Injecting CMA-ES into MOEA/D

Saúl Zapotecas-Martínez, Bilel Derbel, Arnaud Liefooghe, Dimo Brockhoff, Hernán Aguirre, Kiyoshi Tanaka

Injecting CMA-ES into MOEA/D

Saúl Zapotecas-Martínez
Shinshu University
Faculty of Engineering
4-17-1 Wakasato, Nagano,
380-8553, Japan
saul.zapotecas@gmail.com

Bilel Derbel
Univ. Lille, CRIStAL
Inria Lille Nord-Europe
Villeneuve d'Ascq, France
bilel.derbel@univ-lille1.fr

Arnaud Liefooghe
Univ. Lille, CRIStAL
Inria Lille Nord-Europe
Villeneuve d'Ascq, France
arnaud.liefooghe@univ-lille1.fr

Dimo Brockhoff
INRIA Lille Nord-Europe
Dolphin Team
Villeneuve d'Ascq, France
dimo.brockhoff@inria.fr

Hernán E. Aguirre
Shinshu University
Faculty of Engineering
4-17-1 Wakasato, Nagano,
380-8553, Japan
ahernan@shinshu-u.ac.jp

Kiyoshi Tanaka
Shinshu University
Faculty of Engineering
4-17-1 Wakasato, Nagano,
380-8553, Japan
ktanaka@shinshu-u.ac.jp

ABSTRACT

MOEA/D is an aggregation-based evolutionary algorithm which has been proved extremely efficient and effective for solving multi-objective optimization problems. It is based on the idea of decomposing the original multi-objective problem into several single-objective subproblems by means of well-defined scalarizing functions. Those single-objective subproblems are solved in a cooperative manner by defining a neighborhood relation between them. This makes MOEA/D particularly interesting when attempting to plug and to leverage single-objective optimizers in a multi-objective setting. In this context, we investigate the benefits that MOEA/D can achieve when coupled with CMA-ES, which is believed to be a powerful single-objective optimizer. We rely on the ability of CMA-ES to deal with injected solutions in order to update different covariance matrices with respect to each subproblem defined in MOEA/D. We show that by cooperatively evolving neighboring CMA-ES components, we are able to obtain competitive results for different multi-objective benchmark functions.

Categories and Subject Descriptors

I.2.8 [**Computing Methodologies**]: Artificial Intelligence—*Problem Solving, Control Methods, and Search.*

Keywords

Multi-objective Optimization, Decomposition-based MOEAs, Covariance Matrix Adaption Evolution Strategy.

1. INTRODUCTION

A multi-objective optimization problem (MOP) refers to the situation where several conflicting objectives are to be optimized simultaneously. In such a setting, solving a MOP consists in finding a whole set of solutions providing both good and diverse compromises with respect to the corresponding objectives. Computing such a set is actually a challenging task for which one can find different methodological frameworks and different algorithmic concepts. Evolutionary algorithms (EA) are particularly well-suited in order to compute an accurate approximation of the so-called Pareto set, that is the set of the best achievable objective trade-offs. Generally speaking, multi-objective EAs can be classified in different classes, ranging from dominance-based algorithms [4, 14], indicator-based algorithms [21, 18], and aggregation-based algorithms [12, 19]. In this paper, we are interested in the MOEA/D (multi-objective optimization based on decomposition) framework for which we can witness a growing interest from the community due to its simplicity and to its effectiveness when applied to a broad range of multi-objective optimization problems.

One can find several variants and implementations of MOEA/D, which are all based on seemingly the same concept. Instead of attempting to directly solve the MOP as a whole, all MOEA/D variants decompose the original MOP into several single-objective subproblems, and solve them cooperatively. More specifically, a set of weighting coefficient vectors is used to define different scalarized subproblems for which one solution is maintained and evolved over time. In this respect, any single-objective optimizer can be potentially plugged into MOEA/D as search engine for the single-objective subproblems. This has been done for example with polynomial mutation and SBX crossover [19] as well as differential evolution (DE) [15], where the latter is believed to be among the best performing implementations [15]. However, this inclusion of single-objective optimizers might not always be straightforward due to two additional specificities of the MOEA/D framework related to the *cooperativity* among the search process of the scalarized problems.

Instead of solving every subproblem independently, MOEA/D defines a neighborhood relation among them, and allows solutions to be exchanged between neighboring problems in two ways. On the one hand, the currently best-known solution of a neighboring problem can be employed to change ones own search distribution,

e.g., by using them as parents in a crossover operator. On the other hand, newly generated solutions are also allowed to be actively transferred to a neighboring subproblem in order to replace its current best solution. On an abstract level, we can say that in MOEA/D, the single-objective optimizers applied at each subproblem are not only acting selfishly to improve the solution of their own problem by "stealing information" from others, but they also behave in an altruistic way by helping to improve the solutions of their neighbors.

In this paper, we investigate new opportunities offered by the flexibility of MOEA/D in incorporating novel single-objective optimizers. More specifically, we explore the strengths of the well-established CMA-ES (Covariance Matrix Adaption Evolution Strategy) algorithm considered as a state-of-the-art optimizer for single-objective blackbox continuous problems [9]. Our interest in combining the CMA-ES with MOEA/D stems from two sources. On the one hand, CMA-ES has been shown experimentally to be among the best-performing single-objective blackbox algorithms on the well-established BBOB testbed, with typically superior performance to DE variants and other numerical optimizers—especially when the problems are difficult and the budgets are not too small [7]. On the other hand, a recent paper by Hansen [6] shows that external solutions can be easily *injected* into the algorithm to gain from good candidate solutions that are not sampled directly from the algorithm itself. Both aspects together make the CMA-ES with injection a highly interesting candidate to be used within MOEA/D and its cooperative optimization in which solutions are exchanged between single-objective subproblems.

In this paper, we therefore propose a novel variant of MOEA/D where CMA-ES is used as the core single-objective evolution engine and where the injection idea allows to incorporate information from neighboring scalarizing problems into the search distributions. We show that by appropriately injecting solutions coming from the CMA-ES sampling process into neighboring subproblems, we can derive novel effective variation mechanisms which are compatible with the decomposition-based concepts used with the MOEA/D framework. To assess the validity of our approach, we conduct an experimental study where, in particular, we analyze the performance of the proposed MOEA/D-CMA algorithm compared to MOEA/D-DE using the CEC 2009 box-constrained benchmark functions. Besides being able to obtain competitive results, the investigations conducted in this paper highlight promising alternatives for designing novel efficient multi-objective optimization algorithms falling in the class of aggregation-based EAs.

Note that already several multi-objective versions of the CMA-ES algorithm exist [10, 11] which, however, do not resemble the framework of MOEA/D but instead aim at maximizing the hypervolume of a solution set in the framework of indicator-based algorithms. These algorithms are not in the focus of this paper although a numerical comparison would be interesting for the future.

The rest of this paper is organized as follows. In Section 2, we review some basic concepts related to multi-objective optimization as well as to the MOEA/D framework. In Section 3, we recall the main algorithmic components of the single objective CMA-ES as well as the idea of injection that will serve as a basis towards its successful incorporation into MOEA/D. In Section 4, we describe our proposed MOEA/D-CMA approach and discuss its design components in detail. In Section 5, we describe our experimental study and discuss our main findings. Section 6 finally concludes the paper and discusses related open research directions.

2. BASIC CONCEPTS

Algorithm 1: General Framework of MOEA/D

Input:
N: the number of subproblems to be decomposed;
W: a well-distributed set of weight vectors $\{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$;
T: the neighborhood size.
Output:
P: the final approximation to PS.

```
1  z = (z_1 = +∞, ..., z_k = +∞)^T;
2  Generate a random set of solutions P = {x^1, ..., x^N} in Ω;
3  for i = 1, ..., N do
4      B_i ← {i_1, ..., i_T}, such that: w^{i_1}, ..., w^{i_T} are the T closest weight
       vectors to w^i;
5      z_j ← min(z_j, f_j(x^i));                          // for j = 1, ..., k
6  while stopping criterion is not satisfied do
7      for x^i ∈ P do
8          Reproduction: Randomly select two indexes k, l from B_i, and
           then generate a new solution y from x^k and x^l by using
           genetic operators.
9          Mutation: Apply a mutation operator on y to produce y'.
10         Update of z: z_j ← min(z_j, f_j(x^i));          // for j = 1, ..., k
11         Update of Neighboring Solutions: For each index j ∈ B_i, if
           g(y'|w^j, z) < g(x^i|w^j, z), then set x^j = y';
12 return P = {x^1, ..., x^N};
```

2.1 Multi-objective Optimization

Assuming minimization, a continuous multi-objective optimization problem (MOP), can be stated as:

$$\underset{\mathbf{x}\in\Omega}{\text{minimize}} \quad \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_k(\mathbf{x}))^\intercal \tag{1}$$

where $\Omega \subset \mathbb{R}^n$ defines the decision space and $\mathbf{F}$ is defined as the vector of the objective functions where each $f_j : \Omega \to \mathbb{R}$ $(j = 1, \ldots, k)$ is the function to be minimized. In this paper we consider the box-constrained case, i.e., $\Omega = \prod_{i=1}^{n}[a_j, b_j]$. Therefore, each variable vector $\mathbf{x} = (x_1, \ldots, x_n)^\intercal \in \Omega$ is such that $a_i \leq x_i \leq b_i$ for all $i \in \{1, \ldots, n\}$. In order to describe the concept of optimality in which we are interested, the following definitions are introduced [16].

Definition 1. *Let* $\mathbf{x}, \mathbf{y} \in \Omega$*, we say that* $\mathbf{x}$ dominates $\mathbf{y}$ *(denoted by* $\mathbf{x} \prec \mathbf{y}$*) with respect to the problem defined in equation* (1) *if and only if:* 1) $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$ *for all* $i \in \{1, \ldots, k\}$ *and* 2) $f_j(\mathbf{x}) < f_j(\mathbf{y})$ *for at least one* $j \in \{1, \ldots, k\}$.

Definition 2. *Let* $\mathbf{x}^\star \in \Omega$*, we say that* $\mathbf{x}^\star$ *is a* Pareto optimal *solution, if there is no other solution* $\mathbf{y} \in \Omega$ *such that* $\mathbf{y} \prec \mathbf{x}^\star$.

Definition 3. *The* Pareto optimal set *PS is defined by:* $PS = \{\mathbf{x} \in \Omega | \mathbf{x}$ is a Pareto optimal solution$\}$ *and its image* $PF = \{\mathbf{F}(\mathbf{x}) | \mathbf{x} \in PS\}$ *is called* Pareto front *PF*.

2.2 MOEA/D

The Multi-Objective Evolutionary Algorithm Based on Decomposition (MOEA/D) [19], transforms a MOP into several scalarizing subproblems. Therefore, an approximation of the Pareto set is obtained by solving the N scalarizing subproblems in which a MOP is decomposed.

Considering $W = \{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$ as the well-distributed set of weighting coefficient vectors, MOEA/D seeks the best solution to each subproblem defined by each weight vector using the Penalty Boundary Intersection (PBI) approach [19], which is in the form:

$$\text{minimize:} \quad g^{pbi}(\mathbf{x}|\mathbf{w}^i, \mathbf{z}) = d_1 + \theta d_2 \tag{2}$$

such that

$$d_1 = \frac{|(\mathbf{F}(\mathbf{x}) - \mathbf{z})^\intercal \mathbf{w}|}{\|\mathbf{w}\|} \text{ and } d_2 = \left\| (\mathbf{F}(\mathbf{x}) - \mathbf{z}) - d_1 \frac{\mathbf{w}}{\|\mathbf{w}\|} \right\|$$

where $\mathbf{x} \in \Omega \subset \mathbb{R}^n$ and $z_j = \min\{f_j(\mathbf{x}) | \mathbf{x} \in \Omega\}$ for $i = 1, \ldots, N$ and $j = 1, \ldots, k$. Since $\mathbf{z} = (z_1, \ldots, z_k)^\intercal$ is unknown, MOEA/D states each component z_j by the minimum value for each objective f_j found during the search, for $j = 1, \ldots, k$.

In MOEA/D, a neighborhood of a weight vector $\mathbf{w}^i$ is defined as a set of its closest weight vectors in W. Therefore, the neighborhood of the weight vector $\mathbf{w}^i$ contains all the indexes of the T closest weight vectors to $\mathbf{w}^i$.

Throughout the evolutionary process, MOEA/D finds the best solution to each subproblem maintaining a population of N solutions $P = \{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$ where $\mathbf{x}^i \in \Omega$ is the current solution to the i^{th} subproblem. Algorithm 1 presents the general framework of MOEA/D, however, for a more detailed description of this algorithm the interested reader is referred to [19].

3. CMA-ES

The Covariance Matrix Adaptation Evolution Strategy (CMA-ES, [9]) is one of the state-of-the-art numerical blackbox optimization algorithms available—outperforming other algorithms especially for difficult functions and larger budgets [7]. To minimize a function $f : \mathbb{R}^n \to \mathbb{R}$, CMA-ES iteratively samples λ solutions from a multivariate normal distribution, parameterized by a mean vector $\mathbf{m}_\tau \in \mathbb{R}^n$, a step size $\sigma_\tau > 0$, and a covariance matrix $\mathbf{C}_\tau \in \mathbb{R}^{n \times n}$ at time step τ. After the evaluation and ranking of the λ candidate solutions, their relative steps (between the sampled points and the old mean, sorted according to the solution's objective function value) are used to update the parameters of the sampling distribution. This loop of sampling and sample distribution update is repeated until any of several stopping criteria is met.

More concretely, the CMA-ES version, employed in this paper and following [5], works as follows. In the initialization step at $\tau = 0$, the mean $\mathbf{m}_0$ is typically sampled uniformly at random in a bounded search domain $[a, b] \in \mathbb{R}^n$, the initial covariance matrix $\mathbf{C}_0$ is chosen as the identity matrix, and the evolution paths $\mathbf{p}_\rho$ and $\mathbf{p}_c$ are set to $\mathbf{0} \in \mathbb{R}^n$.

At each iteration, CMA-ES then samples $\lambda \geq 1$ candidate solutions $\mathbf{x}^i \in \mathbb{R}^n$ $(1 \leq i \leq \lambda)$ according to a multivariate normal distribution $(\mathbf{x}^i \sim \mathcal{N}(\mathbf{m}_\tau, \sigma_\tau^2 \mathbf{C}_\tau) \sim \mathbf{m}_\tau + \sigma_\tau \cdot \mathcal{N}(\mathbf{0}, \mathbf{C}_\tau))$.

The candidate solutions $\mathbf{x}^i$ are evaluated with respect to the objective function $f : \mathbb{R}^n \to \mathbb{R}$ and ranked according to f-values where we use the notation $\mathbf{x}^{i:\lambda}$ for the ith best solution (i.e. $f(\mathbf{x}^{1:\lambda}) \leq \cdots \leq f(\mathbf{x}^{\mu:\lambda}) \leq f(\mathbf{x}^{\mu+1:\lambda}) \leq \cdots \leq f(\mathbf{x}^{\lambda:\lambda})$.

The new mean vector is computed via *weighted recombination* of the μ best candidate solutions as:

$$\begin{aligned} \mathbf{m}_{\tau+1} &= \sum_{i=1}^{\mu} \omega_i \mathbf{x}^{i:\lambda} \\ &= \mathbf{m}_\tau + \sum_{i=1}^{\mu} \omega_i (\mathbf{x}^{i:\lambda} - \mathbf{m}_\tau) \end{aligned} \tag{3}$$

where the positive (recombination) weights $\omega_i > 0$ with $\sum_{i=1}^{\mu} \omega_i = 1$ are typically chosen linearly on the log-scale with $\mu \leq \lambda/2$.

The step size σ_τ is updated using *cumulative step size adaptation* (CSA). The evolution path (or search path) $\mathbf{p}_\sigma$ is thereby firstly updated as:

$$\mathbf{p}_\sigma \leftarrow (1 - c_\sigma)\mathbf{p}_\sigma + \sqrt{1 - (1 - c_\sigma)^2}\, \sqrt{\mu_{\text{eff}}}\, \mathbf{C}_\tau^{-1/2} \frac{\mathbf{m}_{\tau+1} - \mathbf{m}_\tau}{\sigma_\tau} \tag{4}$$

with the help of which the step size is finally updated as:

$$\sigma_{\tau+1} = \sigma_\tau \times \exp\left(\frac{c_\sigma}{d_\sigma} \times \left(\frac{\|\mathbf{p}_\sigma\|}{E\|\mathcal{N}(\mathbf{0}, \mathbf{I})\|} - 1 \right) \right) \tag{5}$$

where $c_\sigma^{-1} \approx n/3$ is the backward time horizon for the evolution path $\mathbf{p}_\sigma$ and larger than one, $\mu_{\text{eff}} = \left(\sum_{i=1}^{\mu} \omega_i^2 \right)^{-1}$ is the variance effective selection mass and $1 \leq \mu_{\text{eff}} \leq \mu$ by definition of ω_i, $\mathbf{C}_\tau^{-1/2} = \sqrt{\mathbf{C}_\tau^{-1}}$ is the unique symmetric square root of the inverse of $\mathbf{C}_\tau$, and d_σ is a damping parameter usually close to one. For $d_\sigma = \infty$ or $c_\sigma = 0$ the step size remains unchanged. Note that the step size σ_τ is increased if and only if $\|\mathbf{p}_\sigma\|$ is larger than the expected step length of a fully random sample:

$$\begin{aligned} E\|\mathcal{N}(0, I)\| &= \sqrt{2}\, \Gamma((n+1)/2)/\Gamma(n/2) \\ &\approx \sqrt{n}\, (1 - 1/(4\,n) + 1/(21\,n^2)) \end{aligned} \tag{6}$$

and decreased if it is smaller. For this reason, the step size update tends to make consecutive steps $\mathbf{C}_\tau^{-1}$-conjugate, in that after the adaptation has been successful, $\left(\frac{\mathbf{m}_{\tau+2} - \mathbf{m}_{\tau+1}}{\sigma_{\tau+1}} \right)^\lambda \mathbf{C}_\tau^{-1} \frac{\mathbf{m}_{\tau+1} - \mathbf{m}_\tau}{\sigma_\tau} \approx 0$ holds.

Finally, the covariance matrix is updated by means of *rank-one* and *rank-μ* updates for which, again, the respective evolution path is firstly updated as:

$$\mathbf{p}_c \leftarrow (1 - c_c)\mathbf{p}_c + \mathbf{1}_{[0, \alpha\sqrt{n}]}(\|\mathbf{p}_\sigma\|) \sqrt{1 - (1 - c_c)^2}\, \sqrt{\mu_{\text{eff}}} \frac{\mathbf{m}_{\tau+1} - \mathbf{m}_\tau}{\sigma_\tau} \tag{7}$$

which is used to update the covariance matrix as:

$$\begin{aligned} \mathbf{C}_{\tau+1} = (1 - c_1 - c_\mu + c_s)\mathbf{C}_\tau + c_1 \underbrace{\mathbf{p}_c \mathbf{p}_c^\intercal}_{\text{rank-one update}} \\ + c_\mu \underbrace{\sum_{i=1}^{\mu} \omega_j \frac{\mathbf{x}^{i:\lambda} - \mathbf{m}_\tau}{\sigma_\tau} \left(\frac{\mathbf{x}^{i:\lambda} - \mathbf{m}_\tau}{\sigma_\tau} \right)^\intercal}_{\text{rank-}\mu\text{ update}} \end{aligned} \tag{8}$$

where $^\intercal$ denotes the transpose and $c_c^{-1} \approx n/4$ is the backward time horizon for the evolution path $\mathbf{p}_c$ and larger than one, $\alpha \approx 3/2$ and the indicator function $\mathbf{1}_{[0, \alpha\sqrt{n}]}(\|\mathbf{p}_\sigma\|)$ evaluates to one iff $\|\mathbf{p}_\sigma\| \in [0, \alpha\sqrt{n}]$ or, in other words, $\|\mathbf{p}_\sigma\| \leq \alpha\sqrt{n}$, which is usually the case. The constant $c_s = (1 - \mathbf{1}_{[0, \alpha\sqrt{n}]}(\|\mathbf{p}_\sigma\|)^2)\, c_1 c_c (2 - c_c)$ makes partly up for the small variance loss in case the indicator is zero, $c_1 \approx 2/n^2$ is the learning rate for the rank-one update of the covariance matrix and $c_\mu \approx \mu_{\text{eff}}/n^2$ is the learning rate for the rank-μ update of the covariance matrix and must not exceed $1 - c_1$.

This completes an iteration of CMA-ES which continues with the sampling of new solutions and updates of the sample distribution parameters until a stopping criterion is met (see Sec. 4 for details). For a more detailed description, the interested readers are referred to [5].

CMA-ES with Injection. If external solutions are available to CMA-ES, for example from a gradient step or from evaluating a meta-model of the (expensive) objective function, these solutions can be directly used in the update of the sampling distribution's parameters. The only change to make the above algorithm work when handling such "injections" of solutions—as argued in [6]—is to restrict the distance between injected solution and previous mean and to rescale the corresponding search step accordingly before taking it into account in the updates of mean, step size, and covariance matrix.

4. MOEA/D-CMA

The proposed MOEA/D with Covariance Matrix Adaption Evolution Strategy (MOEA/D-CMA) is given in the template of Algorithm 2. As we can see, the proposed approach consists in an initialization step, recombination step and a specific CMA evolution process. In the following, we will provide a step-by-step descrip-

Procedure `Initialize`$(\langle \mathbf{p}_c^i, \mathbf{p}_\sigma^i, \mathbf{C}^i, \mathbf{m}^i, \sigma_i, \tau_i \rangle, \mathbf{x}^i, \sigma_{\text{init}})$

$\mathbf{p}_c^i \leftarrow \mathbf{p}_\sigma^i \leftarrow \mathbf{0}$;	// Initial cumulative paths
$\mathbf{C}^i \leftarrow \mathbf{I}$;	// Initial covariance matrix
$\mathbf{m}^i \leftarrow \mathbf{x}^i$, $\sigma_i \leftarrow \sigma_{\text{init}}$;	// Initial mean and sigma
$\tau_i \leftarrow 0$;	// Initial iteration

tion of the different components involved in our proposed approach.

Preliminary Considerations and Initialization. MOEA/D-CMA decomposes the problem (1) into N single-objective optimization subproblems. It is in fact based on MOEA/D in the sense that it updates a neighborhood of solutions which solves a set of neighboring subproblems. However, instead of using genetic operators (crossover and mutation), each subproblem evolves the parameters of a multivariate normal distribution and samples according to this distribution, as presented in Section 3. Therefore, analogously to MOEA/D, MOEA/D-CMA employs a well-distributed set of weight vectors $W = \{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$ to define the set of single-objective subproblems to be optimized cooperatively. Each weight vector $\mathbf{w}^i$ defines a scalarized function by using the PBI approach (equation (2)). However, the use of other scalarized functions is also possible, see for example those presented in [16]. Each of the subproblems is then solved by a CMA-ES instance, involving a mean vector, step size, covariance matrix, and the corresponding cumulation paths as presented in Section 3. The i^{th} individual is thereby denoted by the six-tuple $\langle \mathbf{p}_c^i, \mathbf{p}_\sigma^i, \mathbf{C}^i, \mathbf{m}^i, \sigma_i, \tau_i \rangle$ with τ_i being an iteration counter.

Let us consider $P = \{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$ as the set of initial random solutions generated in the feasible search space Ω. At the beginning, the reference point $\mathbf{z}$ is set with infinite values and then, its jth component is updated with the minimum value of the jth objective function f_j found along the optimization process. Analogously to MOEA/D, a neighborhood B_i which contains the indexes of the T closest weight vectors to $\mathbf{w}^i$ is defined. Each solution in P defines the initial mean of each CMA-ES's search distribution. The covariance matrices, step sizes and evolution paths are initialized the same for each subproblem as shown in procedure `Initialize`.

Recombination. In **Step 1** of Algorithm 2, the ith subproblem generates a set of solutions $\mathbf{V}^i$ by using the multivariate normal distribution which is in the form $\mathcal{N}(\mathbf{m}^i, \sigma_i^2 \mathbf{C}^i)$[1]. It is worth noticing that the new solutions in $\mathbf{V}^i$ could be located outside of the feasible region. In this case, we substitute each infeasible solution by its closest vector in the feasible region, which is denoted by $\mathbf{V}^i_{\text{rep}}$. If the solution is feasible we set $\mathbf{v}_{\text{rep}} = \mathbf{v}$, this procedure is referred to as `Repair` in Algorithm 2.

After sampling new candidate solutions, we update the neighboring solutions maintained at each subproblem. For this purpose we adopt the update mechanism proposed in MOEA/D-DE [15], where a maximum number of replacements n_r (lines 20–21) and a dynamic selection of neighborhood (line 18) are employed. In this way, the probability of losing diversity is reduced and the preservation of the best solutions to each neighboring subproblem is maintained during the sampling procedure.

It is important to note that in order to avoid stagnation and allow for restarts of the single optimization runs in each subproblem, we adopt a reset procedure denoted by `ResetCriteria` in line 12. This procedure first checks the following four criteria taken from the literature (e.g., see [1]):

1. `NoEffectCoord`. Reset if adding 0.2-standard deviations in any single coordinate does not change $\mathbf{m}^i$ (i.e. m_j equals $m_j^i + 0.2\sigma c_{j,j}$ for any $j = 1, \ldots, n$).
2. `NoEffectAxis`. Reset if adding a 0.1-standard deviation vector in any principal axis direction of $\mathbf{C}^i$ does not change $\mathbf{m}^i$. More formally, stop if $\mathbf{m}^i$ equals $\mathbf{m}^i + 0.1\sigma\sqrt{d_{jj}}\mathbf{b}_j$, where $j = (\tau \mod n) + 1$ and d_{jj} and $\mathbf{b}_j$ are the j^{th} eigenvalue and eigenvector of $\mathbf{C}^i$, with $\|\mathbf{b}_j\| = 1$, respectively.
3. `TolXUp`. Stop/Reset if $\sigma \cdot \max(\text{diag}(D))$ increased by more than 10^4.
4. `ConditionCov`. Stop/Reset if the condition number of the covariance matrix exceeds 10^{14}.

Therefore, if the components of the ith tuple satisfies one of the above criteria, the tuple $\langle \mathbf{p}_c^i, \mathbf{p}_\sigma^i, \mathbf{C}^i, \mathbf{m}^i, \sigma_i, \tau_i \rangle$ is reset using the same `Initialize` procedure as introduced previously. It is worth noticing that with each reinitialization, we reset the step size σ_i to the initial value divided by a factor of two, and using the current best solution for this subproblem as the new initial mean before continuing. More precisely, the ith mean is set as $\mathbf{m}^i = \mathbf{x}^i$ and $\sigma_i = \sigma_{\text{init}}/2$.

Evolution Strategy. After performing the recombination and replacement stage, we consider to evolve and update the components of every single CMA-ES optimizer at each single-objective subproblem. This is the main aim of **Step 2**. Generally speaking, we adopt a variant of the original CMA-ES presented in [6], where the injection of external solutions into the evolution of the covariance matrix is possible. As it was pointed out by Hansen [6], the injection of solutions in the evolutionary process of CMA-ES could improve the adaptation of the matrix if the injected solutions provide sufficient information about the fitness landscape. Our main concern is then to design an accurate injection mechanism which can deal in a proper way with our multi-objective setting.

We argue that, under some assumptions, from the Karush-Kuhn-Tucker conditions, it can be deduced that the PS of a continuous MOP with k objectives forms a $(k-1)$-dimensional piecewise continuous manifold in decision variable space [16]. It means that an optimal solution of the scalarizing problem defined by a weight vector $\mathbf{w}^p$ is close to the one defined by another weight vector $\mathbf{w}^q$ ($p \neq q$), if $\mathbf{w}^p$ and $\mathbf{w}^q$ are close to each other.

We hypothesize that during the search, the sample distribution of each subproblem converges to the region in which the optimal solution to each subproblem is located. Therefore, considering continuous MOPs and having the reference to the neighboring candidate solutions to each subproblem, the solutions to be injected are chosen precisely from this neighborhood. With that, a set of promising solutions are injected while eventually optimizing each separate subproblem.

Injecting Solutions. In **Step 2** of Algorithm 2, the set $\mathbf{Q}$ denotes the set of candidate solutions to be injected in the adaptation of the covariance matrix. This set of solutions, in fact, contains the samples, generated by the ith subproblem, and the best solution for the ith scalarized subproblem (defined by $\mathbf{w}^i$) found among the samples $\mathbf{V}^j$ (for $j \in B_i$). Since we are injecting (hypothetically) promising solutions, we consider the repaired solutions $\mathbf{V}^j_{\text{rep}}$. Note

[1]Note that the sampling is implemented via an Eigen decomposition of the covariance matrix into $\mathbf{C}^i = \mathbf{B}\mathbf{D}\mathbf{B}^{-1}$ and then sampling according to $\mathbf{m}^i + \sigma^i\mathbf{B}\mathbf{D}\mathcal{N}(\mathbf{0}, \mathbf{I})$ with $\mathbf{B}$ an orthonormal basis of eigenvectors and $\mathbf{D}$ a diagonal matrix containing the corresponding positive eigenvalues.

Algorithm 2: MOEA/D-CMA+I

Input:
N: the number of subproblems to be decomposed;
W: a well-distributed set of weight vectors $W = \{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$;
T: the neighborhood size.
Output:
P: the final approximation to PS;

1 Initialization
2 $\mathbf{z} = (z_1 = +\infty, \ldots, z_k = +\infty)^\intercal$;
3 Generate a random set of solutions $P = \{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$ in Ω;
4 **for** $i = 1, \ldots, N$ **do**
5 $\quad z_l = \min(z_l, f_l(\mathbf{x}^i))$; // for $l = 1, \ldots, k$
6 $\quad B_i \leftarrow \{i_1, \ldots, i_T\}$, such that: $\mathbf{w}^{i_1}, \ldots, \mathbf{w}^{i_T}$ are the T closest weight vectors to $\mathbf{w}^i$;
7 $\quad$ `Initialize` $(\langle \mathbf{p}_c^i, \mathbf{p}_\sigma^i, \mathbf{C}^i, \mathbf{m}^i, \sigma_i, \tau_i \rangle, \mathbf{x}^i, \sigma_{\text{init}})$
8 Evolution Strategy
9 **while** *stopping criterion is not satisfied* **do**
10 $\quad$ **Step 1.** Reproduction and Replacement
11 $\quad$ **for** $i = 1, \ldots, N$ **do**
12 $\quad\quad$ `ResetCriteria`$(\langle \mathbf{p}_c^i, \mathbf{p}_\sigma^i, \mathbf{C}^i, \mathbf{m}^i, \sigma_i, \tau_i \rangle, \mathbf{x}^i, \sigma_{\text{init}}/2)$;
13 $\quad\quad \mathbf{V}^i = \mathbf{V}^i_{\text{rep}} = \emptyset$;
14 $\quad\quad$ **for** $j = 1, \ldots, \lambda$ **do**
15 $\quad\quad\quad \mathbf{V}^i \leftarrow \mathbf{V}^i \cup \{\mathbf{v}^j\}$, where $\mathbf{v}^j \leftarrow\sim \mathcal{N}(\mathbf{m}^i, \sigma_i^2 \mathbf{C}^i)$;
16 $\quad\quad\quad \mathbf{V}^i_{\text{rep}} \leftarrow \mathbf{V}^i_{\text{rep}} \cup \{\mathbf{v}^j_{\text{rep}}\}$, where $\mathbf{v}^j_{\text{rep}} \leftarrow$ `Repair`$(\mathbf{v}^j)$;
17 $\quad\quad\quad z_l \leftarrow \min(z_l, f_l(\mathbf{v}^j_{\text{rep}}))$; // for $l = 1, \ldots, k$
18 $\quad\quad\quad$ **if** $rand() < \delta$ **then** $\pi \leftarrow B_i$ **else** $\pi \leftarrow \{1, \ldots, N\}$;
19 $\quad\quad\quad c \leftarrow 0$;
20 $\quad\quad\quad$ **foreach** $l \in \pi$ **do**
21 $\quad\quad\quad\quad$ **if** $g^{pbi}(\mathbf{v}^j_{rep}|\mathbf{w}^l, \mathbf{z}) < g^{pbi}(\mathbf{x}^l|\mathbf{w}^l, \mathbf{z})$ *and* $c < n_r$ **then** $\mathbf{x}^l \leftarrow \mathbf{v}^j_{\text{rep}}$ and $c \leftarrow c + 1$;
22 $\quad$ **Step 2.** Covariance Matrices Adaptation
23 $\quad$ **for** $i = 1, \ldots, \ldots, N$ **do**
24 $\quad\quad$ **if** $rand() < \delta$ **then** $\pi \leftarrow B_i$ **else** $\pi \leftarrow \{1, \ldots, N\}$;
25 $\quad\quad \mathbf{Q} \leftarrow \mathbf{V}^i$;
26 $\quad\quad$ 2.1. Injecting Solutions
27 $\quad\quad$ **foreach** $j \in \pi$ **do**
28 $\quad\quad\quad \mathbf{Q} \leftarrow \mathbf{Q} \cup \{\mathbf{v}^\star\}$, such that: $\mathbf{v}^\star = \arg\min_{\mathbf{v} \in \mathbf{V}^j_{\text{rep}}} g(\mathbf{v}|\mathbf{w}^i, \mathbf{z})$;
29 $\quad\quad$ 2.2. Covariance matrix adaptation
30 $\quad\quad \mathbf{y}^j \leftarrow \dfrac{\mathbf{q}^{j:|Q|} - \mathbf{m}^i}{\sigma_i}$ where:
$\quad\quad f_{\text{fit}}(\mathbf{q}^{1:|Q|}|\mathbf{w}^i, \mathbf{z}) \leq \cdots \leq f_{\text{fit}}(\mathbf{q}^{\mu:|Q|}|\mathbf{w}^i, \mathbf{z}) \leq \cdots$ and $\mathbf{q}^j \in Q$;
31 $\quad\quad \mathbf{y}^j \leftarrow \alpha_{\text{clip}}\left(c_y, \|(\mathbf{C}^i)^{-1/2}\mathbf{y}^j\|\right) \times \mathbf{y}^j$, if $\mathbf{q}^{j:|Q|}$ was injected;
32 $\quad\quad \Delta\mathbf{m} \leftarrow \sum_{j=1}^{\mu} \omega_j \mathbf{y}^j$;
33 $\quad\quad \mathbf{m}^i \leftarrow \mathbf{m}^i + c_{\text{m}} \sigma_i \Delta\mathbf{m}$;
34 $\quad\quad \mathbf{m}^i_{\text{rep}} \leftarrow \text{Repair}(\mathbf{m}^i)$;
35 $\quad\quad$ `UpdateStepSize`$(\mathbf{C}^i, \Delta\mathbf{m}, \mathbf{p}^i_\sigma, \sigma_i)$;
36 $\quad\quad$ `UpdateCovarianceMatrix`$(\mathbf{C}^i, \mathbf{p}^i_c, \Delta\mathbf{m}, \mathbf{y}^1, \ldots, \mathbf{y}^\mu)$;
37 $\quad\quad \tau_i \leftarrow \tau_i + 1$;
38 $\quad\quad$ **for** $l \in \pi$ **do**
39 $\quad\quad\quad$ **if** $g^{pbi}(\mathbf{m}^i_{rep}|\mathbf{w}^l, \mathbf{z}) < g^{pbi}(\mathbf{x}^l|\mathbf{w}^l, \mathbf{z})$ **then** $\mathbf{x}^l \leftarrow \mathbf{v}^j_{\text{rep}}$;
40 **return** $P \leftarrow \{\mathbf{x}^1, \ldots, \mathbf{x}^N\}$;

that the samples $\mathbf{V}^i$ are the samples given by the ith subproblem and $i \neq j$.

To obtain a ranking among the to-be-injected solutions, the sorting is carried out by using the definition of an auxiliary fitness function adopted from [8], which allows us, to deal with the box-constrained case, and it is stated as follows.

$$f_{\text{fit}}(\mathbf{q}^j|\mathbf{w}^i, \mathbf{z}) = g^{pbi}(\mathbf{q}^j_{\text{rep}}|\mathbf{w}^i, \mathbf{z}) + \alpha\|\mathbf{q}^j - \mathbf{q}^j_{\text{rep}}\|^2 \tag{9}$$

It is worth noticing that line 31 is introduced as part of the updated process of the CMA-ES with injection [6], where $\alpha_{\text{clip}}(c, x) = 1 \wedge \frac{c}{x}$ and the notation $a \wedge bc + d$ refers to the minimum of a and $bc + d$. In this way, the tuple $\langle \mathbf{p}_c^i, \mathbf{p}_\sigma^i, \mathbf{C}^i, \mathbf{m}^i, \sigma_i, \tau_i \rangle$ is updated by means of equations provided by [6], which are referred to as the `UpdateStepSize`. More precisely, the step size σ_i and the corresponding evolution path $\mathbf{p}_c^i$ are updated according to the following equations.

$$\mathbf{p}_\sigma^i \leftarrow (1 - c_\sigma)\mathbf{p}_\sigma^i + \sqrt{c_\sigma(2 - c_\sigma)\mu_{\text{eff}}}(\mathbf{C}^i)^{-1/2}\Delta\mathbf{m} \tag{10}$$

$$\sigma_i \leftarrow \sigma_i \times \exp\left(\Delta_\sigma^{\max} \wedge \frac{c_\sigma}{d_\sigma}\left(\frac{\|\mathbf{p}_\sigma^i\|}{E\|\mathcal{N}(\mathbf{0}, I)\|} - 1\right)\right) \tag{11}$$

The covariance matrix is updated by the procedure `UpdateCovarianceMatrix` by using Equation (8). However, because we are injecting a set of solutions, the learning path $\mathbf{p}_c^i$ is updated by means of the following equation.

$$\mathbf{p}_c^i \leftarrow (1 - c_c)\mathbf{p}_c^i + h_\sigma\sqrt{c_c(2 - c_c)\mu_{\text{eff}}}\Delta\mathbf{m} \tag{12}$$

In fact, as it is shown in [6], Equations (10), (11) and (12) differ from Equations (4), (5) and (7), by introducing the parameter $\Delta\mathbf{m}$ and $\Delta_\sigma^{\max}$. Note however, that using $\Delta_\sigma^{\max} = +\infty$ and $\Delta\mathbf{m} = \frac{\mathbf{m}_{\tau_i} - \mathbf{m}_{\tau_i - 1}}{\sigma_i}$, the original equations of CMA-ES are recovered. It is also possible to inject an arbitrary mean, which shifts the current mean $\mathbf{m}^i$ by means of additional equations. In the study presented herein, we focused only on the injection of already evaluated solutions. However, the injection of a determined mean, is indeed, a possible path for future research.

In the remainder, our proposed approach is denoted as MOEA/D-CMA+I as exactly described in Algorithm 2; however, we shall also consider a second variant of this algorithm, denoted by MOEA/D-CMA, by taking off the injection mechanism, i.e., $\mathbf{Q} = \mathbf{V}^i$ and $\Delta_\sigma^{\max} = +\infty$.

5. EXPERIMENTAL STUDY

In order to analyze the efficiency of the proposed approach *with* and *without* injection, i.e. MOEA/D-CMA+I and MOEA/D-CMA, we compare its performance against two competing algorithms: (i) the conventional *Multi-Objective Evolutionary Algorithm based on Decomposition* (MOEA/D) [19], and (ii) an improved variant of MOEA/D which uses differential evolution operators and a dynamic neighborhood selection. The corresponding algorithm is called MOEA/D-DE [15]. In this section, the benchmark problems and the performance assessment design adopted in our analysis are presented.

5.1 Experimental Setup

We consider the continuous MOPs with complicated Pareto sets proposed in [20], and extracted from the CEC 2009 special session and competition on the performance assessment of multi-objective optimization algorithms. This benchmark test suite has been specifically designed to resemble complicated real-life optimization problems. The MOPs therein present different properties in terms of separability, multi-modality, and shape of the Pareto front, i.e. convexity, concavity, discontinuities, gaps, etc. More particularly, we consider all box-constrained functions UF1–F10 under their original setting [20], with UF1–UF7 being two-objective problems and UF8–UF10 being three-objective problems. Notice that, for all of them, the number of variables is $n = 30$, and the Pareto fronts lie in the hyper-box $[0, 1]^k$, where k denotes the number of objective functions for the problem under consideration.

All the competing algorithms considered in this study were compared by following the performance assessment experimental setup

recommended in [13].

Relative Hypervolume Deviation. The first performance measure indicates the relative hypervolume achieved by the Pareto set approximation given by an algorithm. This Relative Hypervolume (RHV) deviation can be computed by:

$$RHV(A) = \frac{HV(R) - HV(A)}{HV(R)} \tag{13}$$

where HV is the hypervolume indicator [22], A is an approximation set and R is a reference set for the instance under consideration. The reference vector is set to $(2, \ldots, 2)^\intercal$. In this quality indicator, a lower value means a better approximation set.

Inverted Generational Distance. The Inverted Generational Distance (IGD, [3]) indicates how far a given Pareto front approximation is from a reference set. Let R be the true Pareto front, the IGD for a set of approximated solutions A is calculated as:

$$IGD(A) = \frac{1}{|R|} \sum_{i=1}^{|R|} d_i \tag{14}$$

where $d_i = \min_{j=1}^{|A|} \sqrt{\sum_{l=1}^{k} (f_l(i) - f_l(j))^2}$ and k is the number of objective functions. A value of zero in this performance measure, indicates that all the solutions obtained by the algorithm are on the true Pareto front.

Both performance measures (i.e. *RHV* and *IGD*) are computed by using the reference sets available at: http://dces.essex.ac.uk/staff/qzhang/moeacompetition09.htm.

5.2 Parameter Setting

As we mentioned before, we consider two variants of the proposed MOEA/D-CMA paradigm. The first version exactly maps to the one presented in Algorithm 2 (i.e. MOEA/D-CMA+I). The second version, denoted by MOEA/D-CMA, corresponds to the same framework, but without performing the phase where solutions are injected to the neighboring subproblems. That is, **Step 2.1** is not performed in Algorithm 2, i.e. $\mathbf{Q} = \mathbf{V}^i$. This shall allow us to appreciate by how much the search process can actually benefit from injecting solutions sampled from neighboring subproblems.

Besides these two versions, we also consider the conventional MOEA/D [19] as well as the more sophisticated MOEA/D-DE [15] in our comparative study. For all competing algorithms, the set of weighting coefficient vectors is generated following a simplex-lattice design [17]. The settings of N and $W = \{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$ are controlled by a parameter H. More precisely, let $\{\mathbf{w}^1, \ldots, \mathbf{w}^N\}$ be the set of weight vectors. Each individual weight w_j^i, such that $i = 1, \ldots, N$ and $j = 1, \ldots, k$, can take a value from $\left\{\frac{0}{H}, \frac{1}{H}, \ldots, \frac{H}{H}\right\}$. Therefore, the number of vectors in W is given by $N = C_{H+k-1}^{k-1}$, where k is the number of objective functions. In this work, we set $H = 99$ for two-objective problems, and $H = 19$ for three-objective problems, that is 100 and 210 weight vectors for MOPs having two and three objectives, respectively.

To fix the parameters of MOEA/D-CMA+I and MOEA/D-CMA, we essentially get inspired by the standard setting values suggested in [5], which are summarized in Table 1. However, following our empirical observations, we reduce the population size λ to be the same as μ. Nonetheless, the adjustment of the λ parameter, is indeed, one important open issue that would deserve to be investigated. The remaining parameters required by the MOEA/D framework are set as follows: $T = 20$, $\eta_c = \eta_m = 20$, $P_c = 1$ and $P_m = 1/n$; which represent respectively the neighborhood size, crossover index (for Simulated Binary Crossover (SBX)), mutation index (for Polynomial-Based Mutation (PBM)), crossover rate and

Table 1: Parameters for MOEA/D-CMA

$\lambda = 4 + \lfloor 3 \ln n \rfloor$	$\mu = \lfloor \frac{\lambda}{2} \rfloor$
$\omega_i = \frac{\ln\left(\frac{\lambda+1}{2}\right) - \ln i}{\sum_{j=1}^{\mu}\left(\ln\left(\frac{\lambda+1}{2}\right) - \ln j\right)}$ for $i = 1, \ldots, \mu$	
$\mu_{\text{eff}} = \sum_{i=1}^{\mu} (\omega_i^2)^{-1}$	$c_y = \sqrt{n} + 2n/(n+2)$
$c_m = 1$	$c_\sigma = \frac{\mu_{\text{eff}}+2}{n+\mu_{\text{eff}}+3}$
$c_c = \frac{4}{n+4}$	$d_\sigma = 1 + c_\sigma + 2\max\left(0, \sqrt{\frac{\mu_{\text{eff}}-1}{n+1}} - 1\right)$
$c_1 = \frac{\alpha_{\text{cov}} \min(1, \lambda/6)}{(n+1.3)^2+\mu_{\text{eff}}}$	$c_\mu = 1 - c_1 \wedge \alpha_{\text{cov}} \frac{\mu_{\text{eff}}-2+1/\mu_{\text{eff}}}{(n+2)^2+\alpha_{\text{cov}}\mu_{\text{eff}}/2}$
$\alpha_{\text{cov}} = 2$	$\Delta_\sigma^{\max} = 1$

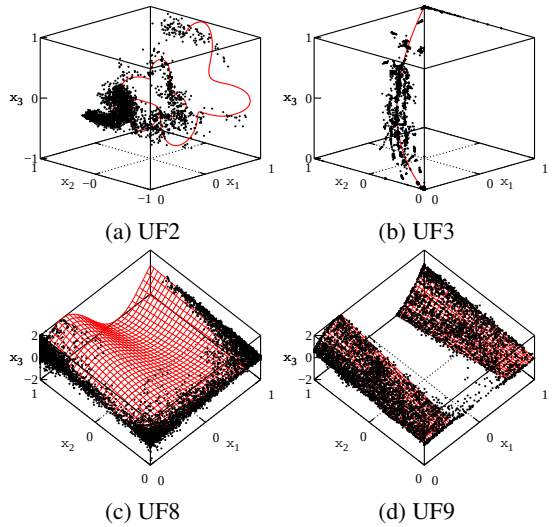
mutation rate. Finally, the parameter θ in the PBI approach was set to $\theta = 5$. For MOEA/D-DE, we adopted the set of parameters given in [15]. More precisely, the differential factor was set as $F = 0.5$, the crossover ratio was set as $CR = 1$, the maximum number of replacements was set as $n_r = 2$, and $\delta = 0.9$.

We define the initial step size as $\sigma_{\text{init}} = \frac{1}{4} \times (U_b - L_b)$, where U_b and L_b are the upper- and the lower-bounds in the search space. Since we consider the problems with the same boundaries in all dimensions, the decision variables of the original CEC 2009 test functions are simply rescaled without modifying the shape of the PS or the PF. Finally, the auxiliary fitness function was computed with $\alpha = 1 \times 10^{-5}$.

For each MOP, we performed 30 independent runs and we measure the performance of the algorithms after $N \times 1\,000$ and $N \times 2\,000$ fitness function evaluations.

5.3 Numerical Results

Our results are summarized in Table 2, where we can see the relative performance of the four competing algorithms using two different stopping conditions and for the two quality indicators described above. Notice that we can read two informations in Table 2. We put in bold the best average indicator-value obtained among all algorithms. We also perform a Mann-Whitney non-parametric statistical test between each pair of algorithms in order to determine whether a given algorithm is significantly outperformed by



(a) UF2 (b) UF3

(c) UF8 (d) UF9

Figure 1: Pareto set approximations given by MOEA/D-CMA+I in UF2, UF3, UF8 and UF9

Table 2: Comparison of the competing algorithms with respect to the relative hypervolume deviation (RHV) and to the inverted generational distance (IGD). The first number stands for the average indicator-value (lower is better). The number in brackets stands for the standard deviation. Bold values correspond to the best average indicator-value for the instance and the indicator under consideration. Underline values correspond to algorithms that are *not* statistically outperformed by any other algorithm for the instance and the indicator under consideration with respect to a Mann-Whitney non-parametric statistical test with a p-value of 0.05 by using a Bonferroni correction [2].

	RHV				IGD			
	MOEA/D-CMA+I	MOEA/D-CMA	MOEA/D	MOEA/D-DE	MOEA/D-CMA+I	MOEA/D-CMA	MOEA/D	MOEA/D-DE
Fitness function evaluations = N x 1000								
UF1	**0.058** (0.01)	0.177 (0.01)	0.216 (0.07)	0.105 (0.05)	**0.003** (0.01)	0.012 (0.01)	0.009 (0.04)	0.005 (0.03)
UF2	**0.065** (0.03)	0.100 (0.02)	0.101 (0.04)	0.081 (0.04)	**0.004** (0.02)	0.007 (0.01)	0.007 (0.03)	0.005 (0.03)
UF3	**0.125** (0.04)	0.227 (0.04)	0.325 (0.02)	0.213 (0.07)	0.008 (0.02)	0.014 (0.02)	0.012 (0.01)	**0.008** (0.02)
UF4	**0.054** (0.00)	0.075 (0.00)	0.085 (0.01)	0.081 (0.01)	**0.002** (0.00)	0.003 (0.00)	0.002 (0.00)	0.003 (0.00)
UF5	0.742 (0.11)	0.951 (0.05)	**0.435** (0.04)	0.488 (0.08)	0.223 (0.40)	0.315 (0.39)	**0.128** (0.25)	0.143 (0.29)
UF6	0.387 (0.08)	0.391 (0.03)	0.405 (0.07)	**0.318** (0.09)	0.024 (0.07)	**0.015** (0.01)	0.021 (0.05)	0.018 (0.08)
UF7	**0.050** (0.02)	0.120 (0.02)	0.379 (0.05)	0.176 (0.14)	**0.002** (0.00)	0.008 (0.01)	0.020 (0.05)	0.009 (0.08)
UF8	0.074 (0.04)	0.093 (0.03)	0.167 (0.12)	**0.074** (0.05)	0.007 (0.02)	0.009 (0.01)	0.010 (0.07)	**0.005** (0.02)
UF9	0.086 (0.04)	0.092 (0.02)	0.223 (0.02)	**0.082** (0.02)	0.007 (0.03)	0.007 (0.01)	0.009 (0.01)	**0.006** (0.02)
UF10	0.918 (0.07)	0.988 (0.04)	**0.510** (0.11)	0.525 (0.05)	0.046 (0.13)	0.077 (0.21)	0.022 (0.05)	**0.019** (0.03)
Fitness function evaluations = N x 2000								
UF1	**0.051** (0.02)	0.157 (0.02)	0.205 (0.07)	0.075 (0.04)	**0.003** (0.01)	0.011 (0.02)	0.008 (0.04)	0.003 (0.02)
UF2	**0.056** (0.02)	0.082 (0.01)	0.092 (0.03)	0.062 (0.03)	**0.004** (0.02)	0.005 (0.01)	0.006 (0.03)	0.004 (0.03)
UF3	**0.085** (0.04)	0.199 (0.03)	0.325 (0.02)	0.122 (0.07)	0.006 (0.02)	0.013 (0.01)	0.012 (0.01)	**0.004** (0.03)
UF4	**0.051** (0.00)	0.072 (0.00)	0.074 (0.01)	0.076 (0.01)	**0.002** (0.00)	0.003 (0.00)	0.002 (0.00)	0.003 (0.00)
UF5	0.738 (0.11)	0.882 (0.06)	0.424 (0.05)	**0.406** (0.06)	0.221 (0.40)	0.275 (0.35)	**0.123** (0.26)	0.128 (0.41)
UF6	0.363 (0.11)	0.331 (0.04)	0.392 (0.04)	**0.314** (0.10)	0.023 (0.09)	**0.013** (0.01)	0.021 (0.05)	0.018 (0.08)
UF7	**0.041** (0.02)	0.117 (0.01)	0.369 (0.06)	0.140 (0.14)	**0.002** (0.00)	0.007 (0.01)	0.019 (0.05)	0.007 (0.08)
UF8	0.054 (0.03)	0.074 (0.02)	0.163 (0.13)	**0.042** (0.03)	0.006 (0.01)	0.008 (0.01)	0.010 (0.07)	**0.004** (0.01)
UF9	0.074 (0.04)	**0.059** (0.02)	0.162 (0.03)	0.071 (0.03)	0.006 (0.03)	0.006 (0.01)	0.008 (0.02)	**0.006** (0.02)
UF10	0.909 (0.07)	0.962 (0.05)	0.510 (0.11)	**0.462** (0.06)	0.045 (0.11)	0.057 (0.16)	0.022 (0.05)	**0.017** (0.04)

any other. If *no* other algorithm is statistically better than the algorithm under consideration, then the corresponding indicator-value is underlined in the table. Several interesting observations can be extracted from Table 2. First, we can see that there are no significant differences between the behavior of the algorithms when considering different stopping conditions. This hopefully informs that all algorithms are rather ranking in the same manner independently of the available function-evaluation budget. More interestingly, the results of Table 2 allows us to validate the accuracy of the introduced approach from several perspectives.

When comparing MOEA/D-CMA+I with MOEA/D-CMA, we can see that MOEA/D-CMA is outperformed by MOEA/D-CMA+I for all instances except for UF6 when using the IGD indicator. This indicates that the way in which solutions are injected from neighboring subproblem into the single-objective CMA-ES engine, and the way in which CMA-ES is taking care of those solutions when adapting its components, is drastically improving the search when compared to the straightforward version where CMA-ES is plugged within MOEA/D to simply sample new points independently at every subproblem. Notice however, that the MOEA/D-CMA version exhibits, in the worst case, very comparable performance with respect to conventional MOEA/D, and the more advanced MOEA/D-CMA+I algorithm is outperforming the conventional version of MOEA/D in almost all instances expect for instance UF5 and UF10. This indicates that the variation operator induced by the MOEA/D-CMA algorithm is performing well. This is confirmed when looking at the relative performance of MOEA/D-CMA+I with respect to MOEA/D-DE which is known to perform extremely well on these benchmark functions. We can see that there is no algorithm that is performing better in all the considered instances and that the difference is most of the time not statistically significant. Notice also that the difference between the two algorithms is more pronounced for the last 3 instances UF8, UF9 and UF10, in favor of MOEA/D-DE—these instances are actually the three-objective problems considered in our experiments. For the bi-objective instances, MOEA/D-CMA+I appears to be relatively competitive compared to MOEA/D-DE.

In Fig. 1, we show the Pareto set approximations obtained by our proposed MOEA/D-CMA+I for the UF2, UF3, UF8 and UF9 problems. As we can see, the benchmark functions exhibit complicated shapes and depending on the considered instance, the algorithm has some potential in accurately approaching the true Pareto set. This is actually a common behavior for the other competing algorithms as well which is to be attributed to the particular difficulty of some of the considered instances.

We recall that the main purpose of this paper is to investigate at what extent the CMA-ES single-objective optimizer could be appropriately integrated within the MOEA/D framework and what are the benefits one can obtain. As such, we can conclude from the previous set of experiments that injecting accurately solutions from

neighbors plays a crucial role to this end. We can also conclude the newly proposed MOEA/D-CMA+I algorithm is a promising candidate to solve hard and complicated optimization problems. In fact, the CMA-ES optimizer enjoys several attractive properties, such as invariance and robustness, which are not verified by several other evolutionary operators. Hence, the experimental study presented in this paper, and the relatively good performance that MOEA/D-CMA+I is able to obtain can be viewed as a first promising step towards the establishment of highly accurate algorithms.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we consider the single-objective CMA-ES algorithm as a plausible optimizer to be incorporated into the MOEA/D framework. We proposed to assign to each subproblem, obtained by decomposition, a single CMA-ES engine and to profit from the solutions of the neighboring subproblems in order to adapt the CMA-ES components accordingly. We thereby rely on the ability of CMA-ES to deal with external solutions in order to derive novel variation operators for the MOEA/D framework. The experimental study conducted in this paper allows us to validate the proposed approach and to show its effectiveness. Besides, the presented algorithm opens the road to further challenging research questions. In fact, the proposed approach is to be viewed as a first step enlightening how the CMA-ES algorithm can specifically be used as a single-objective optimizer when solving multi-objective problems; and this is thanks to the basic concepts introduced by MOEA/D and the ability of CMA-ES of handling external solutions. It is worth noticing that different strategies to take into account the sampled solutions from neighbors could be investigated in the future. Furthermore, it would be insightful to study more deeply the behavior of the so-obtained strategies when considering other benchmark functions. This would enable to fully appreciate the strength of the CMA-ES algorithm on a broad range of problems with different properties. It is our hope that the contribution of this paper can serve as a starting point to derive more powerful aggregation-like EMO methods based on the well established CMA-ES algorithm.

7. REFERENCES

[1] A. Auger and N. Hansen. A restart CMA evolution strategy with increasing population size. In B. McKay et al., editors, *CEC'2005*, volume 2, pages 1769–1776, 2005.

[2] C. E. Bonferroni. Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commerciali di Firenze*, 8:3–62, 1936.

[3] C. A. Coello Coello and N. Cruz Cortés. Solving Multiobjective Optimization Problems using an Artificial Immune System. *Genetic Programming and Evolvable Machines*, 6(2):163–190, June 2005.

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA–II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[5] N. Hansen. The CMA evolution strategy: a comparing review. In J. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, editors, *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, pages 75–102. Springer, 2006.

[6] N. Hansen. Injecting external solutions into CMA-ES. Technical report, INRIA, 2011.

[7] N. Hansen, A. Auger, R. Ros, S. Finck, and P. Posík. Comparing results of 31 algorithms from the black-box optimization benchmarking BBOB-2009. In J. Branke et al., editors, *GECCO workshop on Black-Box Optimization Benchmarking (BBOB'2010)*, pages 1689–1696. ACM, July 2010.

[8] N. Hansen, S. 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*, 13(1):180–197, 2009.

[9] N. Hansen and A. Ostermeier. Completely Derandomized Self-Adaptation in Evolution Strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[10] C. Igel, N. Hansen, and S. Roth. Covariance matrix adaptation for multi-objective optimization. *Evolutionary Computation*, 15(1):1–28, 2007.

[11] C. Igel, T. Suttorp, and N. Hansen. Steady-state selection and efficient covariance matrix update in the multi-objective cma-es. In *Evolutionary Multi-Criterion Optimization*, pages 171–185. Springer Berlin Heidelberg, 2007.

[12] H. Ishibuchi and T. Murata. Multi-Objective Genetic Local Search Algorithm. In T. Fukuda and T. Furuhashi, editors, *Proceedings of the 1996 International Conference on Evolutionary Computation*, pages 119–124, Nagoya, Japan, 1996. IEEE.

[13] J. Knowles, L. Thiele, and E. Zitzler. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland, feb 2006. revised version.

[14] S. Kukkonen and J. Lampinen. GDE3: the third evolution step of generalized differential evolution. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 1, pages 443–450 Vol.1, Sept 2005.

[15] H. Li and Q. Zhang. Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 13(2):284–302, April 2009.

[16] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, Massachuisetts, 1999.

[17] H. Scheffé. Experiments With Mixtures. *Journal of the Royal Statistical Society*, Series B (Methodological), 20(2):344–360, 1958.

[18] S. Zapotecas Martínez, V. A. Sosa Hernández, H. E. Aguirre, K. Tanaka, and C. A. C. Coello. Using a Family of Curves to Approximate the Pareto Front of a Multi-Objective Optimization Problem. In *PPSN XIII*, pages 682–691. Springer, Ljubljana, Slovenia, September 2014.

[19] Q. Zhang and H. Li. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE TEVC*, 11(6):712–731, December 2007.

[20] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari. Multiobjective Optimization Test Instances for the CEC 2009 Special Session and Competition. Technical Report CES-487, University of Essex and Nanyang Technological University, 2008.

[21] E. Zitzler and S. Künzli. Indicator-based selection in multiobjective search. In *PPSN VIII*, pages 832–842. Springer, 2004.

[22] E. Zitzler and L. Thiele. Multiobjective Optimization Using Evolutionary Algorithms – A Comparative Case Study. In A. E. Eiben, editor, *PPSN V*, pages 292–301, Amsterdam, September 1998. Springer-Verlag.