithms preserve certain invariance properties like CMA-ES, yet their distributions are not iteratively updated but estimated directly from the solutions. Thus, they are not fully compatible with state-of-the-art cumulation mechanisms developed for the ES family.

This work concerns the covariance matrix reconstruction based ESs due to their desirable invariance properties and the
robustness in handling different kinds of problems. Among them, we distinguish two different schemes for reconstructing the covariance matrix, according to whether all the \( m \) direction vectors are used simultaneously.

- The first scheme, used in the rank-\( m \) ES (Rm-ES) [14], the limited-memory matrix adaptation ES (LM-MA) [13], and the search direction adaptation ES (SDA-ES) [16], is to utilize all direction vectors simultaneously in generating every solution. The solutions obtained definitely obey a multivariate Gaussian distribution determined by the mean vector and the covariance matrix. Thus, these algorithms possess a probability model of high interpretability. The setting \( m \ll n \), however, becomes mandatory rather than optional, since the runtime of the sampling operations scales linearly with \( m \). This deteriorates the performance, because the number of degrees of freedom in the reconstructed covariance matrix cannot exceed \( nm \) and the algorithms fail to capture all variable correlations.

- The second scheme chooses only a subset of \( l \) ones from all \( m \) direction vectors, where \( l \) may vary for different solutions. This scheme facilitates exploring the rich characteristics of the function landscape, as \( m \) can be sufficiently large while the time complexity relies mainly on \( l \) (which is usually set as \( l \ll m \)). The limited-memory CMA-ES (LM-CMA) [12] and the fast CMA-ES (fast-CMA) [15] adopt this scheme and show significant performance improvement. On the other hand, the reconstructed Gaussian distribution is actually conditioned on the specified \( l \) vectors, so the true probability model is not Gaussian. Additionally, the used vectors are usually chosen heuristically and few studies derive an explicit form for the true probability model which guides the search. Thus, we do not obtain new insights into how these variants approximate the original CMA-ES.

Considering the properties of the above two schemes, one may wonder whether it is possible to combine the merits of them. Therefore, the first research question that motivates this study is:

**Q1: How to reconstruct a probability distribution, with a closed form expression, from an arbitrary number of direction vectors while keeping the sampling operations efficient?**

This work proposes a fast mixture sampling method (FMS) as an answer to **Q1**. FMS samples solutions from a Gaussian distribution whose covariance matrix is the regularized arithmetic mean of \( l \) rank-1 matrices randomly constructed from a set of \( m \) direction vectors. The solutions turn out to obey the well-known Gaussian mixture model that has a closed form expression. Then, one may take this model one step further and ask:

**Q2: How can the reconstructed probability distribution approximate the Gaussian distribution of the standard CMA-ES?**

Surprisingly, we find when \( l \to \infty \) the reconstructed probability model converges to a multivariate Gaussian distribution which approximates the one used in the standard CMA-ES. Due to this property, we call the parameter \( l \) “mixing strength” and regard it as the most important parameter in further analysis. However, the infinity assumption on the mixing strength is too strong, so it is natural to ask how accurate can this approximation be when the mixing strength is finite. That is, we investigate:

**Q3: To what extent can the reconstructed probability distribution, with a finite mixing strength, approximate the Gaussian distribution of the standard CMA-ES?**

Our analysis shows that the probability distribution in FMS and a Gaussian distribution which approximates the one in CMA-ES only differ in high order statistical information. More precisely, their difference in the high order statistical moments decreases linearly when the mixing strength increases. This demonstrates the rationality of setting \( l \ll m \) and the ability of FMS to explore a large amount of variable correlations within a limited time.

To demonstrate the effectiveness of FMS, we incorporate it into the standard ES framework and design a mixture model based evolution strategy (MMES) to handle large-scale black-box optimization problems. We also propose a simple rule, called paired test adaptation (PTA), to adjust the mutation strength. The innovations of MMES are as follows:

- MMES reconstructs the probability model from an arbitrarily large number of direction vectors without increasing the time complexity.
- MMES has a nice theoretic property that its probability model converges to an approximation of the Gaussian distribution in the standard CMA-ES and the asymptotic error is inversely proportional to the mixing strength.
- MMES adapts the mutation strength in a way that relies only on the objective function and exhibits the so-called derandomization property.

In the remainder of this paper, we first provide the basic idea of FMS in Section II. The implementation details of MMES are given in Section III. Thereafter, we present the simulation results on two benchmark suites in Section IV. Finally, Section V concludes this paper and gives some remarks for future studies.

II. Fast Mixture Sampling (FMS)

This section describes the FMS method and analyzes how and to what extent can its underlying probability model approximate that of the standard CMA-ES.

A. Target Probability Distribution

The early implementation of CMA-ES [2] adapts the covariance matrix using the so-called rank-1 update:

\[
\begin{align*}
C(0) &= I_n \\
C(g) &= (1 - c_{\text{cov}})C(g) + c_{\text{cov}} p(g) (p(g))^T
\end{align*}
\]  

(1)

where \( C(g) \) is the maintained covariance matrix, \( c_{\text{cov}} \in [0, 1] \) is the learning rate, \( p(g) \in \mathbb{R}^n \) is a random vector referred to as the evolution path, \( I_n \) is the \( n \)-dimensional identity matrix, and the superscript \( g \) denotes the generation index. The recursive form of Eq. (1) requires explicitly storing the full covariance matrix, which eventually becomes the performance bottleneck. By noticing that the evolution paths are smoothed...
with a very small decaying coefficient, one can approximate
the covariance matrix efficiently in a non-recursive manner.
Such an approximation takes the following form:
\[
C_a = (1 - c_a)^m I + c_a \sum_{j=1}^{m} (1 - c_a)^{m-j} q_j q_j^T
\]  
(2)
where the vectors \( q_1, \ldots, q_m \in \mathbb{R}^n \) are designed to approx-
imate the evolution paths and \( c_a \) is a new parameter analogous
to the learning rate. Then, one can define an approximate
Gaussian distribution
\[
\mathcal{P}_a : \mathcal{N}(0_n, C_a)
\]
to perturb the population mean and generate candidate solu-
tions, where \( 0_n \) denotes the \( n \)-dimensional zero vector.
\( \mathcal{P}_a \) is the target probability distribution which FMS would
approximate. It has been used in various papers [14], [15],
[28] and extensive numerical results suggest that using a larger
\( m \) leads to a better performance. However, it takes \( O(nm) \)
time to sample a solution from \( \mathcal{P}_a \) and so one has to keep
\( m \) sufficiently small in large-scale optimization, causing the
dilemma on performance versus efficiency. The goal of this
section is to derive an efficient method that samples solutions
approximately obey \( \mathcal{P}_a \) while allowing \( m \) to be arbitrarily
large.

**B. Working Procedure of FMS**

The proposed FMS method firstly generates a random
vector \( i = (i_1, \ldots, i_l)^T \in \{1, \ldots, m\}^l \) with the probability
distribution
\[
\mathcal{P}_i : \quad p(i_j = k) = \alpha_k, j \in \{1, \ldots, l\}, k \in \{1, \ldots, m\}
\]
where \( \alpha_1, \ldots, \alpha_m \in (0, 1) \) and \( \sum_{k=1}^{m} \alpha_k = 1 \). Then, use
\( i_1, \ldots, i_l \) as indexes to select \( l \) ones from the \( q \) vectors and
construct the following symmetric matrix
\[
\Sigma_i = (1 - \gamma)I_n + \frac{\gamma}{l} \sum_{j=1}^{l} q_{i_j} q_{i_j}^T
\]  
(3)
where \( \gamma \in (0, 1) \) is a regularization parameter for ensuring
positive definiteness. Finally, define a multivariate Gaussian
distribution with \( \Sigma_i \) being the covariance matrix
\[
\mathcal{P}_{\Sigma_i} : \mathcal{N}(0_n, \Sigma_i)
\]
and use this distribution to sample a mutation vector.

Note that, the distribution \( \mathcal{P}_{\Sigma_i} \) is actually conditioned on the
chosen \( q \) vectors, and thus, the true probability model of FMS
is a mixture of one multivariate isotropic Gaussian distribution
and \( m \) linear Gaussian distributions. For simplicity, denote this
probability model by \( \mathcal{P}_m \):
\[
\mathcal{P}_m : \quad \text{The probability model reconstructed in FMS.}
\]
The following lemma provides a closed form expression for
\( \mathcal{P}_m \).

**Lemma 1.** The distribution \( \mathcal{P}_m \) is uniquely determined by its
moment generating function given by
\[
M(t) = \exp \left( \frac{1}{2} (1 - \gamma) |t|^2 \right) \left( \sum_{j=1}^{m} \alpha_j \exp \left( \frac{\gamma}{2j} (t^T q_j^T)^2 \right) \right)^l.
\]  
(4)

On the one hand, Lemma [1] implies that \( \mathcal{P}_m \) is directly relevant to all \( p \) vectors, independent of the specification of \( i \). On the other hand, FMS takes \( O(ln) \) time per solution, since any mutation vector \( z \) sampled from \( \mathcal{P}_m \) can be written as
\[
z = \sqrt{1 - \gamma} z_0 + \sqrt{\frac{\gamma}{l}} \sum_{j=1}^{l} q_{i_j} z_j,
\]  
(5)
where \( z_0 \) is sampled from \( \mathcal{N}(0_n, I_n) \) and \( z_1, \ldots, z_l \) are independent identically distributed (i.i.d.) random variables
sampled from \( \mathcal{N}(0, 1) \). Thus, the time complexity of FMS is independent of \( m \). By setting \( l \) to a small number, the running
time of the sampling operations can be significantly reduced.
This gives the answer to **Q1**.

**C. Approximation Property of FMS**

We show how \( \mathcal{P}_m \) approximates \( \mathcal{P}_a \). Before further analysis,
we specify the parameters of \( \mathcal{P}_i \) as follows:
\[
\begin{align*}
\gamma &= 1 - (1 - c_a)^m, \\
\alpha_k &= \frac{1}{m} c_a (1 - c_a)^{m-k}, k = 1, \ldots, m.
\end{align*}
\]  
(6)
This setting leads to a nice theoretic property:

**Theorem 1.** \( \mathcal{P}_m \) converges to \( \mathcal{P}_a \) when the mixing strength
approaches infinity, provided that the parameters in Eq. [6]
are used.

Theorem [1] provides an answer to **Q2** by showing that \( \mathcal{P}_a \)
can be exactly recovered from \( \mathcal{P}_m \) if the mixing strength is suf-
ficiently large. This is the approximation property in the ideal
case. The infinite mixing strength assumption is, however, too
strong to be practically useful as it could make FMS even
more expensive than directly sampling from \( \mathcal{P}_a \). Fortunately,
this assumption can be relaxed without significantly affecting
the similarities between \( \mathcal{P}_m \) and \( \mathcal{P}_a \). This will be discussed in
the next subsection.

**D. Approximation Accuracy Analysis**

We answer **Q3** and show, with a finite mixing strength, how
accurate can the approximation be. Since Lemma [1] states that
\( \mathcal{P}_m \) can be completely characterized by its moment generating
function, we perform the analysis by comparing the moments
of \( \mathcal{P}_m \) and \( \mathcal{P}_a \).

Firstly, it is easy to conclude from the symmetry property
that all odd order moments of \( \mathcal{P}_m \) equal to 0, being coincident
with \( \mathcal{P}_a \).

Moreover, the second order moments of \( \mathcal{P}_m \) and \( \mathcal{P}_a \) are closely related:
Theorem 2. \( P_m \) and \( P_n \) have the same covariance matrix given by Eq. (2), provided that the parameters in Eq. (6) are used.

Thus, \( P_m \) approximates \( P_o \) in the sense that they have the identical second order moment. The special instance of Theorem 2 when \( l = 1 \) coincides with a recent study [29]. However, Theorem 2 further suggests that such a property holds for any mixture strength, and so, it is a more generic result.

The above approximation properties hold for any mixing strength, so we can infer that the mixing strength only affects the higher moments of even order. The following theorem gives a rigorous statement.

Theorem 3. Provided that the parameters in Eq. (6) are used. The difference of \( P_m \) and \( P_o \) in the \( k \)-th order moments is on the order of \( O(1/l) \), where \( k = 4, 6, 8, \cdots \).

Theorem 3 is critical in practice. It states that there is no need to choose a large mixing strength since the approximation accuracy, measured by the difference in high order statistical moments, increases at a linear rate when increasing the mixing strength. Therefore, it is reasonable to choose a small mixing strength to reduce the runtime without significantly deteriorating the performance. The numerical studies show that the setting \( l = 4 \) works very well in various scenarios.

The following theorem facilitates an intuitive understanding of the characteristics of \( P_m \).

Theorem 4. The projected distribution of \( P_m \) onto any one-dimensional subspace has a non-negative excess kurtosis.

Theorem 4 does not rely on the parameters of \( P_o \); but when the setting in Eq. (6) is used, it implies that \( P_m \) has tails fatter than \( P_o \) and is more likely to produce outliers. In this sense, \( P_m \) is analogous to the Cauchy distribution or the t-distribution which have been studied in the evolutionary algorithm literatures [26], [30]. Nevertheless, this would not mean \( P_m \) belongs to the class of heavy-tailed distributions as we can infer from Lemma 1 that its tails decay exponentially fast.

E. Empirical Verification

We conduct numerical simulations to verify the above theoretical analyses regarding the approximation accuracy. In the simulations, we firstly create \( N_s = 10000 \) samples from \( P_m \) and then compare their empirical distribution to the target distribution \( P_o \). The sample set is denoted by \( \{y_1, \cdots, y_m\}, y_i \in \mathbb{R}^n \}. Parameters of the distributions are set as \( m = 200, n = 1000, \) and \( c_o = 0.1. \) The \( q \) vectors are constructed as \( q_i = (q_{i,1}, \cdots, q_{i,n})^T \) where \( q_{i,j} = 10^3 \times (i-1)/m \) and \( q_{i,j} = 0 \) for all \( j \neq i \) \((i \in \{1, \cdots, m\}, j \in \{1, \cdots, n\}\). This setting makes sure that both \( P_m \) and \( P_o \) can be decomposed into \( n \) univariate ones, which facilitates the comparison and visualization of the high order statistical moments.

We propose two metrics to qualify how \( P_m \) differs from \( P_o \). The first metric is the normalized variance (NV), calculated as

\[
NV = \frac{1}{n \cdot N_s} \sum_{i=1}^{N_s} \| \sqrt{C_o} y_i \|^2. \tag{7}
\]

According to Theorem 2, using the matrix \( C_o \) to normalize the samples will lead to a unit diagonal covariance. Consequently, the NV metric given in Eq. (7) must be close to 1, regardless of the mixing strength.

Our second metric is the standardized moment error (SME). It measures the difference between \( P_m \) and \( P_o \), in terms of higher order moments. For the \( k \)-th order moment, the metric (denoted by \( SME_k \)) is given by

\[
SME_k = \frac{1}{n \cdot N_s} \sum_{i=1}^{N_s} \sum_{j=1}^{n} \frac{y_{i,j}^k}{\delta_j^k} - \tau_k
\]

where \( \tau_k \) denotes the \( k \)-th order moment of the standard normal distribution, \( y_{i,j} \) denotes the \( j \)-th element of \( y_i \), and \( \delta_j \) is the standard variance of the samples along the \( j \)-th axis direction. Small absolute values of \( SME_k \) indicate that \( P_m \) is close to \( P_o \) in terms of the \( k \)-th order moment. According to Theorems 1 and 3, \( |SME_k| \) should decrease to 0 at a rate of order \( O(1/l) \).

The simulations are performed with different mixing strengths chosen from \( \{2, 4, 6, \cdots, 32\} \). The corresponding results are plotted in Fig. 1. From Fig. 1a, it is found that all empirical results (depicted by red circles) are very close to the theoretical value (depicted by the blue line). Thus, \( P_m \) and \( P_o \) has similar covariance matrix, regardless of the mixing strength. This conclusion coincides with Theorem 2.

Regarding the SME metric, we focus on the cases \( k = 4 \) and \( k = 6 \). The results are respectively shown in Figs. 1b and 1c. In order to capture the asymptotic characteristics with different mixing strengths, we fit a rational function (shown by the blue line) using least squares regression. In Fig. 1b, the regression curve \( SME_4 = \frac{69.66}{1+0.04 \cdot l} \) perfectly fits all empirical results, implying that the asymptotic error given in Theorem 3 is quite accurate in the 4-th order case. In Fig. 1c, the curve \( SME_6 = \frac{78.64}{1+0.002 \cdot l} \) slightly overestimates the error for \( l \geq 8 \); but on the whole, it still exhibits the \( O(1/l) \) decreasing rate, which coincides with Theorem 3. In addition, both Figs. 1b and 1c show that the SME metric decreases rapidly when the mixing strength is relative small (say \( l < 10 \)) but will approach 0 extremely slow. This supports our claim that the mixing strength needs not to be very large.

III. THE PROPOSED ALGORITHM: MMES

We develop the MMES algorithm by incorporating FMS into a generic \((\mu, \lambda)\)-ES framework. MMES considers an unconstrained function \( f : \mathbb{R}^{n} \rightarrow \mathbb{R} \) to be minimized by sampling in the \( g \)-th generation a population of \( \lambda \) candidate solutions as

\[
x_i^{(g)} = m_i^{(g)} + \sigma^{(g)} z_i^{(g)}, i \in \{1, \cdots, \lambda\} \tag{9}
\]

where \( m_i^{(g)} \) is the mean vector, \( \sigma^{(g)} \) is the mutation strength, and \( z_i^{(g)} \) is a mutation vector sampled from \( P_m \). After evaluating the population, the best \( \mu \) ones are chosen to update
An innovation of PTA is that it permits the preferences to high distribution and only relies on the objective function values. In this section an alternative to CSA, called paired test adaptation (PTA), which makes no assumption about the probability distribution and only relies on the objective function values. An innovation of PTA is that it permits the preferences to high quality solutions and preserves the derandomization property of CSA.

PTA is an improved version of the generalized 1/5-th success rule (GSR) in our previous work [16]. Similar to other success based methods, it adopts a simple rule: increasing the mutation strength if the optimization process is significantly successful, or decreasing it otherwise. The success is measured by the change of the objective function values evaluated in consecutive generations. The major distinction is that PTA uses a weighting scheme to put more importance to top ranked solutions. Specifically, it computes a scalar $L^{(g)}$ at generation $g$ as:

$$L^{(g)} = \sum_{i=1}^{\mu} \omega_i \mathbb{I}[f(x_i^{(g-1)}) > f(x_i^{(g)})]$$  \hspace{1cm} (10)$$

where $\mathbb{I}[\cdot]$ is the indicator function, $x_i^{(g)}$ denotes the $i$-th best solution in the $g$-th generation, and $\omega_i$ is the weights, defined in the standard ES framework, satisfying $1 > \omega_1 > \cdots > \omega_\mu > 0$ and $\sum_{i=1}^{\mu} \omega_i = 1$. The scalar $L^{(g)}$ calculates the weighted percentage of solution pairs that gain improvement of the objective values, and thus, serves as a success metric.

To explicitly measure the success, we show that $L^{(g)}$ obeys a well-defined probability distribution under random selection, so the significance of success can be calculated from a properly designed statistical test. It is easy to see that $\mu L^{(g)}$ obeys a binomial distribution if setting $\omega_1 = \cdots = \omega_\mu$. However, the case would be nontrivial when the weighting scheme is used. Fortunately, we find that with the default weight settings the standard CMA-ES, $L^{(g)}$ converges in distribution to a Gaussian, as the population size approaches infinity.

**Proposition 1.** Set $\omega_i = \omega'_i / \sum_{j=1}^{\mu} \omega'_j$, $(i = 1, \cdots, \mu)$, where $\omega'_i = \ln(\mu + 0.5) - \ln(i)$. Assume that the solutions are randomly sorted. Then, as $\mu \to \infty$,

$$L^{(g)} \overset{d}{\to} \mathcal{N} \left( \frac{1}{2} \left[ 1 - 2 \sum_{i=1}^{\mu} \omega_i^2 \right] \right).$$  \hspace{1cm} (11)$$

Inspired by this property, we design an exponential smoothing rule to improve the robustness of $L^{(g)}$ as

$$W^{(g)} = (1 - c_\sigma)W^{(g-1)} + \sqrt{c_\sigma (2 - c_\sigma)} \frac{2L^{(g)} - 1}{\sqrt{\sum_{i=1}^{\mu} \omega_i^2}}$$  \hspace{1cm} (12)$$

where $c_\sigma \in (0, 1)$ is a learning rate to average $L^{(g)}$ over generations. The second term on the right hand side is designed for normalization since we have $\frac{2L^{(g)} - 1}{\sqrt{\sum_{i=1}^{\mu} \omega_i^2}}$ approximately obey $\mathcal{N}(0, 1)$. Therefore, the smoother $W^{(g)}$ can be considered as an accumulated and normalized version of the success metric. The update rule in Eq. (12) exhibits the derandomization property when Proposition 1 holds. On the one hand, the update is stable in the sense that if $W^{(g-1)} \sim \mathcal{N}(0, 1)$ we also have $W^{(g)} \sim \mathcal{N}(0, 1)$. On the other hand, the distribution of $W^{(g)}$ is independent of other parameters to be adapted. These coincide with the derandomization properties of CSA.

The metric $W^{(g)}$ characterizes the success of the optimization process, and in particular, $W^{(g)} > 0$ means that

### Fig. 1. Influence of mixing strength on the approximation accuracy.

- **(a)** Averaged variance after normalization
- **(b)** Averaged approximation error of the 4-th order standardized moment
- **(c)** Averaged approximation error of the 6-th order standardized moment

$m_l^{(g)}$, $\sigma_l^{(g)}$, and $P_m$. In the update, the $(\mu, \lambda)$-ES framework defines a set of weights and uses a weighting scheme to favor top ranked solutions.

**A. Paired Test based Mutation Strength Adaptation**

Mutation strength adaptation enables ESs to achieve excellent convergence speed and reduce the adaptation time. The cumulative step-size control (CSA) in the standard CMA-ES is not applicable in MMES since it works in an implicit coordinate system defined by the covariance matrix. We present in this section an alternative to CSA, called paired test adaptation (PTA), which makes no assumption about the probability distribution and only relies on the objective function values. An innovation of PTA is that it permits the preferences to high
the algorithm generates more improving pairs than expected. Considering that it approximately obeys \( \mathcal{N}(0,1) \) under random selection, we can verify whether the generation \( g \) is successful with a simple \( z \)-test and use the result to update the mutation strength. This idea is inspired by GSR and is implemented as

\[
\sigma^{(g+1)} = \sigma^{(g)} \exp \left( \frac{1}{d_g} \left( \Phi(W^{(g)}) - 1 + \alpha_z \right) \right)
\]

where \( \Phi(\cdot) \) is the cumulative probability function of \( \mathcal{N}(0,1) \), \( \alpha_z \) is the significance level of the \( z \)-test, and \( d_g \) is a damping factor. When the success probability (i.e., \( \Phi(W^{(g)}) \)) is smaller than \( 1 - \alpha_z \), PTA decreases the mutation strength to encourage local search, which in turn helps to increase the success probability. On the contrary, when the success probability is greater than \( 1 - \alpha_z \), PTA increases the mutation strength to enhance the global exploration ability, thereby causing a reduction in the success probability. By iteratively applying the above procedures, PTA keeps the significance of success at a \( \alpha_z \) level.

\[B. \ Direction \ Vector \ Adaptation \ and \ Selection\]

The direction vectors in reconstructing the probability model are designed to increase the likelihood of generating promising solutions. As the probability model reconstructed in FMS approximates the one in the standard CMA-ES, various well-established methods [12], [14] can be applied without any modification. In this work, we adopt the one proposed in [12]. This method maintains a set of direction vectors \( q_1, \cdots, q_m \in \mathbb{R}^n \), a set of timestamps \( t_1, \cdots, t_m \in \mathbb{Z}^+ \), and a set of logical indexes \( v_1, \cdots, v_m \in \{1, \cdots, m\} \) in a way that \( q_{v_i} \) stores the evolution path generated in the \( t_{v_i} \)-th generation. The adaptation starts by identifying a logic index \( k^* \) given by

\[
k^* = \arg \min_{k \in \{2, \cdots, m\}} t_{v_k} - t_{v_{k-1}}.
\]

The newly obtained evolution path \( k^* \) (denoted by \( p \)) then replaces \( q_{v_k} \) if \( t_{v_k} - t_{v_{k-1}} \) is no larger than a threshold \( T \in \mathbb{Z}^+ \), or replaces \( q_{v_k} \) otherwise. This preserves a certain distance, in terms of number of generations, between consecutive evolution paths to prevent the probability model from degeneracy.

When selecting random direction vectors, MMES generates the corresponding indexes in two steps. The first step is to produce a logic index \( k \) from the distribution \( \mathcal{P}_k \). With the parameters in Eq. (5), it is equivalent to drawing the integer \( m - k \) from a geometric distribution with the success probability \( c_a \), conditioned on the range \([0, m - 1]\). Then, the second step is to transform this index to the physical one. Combining these two steps together, a random index for selecting the \( q \) vectors can be calculated as

\[v_m - j \% m \]

where \( \% \) is the modulo operation and \( j \) is a random integer sampled from a geometric distribution with the success probability \( c_a \) (denoted by \( \mathcal{G}(c_a) \)). This procedure puts more importance to more recent evolution paths as they are logically stored with higher indexes.

\[C. \ Detailed \ Implementation\]

The pseudo-code of MMES is given in Algorithm 1. In the initialization (Lines 2-6), all direction vectors are set to \( \mathbf{0}_n \) such that the initial probability model is an isotropic Gaussian. In the main loop, the FMS procedure is called in Lines 8-14 to generate the population. For each solution, it samples one \( n \)-dimensional isotropic Gaussian vector \( z_0 \) and \( l \) isotropic Gaussian scalars \( z_1, \cdots, z_l \) in Lines 9 and 10. Then, \( l \) indexes are generated in Lines 11 and 12 by sampling the geometric distribution \( \mathcal{G}(c_a) \), followed by the transformation described Section III-B. Using these indexes, Line 13 chooses \( l \) direction vectors to construct the mutation vector, according to Eq. (5), with \( z_0 \) and \( z_1, \cdots, z_l \) being the mixture components. Finally, in Line 14, the mutation vector is rescaled and imposed on the population mean to get a candidate solution.

Lines 16-18 are from the standard CMA-ES. All solutions are sorted according to their objective function values in Line 16. The new mean is recomputed from the best \( \mu \) solutions in Line 17, with a rank-based weighting strategy to favor high quality solutions. Line 18 updates the evolution path by cumulating the mutation step of the population mean, where \( c_a \) is the learning rate.

Lines 19-26 adapt the direction vectors as well as their indexes and timestamps according to [12]. It chooses the \( k^* \)-th vector having the minimal distance to the previous one (Line 19), removes it from the vector record (Lines 23 and 24), and then appends the evolution path to the record (Lines 25 and 26). The index \( k^* \) is reset to 1 to drop the earliest information whenever no pair of consecutive vectors has a distance smaller than \( T \) (Lines 20-22).

In the end of the loop, Lines 27-29 implement the PTA rule. The implementation is the same as described in Section III-A except that the term \( 1/\sqrt{\sum_{i=1}^{\mu} \omega_i^2} \) in Eq. (12) is replaced by \( \mu_{eff} \), by convention of the standard CMA-ES.

D. Parameters

All parameters of MMES are summarized in Table I. \( \lambda, \mu, \omega_1, \cdots, \omega_\mu, \mu_{eff}, \) and \( c_a \) are from the standard CMA-ES, so they are set as suggested in [31]. Other parameters are discussed as below:

- \( c_a \) is the success probability of the geometric distribution in choosing the direction vectors. Theorem 2 states that it serves as a learning rate for the covariance matrix of the reconstructed probability model. Since a standard ES requires \( O(n) \) expected time for a fixed relative improvement on a spherical function [32], setting \( c_a \geq O(1/n) \) seems to be necessary to obtain the same convergence speed for MMES on a convex quadratic function. Therefore, we use the setting \( c_a = 4/n \).

- \( T \) is the minimal distance between the consecutive evolution paths maintained by MMES. As suggested in [14], we set \( T = \lceil 1/c_a \rceil \) which tends to keep the evolution paths uncorrelated.

\[\text{The source code is publically available at https://github.com/hxyokokok/}\]

[1]
Algorithm 1 MMES

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

1: $g = 0$
2: $p^{(0)} = 0$
3: $W^{(0)} = 0$
4: $q_i^{(0)} = 0$, for $i = 1, \ldots, m$
5: $t_i = 0$, for $i = 1, \ldots, m$
6: $v_i = i$, for $i = 1, \ldots, m$
7: while the termination criterion is not met do
8: \hspace{1em} for $i = 1$ to $\lambda$ do
9: \hspace{2em} $z_i \sim N(0, f_i)$
10: \hspace{2em} $z_1, \ldots, z_i \sim N(0, 1)$
11: \hspace{2em} $j_1, \ldots, j_t \sim \mathcal{U}(c_d)$
12: \hspace{2em} $j_k = v_{n+t_j} g_{(0)}$ for $k = 1, \ldots, m$
13: \hspace{2em} $z = \sqrt{1 - \gamma} z_0 + \sqrt{\gamma} \sum_{k=1}^t z_k q_{jk}$
14: \hspace{2em} $x^{(g)} = m^{(g)} + \sigma^{(g)} z$
15: \hspace{1em} end for
16: Sort the solutions such that $f(x^{(g)}(1)) \leq \cdots \leq f(x^{(g)}(n))$
17: $p^{(g+1)} = \sum_{j=1}^n \omega_j x^{(g)}(j)$
18: $p^{(g+1)} = (1-c_h)p^{(g)} + \sqrt{c_h(2-c_h)} \mu_{eff} W^{(g+1)}/\sigma^{(g)}$
19: $k^* = \arg \min t_{yk} - t_{yk-1}$
20: if $t_{yk} - t_{yk-1} \geq T$ then
21: \hspace{1em} $k^* = 1$
22: end if
23: $v_k = v_{k+1}$ for $k = k^*, \ldots, m - 1$
24: $v_m = v_k$
25: $t_{vm} = g + 1$
26: $q_{vm} = p^{(g+1)}$
27: $L = \sum_{j=1}^n \omega_j [f(x^{(g)}(j)) > f(x^{(g)}(j))]$
28: $W^{(g+1)} = (1-c_h)W^{(g)} + \sqrt{c_h(2-c_h)} \mu_{eff} (2L - 1)$
29: $\sigma^{(g+1)} = \sigma^{(g)} \exp \left( \frac{1}{\tau x} f(W^{(g)} - 1 + \alpha_s) \right)$
30: $g = g + 1$
31: end while
32: return

- $\gamma$ is the regularization coefficient in building the conditional Gaussian distribution $\mathcal{P}_{x_i}$. We choose the setting $\gamma = 1 - (1 - c_h) m$ since it leads to the theoretical properties discussed in Section II-C.
- $m$ is the number of candidate direction vectors. It constrains the number of variable correlations that can be learned. In this work, we set $m = 2 \sqrt{n}$. This is relatively larger than the usual settings in many CMA-ES variants, but is still affordable in most scenarios where evolutionary algorithms are applied.
- $l$ is the mixing strength affecting the approximation accuracy of the probability model. As analyzed in Section II-D, a small value is enough to attain considerable accuracy. Thus, we recommend setting $l$ to $O(1)$. In this work, we choose $l = 4$ and the numerical experiments show that it works well in various optimization tasks.
- $c_h$ is the decay factor of the exponential smoothing for $W^{(g)}$ and $d_\sigma$ is the corresponding damping constant. They both control the changing rate of the mutation strength. As the distribution of $W^{(g)}$ is independent of the other parameters (see Section III-A), it is reasonable to set both $c_h$ and $d_\sigma$ to $O(1)$. We choose the setting $c_h = 0.3, d_\sigma = 1$ from our previous study [16] in which the same smoothing rule is adopted.
- $\alpha_s$ is the target significance level in PTA. It is essentially a parameter of the $z$-test rather than of MMES itself. We set $\alpha_s = 0.05$, the most popular setting in statistical tests.

### E. Complexity

MMES stores $\lambda$ solutions and $m$ direction vectors and so its space complexity is $O(n^{1.5})$. The most time-consuming steps of MMES are in the FMS procedure. Precisely, it requires $O(n)$ operations for sampling the multivariate isotropic Gaussian vector in Line 9, $O(n^2)$ operations for constructing the mutation vector in Line 13, and $O(n)$ operations for calculating the candidate solution in Line 14. All other lines can be performed in $O(n)$ time. Thus, when adopting the parameters in Table I, the time complexity of MMES is $O(n^2)$ per solution.

### IV. COMPARATIVE STUDIES

We investigate the performance of MMES and other eleven large-scale evolutionary algorithms on two benchmark sets.

#### A. Experimental Settings

1) Test Problems: Two sets of test problems are selected in the numerical study. Their objective functions and brief descriptions are summarized in Table I.

a) Set 1: Basic Test Problems: The first test set is intended to test the algorithms’ rotational invariance and scalability, the most favorable properties in designing new ESs. It contains 11 basic test problems, all of which have the global minimum 0. $f_{\text{Sphere}}$ is the simplest and serves as a base in performance analysis. $f_{\text{Ellips}}, f_{\text{Rosen}}, f_{\text{Discus}}, f_{\text{Cigar}},$ and $f_{\text{DiffPow}}$ are separable or only have very weak variable correlations. Based on them, another five fully non-separable problems, namely $f_{\text{RotEllips}}, f_{\text{RotRosen}}, f_{\text{RotDiscus}}, f_{\text{RotCigar}},$ and $f_{\text{RotDiffPow}}$, are constructed respectively by imposing a rotational transformation on the decision space. These problems differ in the landscape characteristics, and hence, mainly challenge the ESs’ adaptation ability in different optimization tasks. For example, the Rosenbrock problem and the Cigar problem have a low-rank structured landscape in the sense that there exists a significant eigengap in the Hessian such that the landscape shape can be well approximated by only a few direction vectors. Thus, ESs based on the probability model reconstruction techniques are expected to solve these problems easily, provided that the used direction vectors are appropriately adapted. On the contrary, the Ellipsoid, the Discus, and the Different Powers problems do not have this property, and they are likely to cause considerable difficulties to the ESs chosen in the comparative study.

| Parameters for MMES |
|----------------------|
| $\lambda = 4 + [3 \ln n]$, $\mu = [\frac{1}{2}]$, $m = 2 \sqrt{n}$, $c_h = \frac{4}{\sqrt{n}}$, $c_c = \frac{0.4}{\sqrt{n}}$, $\omega_1 = \frac{\ln(\mu + 0.5) - \ln(1)}{\mu} - \frac{1}{\sum_{i=1}^n \mu}$, $c_e = 0.3$, $d_a = 1$, $\alpha_s = 0.05$. |

- In this study, we set $\alpha_s = 0.05$, which is the most popular setting in statistical tests.
correlations between groups or in unchosen groups. For variables are uniformly divided into 20 groups, some of which non-separable. In the partially non-separable problems, the explicitly controlled. Specifically, scale global optimization (LSGO). It contains 20 test problems the benchmark suite for the CEC’2010 competition on large-
definitions and properties, please refer to [33].

| Mean vector is uniformly sampled in the range |
|-----------------------------------------------|
| Set 1: Basic Test Problems                      |
| **Name** | **Function** |
| Sphere  | $f_{\text{Sphere}}(x) = \sum_{i=1}^{n} x_i^2$ |
| Ellipsoid| $f_{\text{Ellipsoid}}(x) = \sum_{i=1}^{n} \frac{1}{10^8} x_i^2$ |
| Rosenbrock| $f_{\text{Rosenbrock}}(x) = \sum_{i=2}^{n} (100(x_i^2 - x_{i-1})^2 + (x_{i-1} - 1)^2)$ |
| Discus  | $f_{\text{Discus}}(x) = 10^{-6} x_1^2 + \sum_{i=2}^{n} x_i^2$ |
| Cigar   | $f_{\text{Cigar}}(x) = x_1^2 + 10^6 \sum_{i=2}^{n} x_i^2$ |
| Different Powers | $f_{\text{Different Powers}}(x) = \sum_{i=1}^{n} |x_i|^{1+10^{-8}}$ |
| Rotated Ellipsoid | $f_{\text{Rotated Ellipsoid}}(x) = f_{\text{Sphere}}(Rx)$ |
| Rotated Rosenbrock | $f_{\text{Rotated Rosenbrock}}(x) = f_{\text{Rosenbrock}}(Rx)$ |
| Rotated Discus | $f_{\text{Rotated Discus}}(x) = f_{\text{Discus}}(Rx)$ |
| Rotated Cigar | $f_{\text{Rotated Cigar}}(x) = f_{\text{Cigar}}(Rx)$ |
| Rotated Different Powers | $f_{\text{Rotated Different Powers}}(x) = f_{\text{Different Powers}}(Rx)$ |

Set 2: CEC’2010 LSGO Problems

| **Function** | **Description** |
|----------------|-----------------|
| $f_1$ to $f_3$ | Fully separable |
| $f_4$ to $f_8$ | Non-separable in a single group |
| $f_9$ to $f_{13}$ | Non-separable in 10 groups |
| $f_{14}$ to $f_{18}$ | Non-separable in all 20 groups |
| $f_{19}$ to $f_{20}$ | Fully non-separable |

1 $R$ is an orthogonal matrix generated by applying the Gram-Schmidt procedure on a random matrix with standard normally distributed entries. 2 The condition number of $f_{\text{Ellipsoid}}$ is customizable by tuning the parameter $\alpha$. Unless stated otherwise, we choose $\alpha = 6$ to render the problem landscape ill-conditioned.

b) Set 2: CEC’2010 LSGO Problems: The second set is the benchmark suite for the CEC’2010 competition on large-scale global optimization (LSGO). It contains 20 test problems and covers a variety of difficulties such as multimodality, non-separability, and boundary constraints. We choose this set to verify the overall performance of the algorithms. A distinct feature of this test set is that its non-separability can be explicitly controlled. Specifically, $f_1$ to $f_3$ are fully separable, $f_{19}$ to $f_{20}$ are fully non-separable, and the rest are partially non-separable. In the partially non-separable problems, the variables are uniformly divided into 20 groups, some of which are chosen and then made non-separable, such that the variable correlations only exist in certain groups and there are no correlations between groups or in unchosen groups. For $f_1$ to $f_3$, $f_4$ to $f_8$, and $f_{14}$ to $f_{18}$, the numbers of non-separable groups are set to 1, 10, and 20, respectively. For their detailed definitions and properties, please refer to [33].

2) Algorithms for Comparison: CMA-ES [3] and its five large-scale variants, namely sep-CMA [11], LM-CMA [12], LM-MA [13], Rm-ES [14], SDA-ES [15] are used in the first test set. In all these algorithms, as well as in MMES, the initial mean vector is uniformly sampled in the range $[-5, 5]^n$ and the initial mutation strength is set to 3. We benchmark these algorithms with their default parameter settings.

Another five large-scale evolutionary algorithms that are not based on the ES framework are used in the second test set. They include DECC-G [19], MA-SW [34], MOS [35], CPSSO2 [36], and DECC-DG [37]. DECC-G, DECC-DG, and CPSSO2 are CC-based algorithms and are chosen to show the advantages and disadvantages of the ES framework over the CC framework. MA-SW and MOS are state-of-the-art memetic algorithms. The former is the winner of the CEC’2010 LSGO competition while the latter wins all the CEC competitions held during 2013-2017. For the above five competitors, the results are directly from [14], [16], [38], [39], measured in the standard settings for the CEC competitions.

3) Performance Metrics: The experiments on the first test set mainly concern the convergence ability of the algorithms. Thus, the number of function evaluations required to converge (denoted by $FEs$) is utilized as the performance indicator. We consider an algorithm as converged if it finds an objective function value smaller than $10^{-8}$ before $FEs$ reaches a pre-defined threshold $maxFEs$. For those that fail to converge, their $FEs$ values are directly set to $maxFEs$. $maxFEs$ is set to $10^8$ for $n = 1000$ and $2 \times 10^8$ for $n > 1000$. All algorithms are independently run 20 times and the median results are reported.

For the experiments on the second test set, the performance of algorithms are measured by the best objective function values found within a computational budget of $3 \times 10^6$ function evaluations. MMES is independently run 25 times, as recommended for the CEC competition.

On each test instance, we use the Wilcoxon rank sum test [40] to verify whether the results of MMES and those of the others are significantly different. For MA-SW, MOS, CPSSO, and DECC-DG where only the median results are available, the Wilcoxon signed rank test is used instead. To have an overall view of the performance in a certain set of test instances, we calculate for each algorithm the rank averaged over all test instances, according to the Friedman test [41]. The significance of difference between MMES and the other competitors are reported, with a correction by the Bonferroni procedure [42] to eliminate the family-wise error.

B. Effectiveness of PTA

We verify the potential of PTA as an alternative to CSA and other methods for mutation strategy adaptation. To this end, we compare PTA with the CSA from LM-MA, the population success rule (PSR) from LM-CMA, the rank success rule (R5) from Rm-ES, and the GSR from SDA-ES. For a fair comparison, these methods are extracted from the corresponding algorithms and incorporated into the standard ES.

We first choose 1000-dimensional $f_{\text{Sphere}}$ as the test problem. Since the function landscape can be perfectly fitted by the density contours of the isotropic Gaussian model, this test is to reveal the behavior of PTA in isolation from the covariance matrix adaptation or the sampling procedure. Fig. [24] presents the Euclidean distance between the mean and the global optimum, calculated as $|m|$, versus the number of function evaluations. We can observe from this semi-log plot that PTA enables ES to attain the linear convergence speed, as its trajectory is rendered by a descending line. This is consistent with the theoretical results about classic methods such as the 1/5-th success rule [43] and the CSA without cumulation [44]. Other comparative methods also exhibit linear convergence and no significant difference is observed in terms of the convergence rate.

Compared with the simplest Sphere problem, well-conditioned Ellipsoid problem is more interesting because...
it simulates a real scenario where the optimal covariance matrix cannot be learned. We thus consider 1000-dimensional $f_{\text{Elli}}$, with the conditioning parameter $\alpha$ varying in $\{1, 2, 3\}$. The results are presented respectively in Figs. 26 to 28. It is seen that all algorithms converge linearly except in the early stage. GSR exhibits the slowest convergence rate and is significantly surpassed by PTA, which is exactly opposite to the results on $f_{\text{Sphere}}$. The performance degradation of GSR is probably due to that the solutions give less information than on $f_{\text{Sphere}}$ about how success the progress is made, and hence, GSR tends to accept small mutation strengths leading to a slow adaptation rate. PTA seems to alleviate this problem by using a weighting scheme to favor top ranked solutions. The above tests demonstrate that PTA can serve as an alternative to existing methods for mutation strength adaptation and its weighting scheme may also save computational effort on problems with small or moderate condition numbers.

C. Rotational Invariance

Rotational invariance refers to the property of an algorithm that its performance does not change after rotating the decision space, provided that the algorithm is properly initialized. An algorithm possessing this property is robust to the non-separability of the problem, as the variable correlations can be linearly approximated by the rotational transformations on the decision space. For the standard CMA-ES, the rotational invariance is built-in. It explicitly maintains all pairwise linear correlations in a covariance matrix such that the rotations can be completely captured. On the contrary, sep-CMA does not possess this property since it does not try to explore the variable correlations at all. Other comparative algorithms including MMES use only a set of direction vectors to reconstruct the probability model and therefore cannot learn all variable correlations. Thus, in this subsection, we verify whether they are invariant against rotations with numeric simulations.

Fig. 5 shows the convergence behaviors of each algorithm on the 1000-dimensional unrotated basic test problems and on the corresponding rotated versions. It is evident that sep-CMA is sensitive to the rotations, as it performs well on most unrotated problems but fails to converge in all rotated problems. Such a performance deterioration is not seen for MMES or other modern CMA-ES variants, demonstrating that they are rotationally invariant. In fact, for MMES, the rotational invariance is guaranteed by design: both of its two key components, FMS and PTA, contain purely linear operations (see Lines 8-15 and 27-29 in Algorithm 1) and are independent of certain Euclidean coordinates which may be changed by the rotations. The change in performance (if exists) is mainly caused by the non-invariant initialization and seems to be negligible according to the experiments.

Table III presents the ranking results for the algorithms that are invariant against rotations. The standard CMA-ES are surpassed by its variants, probably due to the large number of strategy parameters required to be adapted. Applying more advanced adaptation and sampling schemes (e.g., [45], [46]) or carefully tuning the hyperparameters (e.g., [47], [48]) is likely to address this issue. MMES performs consistently well on all test problems and ranks first according to the Friedman test. The win/loss/tie record of the Wilcoxon test also shows it is competitive with LM-CMA and superior to the others on the majority of the problems. However, the statistical difference between MMES and the four variants are insignificant as reported by the Bonferroni post hoc procedure. Thus, no algorithm has an absolute advantage over the others. More detailed numerical results are found in Table S-I in the supplement.

### Table III

| CMA-ES | LM-MA | LM-CMA | Rm-ES | SDA-ES | MMES |
|--------|-------|--------|-------|--------|------|
| $f_{\text{Elli}}$ | 6 | 5 | 4 | 3 | 2 |
| $f_{\text{Rosen}}$ | 6 | 5 | 4 | 3 | 2 |
| $f_{\text{Hess}}$ | 6 | 5 | 4 | 3 | 2 |
| $f_{\text{Cigar}}$ | 6 | 5 | 4 | 3 | 2 |
| $f_{\text{Discus}}$ | 6 | 5 | 4 | 3 | 2 |
| $f_{\text{Elli}}$ | 6 | 5 | 4 | 3 | 2 |
| $f_{\text{Rosen}}$ | 6 | 5 | 4 | 3 | 2 |
| $f_{\text{Hess}}$ | 6 | 5 | 4 | 3 | 2 |
| $f_{\text{Cigar}}$ | 6 | 5 | 4 | 3 | 2 |

• / ◦ / † 10 / 0 / 0 6 / 3 / 1 4 / 4 / 2 6 / 2 / 2 8 / 2 / 0

Avg Rank 5.8 3.8 2.7 3.3 3.1 2.3

p-Value 0.00 1 1 1 1

1 * indicates that MMES significantly outperforms the peer algorithm at a 0.05 significance level by the Wilcoxon test, whereas ◦ indicates the opposite. If no significant difference is detected, it will be marked by the symbol “†”.
2 “Avg Rank” denotes the ranking results averaged over all problems according to the Friedman test.
3 p-Value denotes the significance of difference between the averaged ranks of MMES and the pair algorithms, corrected by the Bonferroni procedure.
D. Scalability

We investigate the scalability of MMES on the unrotated basic test problems with \( n = 2500, 5000, 7500, \) and 1000. The number of function evaluations required to reach the accuracy \( 10^{-8} \) are plotted in Fig. 4.

On the Ellipsoid problem \( f_{\text{Elli}} \), MMES outperforms all except LM-CMA on the 2500- and 5000-dimensional instances and is the best performer on the higher-dimensional instances. This problem is characterized by a Hessian with uniformly spread eigenvalues in the log scale, and hence, its landscape cannot be well learned unless there are sufficiently many direction vectors for probability model reconstruction. The good performance of MMES demonstrates its advantage in exploiting the rich correlations with no extra time cost. All the other algorithms scale linearly with a constant factor varying with different methods for probability model adaptation; the linear scaling is because of the fixed number of direction vectors to be adapted. An interesting observation is that MMES also scales linearly although the number of direction vectors increases sub-linearly with the increasing dimension, possibly because of the non-linear setting for the adaptation rate.

MMES is better than or as good as LM-CMA and LM-MA on \( f_{\text{Rosen}} \). This problem is much more difficult than \( f_{\text{Elli}} \) as the solutions have to pass through a long and narrow valley which cannot be rendered by a single linear transformation [49]. Rm-ES and SDA-ES achieve better performance because of the fewer direction vectors to be adapted in exploiting the landscape structure. However, some comparison results are insignificant because the landscape is multimodal and the algorithms seem to suffer from premature convergence.

On \( f_{\text{Cigar}} \), MMES achieves competitive results compared with the best performer, Rm-ES. This result is different from that observed on \( f_{\text{Rosen}} \), although their landscapes are both low-rank structured. In fact, \( f_{\text{Cigar}} \) is much simpler in that it is convex quadratic and the shape of the landscape can be completely described by only one direction vector. The relatively good performance of MMES suggests that it sustains the capability of capturing the most promising search direction even when the used direction vectors are more than required.

MMES produces surprisingly good results on \( f_{\text{Discus}} \). The spectrum of the Hessian suggests that the optimal covariance matrix for capturing the variable correlations should have \( n - 1 \) identical eigenvalues and a single one that is one million times smaller. Thus, the ESs have to learn all the difference in scalings between the most sensitive direction (i.e., \( x_1 \)) and the other \( n - 1 \) ones, which is, obviously, not the case for the algorithms chosen in the experiments. The good performance of MMES may be contributed to the fatter tails of the probability distribution which leads to a higher probability of generating long jumps on the insensitive directions. A parameter sensitiveness test presented later will verify this statement.

The problem \( f_{\text{DiffPow}} \) is similar to \( f_{\text{Elli}} \) except that the variable correlations are non-linear and vary with position. The closer to the optimum, the more difficult an algorithm gets to approach it further. MMES and LM-CMA both perform well on this problem and demonstrate their local exploitation ability.

The ranking results of all the algorithms are summarized in Table IV. It suggests that MMES ranks firstly according to the Friedman test and significantly surpasses LM-MA, LM-CMA, and Rm-ES. More detailed results are found in Table S-II in
Fig. 4. Median results on the 2500-, 5000-, 7500-, and 10000-dimensional non-rotated problems. The curves present the number of function evaluations required to reach the accuracy $10^{-8}$. The supplement.

E. Runtime

We verify the practical efficiency of MMES in terms of runtime. The runtime of an algorithm refers to its consumed CPU time divided by the number of function evaluations. We carry out the simulation on $f_{\text{Elli}}$. However, to make the results independent of the specification of the test problems, the processing time for function evaluations are excluded from the measurement.

Fig. 5 provides a visual comparison for the runtime results obtained on a PC with a 3.10-GHz Intel Core i5-10500 CPU. It is clear that CMA-ES is the slowest. The regression line fitted by its associated data points, depicted by a red dotted line, suggests that its runtime grows at the order of $O(n^{2.1})$, being consistent with the fact that a full covariance matrix are required to be adapted. sep-CMA, RmES, SDA-ES, and MMES are the fastest on all dimension settings and are about 500 times faster than CMA-ES in the 10000-dimensional case. LM-MA and LM-CMA are slower, but their runtimes only increase by a small constant. The regression line fitted for all these variants, shown by a blue dotted line, states that their asymptotic runtimes can be well approximated by a polynomial of order 1.0. This leads to the conclusion that CMA-ES scales approximately quadratically while MMES and all the others scale approximately linearly. But note that the obtained results can heavily depend on programming skills and operating environments. In general, all the considered CMA-ES variants are time-efficient in large-scale settings.

F. Sensitiveness to Mixing Strength

The mixing strength $l$ is a key parameter in the proposed MMES algorithm. A larger mixing strength leads to a more

| TABLE IV | RANKS OF LARGE-SCALE CMA-ES VARIANTS ON THE 2500-, 5000-, 7500-, AND 10000-DIMENSIONAL UNROTATED PROBLEMS IN TERMS OF THE NUMBER OF FUNCTION EVALUATIONS REQUIRED TO REACH THE ACCURACY $10^{-8}$. BETTER RESULTS ARE HIGHLIGHTED WITH DARKER GRAY BACKGROUND. |
|-----------|-------------|-------------|-------------|-------------|-------------|
|           | $n$         | LM-MA       | LM-CMA      | Rm-ES       | SDA-ES      | MMES        |
| $f_{\text{Elli}}$ | 2500        | 4           | 1           | 5           | 3           | 2           |
|            | 5000        | 4           | 1           | 5           | 3           | 2           |
|            | 7500        | 4           | 2           | 5           | 3           | 1           |
|            | 10000       | 4           | 3           | 5           | 2           | 1           |
| $f_{\text{Rosen}}$ | 2500        | 4           | 5           | 2           | 1           | 3           |
|            | 5000        | 4           | 5           | 2           | 1           | 3           |
|            | 7500        | 2           | 5           | 4           | 3           | 1           |
|            | 10000       | 2           | 5           | 4           | 3           | 1           |
| $f_{\text{Discus}}$ | 2500        | 4           | 5           | 3           | 2           | 1           |
|            | 5000        | 3           | 5           | 4           | 2           | 1           |
|            | 7500        | 3           | 5           | 4           | 2           | 1           |
|            | 10000       | 3           | 5           | 4           | 2           | 1           |
| $f_{\text{Cigar}}$ | 2500        | 4           | 1           | 3           | 2           | 1           |
|            | 5000        | 5           | 1           | 3           | 2           | 1           |
|            | 7500        | 5           | 1           | 3           | 2           | 1           |
|            | 10000       | 5           | 1           | 3           | 2           | 1           |
| $f_{\text{DiffPow}}$ | 2500        | 3           | 1           | 5           | 4           | 2           |
|            | 5000        | 3           | 1           | 5           | 4           | 2           |
|            | 7500        | 3           | 1           | 5           | 4           | 2           |
|            | 10000       | 3           | 1           | 5           | 4           | 2           |

Avg Rank 3.65 3.35 3.35 2.75 1.9

p-Value 0.00 0.04 0.04 0.89

Fig. 5. Runtime results on the $f_{\text{Elli}}$ problem. The dotted lines are obtained by the log-log linear regression.
accurate probability model, but in the meanwhile, makes the runtime longer. The analysis in Section II-D shows that a relatively small $l$ is enough for practical use and thus we verify this statement with simulations.

We choose four unimodal test problems, namely $f_{\text{Elli}}$, $f_{\text{Discus}}$, $f_{\text{Cigar}}$, and $f_{\text{DiffPow}}$, and investigate the scalability of MMES when different settings of $l$ are used. Fig. 6 shows the median results produced by MMES when $l$ is chosen from \{2, 4, 8, 16, 32\}. We observe that $l = 2$ is sufficient to solve $f_{\text{Discus}}$ while increasing $l$ can even deteriorate the performance. As discussed in Section IV-D, this is probably due to the fat-tailed distribution that has a higher probability of generating long jumps on insensitive search directions. This is not the case for $f_{\text{DiffPow}}$ where the settings of $l$ seem to have no influence on the performance. On $f_{\text{Elli}}$ and $f_{\text{Cigar}}$ whose landscapes can be well learned by the approximate distribution, it is observed that the performance can be improved by increasing $l$. This is consistent with the theoretical result that a larger mixing strength leads to a better approximation. Nevertheless, this does not mean a larger mixing strength is necessary, as the performance improvement seems to be quite insignificant. Thus, the default setting $l = 4$ would be a reasonable choice for balancing the approximation accuracy and the sampling efficiency.

G. Performance on CEC’2010 LSGO Problems

The CEC’2010 problems from the second test set are characterized with controllable non-separability and multimodality. The aim of the experiments on this test set is to 1) verify the potential of MMES as a generic global optimizer and 2) demonstrate its pros and cons compared with non-ES-based algorithms. Note that, ESs are usually designed for local search and should be used with restart strategies in order to solve multimodal problems \cite{50}. Thus, in this part of experiments, we restart MMES whenever a stagnation is detected. An instance of MMES is considered as stagnated if the improvement of the objective values during the last $n$ generation is smaller than $10^{-8}$. At each restart, we increase both the population size and the damping factor by a factor of 2, according to the suggestions from \cite{14}. This encourages MMES to explore the global structure of the landscape and prevents being trapped into local optima, thereby improving the global exploration ability.

Table \ref{table:ranking} summarizes the ranking results on the 1000-dimensional CEC’2010 LSGO test problems. The multiple comparison based on the Friedman test and the Bonferroni post hoc procedure shows that MMES ranks first among all algorithms and is superior to the three CC-based algorithms (i.e., DECC-G, CCPSO2, and DECC-DG). The Wilcoxon test also shows that it is significantly better than the other competitors on the majority of the test problems. In particular, MMES performs the best or the second best on highly multimodal problems with a huge number of local optima (e.g., $f_{15}$ and $f_{16}$). These results suggest that the MMES with restarts can serve as a competitive alternative to existing large-scale global optimization algorithms.

The simulation provides some hints about how MMES behaves differently from the other algorithms that are not based on the ES framework. For example, MMES is more robust to the non-separability than the CC-based algorithms, as the latter perform well on the fully separable problems but suffer from an obvious performance degradation on the other problems. The effectiveness of the CC-based algorithms is also limited by the degrees of the non-separability, as we see on the fully separable or fully non-separable problems that a more accurate grouping strategy (i.e., that in DECC-DG) does not guarantee achieving a better performance than the purely random grouping strategy (i.e., that in DECC-G). MA-SW and MOS perform generally better than the CC-based algorithms and can produce quite good results on the fully separable problems. This is due to their local search strategies performed individually on each variable in the way that the difference in the scalings of the coordinates can be easily explored. Nevertheless, they exhibit the strong dependence on the separable topology of the problems. MMES, on the contrary, does not explore the separable structure of the problem landscape and is less sensitive to the non-separability. In fact, the more non-separable subgroups are, the greater advantage over the others MMES possesses. This can be observed on the Ellipsoid problem where MMES performs the worst on the fully separable instance (i.e., $f_1$) but becomes the best performer whenever the non-separability is introduced (i.e., on $f_4$, $f_9$, or $f_{14}$).

The detailed numerical results are found in Table S-III in the supplement.

V. Conclusion

In this paper, we develop a large-scale CMA-ES variant, named MMES, based on a mixture modeling method for solution sampling and a paired test method for mutation strength adaptation. MMES permits reconstructing the probability model from an arbitrarily large number of direction
TABLE V
RANKS OF DIFFERENT ALGORITHMS, IN TERMS OF THE FINAL OBJECTIVE VALUES, ON THE 1000-DIMENSIONAL CEC’2010 LSGO PROBLEMS. BETTER RESULTS ARE HIGHLIGHTED WITH DARKER GRAY BACKGROUND.

|       | DECC-G | MA-SW | MOS | CCPSO2 | DECC-DG | MMES |
|-------|--------|-------|-----|--------|---------|------|
| f₁    | 3      | 3     | 1   | 4      | 5       | 6    |
| f₂    | 5      | 4     | 2   | 1      | 6       | 3    |
| f₃    | 4      | 2     | 5   | 3      | 6       | 1    |
| f₄    | 6      | 3     | 2   | 4      | 5       | 1    |
| f₅    | 4      | 3     | 6   | 5      | 2       | 1    |
| f₆    | 4      | 1     | 6   | 5      | 20      | 3    |
| f₇    | 6      | 5     | 1   | 40     | 4       | 2    |
| f₈    | 6      | 3     | 2   | 4      | 5       | 3    |
| f₉    | 6      | 2     | 5   | 4      | 3       | 1    |
| f₁₀   | 6      | 2     | 5   | 4      | 3       | 1    |
| f₁₁   | 3      | 4     | 6   | 5      | 1       | 2    |
| f₁₂   | 6      | 3     | 11  | 5      | 4       | 1    |
| f₁₃   | 5      | 2     | 3   | 4      | 6       | 1    |
| f₁₄   | 6      | 3     | 2   | 4      | 5       | 1    |
| f₁₅   | 5      | 2     | 6   | 4      | 3       | 1    |
| f₁₆   | 3      | 4     | 5   | 5      | 1       | 2    |
| f₁₇   | 6      | 3     | 2   | 5      | 4       | 1    |
| f₁₈   | 4      | 3     | 2   | 5      | 6       | 1    |
| f₁₉   | 5      | 3     | 2   | 4      | 6       | 1    |

Avg Rank 4.9 2.7 3.25 4.28 4.1 1.77
p-Value 0.1 0.188 0.0003 0.0012

vectors in limited operation time such that the rich variable correlations can be exploited efficiently. The theoretical analyses show that MMES well approximates CMA-ES in terms of the underlying probability model while the numerical simulations suggest that MMES demonstrates state-of-the-art performance in both local and global optimization.

Further development of MMES will be two-directional. The first direction is to develop new mechanisms for adapting the direction vectors. The current version of MMES does not involve this aspect but directly applies well established techniques. However, the proposed mixture modeling method should not be limited in the sampling process; using it to accelerate the distribution adaptation would also be an interesting idea. The second direction is to establish the convergence theorem for MMES. The PTA method in MMES relies on the comparison of the populations in consecutive generations, and thus, implies some types of elitism. We plan to search for an efficient indicator to quantify the population which may guarantee sufficient descents and lead to a global convergence theorem.

REFERENCES
[1] H.-G. Beyer and H.-P. Schwefel, “Evolution strategies–A comprehensive introduction,” Natural computing, vol. 1, no. 1, pp. 3–52, 2002.
[2] N. Hansen and A. Ostermeier, “Completely Derandomized Self-Adaptation in Evolution Strategies,” Evolutionary Computation, vol. 9, no. 2, pp. 159–195, Jun. 2001.
[3] Nikolaus Hansen, Sibylle D. Müller, and Petros Koumoutsakos, “Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES),” Evolutionary Computation, vol. 11, no. 1, pp. 1–18, Mar. 2003.
[4] D. V. Arnold, “Weighted multi-Combination evolution strategies,” Theoretical Computer Science, vol. 361, no. 1, pp. 18–37, Aug. 2006.
[5] A. Abdolmaleki, B. Price, N. Luan, L. P. Reis, and G. Neumann, “Deriving and improving CMA-ES with information geometric trust regions,” in Proceedings of the Genetic and Evolutionary Computation Conference. Berlin, Germany: ACM, Jul. 2017, pp. 657–664.
[6] Z. Chen, Y. Zhou, X.-y. He, and S. Jiang, “A Restart-based Rank-1 Evolution Strategy for Reinforcement Learning,” in Proceedings of the Twenty-Eight International Joint Conference on Artificial Intelligence. Macao, China: International Joint Conferences on Artificial Intelligence Organization, Aug. 2019, pp. 2130–2136.
[7] E. Mezura-Montes and C. A. C. Coello, “An empirical study about the usefulness of evolution strategies to solve constrained optimization problems,” International Journal of General Systems, vol. 37, no. 4, pp. 443–473, Aug. 2008.
[8] M. D. Gregory, Z. Bayraktar, and D. H. Werner, “Fast Optimization of Electromagnetic Design Problems Using the Covariance Matrix Adaptation Evolutionary Strategy,” IEEE Transactions on Antennas and Propagation, vol. 59, no. 4, pp. 1275–1285, Apr. 2011.
[9] 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 Transactions on Evolutionary Computation, vol. 13, no. 1, pp. 180–197, Feb. 2009.
[10] O. M. Shiri, J. Roslund, D. Whitley, and H. Rabitz, “Efficient retrieval of landscape Hessians: Forced optimal covariance adaptive learning,” Physical Review E, vol. 89, no. 6, pp. 063306, Jun. 2014.
[11] R. Ros and N. Hansen, “A Simple Modification in CMA-ES Achieving Linear Time and Space Complexity,” in Proceedings of the 10th International Conference on Parallel Problem Solving from Nature: PPSN X. Dortmund, Germany: Springer-Verlag, 2008, pp. 296–305.
[12] I. Loshchilov, “LM-CMA: An Alternative to L-BFGS for Large-Scale Black Box Optimization,” Evolutionary Computation, vol. 25, no. 1, pp. 143–171, Mar. 2017.
[13] I. Loshchilov, T. Glasmachers, and H. G. Beyer, “Large Scale Black-box Optimization by Limited-Memory Matrix Adaptation,” IEEE Transactions on Evolutionary Computation, vol. 23, no. 2, pp. 353–358, Apr. 2019.
[14] Z. Li and Q. Zhang, “A Simple yet Efficient Evolution Strategy for Large Scale Black-Box Optimization,” IEEE Transactions on Evolutionary Computation, vol. 22, no. 5, pp. 637–646, Oct. 2018.
[15] Z. Li, Q. Zhang, X. Lin, and H. Zhen, “Fast Covariance Matrix Adaptation for Large-Scale Black-Box Optimization,” IEEE Transactions on Cybernetics, vol. 50, no. 5, pp. 2073–2083, May 2020.
[16] X. He, Y. Zhou, Z. Chen, J. Zhang, and W. Chen, “Large-Scale Evolution Strategy Based on Search Direction Adaptation,” IEEE Transactions on Cybernetics, vol. in press, 2020.
[17] Y. Akimoto, A. Auger, and N. Hansen, “Comparison-based natural gradient optimization in high dimension,” in GECCO ’14 Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation Pages 373-380. Vancouver, BC, Canada: ACM Press, July 12 - 16, 2014, pp. 373–380.
[18] Y. Akimoto and N. Hansen, “Projection-Based Restricted Covariance Matrix Adaptation for High Dimension,” in GECCO ’16 Proceedings of the Genetic and Evolutionary Computation Conference 2016. Denver, Colorado, USA: ACM Press, July 20 - 24, 2016, pp. 197–204.
[19] Z. Yang, K. Tang, and X. Yao, “Large scale evolutionary optimization using cooperative coevolution,” Information Sciences, vol. 178, no. 15, pp. 2985–2999, Aug. 2008.
[20] 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. Hefei, China: Springer, 2013, pp. 350–357.
[21] Y. Mei, M. N. Omidvar, X. Li, and X. Yao, “A Competitive Divide-and-Conquer Algorithm for Unconstrained Large-Scale Black-Box Optimization,” ACM Transactions on Mathematical Software, vol. 42, no. 2, pp. 1–24, Jun. 2016.
[22] X. Kong, B. Yuan, and B. Li, “Model complex control CMA-ES,” Swarm and Evolutionary Computation, vol. 50, p. 100558, Nov. 2019.
[23] Y. Sun, X. Li, A. Ernst, and M. N. Omidvar, “Decomposition for Large-scale Optimization Problems with Overlapping Components,” in 2019 IEEE Congress on Evolutionary Computation (CEC), Jun. 2019, pp. 326–333.
[24] X. Ma, X. Li, Q. Zhang, K. Tang, Z. Liang, W. Xie, and Z. Zhu, “A Survey on Cooperative Co-evolutionary Algorithms,” IEEE Transactions on Evolutionary Computation, pp. 1–1, 2018.
[25] A. Kabán, J. Bookkrajc, and R. J. Durrant, “Toward Large-Scale Continuous EDA: A Random Matrix Theory Perspective,” IEEE Transactions on Evolutionary Computation, vol. 24, no. 2, pp. 255–291, Jun. 2016.
[26] M. Sanyang and A. Kabán, “Heavy tails with parameter adaptation in random projection based continuous EDA,” in 2015 IEEE Congress on Evolutionary Computation (CEC). Sendai: IEEE, May 2015, pp. 2074–2081.
[27] W. Dong, Y. Wang, and M. Zhou, “A latent space-based estimation of distribution algorithm for large-scale global optimization,” Soft Comput., vol. 23, no. 13, pp. 4593–4615, Jul. 2019.
[28] D. Jagodzinski and J. Arabas, “A differential evolution strategy,” in 2017 IEEE Congress on Evolutionary Computation (CEC), Jun. 2017, pp. 1772–1776.
[29] J. Aradas and D. Jagodzinski, “Towards a Matrix-free Covariance Matrix Adaptation Evolution Strategy,” IEEE Transactions on Evolutionary Computation, pp. 1–1, 2019.
[30] N. Hansen and N. Hansen, “Adaptive Encoding: How to Render Search Coordinate Invariant,” in Parallel Problem Solving from Nature – PPSN V, T. Jansen, N. Beume, S. Lucas, and C. Poloni, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, vol. 5199, pp. 205–214.
[31] N. Hansen, “The CMA Evolution Strategy: A Tutorial,” arXiv:1604.00772 [cs, stat], Apr. 2016.
[32] D. Arnold and H. G. Beyer, “Performance Analysis of Evolutionary Optimization With Cumulative Step Length Adaptation,” IEEE Transactions on Automatic Control, vol. 49, no. 4, pp. 617–622, Apr. 2004.
[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,” Nature Inspired Computation and Applications Laboratory, USTC, China, Tech. Rep., 2009.
[34] D. Molina, M. Lozano, A. M. Sanchez, and F. Herrera, “Mimetic algorithms based on local search chains for large scale continuous optimization problems: MA-SSW-Chains,” Soft Computing, vol. 15, no. 11, pp. 2201–2220, Nov. 2011.
[35] A. LaTorre, S. Muelas, and J.-M. Peña, “Large scale global optimization: Experimental results with MOS-based hybrid algorithms.” Cancun, Mexico: IEEE, Jun. 2013, pp. 2742–2749.
[36] X. Li and X. Yao, “Cooperatively Coevolving Particle Swarms for Large Scale Optimization,” IEEE Transactions on Evolutionary Computation, vol. 16, no. 2, pp. 210–224, Apr. 2012.
[37] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, “Cooperative Co-Evolution With Differential Grouping for Large Scale Optimization,” IEEE Transactions on Evolutionary Computation, vol. 18, no. 3, pp. 378–393, Jun. 2014.
[38] A. LaTorre, S. Muelas, and J.-M. Pena, “A comprehensive comparison of large scale global optimizers,” Information Sciences, vol. 316, pp. 517–549, Sep. 2015.
[39] Q. Yang, W.-N. Chen, J. Da Deng, Y. Li, T. Gu, and J. Zhang, “A Level-based Learning Swarm Optimizer for Large Scale Optimization,” IEEE Transactions on Evolutionary Computation, vol. 22, no. 4, pp. 578 – 594, Aug. 2018.
[40] F. Wilcoxon, “Individual comparisons by ranking methods,” Biometrics bulletin, vol. 1, no. 6, pp. 80–83, 1945.
[41] J. Derrac, S. Garcia, D. Molina, and F. Herrera, “A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms,” Swarm and Evolutionary Computation, vol. 1, no. 1, pp. 3–18, Mar. 2011.
[42] O. J. Dunn, “Multiple Comparisons among Means,” Journal of the American Statistical Association, vol. 56, no. 293, pp. 52–64, Mar. 1961.
[43] D. Morinaga and Y. Akimoto, “Generalized drift analysis in continuous domain: Linear convergence of (1 + 1)-ES on strongly convex functions with Lipschitz continuous gradients,” in Proceedings of the 15th ACM/SIGEVO Conference on Foundations of Genetic Algorithms - FOGA ’19. Potdam, Germany: ACM Press, 2019, pp. 13–24.
[44] A. Auger and N. Hansen, “Linear convergence of comparison-based Step-size Adaptive Randomized Search via Stability of Markov Chains,” SIAM Journal on Optimization, vol. 26, no. 3, pp. 1589–1624, Jan. 2016.
[45] G. A. Jastrebski and D. V. Arnold, “Improving Evolution Strategies through Active Covariance Matrix Adaptation,” in IEEE Congress on Evolutionary Computation, Vancouver, BC, Canada: IEEE, Jul. 2006.
[46] H. Wang, M. Emmerich, and T. Back, “Mirrored Orthogonal Sampling for Covariance Matrix Adaptation Evolution Strategies,” Evolutionary Computation, vol. 27, no. 4, pp. 699–725, Feb. 2019.
[47] O. Krause, T. Glasmachers, and C. Igel, “Qualitative and Quantitative Assessment of Step Size Adaptation Rules,” in Proceedings of the 14th ACM/SIGEVO Conference on Foundations of Genetic Algorithms - FOGA ’17. Copenhagen, Denmark: ACM Press, 2017, pp. 139–148.
[48] J. E. Rowe and D. Sudholt, “The choice of the offspring population size in the (1,λ) evolutionary algorithm,” Theoretical Computer Science, vol. 545, pp. 20–38, Aug. 2014.
[49] Y.-W. Shang and Y.-H. Qiu, “A Note on the Extended Rosenbrock Function,” Evol. Comput., vol. 14, no. 1, pp. 119–126, Mar. 2006.
[50] A. Auger and N. Hansen, “A Restart CMA Evolution Strategy With Increasing Population Size,” in 2005 IEEE Congress on Evolutionary Computation, vol. 2. IEEE, 2005, pp. 1769–1776.

APPENDIX A
PROOF OF LEMA 1

Proof. Denote
\[ p(z) \] as the probability density function (p.d.f.) value of a mutation vector \( z \) sampled from \( P_m \),
\[ \phi(z|\Sigma_i) \] as the conditional p.d.f. of \( P_{\Sigma_i} \),
\[ \phi(z|\Sigma_i) \] as the joint probability of choosing \( i_1, \cdots, i_m \) from \( P_i \),
\[ [m] \] as the set \{1, 2, \cdots, m\}.

By the law of total probability, \( P_m \) can be expressed with \( P_{\Sigma_i} \) and \( P_i \) using the relation
\[ p(z) = \sum_{i \in [m]} p(i) \phi(z|\Sigma_i) \] (15)
Then, we have
\[
M(t) = E[\exp(t^T z)] \\
= \int \exp(t^T z)p(z)dz \\
= \sum_{i \in [m]} p(i) \int \exp(t^T z)\phi(z|\Sigma_i)dz \\
= \sum_{i \in [m]} p(i)E_i[\exp(t^T z)]
\]
where \(E[\cdot]\) denotes the expectation with respect to \(z\) and \(E_i[\cdot]\) denotes the expectation with respect to \(z\) conditioned on \(i\).

\(E_i[\exp(t^T z)]\) in the last equation equals to the moment generating function of \(\mathcal{P}_{\Sigma_i}\), so we have
\[
M(t) = \sum_{i \in [m]} p(i) \exp\left(\frac{1}{2}t^T \Sigma_i t\right)
\]
Substituting Eq. (15) into this yields
\[
M(t) = \sum_{i \in [m]} p(i) \exp\left(\frac{1}{2}(1-\gamma)|t|^2 + \frac{\gamma}{2l} \sum_{j=1}^l (t^T q_{ij})^2\right)
\]
\[
= \exp\left(\frac{1}{2}(1-\gamma)|t|^2\right) \sum_{i \in [m]} p(i) \prod_{j=1}^l \exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right)
\]
(16)

We immediately reach the conclusion if \(l = 1\). When \(l > 1\), the second multiplier of the last equation in Eq. (16) can be written in the following recursive form:
\[
\sum_{i \in [m]} p(i) \prod_{j=1}^l \exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right) \\
= \sum_{i \in [m]} p(i, i_1) \prod_{j=1}^{l-1} \exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right) p(i) \exp\left(\frac{\gamma}{2l} (t^T q_{il})^2\right) \\
= \sum_{i \in [m]} p(i) \prod_{j=1}^{l-1} \exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right) \left(\sum_{i_l \in [m]} p(i) \exp\left(\frac{\gamma}{2l} (t^T q_{il})^2\right)\right)
\]
Repeat this for \(l - 1\) times and we obtain
\[
\sum_{i \in [m]} p(i) \prod_{j=1}^l \exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right) \\
= \prod_{j=1}^l \left(\sum_{i_j \in [m]} p(i) \exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right)\right) \\
= \prod_{j=1}^l \left(\sum_{i_j \in [m]} \alpha_j \exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right)\right)
\]
Substitute this into Eq. (16) and the target form of the moment generating function is reached.

Then, we prove that \(M(t)\) uniquely determines \(\mathcal{P}_m\) by showing there exists a neighborhood of \(0_n\) in which \(M(t)\) is finite.

Let \(\delta\) be a positive scalar. Then, for any \(t\) satisfying \(|t| \leq \delta\), we have
\[
M(t) \leq \exp\left(\frac{1}{2}(1-\gamma)\delta^2\right) \sum_{j=1}^m \alpha_j \exp\left(\frac{\gamma}{2l} \delta^2 |q_{ij}|^2\right) \\
\leq \exp\left(\frac{1}{2}(1-\gamma)\delta^2\right) \sum_{j=1}^m \alpha_j \exp\left(\frac{\gamma}{2l} \delta^2 |q_{ij}|^2\right)
\]
where the second inequality is derived from the Jensen’s inequality. This gives a positive radius of convergence for the moment generating function, which completes the proof. □

APPENDIX B

PROOF OF THEOREM 1

Proof. Denote the moment generating function of \(\mathcal{P}_a\) by \(U(t)\). With Eqs. (2) and (15), we have
\[
U(t) = \exp\left(\frac{1}{2} t^T \left(1 - c_a\right) I_n + c_a \sum_{j=1}^m (1 - c_a)^{m-j} q_j q_j^T\right) t
\]
\[
= \exp\left(\frac{1}{2} t^T \left(1 - c_a\right) I_n + \gamma \sum_{j=1}^m \alpha_j q_j q_j^T\right) t
\]
Using the identity of zeroth weighted power mean (i.e., \(\lim_{r \to 0} (\omega_1 x_1^r + \cdots + \omega_n x_n^r)^{1/r} = x_1^\omega_1 \cdots x_n^\omega_n\) for \(\sum_{i=1}^n \omega_i = 1\) and \(\omega_1, \cdots, \omega_n > 0\), we have
\[
\lim_{t \to \infty} M(t) \\
= \lim_{t \to \infty} \exp\left(\frac{1}{2}(1-\gamma)|t|^2\right) \left(\sum_{j=1}^m \alpha_j \exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right)\right) \\
= \lim_{t \to \infty} \exp\left(\frac{1}{2}(1-\gamma)|t|^2\right) \left(\sum_{j=1}^m \alpha_j \left(\exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right)\right)^{\frac{\gamma}{2l}}\right) \\
= \exp\left(\frac{1}{2}(1-\gamma)|t|^2\right) \prod_{j=1}^m \left(\exp\left(\frac{\gamma}{2l} (t^T q_{ij})^2\right)^{\alpha_j}\right) \\
= \exp\left(\frac{1}{2}(1-\gamma)|t|^2\right) \exp\left(\sum_{j=1}^m \gamma \alpha_j (t^T q_{ij})^2\right) \\
= \exp\left(\frac{1}{2} t^T \left(1 - c_a\right) I_n + \gamma \sum_{j=1}^m \alpha_j q_j q_j^T\right) t
\]
As \(\mathcal{P}_m\) can be fully characterized by its moment generating function (see Lemma [1]), we can conclude that its limiting distribution is the same as \(\mathcal{P}_a\). This completes the proof. □
APPENDIX C
PROOF OF THEOREM 2

Proof. According to Eq. (15), the covariance matrix of \( \mathcal{P}_m \) can be written as the expectation of the covariance matrix of \( \mathcal{P}_i \):

\[
Var[z] = E[zz^T] = \int zz^T p(z) dz
\]

\[
= \int zz^T \sum_{i \in [m]} p(i) \phi(z | \Sigma_i) dz = \sum_{i \in [m]} p(i) \int zz^T \phi(z | \Sigma_i) dz = \sum_{i \in [m]} p(i) \Sigma_i\]

where \( Var[\cdot] \) denotes the variance.

By Eq. (3), we have

\[
Var[z] = \sum_{i \in [m]} p(i) \left( (1 - \gamma) I_n + \frac{\gamma}{l} \sum_{j=1}^{l} q_{ij} (q_{ij})^T \right)
\]

(17)

(18)

With the same method as in proving Lemma [1] we have

\[
\sum_{i \in [m]} p(i) \sum_{j=1}^{l} q_{ij} q_{ij}^T = \sum_{i \in [m]} p(i) \left( \sum_{j=1}^{l-1} q_{ij} q_{ij}^T + q_0 q_0^T \right)
\]

\[
= \sum_{i \in [m]} p(i) \sum_{j=1}^{l-1} q_{ij} q_{ij}^T + \sum_{i \in [m]} p(i) q_0 q_0^T
\]

\[
= \cdots
\]

\[
= \sum_{i \in [m]} p(i_1) q_{i_1} q_{i_1}^T + \cdots + \sum_{i \in [m]} p(i_l) q_{i_l} q_{i_l}^T
\]

\[
= \sum_{j \in [m]} \alpha_j q_j q_j^T
\]

Substituting this into Eq. (17) and adopting the parameters in Eq. (6) lead to the conclusion.

APPENDIX D
PROOF OF THEOREM 3

Proof. We first show that

\[
\frac{M(t)}{U(t)} = \left( \sum_{j=1}^{m} \alpha_j \exp \left( \frac{\gamma}{l} (q_j^T t)^2 \right) \right)^l
\]

(19)

It follows from Eq. (6) and Lemma [1] that

\[
M(t) = \left( \sum_{j=1}^{m} \alpha_j \exp \left( \frac{\gamma}{2l} (q_j^T t)^2 \right) \right)^l
\]

\[
U(t) = \exp \left( \frac{\gamma}{2l} \sum_{i=1}^{m} \alpha_i (t^T q_i)^2 \right)
\]

Then \( \nabla^k M(0_n) \) is the \( k \)-th order moment of \( \mathcal{P}_m \).

Let \( \nabla_k^r \) be the operator of summing up all \( r \)-th order partial derivative with respect to the elements of \( t \), where the indexes are combinations of \( i_1, i_2, \cdots, i_k \) taken \( r \) at a time. That is,

\[
\nabla_k^r M(t) = \sum_{j_1, \cdots, j_r \in \{1, \cdots, n\}} \frac{\partial^r}{\partial t_{j_1} \cdots \partial t_{j_r}} M(t)
\]

With the above notations, we have \( \nabla_k^r = \nabla^k \) and
\[ \nabla^k M(t) = \nabla^k \left( U(t) \left( 1 + O \left( \frac{1}{t} |t|^4 \right) \right) \right) \]
\[ = \nabla^k U(t) + \nabla^k \left( U(t) O \left( \frac{1}{t} |t|^4 \right) \right) \]
\[ = \nabla^k U(t) + \sum_{j=k-4} \left( \nabla^j U(t) \right) O \left( \frac{|t|^{4-k+j}}{l} \right) \]
\[ = \nabla^k U(t) + \sum_{j=k-3} \left( \nabla^j U(t) \right) O \left( \frac{|t|^{4-k+j}}{l} \right) \]
\[ + \left( \nabla^k U(t) \right) O \left( \frac{1}{l} \right) \]

Set \( t = 0 \) on both sides and we obtain the \( k \)-th order moment

\[ \nabla^k M(0) = \nabla^k U(0) + \sum_{j=k-3} \left( \nabla^j U(0) \right) O \left( \frac{|0|^{4-k+j}}{l} \right) \]
\[ + \left( \nabla^k U(0) \right) O \left( \frac{1}{l} \right) \]

\[ \nabla^k U(0) \] is finite due to the existence of the moments of \( P_m \). We thus obtain

\[ \nabla^k M(0) = \nabla^k U(0) + O \left( \frac{1}{l} \right) \]

and the theorem is proved.

**APPENDIX E**

**PROOF OF THEOREM 4**

**Proof.** It follows from Lemma [1] that the moment generating function of the projected distribution onto any unit vector \( v \) is

\[ M(t|v) = \exp \left( \frac{1}{2l} (1 - \gamma) t^2 \right) \left( \sum_{j=1}^m \alpha_j \exp \left( \frac{\gamma}{2l} (v^T q_j)^2 t^2 \right) \right)^l \]
\[ = \left( \exp \left( \frac{1}{2l} (1 - \gamma) t^2 \right) \right)^l \]
\[ \left( \sum_{j=1}^m \alpha_j \exp \left( \frac{\gamma}{2l} (v^T q_j)^2 t^2 \right) \right)^l \]
\[ = \left( \sum_{j=1}^m \alpha_j \exp \left( \frac{t^2}{2l} ((1 - \gamma) + \gamma (v^T q_j)^2) \right) \right)^l \]
\[ = \left( \sum_{j=1}^m \alpha_j \exp \left( \frac{t^2}{2l} d_j \right) \right)^l \]

where we denote \((1 - \gamma) + \gamma (v^T q_j)^2\) by \(d_j\) to simplify the notations. Expand the exponential term as

\[ \exp \left( \frac{t^2}{2l} d_j \right) = 1 + \frac{t^2}{2l} d_j + \frac{t^4}{8l^2} d_j^2 + O(t^6) \]

and we obtain

\[ M(t|v) = \left( \sum_{j=1}^m \alpha_j \left( 1 + \frac{t^2}{2l} d_j + \frac{t^4}{8l^2} d_j^2 + O(t^6) \right) \right)^l \]
\[ = 1 + t^2 \sum_{j=1}^m \alpha_j d_j + t^4 \sum_{j=1}^m \alpha_j d_j^2 + O(t^6) \]
\[ = t^4 \left( \frac{l(l-1)}{2} \left( \sum_{j=1}^m \alpha_j d_j \right)^2 + l \sum_{j=1}^m \alpha_j d_j^2 \right) \]
\[ + 1 + At^2 + O(t^6) \]

where \( A \) is some constant independent of \( t \) and the last equality is derived from the multinomial theorem.

The fourth order moment can thus be obtained as

\[ \frac{\partial^4}{\partial t^4} M(t|v)|_{t=0} \]
\[ = 24 \left( \frac{l(l-1)}{2} \left( \sum_{j=1}^m \alpha_j d_j \right)^2 + l \sum_{j=1}^m \alpha_j d_j^2 \right) \]
\[ = 3 \frac{l(l-1)}{l} \left( \sum_{j=1}^m \alpha_j d_j \right)^2 + 3 \frac{l}{l} \sum_{j=1}^m \alpha_j d_j^2 \]

On the other hand, it follows from Eq. (3) that the variance of the projected distribution is \( \sum_{j=1}^m \alpha_j d_j \). So the excess kurtosis can be calculated as

\[ \frac{\partial^4}{\partial t^4} M(t|v)|_{t=0} = 3 \left( \sum_{j=1}^m \alpha_j d_j^2 - \left( \sum_{j=1}^m \alpha_j d_j \right)^2 \right) \]

Applying the Jensen’s inequality

\[ \sum_{j=1}^m \alpha_j d_j^2 \geq \left( \sum_{j=1}^m \alpha_j d_j \right)^2 \]

completes the proof.

**APPENDIX F**

**PROOF OF PROPOSITION 1**

**Proof.** Set \( A_i = \omega_i I \{ f(x_{\omega_i}^{(g-1)}) > f(x_{\omega_i}^{(g)}) \} \). We show that the sequence \( \{A_i\} \) has bounded fourth order moment such that the Lyapunov’s condition holds and we can apply the central limit theorem to reach the conclusion.

Firstly, it is easy to see that \( E[A_i] = \frac{1}{2} \omega_i \) and \( Var[A_i] = \frac{1}{4} \omega_i^2 \). Then, we check the Lyapunov’s condition by investigat-
ing the growing rate of the fourth order moment compared with the second order moment

\[
T_\mu = \frac{1}{(\sum_{i=1}^\mu Var[A_i])^2} \sum_{i=1}^\mu E[|A_i - E[A_i]|^4]
\]

\[
= 2 \frac{1}{(\sum_{i=1}^\mu \omega_i^2)^2} \sum_{i=1}^\mu \omega_i^4
\]

\[
= 2 \frac{1}{(\sum_{i=1}^\mu (\omega_i')^2)^2} \sum_{i=1}^\mu (\omega_i')^4
\]

Then, define a function

\[
v(x) = (\ln(\mu + 0.5) - \ln(x))^2.
\]

It is easy to see that \(v(x)\) is convex on the range \([1, \mu]\). Then we have, from this convexity,

\[
\sum_{i=1}^\mu (\omega_i')^2 = \sum_{i=1}^\mu v(i) \geq \mu v\left(\frac{1 + \mu}{2}\right)
\]

\[
= \mu \left(\ln\left(\frac{2 \mu + 1}{\mu + 1}\right)\right)^2
\]

Therefore, we have

\[
T_\mu \leq 2 \frac{\sum_{i=1}^\mu (\omega_i')^4}{\mu^2 \left(\ln\left(\frac{2 \mu + 1}{\mu + 1}\right)\right)^4} \leq 2 \frac{\mu (\omega_1')^4}{\mu^2 \left(\ln\left(\frac{2 \mu + 1}{\mu + 1}\right)\right)^4}
\]

and thus \(\lim_{\mu \to \infty} T_\mu = 0\). This indicates that the rate of growth of the fourth order moment is limited, i.e., the Lyapunov’s condition holds. Finally, apply the central limit theorem and we get

\[
\frac{\sum_{i=1}^\mu (A_i - E[A_i])}{\sqrt{\sum_{i=1}^\mu Var[A_i]}} \xrightarrow{d} N(0, 1)
\]

which is equivalent to the conclusion.

**APPENDIX G**

**Detailed numerical results associated with Tables III to V**

Tables VI to VIII provides respectively the detailed numerical results associated with Tables III to V in the main manuscript.

**APPENDIX H**

**Additional experiments for verifying the impact of mixing strength**

While in the main manuscript we have investigated the performance of MMES with mixing strength set to different small constants (i.e., \(l \in \{2, 4, 8, 16, 32\}\)), here we consider another setting, \(l = m\). That is, totally \(m\) vectors are randomly selected from \(q_1, \cdots, q_m\) for constructing the mixture distribution \(P_n\). Please note that since the selection is with replacement this setting does not guarantee that every \(q\) vector is chosen exactly once. It neither produces the target distribution \(P_a\), as the latter can only be achieved in the case of \(l \to \infty\) (see Theorem 1). The aim of this section is to verify the impact of mixing strength when set to a relatively large value and to provide more evidences to our statements about its practical setting.

**A. Impact on the algorithm efficiency**

We firstly show the runtime of MMES versus different \(l\) values, where runtime is defined and measured in the same way as in Section IV-E. To reflect the asymptotic performance, we plot for each \(l\) a log-log regression line by fitting the experimental results. Fig. 7 shows that the runtime of MMES increases consistently as \(l\) increases. For \(l \in \{2, 4, 8, 16, 32\}\) the runtime scales approximately linearly because the mixing strengths are constant. Choosing \(l = m\) leads to an obvious jump in the order of the runtime (i.e., 1.39 as shown by the regression line). It is because the parameter \(m\) is set to \(O(\sqrt{n})\), so the runtime increases super-linearly with the increasing \(n\). In general, these observations coincide with the \(O(ln)\) time complexity of MMES. We also consider another setting where the solutions are directly drawn from the target distribution, denoted by \(P_\alpha\), in the plot. This setting leads to a super-linearly increasing runtime but is significantly faster than \(l = m\), probably because the sampling can be performed within a simple matrix-vector multiplication. However, it requires nearly triple the time compared to the setting \(l = 4\), demonstrating the time efficiency improvement of the proposed mixture sampling method.

![Fig. 7. Runtime results measured for different mixing strengths. The instance of MMES where the mixture distribution is directly replaced with the target Gaussian distribution is also considered and is denoted by \(P_\alpha\). The dashed lines are obtained by the log-log linear regression where the order of \(n\) estimates the asymptotic order of runtime growth.](image)

**B. Impact on the scalability**

We then assess the impact of mixing strength on the scalability performance of MMES. The experiment adopts the same setting as in Section IV-F. However, setting \(l\) to \(m\) leads
to a super-linearly increase of the execution time and so we have to modify the stopping criteria such that MMES can stop in a reasonable time budget. Concretely, the target solution accuracy \( \epsilon_{\text{acc}} \) for terminating MMES is set to \( 10^{-5} \), and \( 10^{-4} \) for \( f_{\text{Ell}} \), \( f_{\text{Discus}} \), and \( f_{\text{DiffPow}} \), respectively. The \( f_{\text{Cigar}} \) function is relatively simple, so we use the same setting \( \epsilon_{\text{acc}} = 10^{-8} \) as in Section IV-E.

Fig. 8 shows how mixing strength influences the effectiveness of MMES in decreasing the objective function. The presented results measure the number of function evaluations required to achieve the target accuracy. On \( f_{\text{Ell}} \) and \( f_{\text{Cigar}} \), the performance improves as \( l \) increases; but these improvements are quite small. For example, in the 10000-dimensional case for both \( f_{\text{Ell}} \) and \( f_{\text{Cigar}} \), increasing \( l \) from 2 to \( m \) saves no more than 5\% function evaluations. Using \( l = m \) leads to a moderate improvement on 2500-dimensional \( f_{\text{Discus}} \), but this improvement becomes insignificant for higher dimensions.

On \( f_{\text{Discus}} \), contrarily, the increase of \( l \) causes an obvious performance degradation; the setting \( l = m \) leads to nearly 25\% more function evaluations than \( l = 2 \) does in the 10000-dimensional instance. Linking these observations and those in Fig. 7 we conclude that a small constant value for the mixing strength is reasonable and can serve as a good trade-off between performance and efficiency.

C. Impact on the rotational invariance

The proposed FMS method is obviously invariant against rotations since it relies only on basic linear operations. Therefore, any rotational transformations applied to the decision space will not influence the performance of MMES, if the initial distribution can be transformed correspondingly. In Section IV-C we have demonstrated that, with \( l = 4 \), this statement holds in practice and the non-invariant initialization has little influence on the algorithm performance. One may,
however, argue that with a larger $l$, MMES may require more computational effort to offset the possible influence caused by the non-invariant initialization. This accords with the intuition that with a large $l$ the search distribution of MMES in the early stage may behave like a Gaussian which is non-invariantly initialized. Below we will show that this is not the case and MMES still possesses the rotational invariance even when $l$ is large.

The experiment considers the setting $l = m$. We perform MMES on the 1000-dimensional non-rotated problems $f_{\text{Ell}i}$, $f_{\text{Rosen}}$, $f_{\text{Discus}}$, $f_{\text{Cigar}}$, $f_{\text{DiffPow}}$ and their rotated versions $f_{\text{RotEll}i}$, $f_{\text{RotRosen}}$, $f_{\text{RotDiscus}}$, $f_{\text{RotCigar}}$, $f_{\text{RotDiffPow}}$ and then compare the corresponding convergence curves. As shown in Fig. 9, MMES behaves almost the same on a non-rotated problem and its rotated counterpart. That is, increasing $l$ does not impact the rotational invariance of MMES. One possible reason is that, by initializing the $q$ vectors to zeros, MMES imposes a strict limit to the degrees of freedom in the initial distribution which in turn reduces its impact on the algorithm performance.

**D. Relation to the problem**

The results in Fig. 8 show that, even when the time budget is not limited, no one value of the mixing strength $l$ is uniformly preferable. A larger $l$ does not necessary improve performance, but may, in fact, degrade it. Note that this does not contradict the theoretical analyses provided in Section 11 since $l$ only influences the approximation to the Gaussian distribution $P_\alpha$ while the latter is not guaranteed to be the best distribution for guiding the search. The strict dependence of the optimal $l$ on the problem type is unclear; but below we carry out a primary study which may provide practical hints for choosing the mixing strength.

**a) Irrelevance of approximation accuracy and algorithm performance**

It should be noted firstly that a better approximation to the Gaussian distribution does not lead to a better performance. We state here two reasons for which the Gaussian distribution is so widely used in modern ESs, with the aim of illustrating that the problem characteristics are usually not taken into consideration in the design of Gaussian based ESs.

The first reason is that the Gaussian distribution is preferable for black-box optimization as it has maximal entropy among all continuous distributions [51]. The use of the Gaussian distribution is mainly to achieve uniformly good performance and have no explicit relation to specific characteristics of the problems. According to the No Free Lunch theorem [52], one can always find a certain type of problems on which ESs with a non-Gaussian distribution works better than with a Gaussian
distribution. Thus, using a worse approximation to achieve a better performance does not come as a surprise.

The second reason is more concrete and limited to the class of quadratic problems. Existing studies suggested that the standard ES with an isotropic Gaussian exhibits linear convergence on quadratic problems and the convergence rate is inversely related to the condition number of the Hessian \[53\]. Modern ESs generally adapt a Gaussian distribution to the problem landscape which then acts as a pre-conditioner to eliminate the adverse effect of ill-conditioning \[54\]. However, the usefulness of the Gaussian distribution is heuristic based and often relies on the assumption that the variable correlations can be fully captured by the maintained covariance matrix. In real scenes of large-scale optimization, there is no theoretical guarantee that the Gaussian distribution works the best when the correlations can only be partially learned.

b) An example for which a worse approximation may work better: We give an example to show that the approximation distribution \(P_m\) can work better than the target distribution \(P_a\). Consider a problem \(f(x) = x^T H x\) to be minimized from a starting point \(x_0\) sufficiently far from the optimum 0, where \(H\) is the symmetric positive definite Hessian. Denote \(u\) as the eigenvector of \(H\) corresponding to the smallest eigenvalue. We omit the mutation strength for simplicity and focus on analyzing the samples in a single step. The quality of a sample \(y\) is dominated by the distance of its projection on \(u\) to 0, given by \(|u^T y|\). We further assume \(u \notin \text{span}(q_1, \ldots, q_n)\), which simulates the situation when the probability model fails to capture the most promising search direction. Now consider two samples \(y_a\) and \(y_m\), sampled from \(P_a\) and \(P_m\), respectively. From the properties of multivariate Gaussian distribution we have \(u^T y_a \sim \mathcal{N}(u^T x_0, u^T C_a u)\) which is reduced to \(\mathcal{N}(u^T x_0, (1 - c_a)^m)\). Regarding \(y_m\) we have \(E[u^T y_m] = u^T x_0\) and \(\text{Var}[u^T y_m] = (1 - c_a)^m\), where the first equality holds trivially and the second is according to Theorem \(\ref{theorem1}\). That is, the improvements made by \(y_a\) and \(y_m\), along the direction of \(u\), have the same first- and second-order moments. Despite this similarity, \(u^T y_a\) has zero excess kurtosis since it is normally distributed while \(u^T y_m\) has positive excess kurtosis, as proved by Theorem \(\ref{theorem2}\). This means there exists a region around 0 in which the density of \(P_m\) is higher while simultaneously decaying slower than \(P_a\), and therefore, we can find a positive \(\epsilon\) such that \(P\{−\epsilon < u^T y_a < \epsilon\} < P\{−\epsilon < u^T y_m < \epsilon\}\). This indicates \(P_m\) is more likely to produce samples close to the optimum than \(P_a\).

The above example shows that \(P_m\) allows producing longer jumps on promising directions that have not been captured by the search distribution. This also accounts for the situation where there exists more promising search directions than the direction vectors used in the search distribution. A concrete example reflecting this situation is the Discus function \(f_{\text{Discus}}\). The Hessian of this problem has two distinct eigenvalues: 2 of multiplicity \(n - 1\) and \(2 \times 10^6\) of multiplicity 1. Thus, it has \(n - 1\) promising directions required to be explored. However, most large-scale variants of CMA-ES only utilize \(m \ll n\) direction vectors and will in no way explore these directions simultaneously.

The use of \(P_m\) will benefit faster exploration on these directions, which explains why MMES performs better than other Gaussian based ESs. In addition, Theorem \(\ref{theorem3}\) states that the decay rate of the density of \(P_m\)
is proportional to the mixing strength. This explains why on \( f_{\text{Discus}} \) MMES works better with a small \( l \) than with a large \( l \).

\( \text{c) Numerical simulations:} \) In the next we give empirical evidences to the above discussion. The aim of the performed experiment is to verify it is the ability of producing long jumps allowing MMES to perform better than other algorithms on \( f_{\text{Discus}} \). To this end, we modify the sampling operation of MMES which we restate below

\[
z = \sqrt{1 - \gamma z_0} + \frac{\gamma}{l} \sum_{j=1}^{l} q_j z_j.
\]

In this experiment, the \( q \) vectors, the mixing components \( z_1, \ldots, z_m \), and the hyper-parameter \( \gamma \) remain unchanged. \( z_0 \) is the \( n \)-dimensional isotropic Gaussian in MMES but here we replace it with two other types of random variable. The first is a random vector with each element drawn from a \( t \)-distribution with 5 degrees of freedom, followed by a proper normalization to have a unit variance. The resultant distribution has the same variance as that of MMES but will possess a slightly larger excess kurtosis than MMES. The second is a random vector drawn from the Rademacher distribution \([12]\).

The Rademacher distribution mimics the statistical property of Gaussian in high-dimensional space but has a negative excess kurtosis. The MMES variants with \( t \)-distribution and Rademacher distribution are denoted by MMES-T and MMES-R, respectively. Intuition suggests that, on \( f_{\text{Discus}} \), due to the different behaviors in producing long jumps, MMES-T will perform better than MMES while MMES-R will perform worse.

Fig. 10 plots for MMES and its two variants the convergence curves on four 1000-dimensional problems. As expected, the \( t \)-distribution leads to a better performance on \( f_{\text{Discus}} \) while the Rademacher distribution degrades the performance significantly. On the contrary, the \( f_{\text{Cigar}} \) problem has only one promising search direction and is less influenced by the distribution type, so the two variants show no difference from MMES. These observations confirm our statement about the relation between distribution type and landscape characteristics. \( f_{\text{Ell}} \) and \( f_{\text{DiffPow}} \) can be regarded as between the above two extreme cases: the Hessian has widely spread eigenvalues. A properly designed ES usually capture some of the directions that contribute mostly to decreasing the objective value. Producing long jumps may degrade the exploitation ability, which illustrates the better performance of MMES-R and the worse performance of MMES-T. Nevertheless, the difference in convergence behaviors of these three competitors is insignificant, probably due to that all their distributions have densities decaying exponentially fast.

\( \text{d) Further discussion:} \) The above experiments show that a small mixing strength enhances the exploration ability by increasing the likelihood of producing long jumps. In fact, similar studies can be found in the field of multimodal optimization. In \([26],[55],[56]\) the authors used long-tailed distributions, such as the \( t \)-distribution and the Cauchy distribution, to prevent from being trapped into local optima. The difficulty they were to solve, albeit in a different context, share an essential similarity to large-scale optimization. Specifically, the difficulty in handling multimodality for a Gaussian based ES (or EDA) comes from the fact that the Hessian of local landscape may be unbounded but the covariance matrix is bounded by design. On the other hand, the difficulty for solving large-scale optimization is due to the Hessian may have many sensitive directions (i.e., the directions of eigenvectors corresponding to smallest eigenvalues) while the covariance matrix has a low-rank structure and hence lacks the ability of capturing all these directions. In a unified view, the above two difficulties can be considered as caused by the inconsistence in characteristics between the problem landscape and the probability distribution, or more concretely, the inconsistence in the principal components of the inverse of Hessian and the covariance matrix. Experiments presented above show that using a small mixing strength efficiently handles this inconsistence issue while causing no significant adverse effect in other situations.

\( \text{E. Practical hints for mixing strength selection} \)

When applying MMES in practice we suggest always choosing a small constant value for the mixing strength, say \( l < 10 \). The benefit, as revealed by the experiments, is threefold: 1) it improves the performance in coping with the inconsistence between the characteristics of problem landscape and probability distribution; 2) it is robust on other types of problems where the aforementioned inconsistence is insignificant; 3) it reduces the computational time. Manually tuning \( l \) for certain problems may be useless, as the improved performance may not compensate for the increased execution time. Thus, it is recommended using MMES with a fix and small \( l \) in all application scenarios.
APPENDIX I  
EXPERIMENT ON THE CEC’2013 TEST SET

To further investigate the performance of MMES in handling complicated problems, we carry out more experiments using the recent CEC’2013 large-scale test set [57].

A. Test Problems

The CEC’2013 test set contains three fully separable problems (f1 to f3), eight partially non-separable problems (f4 to f11), three overlapping problems (f12 to f14), and one fully non-separable problem (f15). Among the partially non-separable problems, f8 to f11 consist of several fully non-separable subcomponents and no correlation exists between different subcomponents. f3 to f7 are respectively identical to f8 to f11 except that one of the subcomponents is made fully separable. Hence, they exhibit more separability than f3 to f11 do. The overlapping problems are constructed in a similar way; the only difference is that the adjacent subcomponents are not disjoint but have overlapping variables. Generally, this test set is more challenging than the CEC’2010 test set in that 1) the subcomponent partitioning is non-uniform and imbalanced and 2) the local landscape near the optimum is asymmetric and irregular.

B. Experimental Settings

The same algorithms in the previous experiment with the CEC’2010 test set, namely DECC-G [19], MA-SW [34], MOS [35], CCPSO2 [36], and DECC-DG [37], are selected as competitors. Their corresponding results are from [35], [39], measured in the standard settings for the CEC competitions [57]. MMES is configured as described in Section IV-A. The final results are obtained from 25 independent runs with a fixed budget of $3 \times 10^6$ function evaluations, which is also according to the guideline of the CEC competitions.

C. Numerical Results

Table X presents the median results of MMES and the five competitors. MMES ranks first and has a significantly better overall performance than the three CC based algorithms, as suggested by the multi-comparison. We see that MMES is robust to the non-separability of the problem, and can produce the best or the second best results once the problems are not fully separable. The two memetic algorithms (i.e., MA-SW and MOS), on the contrary, appear to show clear preference for solving fully separable problems (e.g., f1) or partially separable problems that involve separable subcomponents (e.g., f7 and f8).

The above observations regarding MMES largely coincide with what we have obtained on the CEC’2010 test set. It is probably due to the implicit similarities between the CEC’2010 test set and the 2013 one. In deed, the major distinction between the two sets is the non-uniform and overlapping scheme in partitioning the subcomponent, which is in essence equivalent to an affine transformation on the decision space. Therefore, MMES behaves similarly on these two sets due to its inherent affine invariance.

The CEC’2013 test set also introduces local irregularity and asymmetry to render deceptive landscape for multimodal problems, causing difficulties for algorithms that exploit the search space symmetrically. Nevertheless, their influence on the performance of MMES appears to be marginal. For example, MMES can produce competitive results on all instances of the Ackley’s problem (f3, f6, and f10), regardless of its local irregularity and asymmetry. One possible reason is that the landscape of this problem has a strong global structure while MMES with restarts has enhanced ability in global exploration. The difficulties caused by the local irregularity and asymmetry can also be compensated by the capacity in handling variable correlations. This can be verified by observing that, on the Rastrigin’s problem, MMES performs relatively poor on the separable instance f2 while becoming much better on the partially non-separable instances f5 and f9. We may conclude that the local irregularity and asymmetry are not the crucial factors in determining the hardness of the test problems and their impact on MMES seems to be insignificant.

APPENDIX J  
PRACTICAL APPLICATION ON SPARSE SPECTRAL CLUSTERING

In this section, we test the effectiveness of MMES on sparse spectral clustering, an emerging application in unsupervised learning.

A. Problem Descriptions

Spectral clustering (SC) is a pioneered clustering method that has numerous applications in the field of data mining [58]. It has a easy-to-compute solution and facilitates exploiting the underlying data structure. Given a set of $N_f$-dimensional data points $x_1, x_2, \cdots, x_N$, and a matrix $W \in R^{N_c \times N_c}$ with the entry $W_{i,j}$ measuring some notion of similarity between $x_i$ and $x_j$, SC aims to find a partition of the data points that minimizes the between-cluster similarities and maximizes the within-cluster similarities. The partition result is usually stored in an indicator matrix $U \in R^{N_c \times N_c}$, where $N_c$ is the number of clusters and $U_{i,j}$ indicates the possibility of $x_i$ belonging to the $j$-th cluster. The $i$-th row of $U$ then represents a low-dimensional embedding of $x_i$ and the final partition result can be obtained by performing a simple clustering (e.g., $k$-means) in the embedding space.

In the standard implementation of SC, the matrix $U$ can be obtained by solving $\min_{U^T U = I_{N_c}} Tr[U^T L U]$ where $L \in R^{N_c \times N_c}$ is the normalized Laplacian determined by $W$ and $Tr[.]$ denotes the matrix trace operation. A recent study [59] suggested that, in a perfect clustering, the covariance matrix of the data embeddings is a permutation of the identity matrix, thereby being sparse in nature. This implies that imposing additional sparsity constraint on $UU^T$ may improve the clustering performance, leading to the sparse SC problem in the following form:

$$\min_{U \in R^{N_c \times N_c}} Tr[U^T L U] + \beta |UU^T|_1$$
$$s.t. \ U^T U = I_{N_c},$$

(21)
MMES works in a certain tangent space and 2) the probability manifold, due to the local homeomorphism property. To work appropriately defined on the tangent spaces of the manifold. In solving the sparse SC problem.

\[ \text{MMES} \text{ is still applicable as long as all its operations are} \]

2. Adaption of MMES in Riemannian Settings

Special treatment is required in handling the orthogonal constraint in Eq. (21) since MMES is proposed for unconstrained optimization. One can realize that the sparse SC is actually defined on a Riemannian manifold termed as the Grassmann manifold [60]. On the other hand, as mentioned in Section IV-C, all genetic operations of MMES are linear. This indicates that MMES is still applicable as long as all its operations are appropriately defined on the tangent spaces of the manifold.

In this experiment, we choose the retraction based framework [61] to handle the Grassmann manifold constraint. Retraction is an approximation of the geodesic which maps a matrix from the tangent space to the manifold. With retraction, MMES may operate on the tangent spaces as if on the manifold, due to the local homeomorphism property. To work with retraction, we modify MMES as follows:

- At the iteration \( g \), choose the tangent space corresponding to the mean \( \mathbf{m}(g) \) as the workspace. Sample a subset of direction vectors with the probability distribution \( P_t \) and then parallel translate them to this workspace.
- In the chosen workspace, perform the same sampling and updating operations as in the standard MMES, except for changing the objective function from \( f \) to \( f \circ r \) where \( r \) is the retraction operator.
- At the end of the iteration, retract the new mean \( \mathbf{m}_{t+1} \) onto the manifold to enforce feasibility.

The above modifications make sure that 1) each iteration of MMES works in a certain tangent space and 2) the probability model reconstructed from the direction vectors are well-defined in that space. The specification of the tangent space is to preserve the unbiasedness of ESs. That is, under random selection, the mean will stay unchanged in expectation and the Riemannian MMES will degenerate to a Euclidean one which runs in a fixed tangent space. The Riemannian version of MMES is implemented in the Manopt toolbox [62].

C. Comparative Algorithm

We select the recently proposed Grassmann manifold optimization assisted sparse spectral clustering (GSC) [63] as a competitor. GSC incorporates a Riemannian trust region framework to improve the global exploration ability on the highly non-convex landscape. Each subproblem in GSC is solved by the truncated Newton method; the termination criteria of the inner conjugate gradient iterations are chosen to achieve superlinear convergence. Unlike MMES, GSC is configured as a white-box solver. However, since the objective function of Eq. (21) is not differentiable, the subgradient is supplied instead in calculating the Riemannian gradient. In this work, GSC is also implemented with the Manopt toolbox.

D. Experimental Settings

The experiment is performed on 18 real-world data sets consisting of numeric data, figures, and text. All these sets are collected in the scikit-learn toolbox [64] and are briefly summarized in Table X. We construct the similarity matrix \( W \) using the \( k \)-nearest neighbor graph with \( k = 5 \) and the local similarity is measured using the Gaussian similarity function. The coefficient \( \beta \) in Eq. (21) is usually problem-dependent, so we choose it from a candidate set \( \{0.01, 0.05, 0.001, 0.0005, 0.0001, 0.00005, 0.00001\} \).

The GSC terminates when the number of iterations exceeds 300 or the norm of gradient is smaller than \(10^{-6}\). Other hyperparameters are set according to the literature [63] and...
the default settings of the Manopt toolbox. For MMES, we stop the algorithm when the mutation strength is smaller than $10^{-6}$ or the number of function evaluations exceeds $3 \times 10^5$. Other settings are the same as in previous experiments. Both MMES and GSC are run 20 times independently and the median results are reported.

| Data sets       | Samples ($N_s$) | Features ($N_f$) | Classes ($N_c$) |
|-----------------|-----------------|-----------------|-----------------|
| orlraws10P      | 100             | 10304           | 10              |
| pixraw10P       | 100             | 10000           | 10              |
| Prostate-GE     | 102             | 5966            | 2               |
| TOX-171         | 171             | 5748            | 4               |
| warpAR10P       | 130             | 2400            | 10              |
| Yale            | 165             | 1024            | 15              |
| ALLAML          | 72              | 7129            | 2               |
| warpPIE10P      | 210             | 2420            | 10              |
| GLI-85          | 85              | 22283           | 2               |
| GLOMMA          | 50              | 4434            | 4               |
| leukemia        | 72              | 7070            | 2               |
| lymphoma        | 96              | 4026            | 9               |
| nci9            | 60              | 9712            | 9               |
| SMK-CAN-187     | 187             | 19993           | 2               |
| arcene          | 200             | 10000           | 2               |
| CLL-SUB-111     | 111             | 11340           | 3               |
| colon           | 62              | 2000            | 2               |

E. Numerical Results

The median results of the objective values are reported in Table XII. It is found that on the majority of the test instances MMES is obviously superior to GSC. An interesting observation is that MMES surpasses GSC for large or moderate $\beta$ values while GSC achieves better results only for small $\beta$ values. The reason is that the objective of the sparse SC is a combination of two subproblems, namely 1) the SC problem $\min Tr[U^T LU]$ which is relative easy due to the smoothness and 2) the sparsity satisfaction problem $\min |UU^T|$ which is much harder due to the non-differentiability. When $\beta$ is small enough, the sparse SC problem will degenerate to be smooth and the trust-region framework of GSC is guaranteed to solve the problem efficiently. Contrarily, when $\beta$ is large, the objective is dominated by its non-differentiable part and the use of subgradient may produce a wrong step deviating from the optimal direction in the subproblem. In this case, MMES exhibits better performance since it does not rely on the differentiability of the objective function.
| $\beta$ | MMES  | GSC   | $\beta$ | MMES  | GSC   |
|--------|-------|-------|--------|-------|-------|
| ohwars10P | 0.01  | 1.27018E+00 | Glioma  | 0.01  | 6.82420E-01 |
|         | 0.005 | 7.30588E-01 |        | 0.005 | 4.72111E-01 |
|         | 0.001 | 3.20149E-01 |        | 0.001 | 2.67244E-01 |
|         | 0.0005| 2.65394E-01 |        | 0.0005| 2.38068E-01 |
|         | 0.0001| 2.20797E-01 |        | 0.0001| 2.14111E-01 |
|         | 0.0005| 2.15156E-01 |        | 0.0005| 2.11078E-01 |
|         | 0.0001| 2.10628E-01 |        | 0.0001| 2.08645E-01 |
| pixwars10P | 0.01  | 1.04366E+00 | Leukemia | 0.01  | 4.51614E-01 |
|         | 0.005 | 5.56284E-01 |        | 0.005 | 2.99835E-01 |
|         | 0.001 | 1.20428E-01 |        | 0.001 | 1.01505E-01 |
|         | 0.0005| 7.04549E-02 |        | 0.0005| 6.72590E-02 |
|         | 0.0001| 3.00057E-02 |        | 0.0001| 1.74672E-02 |
|         | 0.0005| 2.50245E-02 |        | 0.0005| 3.35935E-02 |
|         | 0.0001| 2.09593E-02 |        | 0.0001| 3.04703E-02 |
| Prostate-GE | 0.01  | 4.21246E+01 | Lymphoma | 0.01  | 1.83488E+00 |
|         | 0.005 | 2.52531E-01 |        | 0.005 | 1.33992E+00 |
|         | 0.001 | 9.07137E-02 |        | 0.001 | 8.59514E-02 |
|         | 0.0005| 5.58994E-02 |        | 0.0005| 7.93014E-02 |
|         | 0.0001| 2.22608E-02 |        | 0.0001| 7.37019E-02 |
|         | 0.0005| 1.65801E-02 |        | 0.0005| 7.29787E-02 |
|         | 0.0001| 1.19131E-02 |        | 0.0001| 7.23958E-02 |
| TOX-171 | 0.01  | 7.55167E+01 | NCI9    | 0.01  | 2.93831E+00 |
|         | 0.005 | 5.01975E+00 |        | 0.005 | 3.03356E+00 |
|         | 0.001 | 1.07139E+00 |        | 0.001 | 1.25723E+01 |
|         | 0.0005| 9.77663E+01 |        | 0.0005| 9.39594E-02 |
|         | 0.0001| 8.96071E+01 |        | 0.0001| 4.09850E-02 |
|         | 0.0005| 8.85327E+01 |        | 0.0005| 3.07207E-02 |
|         | 0.0001| 8.76602E+01 |        | 0.0001| 2.21933E-02 |
| Yale | 0.01  | 4.15590E+00 | Arcene  | 0.01  | 3.38271E+01 |
|         | 0.005 | 3.16563E+00 |        | 0.005 | 1.96669E+00 |
|         | 0.001 | 2.15375E+00 |        | 0.001 | 8.17728E-02 |
|         | 0.0005| 2.01013E+00 |        | 0.0005| 7.81464E-02 |
|         | 0.0001| 1.88850E+00 |        | 0.0001| 1.28815E-02 |
|         | 0.0005| 1.87310E+00 |        | 0.0005| 6.07431E-03 |
|         | 0.0001| 1.86068E+00 |        | 0.0001| 1.29089E-03 |
| ALLAML | 0.01  | 4.72313E+01 | Carcinom | 0.01  | 2.28349E+00 |
|         | 0.005 | 3.60565E+00 |        | 0.005 | 1.60812E+00 |
|         | 0.001 | 1.10831E+01 |        | 0.001 | 8.36831E-01 |
|         | 0.0005| 7.57444E+02 |        | 0.0005| 7.34524E-01 |
|         | 0.0001| 4.57507E+02 |        | 0.0001| 6.46545E-01 |
|         | 0.0005| 4.19256E+02 |        | 0.0005| 6.35302E-01 |
|         | 0.0001| 3.88193E+02 |        | 0.0001| 6.25711E+02 |
| warpFR10P | 0.01  | 1.24388E+00 | CLL-SUB-111 | 0.01  | 5.87994E-01 |
|         | 0.005 | 8.35624E-01 |        | 0.005 | 3.96819E+00 |
|         | 0.001 | 2.87847E-01 |        | 0.001 | 1.77443E-01 |
|         | 0.0005| 1.82621E-01 |        | 0.0005| 5.62751E-02 |
|         | 0.0001| 8.85037E-02 |        | 0.0001| 1.29412E-02 |
|         | 0.0005| 7.24649E-02 |        | 0.0005| 7.49887E-03 |
|         | 0.0001| 6.00220E-02 |        | 0.0001| 3.12265E-03 |
| GLI-85 | 0.01  | 4.18131E+00 | Colon   | 0.01  | 5.32337E-01 |
|         | 0.005 | 3.20914E-01 |        | 0.005 | 3.03551E-01 |
|         | 0.001 | 1.25968E-01 |        | 0.001 | 8.59321E-02 |
|         | 0.0005| 8.47801E-02 |        | 0.0005| 8.51601E-02 |
|         | 0.0001| 4.76805E-02 |        | 0.0001| 3.15915E-02 |
|         | 0.0005| 4.28831E-02 |        | 0.0005| 2.84815E-02 |
|         | 0.0001| 3.90254E-02 |        | 0.0001| 2.59862E-02 |

* f o f

38 / 16 / 9
44 / 12 / 7