

Covariance Matrix Adaptation Evolution Strategy without a matrix

Jarosław Arabas
Warsaw University of Technology
Institute of Computer Science
Warsaw, Poland
jaroslaw.arabas@pw.edu.pl

Adam Stelmaszczyk
Warsaw University of Technology
Institute of Computer Science
Warsaw, Poland
adam.stelmaszczyk@pw.edu.pl

Eryk Warchulski
Warsaw University of Technology
Institute of Computer Science
Warsaw, Poland
eryk.warchulski@pw.edu.pl

Dariusz Jagodziński
Warsaw University of Technology
Institute of Computer Science
Warsaw, Poland
dariusz.jagodzinski@pw.edu.pl

Rafał Biedrzycki
Warsaw University of Technology
Institute of Computer Science
Warsaw, Poland
rafal.biedrzycki@pw.edu.pl

ABSTRACT

Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is a highly effective optimization technique. A primary challenge when applying CMA-ES in high dimensionality is sampling from a multivariate normal distribution with an arbitrary covariance matrix, which involves its decomposition. The cubic complexity of this process is the main obstacle to applying CMA-ES in high-dimensional spaces.

We introduce a version of CMA-ES that uses no covariance matrix at all. In the proposed matrix-free CMA-ES, an archive stores the vectors of differences between individuals and the midpoint, normalized by the step size. New individuals are generated as the weighted combinations of the vectors from the archive. We prove that the probability distribution of individuals generated by the proposed method is identical to that of the standard CMA-ES.

Experimental results show that reducing the archive size to store only a fixed number of the most recent populations is sufficient, without compromising optimization efficiency. The matrix-free and matrix-based CMA-ES achieve comparable results on the quadratic function when the step-size adaptation is turned off. When coupled with the step-size adaptation method, the matrix-free CMA-ES converges faster than the matrix-based, and usually yields the results of a comparable or superior quality, according to the results obtained for the CEC'2017 benchmark suite.

Presented approach simplifies the algorithm, offers a novel perspective on covariance matrix adaptation, and serves as a stepping stone toward even more efficient methods.

KEYWORDS

Covariance Matrix Adaptation Evolution Strategy, step-size adaptation, black-box optimization

ACM Reference Format:

Jarosław Arabas, Adam Stelmaszczyk, Eryk Warchulski, Dariusz Jagodziński, and Rafał Biedrzycki. 2025. Covariance Matrix Adaptation Evolution Strategy without a matrix. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2025)*. ACM, New York, NY, USA, 10 pages.

1 INTRODUCTION

For continuous optimization problems, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [6] stands as one of the leading methods within the stochastic algorithms family. CMA-ES adapts the covariance matrix of the search distribution, allowing it to learn and exploit the underlying structure of the objective function. This capability to navigate complex optimization landscapes has garnered significant attention across various domains, including scientific research, engineering and industrial applications.

Each iteration of CMA-ES involves generating new candidate solutions from a multivariate normal distribution. This distribution is dynamically updated based on the performance of the generated solutions. However, sampling from this distribution necessitates the decomposition of the covariance matrix, resulting in a computational complexity of $O(n^3)$ per iteration, where n is the number of dimensions. This cubic scaling presents a significant bottleneck when applying CMA-ES to high-dimensional problems involving thousands of variables. To address this computational challenge, several approaches have been proposed.

In [13] it was observed that the overall cost of running CMA-ES can be reduced by decomposing the covariance matrix not in every but, for example, every tenth iteration. According to the authors, this modification introduces only a small deterioration of the overall optimization efficiency. In another approach [10], the authors propose to change the cumulative step-size adaption (CSA) mechanism in the CMA-ES and replace the inverse of the square root of the covariance matrix by the inverse of the triangular Cholesky factor. Despite these reductions, the matrix operations used in the CMA-ES algorithms for generating basis vectors or σ step adaptation, remain expensive.

Another approach to reducing the computational effort is to simplify the covariance matrix into a form that is more convenient for processing. In the MVA-ES method [12], the covariance matrix is defined as the sum of the identity matrix and the outer product of the cumulative midpoint shift. As a result, the process of generating difference vectors does not require matrix decomposition. While the MVA-ES method achieves results comparable to those of CMA-ES, this is true only when there is a single dominant direction for the adaptation of distribution parameters. Consequently, MVA-ES performs well for optimizing unimodal functions with a clear

direction of steepest descent in the target values. However, for multimodal functions, the performance of MVA-ES is significantly worse than that of the baseline CMA-ES.

A similar approach to simplifying the covariance matrix, as in MVA-ES, is employed in the R1-NES algorithm [14]. In this method, the covariance matrix is defined as the sum of the identity matrix and the outer product of a vector that represents the weighted average of recent gradients of the objective function. Consequently, the covariance matrix will have at most one eigenvector with a large eigenvalue. This limitation prevents the proposed procedure from effectively approximating the actual shape of the objective function. The study's results indicate that the R1-NES method is only effective for specific types of objective function shapes.

An alternative approach to reducing computational effort is to redefine the method for updating the covariance matrix. The MA-ES algorithm [3] eliminates the need for decomposition by approximating the square root of the covariance matrix, which is required to generate the population. Instead of accumulating midpoint shift vectors to update the covariance matrix, the method accumulates multivariate normal vectors that, after transformation into the search space, produced the best-fit points. These assumptions allow the algorithm to bypass matrix factorization without significantly compromising optimization efficiency. However, the proposed formula for the square root of the covariance matrix requires matrix multiplication in each generation. By employing the Coppersmith-Winograd matrix multiplication method, the computational complexity of one iteration of the MA-ES algorithm can be reduced to $O(n^{2.37})$.

The LM-MA-ES method [11] is a modification of the MA-ES algorithm designed to further reduce computational effort. Instead of adapting a matrix that approximates the square root of the covariance matrix for subsequent multiplication by a realization of a unit normal random variable, the authors propose directly estimating and adapting the result of this multiplication. This approach reduces the computational complexity of the method to $O(n \log(n))$. However, since LM-MA-ES approximates the basis vectors produced by the MA-ES algorithm, the resulting covariance matrix of the generated points may differ from that of the original CMA-ES algorithm.

A different approach to simplifying CMA-ES is presented in [1], where the authors propose an algorithm called the Differential Evolution Strategy. Rather than using Gaussian mutation to generate new individuals, this method is based on the differential mutation. New individuals are generated by adding difference vectors to the population midpoint, and each difference vector is a weighted combination of three elements: the difference vector between two randomly selected points from a randomly chosen past population, a cumulative midpoint shift observed in one randomly selected past generation, and a midpoint shift from a past population. These three vectors are combined using weights that are random values drawn from the standard normal distribution. The authors acknowledge that the resulting probability distribution differs from that of CMA-ES. In contrast, in this work, we propose a method whose distribution is equivalent to that of CMA-ES.

In most of the approaches mentioned above, the authors focus on addressing the consequences of aggregating the search history into a matrix. Rather than mitigating the consequences, it may be

more effective to eliminate the root cause of the problem. To this end, we propose a solution where no matrix is needed at all.

We utilize an archive of populations to define new points by combining individuals with cumulated midpoint shift vectors, using randomly assigned weights for these combinations. The probability distribution of the generated points in our approach matches that of CMA-ES, i.e., the resulting distribution is multivariate normal with the same mean vector and covariance matrix as in the case of CMA-ES. It is important to emphasize that our motivation was to develop a method that is compliant with, rather than superior in terms of convergence, to the matrix-based CMA-ES.

The resulting algorithm cannot be coupled with the Cumulative Step-Size Adaptation (CSA) rule for adapting the mutation step size. Therefore, we employ the Previous Population Midpoint Fitness (PPMF) rule [17] for step-size adaptation. When used alongside the matrix-based CMA-ES, PPMF has demonstrated competitiveness with CSA, particularly in optimization problems with a high number of dimensions.

The paper is organized as follows. In Section 2, we introduce the Vanilla CMA-ES method, outlining its core functionality. Section 3 presents the matrix-free CMA-ES (MF-CMA-ES), where we discuss the modification to CMA-ES that eliminates the need for matrix decomposition while maintaining the same statistical properties. Section 4 compares the performance of matrix-based and matrix-free CMA-ES under conditions where step-size adaptation is not applied. Section 5 extends the discussion by including step-size adaptation. Section 6 provides concluding remarks and suggestions for future work.

2 VANILLA CMA-ES

The starting point for our considerations is the vanilla CMA-ES method [6] — see Fig. 1. CMA-ES operates by maintaining a multivariate normal distribution to sample candidate solutions during the optimization process. This distribution is parameterized by three components: the mean vector $\mathbf{m}^{(t)}$, the covariance matrix $\mathbf{C}^{(t)}$, and the step-size $\sigma^{(t)}$, where t represents the iteration index. Initially, $\mathbf{C}^{(1)}$ is set to the identity matrix \mathbf{I} , while $\mathbf{m}^{(1)}$ and $\sigma^{(1)}$ are specified by the user.

At each iteration, the covariance matrix $\mathbf{C}^{(t)}$ is decomposed into its lower triangular form, $\mathbf{L}^{(t)}$, such that $\mathbf{L}^{(t)}(\mathbf{L}^{(t)})^T = \mathbf{C}^{(t)}$. The recommended approach for decomposing the covariance matrix is eigendecomposition [6] since it works properly even when the matrix $\mathbf{C}^{(t)}$ is ill-conditioned.

Using $\mathbf{L}^{(t)}$, the algorithm generates a population of λ candidate solutions by first sampling λ independent standard normal vectors $\mathbf{z}_i^{(t)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. These vectors are then transformed into difference vectors $\mathbf{d}_i^{(t)} = \mathbf{L}^{(t)}\mathbf{z}_i^{(t)}$. Finally, the candidate solutions, or individuals, are computed as $\mathbf{x}_i^{(t)} = \mathbf{m}^{(t)} + \sigma^{(t)}\mathbf{d}_i^{(t)}$.

Once the population is created, the candidates are sorted based on their fitness and the best μ individuals are selected. The difference vectors $\mathbf{d}_i^{(t)}$ and normal vectors $\mathbf{z}_i^{(t)}$ are reordered in compliance with $\mathbf{x}_i^{(t)}$. The difference vectors, which correspond to μ point with the best fitness, are weighted and aggregated to update the parameters of the distribution for the next iteration. Specifically, the mean vector $\mathbf{m}^{(t)}$ is adjusted using $\Delta^{(t)}$ — a weighted sum of

```

1:  $\mathbf{p}_c^{(1)} \leftarrow \mathbf{0}, \mathbf{C}^{(1)} \leftarrow \mathbf{I}$ 
2:  $t \leftarrow 1$ 
3: initialize( $\mathbf{m}^{(1)}, \sigma^{(1)}$ )
4: while !stop do
5:   decompose  $\mathbf{C}^{(t)} : (\mathbf{L}^{(t)})(\mathbf{L}^{(t)})^\top = \mathbf{C}^{(t)}$ 
6:   for  $i = 1$  to  $\lambda$  do
7:      $\mathbf{z}_i^{(t)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
8:      $\mathbf{d}_i^{(t)} = \mathbf{L}^{(t)} \mathbf{z}_i^{(t)}$ 
9:      $\mathbf{x}_i^{(t)} = \mathbf{m}^{(t)} + \sigma^{(t)} \mathbf{d}_i^{(t)}$ 
10:   end for
11:   evaluate ( $X^{(t)}$ )
12:   sort ( $X^{(t)}$ ) according to fitness
13:   reorder ( $D^{(t)}$ ) and ( $Z^{(t)}$ ) according to ( $X^{(t)}$ )
14:    $\Delta^{(t)} \leftarrow \sum_{i=1}^{\mu} w_i \mathbf{d}_i^{(t)}$ 
15:    $\mathbf{m}^{(t+1)} \leftarrow \mathbf{m}^{(t)} + \sigma^{(t)} \Delta^{(t)}$ 
16:    $\mathbf{p}_c^{(t+1)} \leftarrow (1 - c_c) \mathbf{p}_c^{(t)} + \sqrt{\mu_{\text{eff}} c_c (2 - c_c)} \cdot \Delta^{(t)}$ 
17:    $\mathbf{C}_1^{(t)} = \mathbf{p}_c^{(t)} (\mathbf{p}_c^{(t)})^\top$ 
18:    $\mathbf{C}_\mu^{(t)} = \sum_{i=1}^{\mu} w_i \mathbf{d}_i^{(t)} (\mathbf{d}_i^{(t)})^\top$ 
19:    $\mathbf{C}^{(t+1)} \leftarrow (1 - c_{cov}) \mathbf{C}^{(t)} + c_1 \mathbf{C}_1^{(t)} + c_\mu \mathbf{C}_\mu^{(t)}$ 
20:    $\sigma^{(t+1)} \leftarrow \text{update}(\sigma^{(t)})$ 
21:    $t \leftarrow t + 1$ 
22: end while

```

Figure 1: Outline of the matrix-based CMA-ES

the top μ difference vectors, scaled by the step-size $\sigma^{(t)}$. Value of $\Delta^{(t)}$ is accumulated into the so called evolution path vector $\mathbf{p}_c^{(t+1)}$. Value of μ_{eff} is defined as $\mu_{\text{eff}} = \left(\sum_{i=1}^{\mu} (w_i)^2 \right)^{-1}$.

The covariance matrix $\mathbf{C}^{(t)}$ is updated using a weighted combination of three components: the existing covariance matrix, a rank-1 update matrix $\mathbf{C}_1^{(t)}$ derived from the evolution path $\mathbf{p}_c^{(t)}$, and a rank- μ update matrix $\mathbf{C}_\mu^{(t)}$ formed from the top μ difference vectors.

The update rule for $\mathbf{C}^{(t)}$ is controlled by coefficients c_1 , c_μ , and c_{cov} , which satisfy the relations $c_{cov} = c_1 + c_\mu$ and $0 \leq c_1, c_\mu, c_{cov} \leq 1$.

3 MATRIX-FREE CMA-ES (MF-CMA-ES)

According to the formula in Fig. 1, line 19, the covariance matrix $\mathbf{C}^{(t)}$ memorizes the history of populations of difference vectors $\mathbf{d}_i^{(t)}$ as well as the history of the difference vectors mean $\Delta^{(t)}$. The formula has the form of a recursive expression that implements the exponential smoothing. Following the idea presented in [1], we substitute the recursive formula that defines the covariance matrix with a non-recursive version.

OBSERVATION 1. *The covariance matrix update rule (Fig. 1, line 19) can be expressed as:*

$$\mathbf{C}^{(t+1)} = \sum_{\tau=1}^t (1 - c_{cov})^{t-\tau} \left(c_1 \mathbf{C}_1^{(\tau)} + c_\mu \mathbf{C}_\mu^{(\tau)} \right) + (1 - c_{cov})^t \mathbf{I} \quad (1)$$

where \mathbf{C}_1 and \mathbf{C}_μ are defined in Fig. 1, lines 17 and 18, respectively.

Next, we formulate the following theorem.

THEOREM 1. *Consider the random vector:*

$$\delta^{(t+1)} = \sum_{\tau=1}^t (1 - c_{cov})^{\frac{t-\tau}{2}} \left(c_\mu \sum_{j=1}^{\mu} w_j \mathbf{d}_j^{(\tau)} \mathcal{N}(0, 1) + c_1 \mathbf{p}_c^{(\tau)} \mathcal{N}(0, 1) \right) + (1 - c_{cov})^{\frac{t}{2}} \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (2)$$

where $\mathbf{d}_j^{(t)}$ and $\mathbf{p}_c^{(t)}$ are defined in Fig. 1, lines 8 and 16, respectively. Vectors $\mathbf{d}_j^{(t)}$ are ordered according to the fitness of their corresponding points $\mathbf{x}_j^{(t)}$. Symbols $\mathcal{N}(0, 1)$ and $\mathcal{N}(\mathbf{0}, \mathbf{I})$ stand for the mutually independent standard normal variates in one and in n dimensions, respectively.

Probability distribution of the vector $\delta^{(t+1)}$ is Gaussian with the covariance matrix equal $\Sigma[\delta^{(t+1)}] = \mathbf{C}^{(t+1)}$, where $\mathbf{C}^{(t+1)}$ is defined by the CMA-ES algorithm (Fig. 1, line 19).

Sketch of proof: The vector $\delta^{(t+1)}$ is a linear combination of mutually independent multivariate normal variables, therefore it is multivariate normal.

Expected values of summands defining $\delta^{(t+1)}$ equal

$$E \left[\mathbf{d}_j^{(\tau)} \mathcal{N}(0, 1) \right] = \mathbf{0} \quad (3)$$

$$E \left[\mathbf{p}_c^{(\tau)} \mathcal{N}(0, 1) \right] = \mathbf{0} \quad (4)$$

$$E \left[(1 - c_{cov})^{\frac{t}{2}} \mathcal{N}(\mathbf{0}, \mathbf{I}) \right] = \mathbf{0} \quad (5)$$

therefore

$$E \left[\delta^{(t+1)} \right] = \mathbf{0} \quad (6)$$

Note that for a scalar a , a vector \mathbf{d} , and a standard normal variate $r \sim \mathcal{N}(0, 1)$, the mean and covariance matrix of the vector $a\mathbf{d}$ is defined as

$$E[a\mathbf{d}] = E[r] \cdot E[\mathbf{d}] = \mathbf{0} \quad (7)$$

$$\Sigma[a\mathbf{d}] = E[(a\mathbf{d})(a\mathbf{d})^\top] = a^2 (\mathbf{d})(\mathbf{d})^\top \quad (8)$$

For a vector \mathbf{d} , a standard normal variate $r \sim \mathcal{N}(0, 1)$, and a standard normal multivariate $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, it holds upon their mutual independence

$$E[r\mathbf{v}^\top] = \mathbf{0} \quad (9)$$

$$E[(r\mathbf{d})\mathbf{v}^\top] = \mathbf{0} \quad (10)$$

If $\mathbf{d}_1, \mathbf{d}_2$ are vectors and $r_1, r_2 \sim \mathcal{N}(0, 1)$ are independent standard normal variates then

$$E[r_1 r_2] = 0 \quad (11)$$

$$E[(r_1 \mathbf{d}_1)(r_2 \mathbf{d}_2)^\top] = \mathbf{0} \quad (12)$$

Since the expectation vector is zero, the covariance matrix of the vector δ^{t+1} is given by:

$$\Sigma[\delta^{(t+1)}] = E[\delta^{(t+1)}(\delta^{(t+1)})^\top] \quad (13)$$

Bearing in mind (8) – (12) we get

$$\Sigma[\delta^{(t+1)}] = \sum_{\tau=1}^t (1 - c_{cov})^{t-\tau} \left(c_\mu \sum_{j=1}^{\mu} w_j \mathbf{d}_j^{(\tau)} (\mathbf{d}_j^{(\tau)})^\top + c_1 \mathbf{p}_c^{(\tau)} (\mathbf{p}_c^{(\tau)})^\top \right) + (1 - c_{cov})^t \mathbf{I} \quad (14)$$

Together with Observation 1, this proves Theorem 1. \square

Theorem 1 allows for the generation of new difference vectors without requiring covariance matrix factorization. Instead, new difference vectors can be defined as a randomized weighted combination of past difference vectors.

Note that the contribution of difference vectors, which have been generated τ iterations before the current one, diminishes by a factor of $(1 - c_{cov})^{\tau/2}$. Consequently, it is sufficient to consider only the difference vectors from the most recent h populations, without incurring a significant difference in the resulting covariance matrix, compared to using all previous difference vectors. Therefore, formula (2) can be approximated as:

$$\delta_h^t \approx \sum_{\tau=t-h+1}^t (1 - c_{cov})^{\frac{t-\tau}{2}} \left(c_{\mu}^{\frac{1}{2}} \sum_{j=1}^{\mu} w_j^{\frac{1}{2}} \mathbf{d}_j^{(\tau)} \mathcal{N}(0, 1) + c_1^{\frac{1}{2}} \mathbf{p}_c^{(\tau)} \mathcal{N}(0, 1) \right) + (1 - c_{cov})^{\frac{t}{2}} \mathcal{N}(\mathbf{0}, \mathbf{I}) \quad (15)$$

The value of h will be referred to as the *history window size*.

Leveraging this approach, we propose the Matrix-free CMA-ES (MF-CMA-ES) algorithm (see Figure 2) that adapts the probability distribution to generate difference vectors using a Gaussian multivariate distribution.

```

1:  $\mathbf{p}_c^{(1)} \leftarrow \mathbf{0}$ 
2:  $t \leftarrow 1$ 
3: initialize( $\mathbf{m}^{(1)}, \sigma^{(1)}$ )
4: while !stop do
5:   for  $i = 1$  to  $\lambda$  do
6:     generate  $\mathbf{d}_i^{(t)} \sim \delta_h^t$  (see (15))
7:      $\mathbf{x}_i^{(t)} = \mathbf{m}^{(t)} + \sigma^{(t)} \mathbf{d}_i^{(t)}$ 
8:   end for
9:   evaluate ( $X^{(t)}$ )
10:  sort ( $X^{(t)}$ ) according to fitness
11:  reorder ( $D^{(t)}$ ) according to ( $X^{(t)}$ )
12:   $\Delta^{(t)} \leftarrow \sum_{i=1}^{\mu} w_i \mathbf{d}_i^{(t)}$ 
13:   $\mathbf{m}^{(t+1)} \leftarrow \mathbf{m}^{(t)} + \sigma^{(t)} \Delta^{(t)}$ 
14:   $\mathbf{p}_c^{(t+1)} \leftarrow (1 - c_c) \mathbf{p}_c^{(t)} + \sqrt{\mu_{\text{eff}} c_c (2 - c_c)} \cdot \Delta^{(t)}$ 
15:  store  $\{\mathbf{d}_1^{(t)}, \dots, \mathbf{d}_{\mu}^{(t)}, \mathbf{p}_c^{(t)}\}$  into archive
16:   $\sigma^{(t+1)} \leftarrow \text{update}(\sigma^{(t)})$ 
17:   $t \leftarrow t + 1$ 
18: end while

```

Figure 2: Outline of the matrix-free CMA-ES (MF-CMA-ES)

MF-CMA-ES does not need matrix decomposition. To generate a population, it requires drawing $\lambda(\mu + 1)h + \lambda n$ samples from the standard normal distribution, as can be seen in formula (15).

4 COMPARISON OF MATRIX-BASED AND MATRIX-FREE CMA-ES WITHOUT STEP-SIZE ADAPTATION

In this section, we compare matrix-based and matrix-free covariance matrix adaptation rule, therefore, we turn off the step-size

adaptation procedure, setting $\sigma = 1$ in all iterations of both compared algorithms. The parameters of both algorithms are set to values recommended in [7], with two exceptions: the weights were equal to $1/\mu$ and the population size was $\lambda = 4n$, following [3]. For the comparison we use the quadratic fitness function [8]:

$$q(\mathbf{x}) = \sum_{i=1}^n 10^{6 \frac{i-1}{n-1}} x_i^2 \quad (16)$$

4.1 Analysis of the convergence curves

We report the convergence curves of both methods, assuming that both methods are allowed the same budget of fitness evaluations. The convergence curves are averaged over 30 independent runs of compared algorithms. The starting points were randomly generated with a uniform distribution from the range $[-0.2, 0.8]^n$. Figure 3 presents the convergence curves for the dimensionality $n = 30$. In the case of MF-CMA-ES, we report convergence curves for the following history window sizes: $h \in \{10, 30, 60, 80\}$.

Depending on the history window size h , the dynamics of MF-CMA-ES becomes similar or different than of vanilla CMA-ES. When h is small, the convergence curve initially goes down faster, but then it stagnates at a certain level. For sufficiently large h , MF-CMA-ES converges linearly. The convergence is faster than that of CMA-ES for smaller h . When the history window size grows, the convergence rate stabilizes at a level that is comparable to the one of CMA-ES.

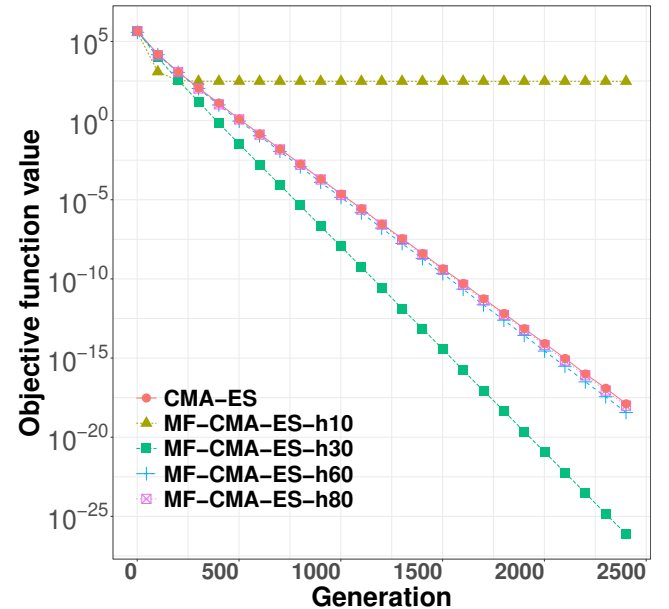


Figure 3: Averaged convergence curves obtained on 30 independent runs of CMA-ES and MF-CMA-ES with different history window sizes on function (16).

4.2 Dynamics of the covariance matrix eigenvalues

An important feature of CMA-ES is its contour-fitting property. After a certain stabilization period of the point-generating distribution, the covariance matrix maintained by CMA-ES becomes proportional to the inverse Hessian of the fitness function.

Following the methodology from [8], in each iteration of both compared algorithms, we computed the eigenvalues of the empirical covariance matrix of the generated difference vectors (MF-CMA-ES) and of the covariance matrix $C^{(t)}$ (vanilla CMA-ES). Figure 4 shows the dynamics of the eigenvalues from a single run of both methods. For MF-CMA-ES, we demonstrate two different history window sizes: $h = 60$ and $h = 10$. The dimension was set to $n = 30$.

The eigenvalues dynamics for the vanilla CMA-ES and the MF-CMA-ES with $h = 60$ are similar, indicating that both methods generate comparable point distributions.

For $h = 10$, the algorithm fails to capture the desired eigenvalues dynamics. The high condition number of the covariance matrix suggests that the population is confined to a low-dimensional subspace, which becomes a fundamental obstacle to reaching the optimum and explains the poor convergence observed for $h = 10$.

5 MATRIX-FREE CMA-ES WITH STEP-SIZE ADAPTATION

For CMA-ES it has been observed that coupling the covariance matrix adaptation procedure with an additional step-size adaptation mechanism can significantly improve the global optimization efficiency of the algorithm. A Cumulative Step Adaptation (CSA) rule [6] is an efficient and widely accepted method to address this problem. Unfortunately, the matrix-free CMA-ES cannot be coupled with CSA, as CSA requires the eigendecomposition of the covariance matrix to compute its inverse square root. Therefore, we decided to use the Previous Population Midpoint Fitness (PPMF) method [17] to control the step size in the matrix-free CMA-ES.

5.1 Cumulative Step Adaptation (CSA)

The CSA procedure (Figure 5) analyzes the set of $z_i^{(t)}$ vectors that are ordered in the same sequence as their corresponding points $x_i^{(t)}$ that are sorted with respect to their fitness. The procedure tracks the value of the vector $p_\sigma^{(t)}$, which is changed in each iteration by accumulating the mean value of μ vectors $z_i^{(t)}$ corresponding to the μ best individuals generated in the current generation. If the fitness function is flat, a random walk of the population midpoint will be observed, and the norm of p_σ will be χ distributed with n degrees of freedom. Then the expectation of the logarithm of step-size change will be zero. When the midpoint m^t is located in a significant distance from the local optimum, consecutive shifts of the midpoint will be positively correlated, which will increase $\|p_\sigma\|$ and the step size will also increase. Contrarily, if the midpoint m^t is located close to the minimum, then, on the average, shorter difference vectors will be correlated with better-fit points. In effect, $\|p_\sigma\|$ and the step size will be decreased.

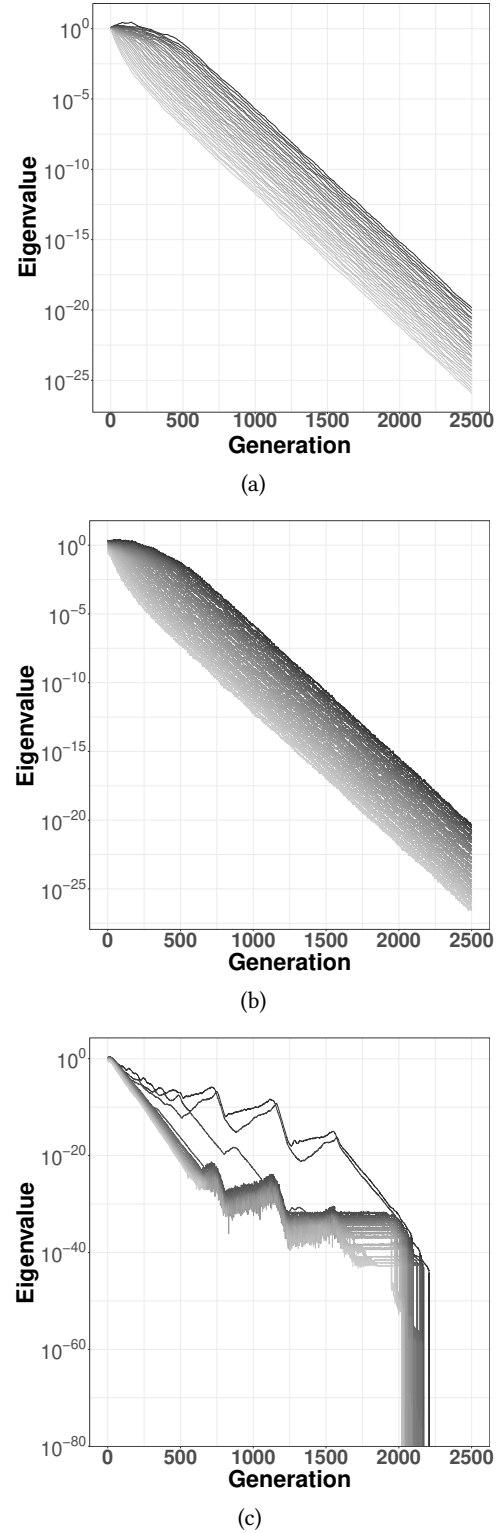


Figure 4: Dynamics of eigenvalues for the vanilla CMA-ES (a) and for the MF-CMA-ES with the window size $h = 60$ (b) and $h = 10$ (c)

- 1: $\Delta_\sigma^{(t)} \leftarrow \sum_{i=1}^{\mu} w_i \mathbf{z}_i^{(t)}$
- 2: $\mathbf{p}_\sigma^{(t+1)} \leftarrow (1 - c_\sigma) \mathbf{p}_\sigma^{(t)} + \sqrt{\mu c_\sigma (2 - c_\sigma)} \cdot \Delta_\sigma^{(t)}$
- 3: $\sigma^{(t+1)} \leftarrow \sigma^{(t)} \exp \left(\frac{c_\sigma}{d_\sigma} \left(\frac{\|\mathbf{p}_\sigma^{(t+1)}\|}{E\|\mathbf{N}(0, \mathbf{I})\|} - 1 \right) \right)$

Figure 5: Outline of the Cumulative Step Adaptation (CSA)

5.2 Previous Point Midpoint Fitness (PPMF)

PPMF is an alternative step-size adaptation method inspired by the one-fifth rule. Pseudocode of the method is given in Figure 6. In each iteration, fitness of the previous population midpoint is evaluated by the fitness function denoted as $q: \mathbb{R}^n \rightarrow \mathbb{R}$. The fitness of the points of the current population is compared to the fitness of the previous population midpoint. If the number of points superior to the midpoint is a small percentage of the whole population then the step-size is reduced. The step-size is increased when the proportion of superior points exceeds a threshold value θ , a PPMF hyperparameter. According to the results reported in [17], the method achieves good efficiency with $\theta = 0.2$. We used this value as is, without any further hyperparameter tuning.

- 1: $\bar{\mathbf{m}}^{(t-1)} \leftarrow \frac{1}{\lambda} \sum_{i=1}^{\lambda} \mathbf{X}_i^{(t-1)}$
- 2: evaluate $(\bar{\mathbf{m}}^{(t-1)})$
- 3: $p_s^{(t)} \leftarrow \left| \{i : q(\mathbf{x}_i^t) < q(\bar{\mathbf{m}}^{(t-1)})\} \right| / \lambda$
- 4: $\sigma^{(t+1)} \leftarrow \sigma^{(t)} \exp \left(\frac{1}{d_\sigma} \cdot \frac{p_s^{(t)} - \theta}{1 - \theta} \right)$

Figure 6: Outline of the Previous Population Midpoint Fitness (PPMF) step-size adaptation

5.3 Benchmarking matrix-free and matrix-based CMA-ES with step-size adaptation

We evaluated the efficiency of the newly introduced MF-CMA-ES algorithm using the CEC'2017 benchmark suite [2]. In line with the CEC'2017 benchmarking guidelines, 51 independent runs were conducted for each optimization problem. For each run, the fitness evaluation budget was set to $\text{MaxFEs} = 10000n$.

In the experiments, MF-CMA-ES was coupled with the PPMF step size adaptation method. For comparison, we used two versions of vanilla CMA-ES, coupled with either CSA or PPMF step-size control. In all optimization problems from the CEC'2017 suite, all common parameters of considered algorithms were set to identical values.

The history window size depended on the dimension number according to the formula $h = 20 + 1.4n$. We derived this heuristic by determining, for each value of n , the smallest values of h that resulted in the convergence rate of MF-CMA-ES similar to that of vanilla CMA-ES on the function (16), assuming constant σ for both methods.

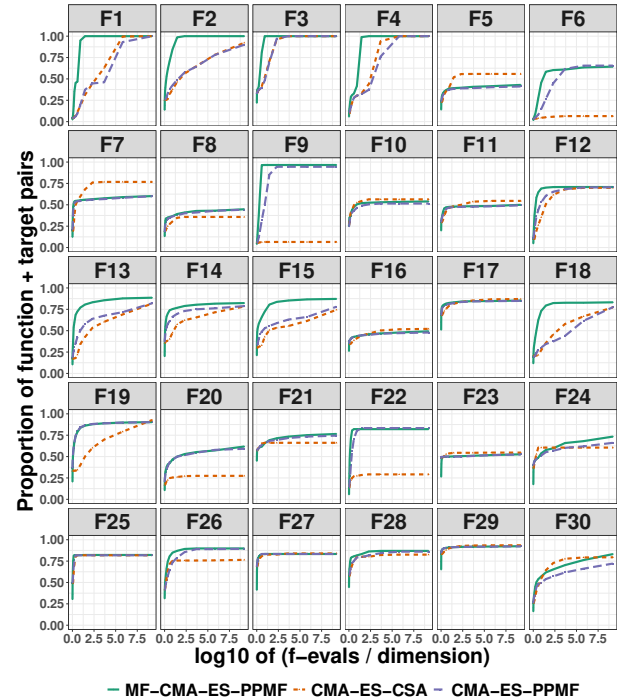
Tables 1 – 3 present the summary of results obtained for each problem from the CEC'2017 suite. The results show the error, i.e. the

difference between the solution objective function value yielded by the algorithm and the objective function value at the global optimum. For each optimization problem, the mean and standard deviation of 51 smallest error values achieved in each run after exceeding the fitness evaluation budget are reported.

To compare the optimization algorithms, we used statistical tests recommended by [4]. First, we applied the Quade test to determine whether a significant performance difference existed between at least one pair of algorithms. The p-values of the null hypothesis were equal 0.0003 for $n = 10$, 0.08 for $n = 30$, and 0.008 for $n = 50$. Hence, for $n = 30$, the null hypothesis cannot be rejected, indicating no significant difference between the algorithms.

The Quade test provides insights into the ranking of the compared methods. Based on this ranking, in $n = 10$ dimensions, MF-CMA-ES outperformed both versions of matrix-based CMA-ES. However, for $n = 50$, CMA-ES-CSA was superior to MF-CMA-ES, while MF-CMA-ES outperformed CMA-ES-PPMF.

Next, we compared MF-CMA-ES with best-performing CMA-ES variant and analyzed the Holm-corrected p-values. The results were produced using the software provided by the authors of [5]. The Holm-corrected p-values were 0.034 for $n = 10$ and 0.013 for $n = 50$. Assuming a significance level of $\alpha = 0.05$, the null hypothesis was rejected for $n = 10$, indicating the superiority of MF-CMA-ES, and for $n = 50$ indicating the superiority of CMA-ES-CSA.

**Figure 7: ECDF curves obtained by MF-CMA-ES (with PPMF) and two versions of CMA-ES (with CSA and PPMF) for CEC'2017 optimization problems in 10 dimensions**

To facilitate a more detailed comparison between algorithms, we analyze their dynamics using Empirical Cumulative Distribution Functions (ECDFs) as described in [9]. For each problem and dimension, we establish a logarithmic scale of target precision values spanning the range of best and worst results achieved by any algorithm. This scale is defined with a ratio of $10^{0.2}$ between neighboring values. Then, for each method and at every percentage of the maximum function evaluations (MaxFEs), we compute the average percentage of fitness levels achieved across all independent runs. This yields a non-decreasing ECDF curve for each method on each problem, providing a measure of its efficiency. These curves are presented in Figures 7 and 8.

From the ECDF curves, we observe that the dynamics of all compared algorithms were similar in many cases. In some cases (e.g. F1 in $n = 10, 30$) the final results were of a comparable quality for all methods, but MF-CMA-ES converged faster. Only in one case (F22, $n = 50$) the CMA-ES-PPMF significantly outperformed MF-CMA-ES. We conclude that MF-CMA-ES should be preferred over CMA-ES-PPMF, especially when the budget for fitness evaluations is smaller than that assumed in CEC'2017.

6 CONCLUDING REMARKS

We introduced MF-CMA-ES, a matrix-free version of CMA-ES that eliminates the use of covariance matrix for generating new individuals. This work provides a fresh perspective on CMA-ES by demonstrating that the covariance matrix adaptation process can be performed implicitly. MF-CMA-ES generates new individuals through a weighted combination of points from the archive of previous generations. The proposed method preserves the distributional properties of the original CMA-ES.

The differences between convergence dynamics of MF-CMA-ES and CMA-ES for the quadratic fitness function were very small when the step-size adaptation was turned off. It is likely that these subtle differences are amplified when the step-size adaptation is enabled, which may lead to discrepancies in the quality of results between matrix-free and matrix-based CMA-ES during the CEC'2017 benchmarking procedure. Overall, MF-CMA-ES typically performed at least as well as CMA-ES-PPMF.

The formula (15) implies that the influence of the archive vectors decays exponentially over time. We plan to explore alternative formulations to express this influence. Specifically, a linear weight decay scheme will be considered, which could allow for smaller history window sizes, ultimately reducing both time and memory complexity.

We also plan to improve the code efficiency of MF-CMA-ES to fully leverage its vector-based form.

IMPLEMENTATION NOTE

All experiments presented in this paper were conducted using the R programming language. We used the `cma-es` package [15] to simulate the matrix-based CMA-ES. The source code of MF-CMA-ES and PPMF was developed by modifying the `cma-es` package. Unless otherwise noted, we used the default parameter values from the `cma-es` package for all tested matrix-based and matrix-free CMA-ES versions. The source code to reproduce all the results is available at <https://doi.org/10.5281/zenodo.14749755>. Benchmarking results

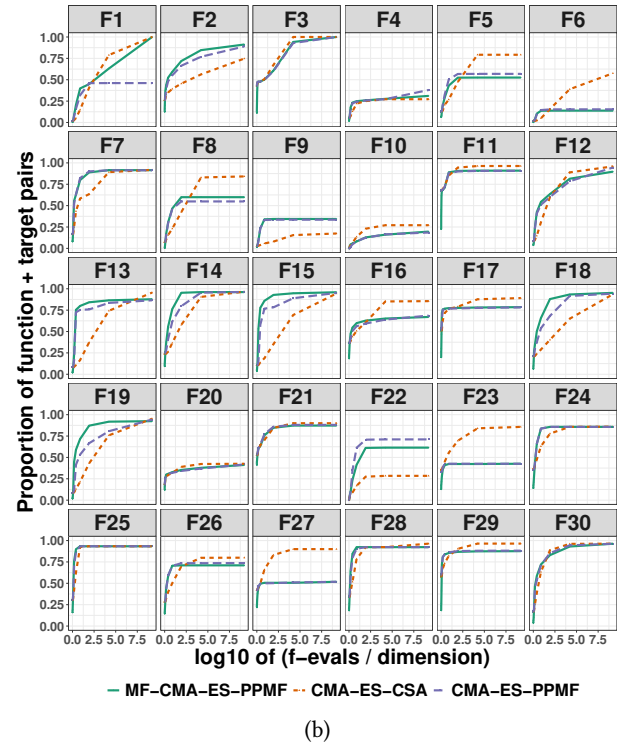
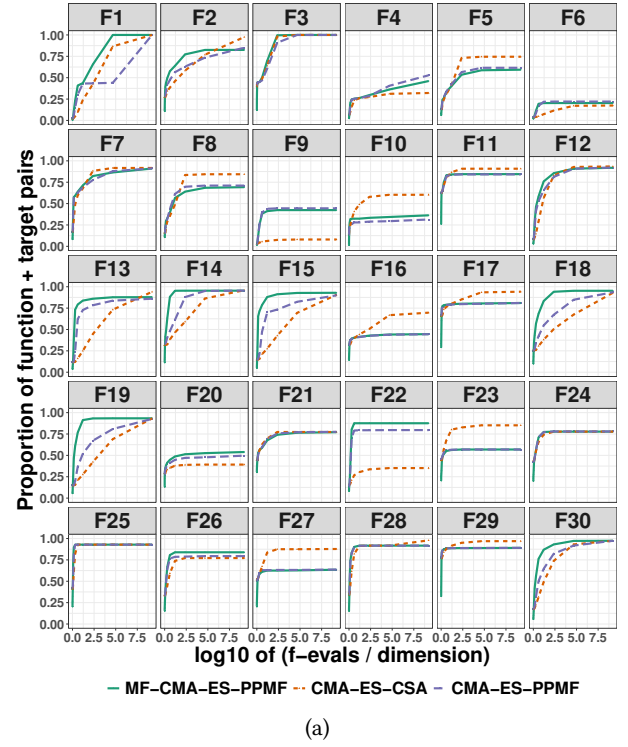


Figure 8: ECDF curves obtained by MF-CMA-ES (with PPMF) and two versions of CMA-ES (with CSA and PPMF) for CEC'2017 optimization problems in 30 dimensions (a) and 50 dimensions (b).

F.	MF-CMA-ES		CMA-ES-CSA		CMA-ES-PPMF	
	Mean	Std	Mean	Std	Mean	Std
1	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00
2	4.1e+46	2.9e+47	8.0e-07	6.0e-07	3.1e+44	1.8e+45
3	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00
4	1.8e+01	2.7e+01	5.5e+01	1.4e+01	1.4e+01	2.4e+01
5	5.5e+01	1.3e+02	3.1e+01	1.1e+02	3.3e+01	7.5e+01
6	1.1e+01	2.7e+01	6.2e+01	3.3e+01	7.0e+00	2.2e+01
7	4.2e+01	2.2e+01	3.6e+01	1.7e+00	4.2e+01	2.4e+01
8	6.1e+01	1.4e+02	2.6e+01	8.7e+01	4.5e+01	1.1e+02
9	1.0e+00	1.1e+00	1.3e+04	3.6e+03	9.8e-01	1.7e+00
10	6.4e+03	1.5e+03	4.0e+03	6.7e+02	6.8e+03	1.5e+03
11	1.3e+02	4.7e+01	3.6e+01	2.9e+01	1.4e+02	4.0e+01
12	1.5e+03	5.1e+02	1.1e+03	4.3e+02	1.4e+03	4.0e+02
13	1.6e+03	7.4e+02	3.8e+02	2.0e+02	2.1e+03	7.8e+02
14	1.4e+02	4.7e+01	1.2e+02	3.5e+01	1.3e+02	3.3e+01
15	2.1e+02	9.4e+01	3.4e+02	8.9e+01	3.8e+02	1.6e+02
16	1.7e+03	3.0e+02	2.1e+02	1.8e+02	1.8e+03	2.9e+02
17	5.8e+02	1.0e+02	8.1e+01	1.3e+02	5.7e+02	1.4e+02
18	1.5e+02	6.3e+01	2.5e+02	8.5e+01	2.3e+02	8.8e+01
19	1.1e+02	3.7e+01	1.2e+02	4.7e+01	1.3e+02	4.4e+01
20	8.1e+02	2.1e+02	1.2e+03	2.9e+02	8.9e+02	2.2e+02
21	2.4e+02	7.6e+01	2.2e+02	4.1e+01	2.3e+02	1.2e+01
22	4.2e+02	1.3e+03	4.1e+03	1.7e+03	1.1e+03	2.3e+03
23	1.1e+03	1.0e+02	3.8e+02	1.6e+02	1.1e+03	1.1e+02
24	4.4e+02	1.5e+01	4.2e+02	6.3e+00	4.3e+02	3.7e+01
25	3.9e+02	7.8e+00	3.9e+02	1.2e-02	3.9e+02	1.5e+01
26	6.2e+02	5.2e+02	7.2e+02	2.9e+02	9.1e+02	8.6e+02
27	1.3e+03	9.4e+01	5.1e+02	9.4e+00	1.4e+03	1.0e+02
28	3.2e+02	4.1e+01	3.3e+02	5.0e+01	3.2e+02	4.1e+01
29	1.7e+03	2.6e+02	4.5e+02	6.5e+01	1.7e+03	2.3e+02
30	2.1e+03	9.6e+01	2.2e+03	1.5e+02	2.5e+03	2.2e+02

Table 1: Statistics of results obtained by MF-CMA-ES (with PPMF) and two versions of CMA-ES (with CSA and PPMF) for CEC'2017 optimization problems in 10 dimensions after spending the budget of $10000n$ fitness evaluations.

F.	MF-CMA-ES		CMA-ES-CSA		CMA-ES-PPMF	
	Mean	Std	Mean	Std	Mean	Std
1	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00
2	4.1e+46	2.9e+47	8.0e-07	6.0e-07	3.1e+44	1.8e+45
3	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00
4	1.8e+01	2.7e+01	5.5e+01	1.4e+01	1.4e+01	2.4e+01
5	5.5e+01	1.3e+02	3.1e+01	1.1e+02	3.3e+01	7.5e+01
6	1.1e+01	2.7e+01	6.2e+01	3.3e+01	7.0e+00	2.2e+01
7	4.2e+01	2.2e+01	3.6e+01	1.7e+00	4.2e+01	2.4e+01
8	6.1e+01	1.4e+02	2.6e+01	8.7e+01	4.5e+01	1.1e+02
9	1.0e+00	1.1e+00	1.3e+04	3.6e+03	9.8e-01	1.7e+00
10	6.4e+03	1.5e+03	4.0e+03	6.7e+02	6.8e+03	1.5e+03
11	1.3e+02	4.7e+01	3.6e+01	2.9e+01	1.4e+02	4.0e+01
12	1.5e+03	5.1e+02	1.1e+03	4.3e+02	1.4e+03	4.0e+02
13	1.6e+03	7.4e+02	3.8e+02	2.0e+02	2.1e+03	7.8e+02
14	1.4e+02	4.7e+01	1.2e+02	3.5e+01	1.3e+02	3.3e+01
15	2.1e+02	9.4e+01	3.4e+02	8.9e+01	3.8e+02	1.6e+02
16	1.7e+03	3.0e+02	2.1e+02	1.8e+02	1.8e+03	2.9e+02
17	5.8e+02	1.0e+02	8.1e+01	1.3e+02	5.7e+02	1.4e+02
18	1.5e+02	6.3e+01	2.5e+02	8.5e+01	2.3e+02	8.8e+01
19	1.1e+02	3.7e+01	1.2e+02	4.7e+01	1.3e+02	4.4e+01
20	8.1e+02	2.1e+02	1.2e+03	2.9e+02	8.9e+02	2.2e+02
21	2.4e+02	7.6e+01	2.2e+02	4.1e+01	2.3e+02	1.2e+01
22	4.2e+02	1.3e+03	4.1e+03	1.7e+03	1.1e+03	2.3e+03
23	1.1e+03	1.0e+02	3.8e+02	1.6e+02	1.1e+03	1.1e+02
24	4.4e+02	1.5e+01	4.2e+02	6.3e+00	4.3e+02	3.7e+01
25	3.9e+02	7.8e+00	3.9e+02	1.2e-02	3.9e+02	1.5e+01
26	6.2e+02	5.2e+02	7.2e+02	2.9e+02	9.1e+02	8.6e+02
27	1.3e+03	9.4e+01	5.1e+02	9.4e+00	1.4e+03	1.0e+02
28	3.2e+02	4.1e+01	3.3e+02	5.0e+01	3.2e+02	4.1e+01
29	1.7e+03	2.6e+02	4.5e+02	6.5e+01	1.7e+03	2.3e+02
30	2.1e+03	9.6e+01	2.2e+03	1.5e+02	2.5e+03	2.2e+02

Table 2: Statistics of results obtained by MF-CMA-ES (with PPMF) and two versions of CMA-ES (with CSA and PPMF) for CEC'2017 optimization problems in 30 dimensions after spending the budget of $10000n$ fitness evaluations.

provided in section 5.3 were obtained using the CEC'2017 implementation in R [16].

F.	MF-CMA-ES		CMA-ES-CSA		CMA-ES-PPMF	
	Mean	Std	Mean	Std	Mean	Std
1	0.0e+00	0.0e+00	0.0e+00	0.0e+00	2.0e+03	2.9e+03
2	3.4e+75	2.4e+76	7.3e+79	5.1e+80	4.8e+81	3.5e+82
3	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00	0.0e+00
4	6.6e+01	4.6e+01	9.5e+01	4.0e+01	5.2e+01	4.6e+01
5	2.2e+02	4.0e+02	9.9e+00	2.7e+00	8.8e+01	1.7e+02
6	2.2e+01	3.6e+01	3.1e+01	3.9e+01	1.1e+01	2.4e+01
7	6.5e+01	8.3e+00	6.0e+01	1.6e+00	6.5e+01	8.0e+00
8	8.8e+01	1.8e+02	1.0e+01	2.3e+00	1.4e+02	2.8e+02
9	1.2e+01	1.9e+01	2.6e+04	1.1e+04	1.3e+01	1.7e+01
10	6.8e+03	1.9e+03	5.1e+03	1.8e+03	7.1e+03	2.1e+03
11	1.8e+02	4.8e+01	4.5e+01	1.7e+01	1.9e+02	5.2e+01
12	1.2e+04	2.7e+04	2.2e+03	5.5e+02	3.0e+03	6.5e+02
13	3.0e+03	1.6e+03	6.1e+02	2.2e+02	3.9e+03	8.7e+02
14	2.1e+02	4.5e+01	1.9e+02	3.6e+01	2.1e+02	4.8e+01
15	4.9e+02	2.5e+02	6.4e+02	1.1e+02	5.8e+02	1.6e+02
16	1.2e+03	5.0e+02	3.2e+02	1.8e+02	1.1e+03	4.4e+02
17	1.6e+03	4.1e+02	2.4e+02	1.1e+02	1.7e+03	4.0e+02
18	2.8e+02	9.7e+01	3.7e+02	9.1e+01	3.3e+02	1.0e+02
19	2.8e+03	1.4e+04	9.6e+01	2.4e+01	1.2e+02	3.3e+01
20	1.8e+03	4.6e+02	1.7e+03	7.1e+02	1.7e+03	3.9e+02
21	3.3e+02	3.1e+02	2.1e+02	3.2e+00	3.0e+02	2.2e+02
22	3.5e+03	4.3e+03	5.6e+03	2.3e+03	2.3e+03	3.7e+03
23	2.0e+03	2.5e+02	4.3e+02	1.1e+01	1.9e+03	1.5e+02
24	5.4e+02	3.4e+01	4.9e+02	6.3e+00	5.4e+02	3.2e+01
25	5.5e+02	3.5e+01	4.8e+02	1.6e+01	5.5e+02	4.3e+01
26	2.3e+03	1.9e+03	8.6e+02	1.2e+02	2.2e+03	2.2e+03
27	3.3e+03	2.5e+02	5.2e+02	1.1e+01	3.3e+03	2.3e+02
28	4.9e+02	1.9e+01	4.8e+02	2.5e+01	5.0e+02	3.0e+01
29	2.4e+03	1.2e+03	4.5e+02	1.6e+02	2.2e+03	1.2e+03
30	6.7e+05	7.8e+04	6.2e+05	4.3e+04	6.7e+05	5.6e+04

Table 3: Statistics of results obtained by MF-CMA-ES (with PPMF) and two versions of CMA-ES (with CSA and PPMF) for CEC'2017 optimization problems in 50 dimensions after spending the budget of 10000n fitness evaluations.

REFERENCES

- [1] ARABAS, J., AND JAGODZIŃSKI, D. Toward a matrix-free Covariance Matrix Adaptation Evolution Strategy. *IEEE Trans. Evol. Comput.* (2019).
- [2] AWAD, N. H., ALL, M., LIANG, J., QU, B., AND SUGANTHAN, P. N. Problem definitions and evaluation criteria for the CEC 2017 special session and competition on real-parameter optimization. Tech. rep., Nanyang Technol. Univ., Singapore and Jordan Univ. Sci. Technol. and Zhengzhou Univ., China, 2016.
- [3] BEYER, H. G., AND SENDHOFF, B. Simplify your Covariance Matrix Adaptation Evolution Strategy. *IEEE Transactions on Evolutionary Computation* 21, 5 (2017), 746–759.
- [4] DERRAC, J., GARCÍA, S., MOLINA, D., AND HERRERA, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation* 1, 1 (2011), 3–18.
- [5] GARCÍA, S., FERNÁNDEZ, A., LUENGO, J., AND HERRERA, F. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences* 180, 10 (2010), 2044–2064. Special Issue on Intelligent Distributed Information Systems.
- [6] HANSEN, N. The CMA evolution strategy: a comparing review. In *Towards a new evolutionary computation: advances on estimation of distribution algorithms*, J. A. Lozano, Ed. Springer, 2006, pp. 75–102.
- [7] HANSEN, N. The CMA evolution strategy: A tutorial. *CoRR abs/1604.00772* (2016).
- [8] HANSEN, N., AND AUGER, A. Evolution strategies and CMA-ES (covariance matrix adaptation). In *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation* (New York, NY, USA, 2014), GECCO Comp '14, Association for Computing Machinery, p. 513–534.
- [9] HANSEN, N., AUGER, A., BROCKHOFF, D., TUSAR, D., AND TUSAR, T. COCO: Performance assessment. *arXiv:1605.03560* (2016).
- [10] KRAUSE, O., ARBONÈS, D. R., AND IGEL, C. CMA-ES with optimal covariance update and storage complexity. In *Advances in Neural Information Processing Systems* (2016), D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29, Curran Associates, Inc.
- [11] LOSCHILOV, I., GLASMACHERS, T., AND BEYER, H.-G. Large scale black-box optimization by limited-memory matrix adaptation. *IEEE Transactions on Evolutionary Computation* 23, 2 (2019), 353–358.
- [12] POLAND, J., AND ZELL, A. Main vector adaptation: A CMA variant with linear time and space complexity. In *Proc. Genet. Evol. Comput. Conf.* (2001), Morgan Kaufmann, pp. 1050–1055.
- [13] RAYMOND, R., AND HANSEN, N. A simple modification in CMA-ES achieving linear time and space complexity. In *Parallel Problem Solving from Nature – PPSN X* (Berlin, Heidelberg, 2008), Springer Berlin Heidelberg, pp. 296–305.
- [14] SUN, Y., SCHAUL, T., GOMEZ, F., AND SCHMIDHUBER, J. A linear time natural evolution strategy for non-separable functions. In *GECCO (Companion)* (2013), ACM, pp. 61–62.
- [15] TRAUTMANN, H., MERSMANN, O., AND ARNU, D. *cmaes: Covariance Matrix Adapting Evolutionary Strategy*, 2022. R package version 1.0-12.
- [16] WARCHULSKI, E. *cecs: R Interface for the C Implementation of CEC Benchmark Functions*, 2021. R package version 0.2.3.
- [17] WARCHULSKI, E., AND ARABAS, J. A new step-size adaptation rule for CMA-ES based on the population midpoint fitness. In *2021 IEEE Congress on Evolutionary Computation (CEC)* (2021), pp. 825–831.