# Large Scale Black-Box Optimization by Limited-Memory Matrix Adaptation

Ilya Loshchilov, Tobias Glasmachers, and Hans-Georg Beyer

***Abstract*—The covariance matrix adaptation evolution strategy (CMA-ES) is a popular method to deal with nonconvex and/or stochastic optimization problems when gradient information is not available. Being based on the CMA-ES, the recently proposed matrix adaptation evolution strategy (MA-ES) establishes the rather surprising result that the covariance matrix and all associated operations (e.g., potentially unstable eigen decomposition) can be replaced by an iteratively updated transformation matrix without any loss of performance. In order to further simplify MA-ES and reduce its $\mathcal{O}(n^2)$ time and storage complexity to $\mathcal{O}(mn)$ with $m \ll n$ such as $m \in \mathcal{O}(1)$ or $m \in \mathcal{O}(\log(n))$, we present the limited-memory MA-ES for efficient zeroth order large-scale optimization. The algorithm demonstrates state-of-the-art performance on a set of established large-scale benchmarks.**

***Index Terms*—Evolutionary computation, optimization.**

## I. Introduction

Evolution strategies (ESs) are optimization methods originally inspired by mutation of organic beings and designed to establish "a reward-based system, to increase the probability of those changes, which lead to improvements of quality of the system" [1]. Going far beyond their biologically inspired roots, they have been developed into state-of-the-art zeroth order search methods [2]. ESs [3] consider an objective function $f : \mathbf{R}^n \mapsto \mathbf{R}, \boldsymbol{x} \mapsto f(\boldsymbol{x})$ to be minimized by sampling $i \in \{1, \ldots, \lambda\}$ candidate solutions at iteration $t$ as

$$\boldsymbol{x}_i^{(t)} \leftarrow \boldsymbol{y}^{(t)} + \sigma^{(t)} \cdot \mathcal{N}\left(\mathbf{0}, \mathbf{C}^{(t)}\right) \tag{1}$$

where $\boldsymbol{y}^{(t)}$ is the current estimate of the optimum, $\mathbf{C}^{(t)} \in \mathbf{R}^{n \times n}$ is a covariance matrix initialized to the identity matrix $\mathbf{C}^{(0)} = \mathbf{I}$, $\sigma^{(t)}$ is a scaling factor for the mutation step, often referred to as the global step size, and $\mathcal{N}$ denotes a random vector following the standard normal distribution. Both $\boldsymbol{y}^{(t)}$ and $\sigma^{(t)}$ are to be adapted or *learned* over time. Modern ESs such as the covariance matrix adaptation ES (CMA-ES) also include the adaptation of $\mathbf{C}^{(t)}$ [4], [5] to the shape of the local landscape, resembling second-order methods. Recent theoretical studies of ES and CMA-ES from the perspective of information geometry [6]–[9], connecting the method to stochastic natural gradient learning, have made significant progress in understanding the principles underpinning the state-of-the-art performance of the algorithm [10]. The variety of algorithms [2] derived from and inspired by the theoretical studies helped to notice that the core component of CMA-ES, the covariance matrix itself (and covariance matrix square root operations) can be removed from the algorithm without any loss of performance [11].[1] The final algorithm called matrix adaptation ES (MA-ES) [11] is conceptually simpler and involves only matrix–matrix and matrix–vector operations, which, however, lead to $\Theta(n^3/\log(n))$ time complexity per sample, which can be reduced to $\Theta(n^2)$ (see below), and $\Theta(n^2)$ space complexity.

A very simple ES style algorithm has recently been applied to the problem of optimizing deep neural network controllers for Atari games [12] in a highly parallel fashion. However, neither the step size nor the covariance matrix were adapted, resulting in extremely poor convergence speed. Because of the high dimensionality of weight spaces of deep neural networks, which can have many thousands and even millions of parameters, CMA-ES and MA-ES are not applicable to the problem. With the steadily increasing dimensionality of real-world optimization problems, the new challenges of large-scale black-box optimization become more pronounced for CMA-ES and MA-ES due to their $\Theta(n^2)$ complexity. To address them, a number of large-scale CMA-ES variants has been proposed [13]–[18] including the limited-memory CMA-ES (LM-CMA-ES) [17] that matches the performance of quasi-Newton methods such as L-BFGS [19] when dealing with large-scale black-box problems. It has a moderate sample cost of $\mathcal{O}(mn)$ time and space complexity, where $m \ll n$ can in principle be as small as 1. Alternative large-scale optimization approaches from the domain of evolutionary computation are based on cooperative coevolution, evolving blocks of coordinates in parallel [20]. This (implicitly) assumes some degree of separability of the problem.

In this letter, we combine the best of two worlds: inspired by LM-CMA-ES we present the limited-memory MA-ES (LM-MA-ES), which matches state-of-the-art results while reducing the time and space complexity of MA-ES to $\mathcal{O}(mn)$ per sample. A key novelty of the algorithm is the introduction of a flexibly sized set of evolution paths, which operate on different time scales.

## II. Variable Metric ESs: CMA-ES and MA-ES

Our discussion of variable metric ESs is based on Algorithm 1, which highlights the similarities and differences between CMA-ES, MA-ES, and the proposed LM-MA-ES.

The sampling of the $\lambda$ candidate solutions in CMA-ES is described by (1) and involves a matrix–vector product between a matrix $\sqrt{\mathbf{C}}$ and a vector $\boldsymbol{z}_i$ sampled from the $n$-dimensional standard normal distribution. This operation (see line 7 in Algorithm 1) requires $\sqrt{\mathbf{C}}$ to be stored [hence, the $\Theta(n^2)$ storage cost] and the matrix–vector multiplication to be performed [hence, the $\Theta(n^2)$ time cost]. The resulting vector $\boldsymbol{d}_i$ represents a direction of the so-called mutation operation. The $i$th candidate solution is obtained by changing (mutating) the current estimate of the optimum $\boldsymbol{y}$ by $\boldsymbol{d}_i$ multiplied by the global mutation step-size $\sigma$ (line 11). The rationale behind parameterizing the sampling distribution by $\mathcal{N}\left(\boldsymbol{y}, \sigma^2\mathbf{C}\right)$ and not just $\mathcal{N}(\boldsymbol{y}, \mathbf{C})$, i.e., decoupling $\mathbf{C}$ and $\sigma$, lies in the observation that $\sigma$ can be learned

Manuscript received January 25, 2018; revised May 15, 2018; accepted July 6, 2018. Date of publication July 11, 2018; date of current version March 29, 2019. The work of H.-G. Beyer was supported by the Austrian Science Fund under Grant P29651-N32. *(Corresponding author: Tobias Glasmachers.)*

I. Loshchilov is with the Research Group on Machine Learning for Automated Algorithm Design, University of Freiburg, 79110 Freiburg im Breisgau, Germany (e-mail: ilya.loshchilov@gmail.com).

T. Glasmachers is with the Institut für Neuroinformatik, Ruhr-Universität Bochum, 44780 Bochum, Germany (e-mail: tobias.glasmachers@ini.rub.de).

H.-G. Beyer is with the Research Center Process and Product Engineering, Vorarlberg University of Applied Sciences, 6850 Dornbirn, Austria (e-mail: hans-georg.beyer@fhv.at).

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

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

Digital Object Identifier 10.1109/TEVC.2018.2855049

[1] Of course, the transformation matrix needed for sampling the multivariate Gaussian is kept, enabling variable metric optimization.

more quickly and more robustly than $\mathbf{C}$ and its adaptation alone enables linear convergence on scale-invariant problems [21].

ESs are invariant to rank-preserving/strictly monotonically increasing transformations of $f$-values because all operations are based on ranks of evaluated solutions. The estimate of the optimum $\boldsymbol{y}$ is updated by a weighted sum of mutation steps taken by the top ranked $\mu$ out of $\lambda$ solutions (line 12).

The currently most commonly applied adaptation rule for the step size is the cumulative step-size adaptation (CSA) mechanism [22]. It is based on the length of an evolution path $\boldsymbol{p}_\sigma$, an exponentially fading record of recent most successful steps $\boldsymbol{z}_{i:\lambda}$ (see line 13). If the path becomes too long (the expected path length of a Gaussian random walk can be approximated by $\sqrt{n}$ when $n$ is large), indicating that recent steps tend to move into the same direction, then the step size is increased. On the contrary, a too short path indicating oscillations due to overjumping the optimum results in a reduction of the step size. A rigorous analysis of CSA with and without cumulation on the ellipsoid function is given in [23].

The seminal CMA-ES algorithm [4], [5] introduced adaptation of the covariance matrix, which renders the algorithm invariant to affine transformations of the search space (achieved in practice after an initial adaptation phase) and hence enables a fast convergence rate independent of the conditioning of the problem, resembling second-order methods. The covariance matrix is adapted toward a weighted maximum likelihood estimate of the $\mu$ most successful samples (rank-$\mu$ update) with learning rate $c_\mu$ and a second evolution path (rank-1 update) with learning rate $c_1$ (see line 15). This update has an alternative interpretation as a stochastic gradient step on the information geometric manifold forming the algorithm's state space [7], [8]. While the default strategy parameter values of CMA-ES given in Algorithm 1 are known to be robust, their optimal values can be adapted online during the optimization process, providing additional speed and robustness [24].

Most implementations of CMA-ES consider eigendecomposition procedures of $\mathcal{O}(n^3)$ time complexity per call to obtain $\sqrt{\mathbf{C}}$ from $\mathbf{C}$ only every $n/\lambda$ iterations (see line 7) to achieve amortized $\Theta(n^2)$ time complexity per sampled solution. Numerical stability of the $\Theta(n^2)$ update can be ensured by maintaining a triangular Cholesky factor [25].

The recently proposed MA-ES greatly simplifies CMA-ES by avoiding the construction of the covariance matrix $\mathbf{C}$. Instead it maintains only a transformation matrix $\mathbf{M}$ representing $\sqrt{\mathbf{C}}$, i.e., fulfilling $\mathbf{M}\mathbf{M}^T = \mathbf{C}$. After removing the approximate redundancy of $\boldsymbol{p}_\sigma$ and $\boldsymbol{p}_c$, $\mathbf{M}$ can be updated multiplicatively (line 16). Matrix multiplication is an $\mathcal{O}(n^3)$ operation, therefore we propose to replace the multiplicative update at iteration $t$ by the equivalent additive update

$$\mathbf{M}^{(t+1)} \leftarrow \Big(1 - \frac{c_1}{2} - \frac{c_\mu}{2}\Big)\mathbf{M}^{(t)} + \frac{c_1}{2}\boldsymbol{d}_\sigma^{(t)}\Big(\boldsymbol{p}_\sigma^{(t)}\Big)^T + \frac{c_\mu}{2}\sum_{i=1}^{\mu} w_i \boldsymbol{d}_{i:\lambda}^{(t)}(\boldsymbol{z}_{i:\lambda}^{(t)})^T \tag{2}$$

which achieves $\mathcal{O}(n^2)$ time cost thanks to precomputing $\boldsymbol{d}_\sigma^{(t)} = \mathbf{M}^{(t)}\boldsymbol{p}_\sigma^{(t)}$ and reusing the vectors $\boldsymbol{d}_{i:\lambda}^{(t)}$. The resulting algorithm is referred to as *fast* MA-ES.

## III. Limited-Memory Matrix Adaptation Evolution Strategy

A number of methods were proposed to reduce the space and time complexity per sample from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ or at least $\mathcal{O}(n\log(n))$ while still modeling the most relevant aspects of the full covariance matrix. Simple approaches like [14] restrict the covariance matrix to its diagonal, while more elaborate methods use a low-rank approximation [15], [17]. Both approaches can be combined [16]. Inspired by the LM-CMA-ES [17], which in turn is inspired by the L-BFGS method [26], we show how to scale up MA-ES to high-dimensional problems. The derivation given below is based on the multiplicative update. The final result for the additive update (2) is equivalent when $\boldsymbol{d}_\sigma^{(t)}$ is not stored but reconstructed as $\mathbf{M}^{(t)}\boldsymbol{p}_\sigma^{(t)}$. At iteration $t$, the main update equation of MA-ES reads

$$\mathbf{M}^{(t+1)} \leftarrow \mathbf{M}^{(t)}\left[\mathbf{I} + \frac{c_1}{2}\left(\boldsymbol{p}_\sigma^{(t+1)}\Big(\boldsymbol{p}_\sigma^{(t+1)}\Big)^T - \mathbf{I}\right) + \frac{c_\mu}{2}\left(\sum_{i=1}^{\mu} w_i \boldsymbol{z}_{i:\lambda}^{(t)}\Big(\boldsymbol{z}_{i:\lambda}^{(t)}\Big)^T - \mathbf{I}\right)\right]$$

(line 16 in Algorithm 1) where $\mathbf{M}^{(t)}$ is adapted multiplicatively based on the rank-one update weighted by $(c_1/2)$ and the rank-$\mu$ update weighted by $(c_\mu/2)$, starting from $\mathbf{M}^{(t=0)} = \mathbf{I}$. By omitting the rank-$\mu$ update for the sake of simplicity (i.e., by setting $c_\mu = 0$), we obtain

$$\begin{aligned}\mathbf{M}^{(1)} &\leftarrow \mathbf{I} + \frac{c_1}{2}\left(\boldsymbol{p}_\sigma^{(1)}\Big(\boldsymbol{p}_\sigma^{(1)}\Big)^T - \mathbf{I}\right)\\ &= \Big(1 - \frac{c_1}{2}\Big)\mathbf{I} + \frac{c_1}{2}\boldsymbol{p}_\sigma^{(1)}\Big(\boldsymbol{p}_\sigma^{(1)}\Big)^T.\end{aligned} \tag{3}$$

The sampling procedure of the $i$th solution $\boldsymbol{x}_i^{(1)}$ follows:

$$\boldsymbol{x}_i^{(1)} \leftarrow \boldsymbol{y}^{(1)} + \sigma^{(1)}\boldsymbol{d}_i^{(1)} = \boldsymbol{y}^{(1)} + \sigma^{(1)}\mathbf{M}^{(1)}\boldsymbol{z}_i^{(1)}$$

where $\boldsymbol{z}_i^{(1)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. One can rewrite $\boldsymbol{d}_i^{(1)} = \mathbf{M}^{(1)}\boldsymbol{z}_i^{(1)}$ based on (3) as

$$\begin{aligned}\boldsymbol{d}_i^{(1)} = \mathbf{M}^{(1)}\boldsymbol{z}_i^{(1)} &= \left(\Big(1 - \frac{c_1}{2}\Big)\mathbf{I} + \frac{c_1}{2}\boldsymbol{p}_\sigma^{(1)}\Big(\boldsymbol{p}_\sigma^{(1)}\Big)^T\right)\boldsymbol{z}_i^{(1)}\\ &= \boldsymbol{z}_i^{(1)}\Big(1 - \frac{c_1}{2}\Big) + \frac{c_1}{2}\boldsymbol{p}_\sigma^{(1)}\left(\Big(\boldsymbol{p}_\sigma^{(1)}\Big)^T\boldsymbol{z}_i^{(1)}\right).\end{aligned} \tag{4}$$

Importantly, $((\boldsymbol{p}_\sigma^{(1)})^T\boldsymbol{z}^{(1)})$ is a scalar (see line 10 in Algorithm 1) and thus (4) does not require $\mathbf{M}^{(1)}$ to be stored in memory. One generally obtains

$$\begin{aligned}\boldsymbol{d}_i^{(t)} &= \mathbf{M}^{(t)}\boldsymbol{z}_i^{(t)} = \mathbf{M}^{(t-1)}\mathbf{P}^{(t)}\boldsymbol{z}_i^{(t)}\\ &= \mathbf{M}^{(t-1)}\underbrace{\left(\Big(1 - \frac{c_1}{2}\Big)\mathbf{I} + \frac{c_1}{2}\boldsymbol{p}_\sigma^{(t)}\Big(\boldsymbol{p}_\sigma^{(t)}\Big)^T\right)}_{:=\mathbf{P}^{(t)}}\boldsymbol{z}_i^{(t)}\end{aligned} \tag{5}$$

leading to a sequence of products

$$\begin{aligned}\boldsymbol{d}_i^{(t)} = &\left(\Big(1 - \frac{c_1}{2}\Big)\mathbf{I} + \frac{c_1}{2}\boldsymbol{p}_\sigma^{(1)}\Big(\boldsymbol{p}_\sigma^{(1)}\Big)^T\right)\\ &\times \cdots \times \left(\Big(1 - \frac{c_1}{2}\Big)\mathbf{I} + \frac{c_1}{2}\boldsymbol{p}_\sigma^{(t-1)}\Big(\boldsymbol{p}_\sigma^{(t-1)}\Big)^T\right)\\ &\times \left(\Big(1 - \frac{c_1}{2}\Big)\mathbf{I} + \frac{c_1}{2}\boldsymbol{p}_\sigma^{(t)}\Big(\boldsymbol{p}_\sigma^{(t)}\Big)^T\right)\boldsymbol{z}_i^{(t)}\end{aligned} \tag{6}$$

which is to be treated from right to left. Thus, the sampling procedure for $\boldsymbol{d}_i^{(t)} = \mathbf{M}^{(t)}\boldsymbol{z}_i^{(t)}$ does neither require matrix–matrix–product operations nor does it require the storage of $\mathbf{M}^{(t)} \in \mathbf{R}^{n\times n}$, but can be performed based on $t$ vectors $\boldsymbol{p}_\sigma^{(t)}$ used to construct $\mathbf{M}^{(t)}$. However, this is efficient only for $t \ll n$. Therefore, in order to reduce the cost of the sampling procedure, (6) must be *approximated* in one way or another by artificially limiting the number $m$ of supporting $\boldsymbol{p}_\sigma^{(t)}$ vectors such that $m \ll n$.

LM-CMA-ES [17] addresses a similar problem of compactly representing the covariance matrix with $m \ll n$ direction vectors: instead of considering the last $m$ vectors, it samples them in a certain temporal

**Algorithm 1** CMA-ES, MA-ES and LM-MA-ES

1: **given** $n \in \mathbb{N}_+$, $\lambda = 4 + \lfloor 3 \ln n \rfloor$, $\mu = \lfloor \lambda/2 \rfloor$, $w_i = \frac{\ln(\mu+\frac{1}{2}) - \ln i}{\sum_{j=1}^{\mu} (\ln(\mu+\frac{1}{2}) - \ln j)}$ for $i = 1, \ldots, \mu$, $\mu_w = \frac{1}{\sum_{i=1}^{\mu} w_i^2}$,
$c_\sigma = \frac{\mu_w+2}{n+\mu_w+5}$, $c_c = \frac{4}{n+4}$, $c_1 = \frac{2}{(n+1.3)^2+\mu_w}$, $c_\mu = \min\left(1 - c_1, \frac{2(\mu_w - 2 + 1/\mu_w)}{(n+2)^2+\mu_w}\right)$,
$m = 4 + \lfloor 3 \ln n \rfloor$, $c_\sigma = \frac{2\lambda}{n}$, $c_{d,i} = \frac{1}{1.5^{i-1} n}$, $c_{c,i} = \frac{\lambda}{4^{i-1} n}$ for $i = 1, \ldots, m$

2: **initialize** $t \leftarrow 0$, $\boldsymbol{y}^{(t=0)} \in \mathbf{R}^n$, $\sigma^{(t=0)} > 0$, $\boldsymbol{p}_\sigma^{(t=0)} = \mathbf{0}$, $\boldsymbol{p}_c^{(t=0)} = \mathbf{0}$, $\mathbf{C}^{(t=0)} = \mathbf{I}$, $\mathbf{M}^{(t=0)} = \mathbf{I}$, $\boldsymbol{m}_i^{(t=0)} \in \mathbf{R}^n$,
$\boldsymbol{m}_i^{(t=0)} = \mathbf{0}$ for $i = 1, \ldots, m$

3: **repeat**

4: **for** $i \leftarrow 1, \ldots, \lambda$ **do**

5: $\boldsymbol{z}_i^{(t)} \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$

6: $\boldsymbol{d}_i^{(t)} \leftarrow \boldsymbol{z}_i^{(t)}$

7: **if** $t \mod \frac{n}{\lambda} = 0$ **then** $\mathbf{M}^{(t)} \leftarrow \sqrt{\mathbf{C}^{(t)}}$ **else** $\mathbf{M}^{(t)} \leftarrow \mathbf{M}^{(t-1)}$ ▷ CMA-ES

8: $\boldsymbol{d}_i^{(t)} \leftarrow \mathbf{M}^{(t)} \boldsymbol{d}_i^{(t)}$ ▷ CMA-ES and MA-ES

9: **for** $j \leftarrow 1, \ldots, \min(t, m)$ **do** ▷ LM-MA-ES

10: $\boldsymbol{d}_i^{(t)} \leftarrow (1 - c_{d,j}) \boldsymbol{d}_i^{(t)} + c_{d,j} \boldsymbol{m}_j^{(t)} \left( (\boldsymbol{m}_j^{(t)})^T \boldsymbol{d}_i^{(t)} \right)$ ▷ LM-MA-ES

11: $\boldsymbol{f}_i^{(t)} \leftarrow f(\boldsymbol{y}^{(t)} + \sigma^{(t)} \boldsymbol{d}_i^{(t)})$

12: $\boldsymbol{y}^{(t+1)} \leftarrow \boldsymbol{y}^{(t)} + \sigma^{(t)} \sum_{i=1}^{\mu} w_i \boldsymbol{d}_{i:\lambda}^{(t)}$ ▷ the symbol $i{:}\lambda$ denotes $i$-th best sample on $f$

13: $\boldsymbol{p}_\sigma^{(t+1)} \leftarrow (1 - c_\sigma) \boldsymbol{p}_\sigma^{(t)} + \sqrt{\mu_w c_\sigma (2 - c_\sigma)} \sum_{i=1}^{\mu} w_i \boldsymbol{z}_{i:\lambda}^{(t)}$

14: $\boldsymbol{p}_c^{(t+1)} \leftarrow (1 - c_c) \boldsymbol{p}_c^{(t)} + \sqrt{\mu_w c_c (2 - c_c)} \sum_{i=1}^{\mu} w_i \boldsymbol{d}_{i:\lambda}^{(t)}$ ▷ CMA-ES

15: $\mathbf{C}^{(t+1)} \leftarrow (1 - c_1 - c_\mu) \mathbf{C}^{(t)} + c_1 \boldsymbol{p}_c^{(t)} (\boldsymbol{p}_c^{(t)})^T + c_\mu \sum_{i=1}^{\mu} w_i \boldsymbol{d}_{i:\lambda}^{(t)} (\boldsymbol{d}_{i:\lambda}^{(t)})^T$ ▷ CMA-ES

16: $\mathbf{M}^{(t+1)} \leftarrow \mathbf{M}^{(t)} \left[ \mathbf{I} + \frac{c_1}{2} \left( \boldsymbol{p}_\sigma^{(t)} (\boldsymbol{p}_\sigma^{(t)})^T - \mathbf{I} \right) + \frac{c_\mu}{2} \left( \sum_{i=1}^{\mu} w_i \boldsymbol{z}_{i:\lambda}^{(t)} (\boldsymbol{z}_{i:\lambda}^{(t)})^T - \mathbf{I} \right) \right]$ ▷ MA-ES

17: **for** $i \leftarrow 1, \ldots, m$ **do** ▷ LM-MA-ES

18: $\boldsymbol{m}_i^{(t+1)} \leftarrow (1 - c_{c,i}) \boldsymbol{m}_i^{(t)} + \sqrt{\mu_w c_{c,i} (2 - c_{c,i})} \sum_{j=1}^{\mu} w_j \boldsymbol{z}_{j:\lambda}^{(t)}$ ▷ LM-MA-ES

19: $\sigma^{(t+1)} \leftarrow \sigma^{(t)} \cdot \exp\left[ \frac{c_\sigma}{2} \left( \frac{\left\| \boldsymbol{p}_\sigma^{(t+1)} \right\|^2}{n} - 1 \right) \right]$

20: $t \leftarrow t + 1$

21: **until** *stopping criterion is met*

TABLE I
TEST FUNCTIONS USED IN THIS LETTER

| Name | Function $f(\boldsymbol{x})$ |
|---|---|
| Sphere | $\sum_{i=1}^{n} \boldsymbol{x}_i^2$ |
| Ellipsoid | $\sum_{i=1}^{n} 10^{6 \frac{i-1}{n-1}} \boldsymbol{x}_i^2$ |
| Rosenbrock | $\sum_{i=1}^{n-1} \left( 100 \cdot (\boldsymbol{x}_i^2 - x_{i+1})^2 + (x_i - 1)^2 \right)$ |
| Discus | $10^6 \boldsymbol{x}_1^2 + \sum_{i=2}^{n} \boldsymbol{x}_i^2$ |
| Cigar | $\boldsymbol{x}_1^2 + 10^6 \sum_{i=2}^{n} \boldsymbol{x}_i^2$ |
| Different Powers | $\sum_{i=1}^{n} \lvert \boldsymbol{x}_i \rvert^{2+4(i-1)/(n-1)}$ |

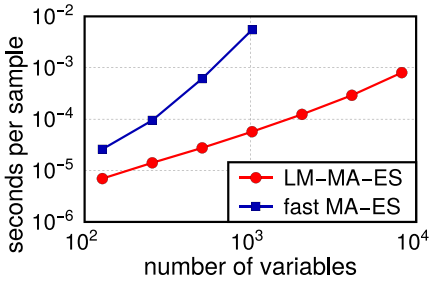


Fig. 1. Internal algorithm cost.

distance in terms of iterations $t$. In principle, the same approach works also for LM-MA-ES (see the supplementary material for details). However, the rather complicated procedure of ensuring a temporal distance between $\boldsymbol{p}_\sigma$ vectors can be simplified by considering different time horizons of their update. This procedure is a viable alternative since $\boldsymbol{p}_\sigma$ itself is anyway incrementally updated with $\sum_{i=1}^{\mu} w_i \boldsymbol{z}_{i:\lambda}$. Thus, instead of a full transformation matrix $\mathbf{M} \in \mathbf{R}^{n \times n}$ and similarly to LM-CMA-ES, LM-MA-ES maintains $m \ll n$ vectors $\boldsymbol{m}_i$ (see lines 17 and 18 in Algorithm 1), modeling the deviation of the transformation matrix from the identity as a rank-$m$ matrix. The learning rates $c_{c,i}$ and $c_{d,i}$ for applying and updating the vectors $\boldsymbol{m}_i$ are chosen to be exponentially decaying, hence the $\boldsymbol{m}_i$ are fading records of mean update steps on exponentially differing time scales. This is in contrast to CMA-ES and MA-ES, which update their matrices $\mathbf{C}$ and $\mathbf{M}$ only with two different learning rates for the rank-1 and rank-$\mu$ updates, and hence operate on a single time scale. LM-MA-ES learns some directions very quickly, while others are kept more stable. This can be advantageous in particular in

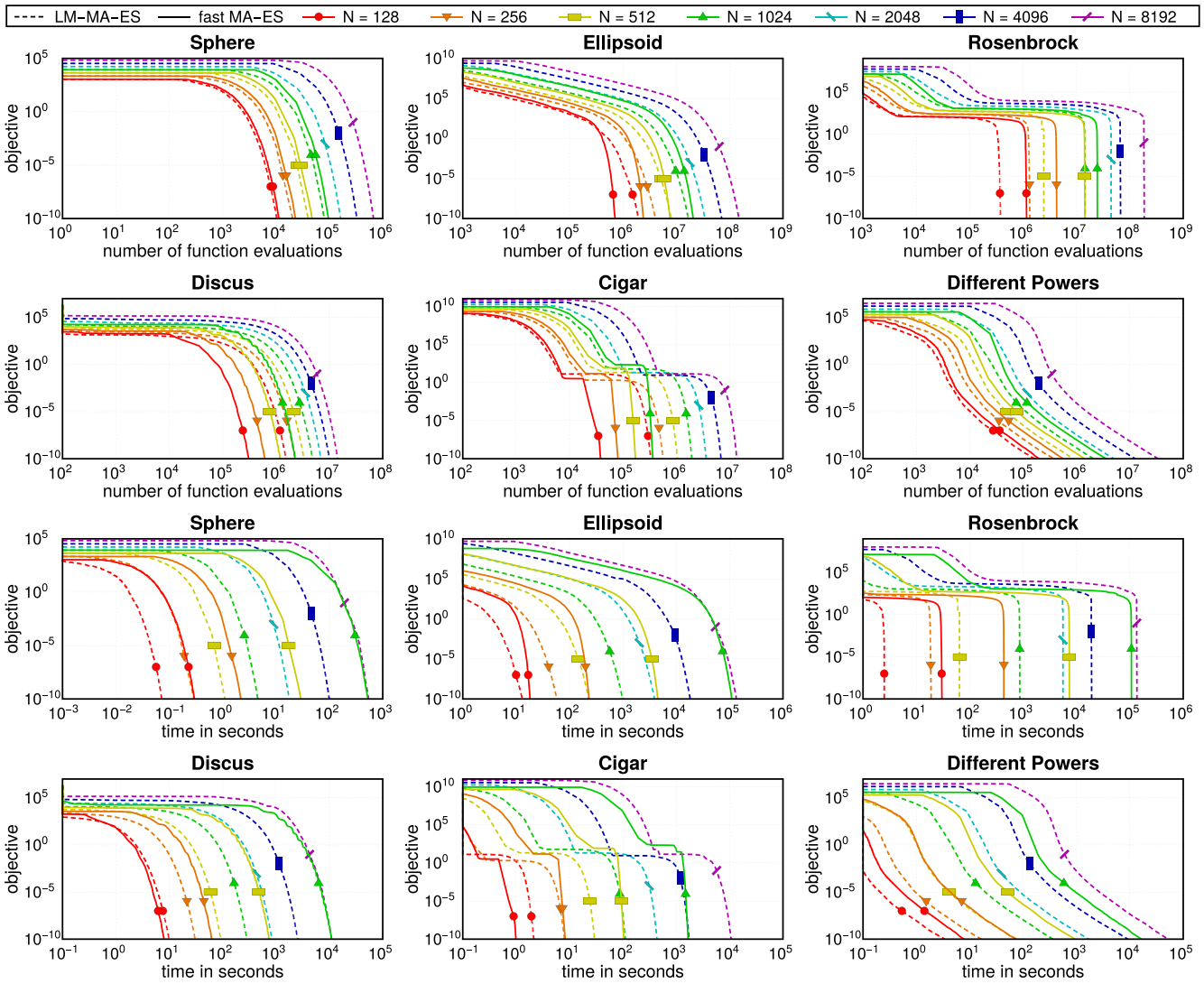


Fig. 2. Runtime in number of function evaluations (rows 1 and 2) and seconds (rows 3 and 4) of LM-MA-ES in 128 to 8192 dimensions and fast MA-ES in 128 to 1024 dimensions on six standard benchmark problems.

high dimensions where learning rates are generally small due to the sublinear sample size.

The setting of the hyperparameter $m$ affects the final time and memory complexity of LM-MA-ES. We suggest to set it to something between $m \in \mathcal{O}(1)$ and $m \in \mathcal{O}(\log(n))$, leading to $\mathcal{O}(n)$ and $\mathcal{O}(n(\log(n))$ time and space complexity, respectively. In this letter, we primarily focus on the latter case. Retuning $c_{d,i}$ and $d_{c,i}$ might be beneficial to adjust the algorithm to work best for a particular choice of $m$.

By construction, LM-MA-ES features all invariance properties of modern ESs, namely invariance to translation and rotation of $\mathbf{R}^n$, and strictly monotonic (rank-preserving) transformations of objective values. That is, when applied to transformed test functions, LM-MA-ES does not show performance degradation. A formal proof and an experimental validation of rotation invariance are found in the supplementary material.

## IV. Experimental Validation

With our experimental evaluation we aim to answer the following questions.

1) How does LM-MA-ES compare to MA-ES, i.e., what is the effect of modeling only an $m$-dimensional subspace?
2) How does LM-MA-ES compare to other algorithms designed for high-dimensional black-box optimization?

To answer these questions, we investigate the performance on large-scale variants ($n \in \{128, 256, \ldots, 8192\}$) of standard benchmark problems [27] (see Table I). Starting from the initial region $[-5,\ 5]^n$ containing the optimum with initial step size $\sigma = 3$ we optimize until reaching the (rather exact) target precision of $f_{\text{tar}} = 10^{-10}$. Similar sets of problems were used in the CEC competition on large scale global optimization [28], however, with a focus on (partial) separability. This is not a concern because LM-MA-ES is invariant to rotations. We focus on unimodal problems. It is straightforward (but outside the scope of this letter) to add well-established wrapper techniques like restarts [29] for improved global search behavior.

All strategy parameters of LM-MA-ES and MA-ES are given in Algorithm 1. We use LM-CMA-ES [17], VD-CMA-ES [16], and the active $(\mu/\mu_w, \lambda)$-CMA-ES [30], [31] (aCMA-ES, known to be up to two times more efficient than the default CMA-ES) as baselines. The source code of LM-MA-ES is available in the supplementary material.

Fig. 1 shows the effect of the $\mathcal{O}(n \log(n))$ scaling of the runtime per sample as compared to $\mathcal{O}(n^2)$ of fast-MA-ES (in the following

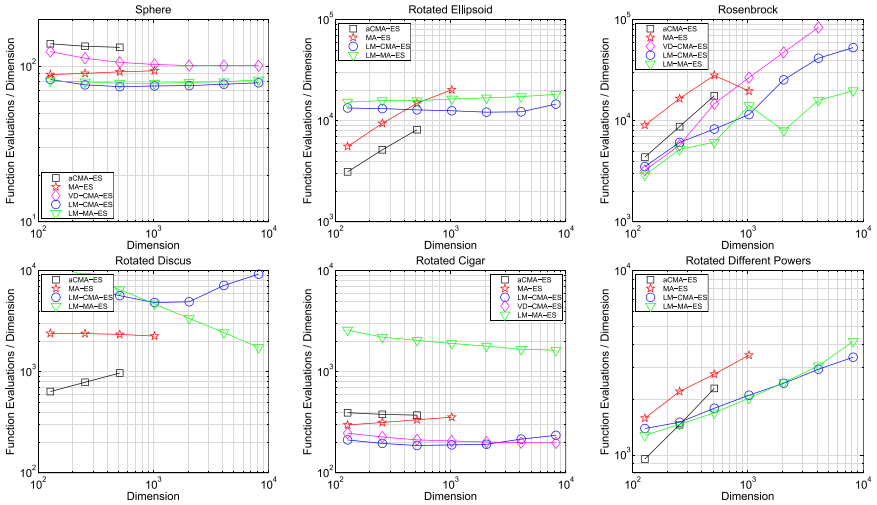


Fig. 3. Median number of function evaluations (out of 5 runs) required to achieve a target objective function value of $f_{\text{tar}} = 10^{-10}$. The results of VD-CMA-ES are shown only on functions where the algorithm succeeded achieving $f_{\text{tar}}$. Some results for CMA-ES and MA-ES are missing due to their extremely long runtimes (note that the vertical axis shows function evaluations, not runtime).

denoted as MA-ES), measured for implementations of the algorithms in plain C.

The much better internal scaling is of value only if the algorithm does not pay a too high price in terms of an increased number of function evaluations required to reach $f_{\text{tar}}$. Fig. 2 shows that LM-MA-ES performs surprisingly well: in some cases it even *saves* function evaluations, and this tends to happen more often for larger $n$. LM-MA-ES is always faster in terms of wall clock time, in some cases by a factor of 100.

Fig. 3 shows that LM-MA-ES scales favorably compared to LM-CMA-ES achieving better scaling on the Rosenbrock and Discus functions, but a worse scaling on Cigar. The latter result might be due to an improper setting of the strategy parameters, a problem that can most probably be fixed with the technique proposed in [24]. VD-CMA is not able to solve some rotated functions efficiently due to the restrictions on the covariance matrix that the algorithm assumes [16], [17].

## V. Conclusion

The recently proposed MA-ES is a simpler variant of the CMA-ES. We showed that the $\mathcal{O}(n^2)$ time and space complexity of MA-ES, which is prohibitive for large $n$, can be reduced to $\mathcal{O}(mn)$ with $m \ll n$ adopting the approach used in [17]. The proposed LM-MA-ES matches state-of-the-art results on large-scale optimization problems while being algorithmically simpler than LM-CMA-ES.

Future work should investigate to which extent the inclusion of the rank-$\mu$ update can improve the performance. The learning rates of LM-MA-ES can be optimized online as it is commonly done in self-adaptive evolutionary algorithms [32] or based on the maximum-likelihood principle [24].

The empirical evaluation presented in this letter is limited to state-of-the-art ESs, and our future work should consider a wider range of algorithms. Moreover, the currently used testbed is limited to relatively simple functions whose difficulty arises from the dimensionality considered here. In our previous work [17] we demonstrated that the performance of LM-CMA-ES on such functions can match L-BFGS and outperform the latter on a nonsmooth Nesterov function. It would be interesting to further analyze advantages of derivative-free algorithms over L-BFGS on nonsmooth functions.

A promising venue for LM-MA-ES would be applications to training of deep neural networks. It may accelerate stochastic gradient descent (SGD) for training deep neural networks by replacing the evolution path vectors by momentum vectors based on noisy batch gradients. The method could potentially represent an alternative to L-BFGS and numerous SGD variants with adaptive learning rates. It is also promising for direct policy search reinforcement learning.

## References

[1] L. Rastrigin, *In the World of Random Events*, Latvian Acad. Sci., USSR, Riga, Latvia, 1963.

[2] N. Hansen, D. V. Arnold, and A. Auger, "Evolution strategies," in *Springer Handbook of Computational Intelligence*. Berlin, Germany: Springer, 2015, pp. 871–898. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-662-43505-2_44#citeas

[3] I. Rechenberg, *Evolutionsstrategie: Optimierung Technischer Systeme Nach Prinzipien Der Biologischen Evolution*. Stuttgart, Germany: Frommann-Holzboog, 1973.

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

[5] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evol. Comput.*, vol. 11, no. 1, pp. 1–18, Mar. 2003.

[6] D. Wierstra *et al.*, "Natural evolution strategies," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 949–980, 2014.

[7] Y. Akimoto, Y. Nagata, I. Ono, and S. Kobayashi, "Bidirectional relation between CMA evolution strategies and natural evolution strategies," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 2010, pp. 154–163.

[8] Y. Ollivier, L. Arnold, A. Auger, and N. Hansen, "Information-geometric optimization algorithms: A unifying picture via invariance principles," *J. Mach. Learn. Res.*, vol. 18, no. 18, pp. 1–65, 2017.

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

[10] N. Hansen, A. Auger, R. Ros, S. Finck, and P. Pošík, "Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009," in *Proc. 12th Annu. Conf. Companion Genetic Evol. Comput. (GECCO)*, 2010, pp. 1689–1696.

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

[12] T. Salimans, J. Ho, X. Chen, and I. Sutskever, "Evolution strategies as a scalable alternative to reinforcement learning," *arXiv preprint arXiv:1703.03864*, 2017.

[13] J. N. Knight and M. Lunacek, "Reducing the space-time complexity of the CMA-ES," in *Proc. Genet. Evol. Comput. Conf.*, 2007, pp. 658–665.

[14] R. Ros and N. Hansen, "A simple modification in CMA-ES achieving linear time and space complexity," in *Parallel Problem Solving From Nature–PPSN*, G. Rudolph, T. Jansen, N. Beume, S. Lucas, and C. Poloni, Eds. Berlin, Germany: Springer, 2008, pp. 296–305.

[15] Y. Sun, F. Gomez, T. Schaul, and J. Schmidhuber, "A linear time natural evolution strategy for non-separable functions," *arXiv preprint arXiv:1106.1998*, 2011.

[16] Y. Akimoto, A. Auger, and N. Hansen, "Comparison-based natural gradient optimization in high dimension," in *Proc. Genet. Evol. Comput. Conf.*, 2014, pp. 373–380.

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

[18] Y. Akimoto and N. Hansen, "Online model selection for restricted covariance matrix adaptation," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 2016, pp. 3–13.

[19] D. Shanno, "Conditioning of Quasi–Newton methods for function minimization," *Math. Comput.*, vol. 24, no. 111, pp. 647–656, 1970.

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

[21] J. Jägersküpper, "Rigorous runtime analysis of the (1+1) ES: 1/5-rule and ellipsoidal fitness landscapes," in *Proc. Int. Workshop Found. Genet. Algorithms*, 2005, pp. 260–281.

[22] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evol. Comput.*, vol. 9, no. 2, pp. 159–195, Jun. 2001. [Online]. Available: http://dx.doi.org/10.1162/106365601750190398

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

[24] I. Loshchilov, M. Schoenauer, M. Sebag, and N. Hansen, "Maximum likelihood-based online adaptation of hyper-parameters in CMA-ES," in *Parallel Problem Solving From Nature–PPSN*, T. Bartz-Beielstein, J. Branke, B. Filipič, and J. Smith, Eds. Heidelberg, Germany: Springer, 2014, pp. 70–79.

[25] O. Krause, D. R. Arbonès, and C. Igel, "CMA-ES with optimal covariance update and storage complexity," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 370–378.

[26] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Math. Program.*, vol. 45, nos. 1–3, pp. 503–528, 1989.

[27] S. Finck, N. Hansen, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2010: Experimental setup," Res. Center PPE, Paris, France, Rep. 2009/21, 2010.

[28] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on large scale global optimization," Nature Inspired Comput. Appl. Lab., Univ. Sci. Technol. China, Hefei, China, Rep., 2009. [Online]. Available: http://staff.ustc.edu.cn/∼ketang/cec2012/lsgo_competition.htm

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

[30] N. Hansen and R. Ros, "Benchmarking a weighted negative covariance matrix update on the BBOB-2010 noiseless testbed," in *Proc. Genet. Evol. Comput. Conf.*, 2010, pp. 1673–1680.

[31] G. A. Jastrebski and D. V. Arnold, "Improving evolution strategies through active covariance matrix adaptation," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 2814–2821.

[32] H.-G. Beyer and K. Deb, "On self-adaptive features in real-parameter evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 5, no. 3, pp. 250–270, Jun. 2001.