A Matrix Adaptation Evolution Strategy for Constrained Real-Parameter Optimization

Michael Hellwig
Vorarlberg University of Applied Science
Research Centre PPE
Campus V, Hochschulstraße 1
6850 Dornbirn, Austria
Email: michael.hellwig@fhv.at

Hans-Georg Beyer
Vorarlberg University of Applied Science
Research Centre PPE
Campus V, Hochschulstraße 1
6850 Dornbirn, Austria
Email: hans-georg.beyer@fhv.at

***Abstract*—By combination of successful constraint handling techniques known within the context of Differential Evolution with the recently suggested Matrix Adaptation Evolution Strategy (MA-ES), a new Evolution Strategy for constrained optimization is presented. The novel MA-ES variant is applied to the benchmark problems specified for the CEC 2018 competition on constrained single objective real-parameter optimization. The algorithm is able to find feasible solutions on more than 80% of the benchmark problems with high accuracy.**

I. Introduction

The field of constrained optimization is concerned with searching for the optimal solution of an objective function with respect to certain limitations on the parameter vector components. Constraints may arise from multiple sources, e.g. limited resources of a problem, problem-specific trade-offs or appropriate physical boundaries. The introduction of constraints into an optimization task adds to the problem complexity. This is particularly true in the context of black-box optimization, where the analytical structure of the optimization problem is unknown.

Organized during the IEEE Congress on Evolutionary Computation (CEC), the competition on constrained real-parameter optimization (2006, 2010, and 2017) introduced specific test environments for evaluation and comparison of state-of-the-art stochastic search algorithms. Apart from the competitions, these test function environments turned out very popular for benchmarking Evolutionary Algorithms (EA).

In particular, Differential Evolution (DE) proved itself successful in a number of constrained settings that include but are not limited to the CEC competitions on constrained real-parameter optimization, see [1]. A survey of recent advances in the field of DE is provided by [2]. DE variants encompass various approaches to deal with constrained problems, e.g. using different mutation and crossover schemes to find the right balance between exploration and exploitation, aiming at optimal control of the intrinsic strategy parameters [3], treating infeasible solutions with diverse constraint handling approaches [4], or even combinations of the mentioned principles [5], [6].

The CEC competition results supported the suitability of recent DE variants for constrained optimization. In contrast, to our knowledge, Evolution Strategies (ES) abstained from competing in the past CEC competitions on constrained real-parameter optimization. While investigations concerning the EA subclass of ES are scarce, a few studies have to be mentioned. Among others, empirical studies on ES on constrained black-box problems involve the application of stochastic ranking, penalty approaches, or the use of Deb's constraint handling method [7] together with a diversity mechanism to preserve a number of infeasible candidate solutions with predefined probability. An overview of constraint handling techniques for ES can be found in [8]. These approaches were evaluated on a set of test functions collected in [9]. An ES variant that makes use of multi-objective techniques to deal with constrained optimization problems is proposed in [10] and evaluated on the CEC2006 benchmark functions [11]. Two ES variants were considered in [12], where a combination of EAs was proposed to cope with constrained optimization problems of different characteristics.

Examples of ES applications to constrained real-world problem include [13]. There, a novel penalty constraint handling technique was incorporated into the CMA-ES [14] to solve nonlinear constrained problems in the context of launcher systems. In [15] an ES variant has been published that deals with a portfolio optimization task subject to mixed linear/nonlinear constraints. The strategy is based on the CMSA-ES [16] and relies on three specific techniques to fulfill the constraints.

Theoretical investigations on qualified constrained test problems also exist. In this context, the active-set ES [17] and the augmented Lagrangian constraint handling approach [18] have to be mentioned. While the former approach only assumes a black-box objective function and known constraints, the latter work concentrates on the case of linear constraints.

The present paper aims at solving the real-valued constrained optimization problems specified in [19] for the CEC competition in 2018. For this purpose, constraint handling methods approved within the context of Differential Evolution are applied to a state-of-the-art Evolution Strategy, the Matrix Adaptation Evolution Strategy (MA-ES) [20]. The MA-ES represents an algorithmically reduced CMA-ES variant [14]. By use of an adequate transformation, the evolution path of the covariance matrix update can be disregarded while the performance is almost retained. The newly proposed ϵMAg-ES handles box-constraints by reflecting exceeding components into the predefined box. Additional in-/equality constraints are dealt with by application of two constraint handling techniques: ϵ-level ordering and a repair step that is based on gradient approximation. The approach demonstrates the mutual benefit of two related fields of research and substantiates the use of the mentioned constraint handling mechanisms within EA variants different from DE. The resulting ϵMAg-ES algorithm turns out to be competitive on the CEC 2017 benchmarks [19].

A. The optimization problem

This paper considers constrained optimization problems of the form

$$\begin{aligned} \min\ & f(\boldsymbol{y}) \\ s.t.\ & g_i(\boldsymbol{y}) \leq 0, \qquad i = 1, \dots, l, \\ & h_j(\boldsymbol{y}) = 0, \qquad j = 1, \dots, k, \\ & \boldsymbol{y} \in S \subseteq \mathbb{R}^N. \end{aligned} \tag{1}$$

Without loss of generality, the optimization goal is the minimization of the real-valued objective function $f(\boldsymbol{y})$. Here, $\boldsymbol{y} \in S$ denotes the N-dimensional search space parameter vector. The set S usually comprises a number of box constraints specifying reasonable intervals of the parameter vector components. We refer to the N-dimensional vectors that specify the lower and upper box constraints of each parameter component as $\check{\boldsymbol{y}}$, and $\hat{\boldsymbol{y}}$, respectively. Additionally, the feasible region of the search space is restricted by $m = l+k$ real-valued constraint functions. These constraint functions are separated into inequality constraints $g_i(\boldsymbol{y})$, $i = 1, \dots, l$ and equality constraints $h_j(\boldsymbol{y})$, $j = 1, \dots, k$. A vector $\boldsymbol{y} \in S$ that satisfies all constraints simultaneously is called feasible. The set of all feasible parameter vectors is denoted

$$M := \{\boldsymbol{y} \in S \colon g_i(\boldsymbol{y}) \leq 0 \wedge h_j(\boldsymbol{y}) = 0, \forall i, j\}. \tag{2}$$

The global optimum of (1) is denoted by $\boldsymbol{y}^* \in M$. Note, that the objective function $f(\boldsymbol{y})$ subject to some constraints is also referred to as *constrained function*.

When considering problem (1) a measure of infeasibility is useful for ranking potentially infeasible candidate solutions. To this end, we compute the constraint violation $\nu(\boldsymbol{y})$ of a candidate solution $\boldsymbol{y}$ as

$$\nu(\boldsymbol{y}) = \sum_{i=1}^{l} G_i(\boldsymbol{y}) + \sum_{j=1}^{k} H_j(\boldsymbol{y}), \tag{3}$$

with functions $G_i(\boldsymbol{y})$ and $H_j(\boldsymbol{y})$ defined by

$$G_i(\boldsymbol{y}) := \max\left(0, g_i(\boldsymbol{y})\right), \tag{4}$$

and

$$H_j(\boldsymbol{y}) := \begin{cases} |h_j(\boldsymbol{y})|, & \text{if } |h_j(\boldsymbol{y})| - \delta > 0 \\ 0, & \text{if } |h_j(\boldsymbol{y})| - \delta \leq 0 \end{cases}. \tag{5}$$

In order to be able to satisfy the equality constraints, the δ term introduces the necessary error margin. This paper considers $\delta = 10^{-4}$. Further, the corresponding mean constraint violation $\overline{\nu}(\boldsymbol{y})$ is given by

$$\overline{\nu}(\boldsymbol{y}) = \frac{\sum_{i=1}^{l} G_i(\boldsymbol{y}) + \sum_{j=1}^{k} H_j(\boldsymbol{y})}{l+k}. \tag{6}$$

The remainder of this paper is organized as follows: Section II introduces the ϵMAg-ES for constrained optimization and its constraint handling techniques. The experimental setup of the CEC 2018 competition on single-objective real-parameter optimization as well as the used parameter settings for the ϵMAg-ES are summarized in Sec. III. In Sec. IV the respective results are presented as specified in [19]. The paper concludes with a discussion of the observations.

II. Algorithm

This section introduces the ϵMAg-ES for constrained real-parameter optimization. Its pseudo code is displayed in Algorithm 1. The strategy is based on the MA-ES [20] which represents an algorithmically simplified CMA-ES variant with comparable performance in unconstrained environments. In order to deal with problem (1), constraint handling techniques are incorporated into the MA-ES.

Given the black-box scenario of the competition, the algorithm has no knowledge about the appropriate step-size at a random location in the search space. Instead of starting from a single point, the algorithm initially samples a uniformly distributed population $\mathcal{P}$ of λ candidate solutions $\boldsymbol{y}_j \in \mathbb{R}^N$, $j = 1, \dots, \lambda$ within the predefined box-constraints $\check{\boldsymbol{y}}$ and $\hat{\boldsymbol{y}}$, respectively. This is performed in lines 2 to 5 of Alg. 1. There, $\boldsymbol{u}(0,1)$ denotes an N-dimensional vector with uniformly distributed components in $(0,1)$. The initial parental recombinant $\boldsymbol{y}^{(0)}$ is then obtained by weighted recombination of the μ best candidate solutions in line 9. Notice, that $\boldsymbol{y}_{m:\lambda}$ denotes the mth best out of λ candidate solutions with respect to the order relation $\leq_\epsilon$, see Eq. (8). The standard weights w_i of the MA-ES as described in Sec. III are used. Regarding the vectors $\boldsymbol{z}_l^{(g)}$ and $\boldsymbol{d}_l^{(g)}$, the vector that contributes to the mth best candidate solution $\boldsymbol{y}_{m:\lambda}$ is considered the mth best, i.e. $\boldsymbol{z}_{m:\lambda}$ and $\boldsymbol{d}_{m:\lambda}$, respectively. The comparison of the candidate solutions always involves the evaluation of the constrained problem and consumes function evaluations. We account one function evaluation per evaluation of a constrained problem, i.e. including the objective function and all related constraint functions (cf. line 8).

The starting population is also used to determine the initial $\epsilon^{(0)}$ value in line 6, and the associated parameter γ, in line 7, that controls the $\epsilon^{(g)}$ decrease. Here, θ_t specifies the percentage of considered candidate solutions and $\lfloor \cdot \rfloor$ denotes the floor function. The relation $\leq_\epsilon$ is described in more detail in Sec. II-A. In line 10, the best individual of the initial population $\boldsymbol{y}_{1:\lambda}$ becomes the best-found solution so far $\boldsymbol{y}_{\text{bsf}}$.

Contrary to CMA-ES, the MA-ES replaces the covariance matrix update as well as the adaptation of the related search path by an updated transformation matrix $\boldsymbol{M}^{(g)}$. Within the

Algorithm 1 Pseudo code of the MA-ES algorithm variant for constrained real-parameter optimization: the ϵMAg-ES.

1: **Initialize:** $\mu,\ \lambda,\ \sigma^{(0)},\ \boldsymbol{p}_\sigma^{(0)} \leftarrow \boldsymbol{0}\ ,\ \boldsymbol{M}^{(0)} \leftarrow \boldsymbol{I}$
2: **for** $j \leftarrow 1 \colon \lambda$ **do**
3: $\quad \boldsymbol{y}_j \leftarrow \check{\boldsymbol{y}} + (\hat{\boldsymbol{y}} - \check{\boldsymbol{y}}) \circ \boldsymbol{u}(0,1)$
4: $\quad \mathcal{P} \leftarrow \mathcal{P} \cup \{\boldsymbol{y}_j\}$
5: **end for**
6: $\epsilon^{(0)} \leftarrow \sum_{i=1}^{\lfloor \theta_t \lambda \rfloor} \nu(\boldsymbol{y}_{i:\lambda}) / \lfloor \theta_t \lambda \rfloor$
7: $\gamma \leftarrow \max\left(\gamma_{\min}, (-5 - \log(\epsilon^{(0)}))/\log(0.05)\right)$
8: $fevals \leftarrow \lambda$
9: $\boldsymbol{y}^{(0)} \leftarrow \sum_{i=1}^{\mu} w_i \boldsymbol{y}_{i:\lambda}$ according to $\leq_\epsilon$
10: $\boldsymbol{y}_{\text{bsf}} \leftarrow \boldsymbol{y}_{1:\lambda}$
11: $g \leftarrow 0$
12: **while** $fevals < fevals_{\max}$ **do**
13: $\quad \boldsymbol{M}^{-1} \leftarrow \texttt{PseudoInverse}(\boldsymbol{M}^{(g)})$
14: $\quad$ Reset $\boldsymbol{M}^{(g)} \leftarrow \boldsymbol{I}$ if $\texttt{PseudoInverse}(\boldsymbol{M}^{(g)})$ fails
15: $\quad$ **for** $l \leftarrow 1 \colon \lambda$ **do**
16: $\quad\quad \boldsymbol{z}_l^{(g)} \leftarrow \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
17: $\quad\quad \boldsymbol{d}_l^{(g)} \leftarrow \boldsymbol{M}^{(g)} \boldsymbol{z}_l^{(g)}$
18: $\quad\quad \bar{\boldsymbol{y}} \leftarrow \boldsymbol{y}^{(g)} + \sigma^{(g)} \boldsymbol{d}_l^{(g)}$
19: $\quad\quad \boldsymbol{y}_l^{(g)} \leftarrow \texttt{KeepRange}(\bar{\boldsymbol{y}})$
20: $\quad\quad fevals \leftarrow fevals + 1$
21: $\quad\quad$ **if** $\operatorname{mod}(g, N) = 0 \ \wedge\ u(0,1) < \theta_p$ **then**
22: $\quad\quad\quad h \leftarrow 1$
23: $\quad\quad\quad$ **while** $h \leq \theta_r \wedge \nu(\boldsymbol{y}_l^{(g)}) > 0$ **do**
24: $\quad\quad\quad\quad h \leftarrow h + 1$
25: $\quad\quad\quad\quad \tilde{\boldsymbol{y}} \leftarrow \texttt{GradientBasedRepair}(\boldsymbol{y}_l^{(g)})$
26: $\quad\quad\quad\quad \boldsymbol{y}_l^{(g)} \leftarrow \texttt{KeepRange}(\tilde{\boldsymbol{y}})$
27: $\quad\quad\quad\quad fevals \leftarrow fevals + N + 1$
28: $\quad\quad\quad$ **end while**
29: $\quad\quad$ **end if**
30: $\quad\quad$ **if** $\bar{\boldsymbol{y}} \neq \boldsymbol{y}_l^{(g)}$ **then**
31: $\quad\quad\quad \boldsymbol{d}_l^{(g)} \leftarrow \left(\boldsymbol{y}_l^{(g)} - \boldsymbol{y}^{(g)}\right)/\sigma^{(g)}$
32: $\quad\quad\quad \boldsymbol{z}_l^{(g)} \leftarrow \boldsymbol{M}^{-1} \boldsymbol{d}_l^{(g)}$
33: $\quad\quad$ **end if**
34: $\quad$ **end for**
35: $\quad$ **if** $\boldsymbol{y}_{1:\lambda}^{(g)} \leq_\epsilon \boldsymbol{y}_{\text{bsf}}$ **then**
36: $\quad\quad \boldsymbol{y}_{\text{bsf}} \leftarrow \boldsymbol{y}_{1:\lambda}^{(g)}$
37: $\quad$ **end if**
38: $\quad \boldsymbol{y}^{(g+1)} \leftarrow \boldsymbol{y}^{(g)} + \sigma^{(g)} \sum_{i=1}^{\mu} w_i \boldsymbol{d}_{i:\lambda}^{(g)}$
39: $\quad \boldsymbol{p}_\sigma^{(g+1)} \leftarrow (1 - c_\sigma)\boldsymbol{p}_\sigma^{(g)} + \sqrt{\mu_w c_\sigma (2 - c_\sigma)} \sum_{i=1}^{\mu} w_i \boldsymbol{z}_{i:\lambda}^{(g)}$
40: $\quad \boldsymbol{M}^{(g+1)} \leftarrow \boldsymbol{M}^{(g)} + \frac{c_1}{2}\boldsymbol{M}^{(g)}\left(\boldsymbol{p}_\sigma^{(g)}(\boldsymbol{p}_\sigma^{(g)})^\top - \boldsymbol{I}\right) \cdots$
$\quad\quad + \frac{c_\mu}{2}\boldsymbol{M}^{(g)}\left(\sum_{i=1}^{\mu} w_i \boldsymbol{z}_{i:\lambda}^{(g)} (\boldsymbol{z}_{i:\lambda}^{(g)})^\top - \boldsymbol{I}\right)$
41: $\quad \sigma^{(g+1)} \leftarrow \min\left(\sigma^{(g)} \exp\left[\frac{c_\sigma}{2}\left(\frac{\|\boldsymbol{p}_\sigma^{(g+1)}\|^2}{N} - 1\right)\right], \sigma_{\max}\right)$
42: $\quad g \leftarrow g + 1$
43: $\quad$ **if** $g < T$ **then**
44: $\quad\quad \epsilon^{(g)} \leftarrow \epsilon^{(0)}(1 - \frac{g}{T})^\gamma$
45: $\quad$ **else** $\epsilon^{(g)} \leftarrow 0$
46: $\quad$ **end if**
47: **end while**
48: **return** $[\boldsymbol{y}_{\text{bsf}},\ f(\boldsymbol{y}_{\text{bsf}}),\ \nu(\boldsymbol{y}_{\text{bsf}})]$

offspring procreation loop of each generation g, λ offspring are generated out of the recombined μ best candidate solution of the previous generation. To this end, the mutation vector $\boldsymbol{d}_l^{(g)}$ of each offspring is obtained by multiplication of this matrix $\boldsymbol{M}^{(g)}$ and a vector $\boldsymbol{z}_l^{(g)}$ with standard normally distributed components, lines 16 and 17. By adding the product of the mutation strength $\sigma^{(g)}$ and $\boldsymbol{d}^{(g)}$ to the recombinant $\boldsymbol{y}^{(g)}$ of the previous generation, an offspring is generated in line 18.

In cases where offspring individuals are generated outside the box constraints, these candidate solutions are reflected into the box. According to Eq. (7), the routine `KeepRange(.)` recomputes each offspring that does not satisfy all box constraints. This step is performed in line 19 before each evaluation of the constrained function.

As suggested by [4], in generations g that are multiples of the dimensionality N, an additional repair step is performed with probability θ_p (lines 21 to 29). In this scenario, infeasible offspring candidate solutions $\boldsymbol{y}_l^{(g)}$ are repaired by application of a step that involves the approximation of the gradient. The repair step requires N function evaluations per execution plus one single evaluation of the repaired candidate solution. While this step is rather costly, it can potentially guide the search into a beneficial region of the search space, or promote the final step towards the optimizer, respectively. While `GradientBasedRepair(.)` does not provide a feasible solution, the repair step is repeated θ_r times.

If an offspring candidate solution $\boldsymbol{y}_l^{(g)}$ has been adjusted by `KeepRange(.)` or `GradientBasedRepair(.)`, it will differ from the originally sampled $\bar{\boldsymbol{y}}$ in at least one component. The corresponding mutation vector $\boldsymbol{d}_l^{(g)}$ and $\boldsymbol{z}_l^{(g)}$ of $\boldsymbol{y}_l^{(g)}$ have to be readjusted in order to adequately take into account the correct quantities in the update of the transformation matrix $\boldsymbol{M}$. This readjustment is executed in lines 31 and 32 of Alg.1. While the repair of $\boldsymbol{d}_l^{(g)}$ is straight forward, that of $\boldsymbol{z}_l^{(g)}$ involves the inverse of the transformation matrix $\boldsymbol{M}^{(g)}$. Since $\boldsymbol{M}^{(g)}$ has no fixed properties, it can easily become singular. Hence, the pseudo inverse $\boldsymbol{M}^{-1}$ of $\boldsymbol{M}^{(g)}$ is used in line 32. It is computed in line 13 at the beginning of each generation.

In the event that the transformation matrix update results in a matrix $\boldsymbol{M}^{(g)}$ that is ill-suited for the determination of the pseudo inverse $\boldsymbol{M}^{-1}$, a regularization step has to be applied in line 14 to prevent the algorithm from breaking off due to numerical instabilities. As a first workaround, we simply reset the transformation matrix to $\boldsymbol{I}$. This way the transformation matrix adaptation can begin anew in the current location of the search space. Notice, that other regularization approaches may be conceivable and need to be evaluated in the future.

After the computation of all λ offspring, the best offspring is compared to the best-found solution so far $\boldsymbol{y}_{\text{bsf}}$ in line 35. The parental recombinant $\boldsymbol{y}^{(g)}$ is updated in line 38. The update involves the selection of the best μ mutation vectors with respect to $\leq_\epsilon$. In line 39, the ϵMAg-ES adapts the search path $\boldsymbol{p}_\sigma^{(g)}$ known from CMA-ES [14]. Its length indicates whether the mutation strength $\sigma^{(g)}$ should be decreased or increased in the next generation. Further, it contributes to the transformation matrix update which is performed in line 40. The corresponding strategy parameters are chosen according to the recommendations in [20]. The mutation strength $\sigma^{(g)}$ is then updated in line 41. It is bounded from above by the parameter $\sigma_{\max}$. On the one hand, this is motivated simply due to empirical observations where the ϵMAg-ES began to gradually increase the mutation strength towards infinity on some constrained test functions. This behavior might be explained with a certain shape of the feasible region, but this requires further examinations. On the other hand, the box constraints bound the range of suitable mutation strength values in any case. That is, the appropriate $\sigma_{\max}$ can be estimated based on the box constraints.

Finally, the $\epsilon^{(g)}$-threshold is gradually decreased in lines 43 to 46 until it reaches zero, or a predefined number of generations T is reached, respectively. For $g > T$, $\epsilon^{(g)}$ is directly set to zero. This procedure continuously increases the accuracy with which the $\leq_\epsilon$ order relation distinguishes feasible from infeasible candidate solutions (see Eq. (8) below).

These steps are repeated until $fevals$ exceeds the budget of function evaluations $fevals_{\max}$. After termination, the algorithm returns the best-found solution $\boldsymbol{y}_{\text{bsf}}$ together with its objective function value $f(\boldsymbol{y}_{\text{bsf}})$, and the corresponding constraint violation $\nu(\boldsymbol{y}_{\text{bsf}})$, respectively.

A. Constraint handling approaches

The proposed ϵMAg-ES algorithm uses multiple techniques to adequately deal with the constraints of problem (1).

a) Treatment of box-constraints: The ES variant uses a reflection method to ensure that the box-constraints are satisfied by every single candidate solution $\boldsymbol{y} \in \mathbb{R}^N$ that is evaluated in the search process. Regarding the upper and lower parameter bounds ($\hat{\boldsymbol{y}}, \check{\boldsymbol{y}} \in \mathbb{R}^N$) of a constrained function, each exceeding component $i \in \{1, \dots, N\}$ of $\boldsymbol{y}$ is reflected into the box according to

$$y_i = \begin{cases} \check{y}_i + \left((\check{y}_i - y_i) - \left\lfloor \frac{\check{y}_i - y_i}{\omega_i} \right\rfloor \omega_i \right), & \text{if } y_i < \check{y}_i, \\ \hat{y}_i - \left((y_i - \hat{y}_i) - \left\lfloor \frac{y_i - \hat{y}_i}{\omega_i} \right\rfloor \omega_i \right), & \text{if } y_i > \hat{y}_i, \\ y_i, & \text{else,} \end{cases} \quad (7)$$

with $\omega_i = (\hat{y}_i - \check{y}_i)$ referring to the component-wise distance between $\hat{\boldsymbol{y}}$, and $\check{\boldsymbol{y}}$. In Alg. 1, repair of parameter vector components according to Eq. (7) is denoted by `KeepRange(.)`.

b) The ϵ-level ordering: The population is driven towards the optimum of (1) by variation and selection. Regarding selection, the generated offspring individuals of a single generation have to be ranked. Feasible solutions are considered superior to infeasible solutions. The usual lexicographic ordering primarily ranks two candidate solutions according to their constraint violations and secondly with respect to their objective function values. The ϵMAg-ES uses another ordering relation: the ϵ-level order relation introduced by [21] in the context of DE. The ϵ-level ordering represents a relaxation that enables the algorithm to treat infeasible candidate solutions with constraint violation below a specific $\epsilon^{(g)}$ value as feasible. A candidate solution $\boldsymbol{y}$ is said to be ϵ-feasible, if its constraint violation $\nu(\boldsymbol{y})$ does not exceed a predefined constraint violation threshold $\epsilon^{(g)}$ in generation g. The threshold $\epsilon^{(g)}$ is continuously reduced to zero with the number of generations. Hence, the strategy is able to move outside the feasible region within the early phase of the search process which can potentially support the convergence to the optimizer $\boldsymbol{y}^*$.

Let $\boldsymbol{y}_i \in \mathbb{R}^N$ and $\boldsymbol{y}_j \in \mathbb{R}^N$ denote two candidate solutions of problem (1) and let the pair $(f_i, \nu_i) = \left(f(\boldsymbol{y}^i), \nu(\boldsymbol{y}^i)\right)$, and (f_j, ν_j) respectively, represent the corresponding objective function values as well as the related constraint violations. The ϵ-level order relation denoted by $\leq_\epsilon$ is then defined by

$$\boldsymbol{y}_i \leq_\epsilon \boldsymbol{y}_j \Leftrightarrow \begin{cases} f_i \leq f_j, & \text{if} \quad (\nu_i \leq \epsilon^{(g)}) \wedge (\nu_j \leq \epsilon^{(g)}), \\ f_i \leq f_j, & \text{if} \quad \nu_i = \nu_j, \\ \nu_i < \nu_j, & \text{otherwise.} \end{cases} \quad (8)$$

Note, that $<_\epsilon$ is defined analogously. Candidate solutions are compared according to the following criteria: Two ϵ-feasible solutions are ranked with respect to their objective function values. Two ϵ-infeasible solutions are ordered on the basis of their constraint violations. Ties are resolved by considering the objective function values.

The initial $\epsilon^{(0)}$ is determined as the average constraint violation of θ_t (percent) of the best candidate solution within the initial population (see Alg. 1, line 6). During the search process, it is gradually reduced with each generation until it is set to zero after a fixed number of generations T. For $\epsilon = 0$, the ϵ level ordering is equal to the lexicographic ordering mentioned above.

c) Gradient-based repair: In addition to ϵ-level order relation, the ϵMAg-ES makes use of a gradient-based repair approach. Both approaches, the ϵ-level ordering and gradient based repair, are adopted from successful DE variants. Together with an archive of inferior candidate solutions, both approaches are part of the ϵDEga strategy [4]. The ϵDEga shows good performance on constrained black-box optimization problems and is the winning strategy of the CEC 2010 competition on constrained real-parameter optimization [22].

Let the vector of all m constraint values according to problem (1) be denoted by

$$\boldsymbol{C}(\boldsymbol{y}) = (g_1(\boldsymbol{y}), \dots, g_l(\boldsymbol{y}), h_1(\boldsymbol{y}), \dots, h_k(\boldsymbol{y}))^\top. \quad (9)$$

By determining the degree of violation of an infeasible candidate solution $\boldsymbol{y}$, the vector of constraint violations reads

$$\Delta \boldsymbol{C}(\boldsymbol{y}) = \left(\max(0, g_1(\boldsymbol{y})), \dots, \max(0, g_l(\boldsymbol{y})), h_1(\boldsymbol{y}), \dots, h_k(\boldsymbol{y})\right)^\top. \quad (10)$$

The infeasible candidate solution $\boldsymbol{y}$ can then be repaired by addition of a correction vector $\Delta \boldsymbol{y}$

$$\tilde{\boldsymbol{y}} = \boldsymbol{y} + \Delta \boldsymbol{y}. \quad (11)$$

The term $\Delta \boldsymbol{y}$ is calculated by solving the linear system

$$\nabla \boldsymbol{C}(\boldsymbol{y}) \Delta \boldsymbol{y} = -\Delta \boldsymbol{C}(\boldsymbol{y}). \quad (12)$$

To this end, the Jacobian matrix $\nabla \boldsymbol{C}(\boldsymbol{y})$ with respect to $\boldsymbol{C}(\boldsymbol{y})$ has to be determined. Since constrained black-box optimization is considered, the Jacobian needs to be approximated, e.g.

by using finite differences. Equation (12) can approximately be solved by making use of the pseudo inverse $\nabla \boldsymbol{C}(\boldsymbol{y})^{-1}$

$$\Delta \boldsymbol{y} = -\nabla \boldsymbol{C}(\boldsymbol{y})^{-1} \Delta \boldsymbol{C}(\boldsymbol{y}). \tag{13}$$

In the case that no feasible solution is found after the correction, the gradient-based repair step is repeated at most θ_r times. Usually, the constraint violation is gradually decreased with every repair step. If the strategy cannot find a feasible solution, the last infeasible candidate solution is considered as the new offspring candidate solution.

III. Experiments

This section summarizes the experimental setup of the CEC competition and provides the parameter settings used by the ϵMAg-ES, Alg. 1, to solve the related problems.

All experiments are carried out on the constrained benchmark problems, and according to the specifications, provided in [19]. Consequently, the algorithm executes 25 independent runs on each test problem $i \in \{1, \dots, 28\}$ and in each dimension $N = \{10, 30, 50, 100\}$. During the runs, equality constraints are considered to be satisfied if the absolute deviation is below the error margin of $\delta = 10^{-4}$, cf. Eq. (5). In order to fully provide the required statistics, the algorithm returns the objective function value, as well as the mean constrained violation (6), of the best candidate solution so far after having consumed 10%, 50%, and 100% of the allowed budget of $fevals_{\max} = 2 \cdot 10^4 \cdot N$ function evaluation.

The ϵMAg-ES uses the standard parameters recommended for the MA-ES in [20]. The recombination weights are

$$w_i = \frac{\ln(\mu + 0.5) - \ln i}{\sum_{j=1}^{\mu} (\ln(\mu + 0.5) - \ln j)}, \text{ for } i \in \{1, \dots, \mu\} \tag{14}$$

and the corresponding effective population size is given as $\mu_w = 1/\sum_{i=1}^{\mu} w_i^2$. The learning rates of the mutation strength, the search path, and the transformation matrix update are specified as

$$c_\sigma = \frac{\mu_w + 2}{N + \mu_w + 5} \tag{15}$$

$$c_1 = \frac{2}{(N + 1.3)^2 + \mu_w}, \text{ and} \tag{16}$$

$$c_\mu = \min\left[1 - c_1, \frac{2(\mu_w - 2 + 1/\mu_w)}{(N+2)^2 + \mu_w}\right] \tag{17}$$

In contrast to the standard choice the population size parameters λ and μ receive different values. The ϵMAg-ES uses offspring population size of $\lambda = 4N$. The parental population size is set to $\mu = \lfloor \lambda/3 \rfloor$. This choice is motivated by empirical observations of inferior performance of the ϵMAg-ES when using the standard population sizes.

Regarding the parameters of the constrained handling techniques used, the recommendations in [4] are considered. The ϵ-threshold of $\leq_\epsilon$ is gradually reduced during the first $T = 1000$ generations of the search, see line 44 of Alg. 1. The parameter γ is computed in line 7 by use of $\gamma_{\min} = 3$. The initial ϵ-threshold is determined with respect to $\theta_t = 0.9$, i.e. according to the mean constraint violation of the best 90% candidate solutions in the initial population. The gradient-based repair is applied at most $\theta_r = 3$ times with probability $\theta_p = 0.2$ every Nth generation.

Considering the mutation strength of the ϵMAg-ES, we use $\sigma^{(0)} = 1$ and limit the growth of $\sigma^{(g)}$ to $\sigma_{\max} = 100$. The maximal mutation strength represents a compromise solution suitable for the considered benchmark problems.

Table I
PC Configuration and algorithm complexity

PC:	Intel Haswell Desktop
CPU:	Intel Core i7-4770 3.40GHz×8
RAM:	16 GB
Language:	Matlab (2017b)
Algorithm:	ϵMAg-ES

Computational complexity

Dimension	$T1(s)$	$T2(s)$	$(T2-T1)/T1$
$N = 10$	0.1752	0.5365	2.0623
$N = 30$	0.2252	0.5973	1.6525
$N = 50$	0.2795	0.6393	1.2873
$N = 100$	0.4353	0.7280	0.6724

IV. Results

This section presents the results of the ϵMAg-ES on the benchmarks specified in [19] for the CEC competition on constrained single objective real-parameter optimization.

According to the guidelines, information on the PC configuration and the measured computational complexity of the ϵMAg-ES are presented in Table I. There, $T1$ represents the mean computation time of 10^4 objective function evaluations of a single candidate solution averaged over all 28 test functions. The average computation time needed by the complete ϵMAg-ES for 10^4 function evaluations is represented by $T2$. For each search space dimensionality $N \in \{10, 30, 50, 100\}$, the algorithm complexity is identified with the relative difference $(T2-T1)/T1$ of these quantities. It can be observed that $T1$ and $T2$ are both increasing with the dimension N. But as the time consumed by the ϵMAg-ES is increasing less quickly, the algorithm complexity is decreasing with the dimension N.

The experimentally obtained statistics of 25 independent ϵMAg-ES runs on all 28 constrained benchmark problems are made available in Tables II to V. Each table displays the final results after $fevals_{\max}$ function evaluations with respect to a predefined dimensionality N. The corresponding results after 10%, and after 50%, of the function evaluation budget are provided in electronic form.

The collected statistics involve the best, median, mean (with standard deviation), and worst objective function value of the 25 independent algorithm runs on a single constrained problem. To this end, the globally best-found solutions are primarily ranked according to their constrained violation, and secondly with respect to their objective function value. In addition to the objective function value of the median solution, the corresponding mean constraint violation $\overline{\nu}$ and the triplet c are monitored. The latter provides the number of unsatisfied constraints with constraint violation larger than 1, 10^{-2}, and

10^{-4}, respectively. Furthermore, the feasibility rate (FR) of the algorithm is computed for each problem and each dimension. It is obtained as the ratio of the number of algorithm runs in which at least one feasible solution is found and the total number of algorithm runs. Finally, $\overline{vio}$ displays the mean constraint violation of all the solutions obtained in the 25 runs.

Regardless of the dimension, the ϵMAg-ES found feasible solutions with feasibility rate $FR = 100\%$ on at least 20 of 28 constrained problems. Disregarding $N = 100$, this number increases to 22. Taking into account dimension $N = 10$ and $N = 30$, only for the five problems C15, C17, C19, C26, and C28 no feasible median solution could be computed. Considering $N = 50$ and $N = 100$, only C17, C19, C26, and C28 were not solved with high reliability. These four problems present the biggest challenge for the ϵMAg-ES. Comparing the ϵMAg-ES results to those of the CEC2017 competition winner strategy, namely LSHADE44 [6], one observes superior performance particularly on the last seven constrained problems that are subject to search space rotations.

V. Discussion

The paper introduces a novel Evolution Strategy for constrained real-parameter optimization. The ϵMAg-ES combines the Matrix Adaptation Evolution Strategy for unconstrained optimization with well-known constraint handling techniques approved in the context of Differential Evolution. Being applied to the constrained benchmark problems of the CEC 2018 competition on constrained real-parameter optimization, our approach exhibits decent performance on most of the test problems. Only 4 problems could never be solved satisfactorily. Further investigations will be concerned with the evaluation of their difficulties and the treatment of these problems. As the ϵMAg-ES adopts mostly standard parameters from the context of unconstrained optimization and Differential Evolution, parameter tuning might improve the algorithm performance and should be considered in future work.

Acknowledgment

This work was supported by the Austrian Science Fund FWF under grant P29651-N32.

References

[1] E. Mezura-Montes, M. Miranda-Varela, and R. del Carmen Gómez-Ramón, "Differential evolution in constrained numerical optimization: an empirical study," *Information Sciences*, vol. 180, no. 22, pp. 4223–4262, 2010.

[2] S. Das, S. S. Mullick, and P. Suganthan, "Recent advances in differential evolution – an updated survey," *Swarm and Evolutionary Computation*, vol. 27, pp. 1 – 30, 2016.

[3] V. L. Huang, A. K. Qin, and P. N. Suganthan, "Self-adaptive differential evolution algorithm for constrained real-parameter optimization," in *IEEE International Conference on Evolutionary Computation, CEC, part of WCCI 2006, Vancouver, BC, Canada, 16-21 July*, 2006, pp. 17–24.

[4] T. Takahama and S. Sakai, "Constrained optimization by the epsilon constrained differential evolution with an archive and gradient-based mutation," in *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2010, Barcelona, Spain, 18-23 July*, 2010, pp. 1–9. [Online]. Available: http://dx.doi.org/10.1109/CEC.2010.5586484

[5] R. Mallipeddi and P. N. Suganthan, "Differential evolution with ensemble of constraint handling techniques for solving CEC 2010 benchmark problems," in *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2010, Barcelona, Spain, 18-23 July*, 2010, pp. 1–8. [Online]. Available: http://dx.doi.org/10.1109/CEC.2010.5586330

[6] R. Polakova and J. Tvrdík, "L-SHADE with competing strategies applied to constrained optimization," in *2017 IEEE Congress on Evolutionary Computation, CEC 2017, Donostia, San Sebastián, Spain, June 5-8, 2017*, 2017, pp. 1683–1689.

[7] K. Deb, "An efficient constraint handling method for genetic algorithms," in *Computer Methods in Applied Mechanics and Engineering*, 1998, pp. 311–338.

[8] O. Kramer, "A novel selection evolutionary strategy for constrained optimization," *Applied Computational Intelligence and Soft Computing*, 2010.

[9] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *Trans. Evol. Comp*, vol. 4, no. 3, pp. 284–294, sep 2000.

[10] L. Jiao, L. Li, R. Shang, F. Liu, and R. Stolkin, "A novel selection evolutionary strategy for constrained optimization," *Information Sciences*, vol. 239, pp. 122 – 141, 2013.

[11] J. J. Liang, T. P. Runarsson, E. Mezura-Montes, M. Clerc, P. N. Suganthan, C. A. Coello Coello, and K. Deb, "Problem definitions and evaluation criteria for the CEC 2006," 2006, Special Session on Constrained Real-Parameter Optimization, Technical Report, Nanyang Technological University, Singapore. [Online]. Available: http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC-06/CEC06.htm

[12] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Adaptive configuration of evolutionary algorithms for constrained optimization," *Applied Mathematics and Computation*, vol. 222, pp. 680 – 711, 2013.

[13] G. Collange, S. Reynaud, and N. Hansen, "Covariance matrix adaptation evolution strategy for multidisciplinary optimization of expendable launcher family," in *13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference, American Institute of Aeronautics and Astronautics, Fort Worth, Texas*, 2010.

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

[15] H. G. Beyer and S. Finck, "On the design of constraint covariance matrix self-adaptation evolution strategies including a cardinality constraint," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 4, pp. 578–596, Aug 2012.

[16] H.-G. Beyer and B. Sendhoff, *Covariance Matrix Adaptation Revisited – The CMSA Evolution Strategy –*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 123–132.

[17] D. V. Arnold, "Reconsidering constraint release for active-set evolution strategies," in *Proceedings of the Genetic and Evolutionary Computation Conference*, ser. GECCO '17. ACM, 2017, pp. 665–672.

[18] A. Atamna, A. Auger, and N. Hansen, "Linearly convergent evolution strategies via augmented lagrangian constraint handling," in *Proceedings of the 14th ACM/SIGEVO Conference on Foundations of Genetic Algorithms*, ser. FOGA '17. New York, NY, USA: ACM, 2017, pp. 149–161.

[19] G. H. Wu, R. Mallipeddi, and P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Single Objective Real-Parameter Optimization," 2016, technical Report, Nanyang Technological University, Singapore. [Online]. Available: http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2017/CEC2017.htm

[20] H.-G. Beyer and B. Sendhoff, "Simplify your covariance matrix adaptation evolution strategy," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 5, pp. 746–759, Oct 2017.

[21] T. Takahama and S. Sakai, "Constrained optimization by the ϵ constrained differential evolution with gradient-based mutation and feasible elites," in *IEEE International Conference on Evolutionary Computation, CEC, part of WCCI 2006, Vancouver, BC, Canada, 16-21 July*, 2006, pp. 1–8. [Online]. Available: http://dx.doi.org/10.1109/CEC.2006.1688283

[22] R. Mallipeddi and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2010 competition on constrained real-parameter optimization," 2010, technical Report, Nanyang Technological University, Singapore. [Online]. Available: http://www3.ntu.edu.sg/home/epnsugan/index_files/CEC10-Const/CEC10-Const.htm

Table II

RESULTS OF 25 INDEPENDENT ALGORITHM RUNS AFTER 2×10^5 FUNCTION EVALUATIONS IN DIMENSION $N = 10$.

	C01	C02	C03	C04
Best	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Median	0.00000e+00	0.00000e+00	0.00000e+00	3.18942e+01
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	1.65316e-30	0.00000e+00	4.73317e-31	2.97766e+01
Std	7.57588e-30	0.00000e+00	1.73430e-30	1.75954e+01
Worst	3.78653e-29	0.00000e+00	7.88861e-30	6.46720e+01
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00

	C05	C06	C07	C08
Best	0.00000e+00	0.00000e+00	-4.37159e+02	-1.34840e-03
Median	0.00000e+00	0.00000e+00	-2.97522e+02	-1.34840e-03
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	0.00000e+00	3.57689e+01	-3.17339e+02	-1.34840e-03
Std	0.00000e+00	3.82482e+01	8.31863e+01	0.00000e+00
Worst	0.00000e+00	9.23767e+01	-1.69987e+02	-1.34840e-03
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00

	C09	C10	C11	C12
Best	-4.97525e-03	-5.09647e-04	-1.68819e-01	3.98790e+00
Median	-4.97525e-03	-5.09647e-04	-1.68819e-01	3.98790e+00
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	-4.97525e-03	-5.09647e-04	-1.67792e-01	6.99570e+00
Std	0.00000e+00	0.00000e+00	5.12778e-03	7.03324e+00
Worst	-4.97525e-03	-5.09647e-04	-1.43178e-01	2.27853e+01
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00

	C13	C14	C15	C16
Best	0.00000e+00	2.37633e+00	2.35619e+00	0.00000e+00
Median	0.00000e+00	2.37633e+00	8.63012e+00	0.00000e+00
c	(0,0,0)	(0,0,0)	(0,1,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	6.54538e-03	0.00000e+00
Mean	1.59463e-01	2.87425e+00	7.61357e+00	0.00000e+00
Std	7.97316e-01	7.63259e-01	6.47258e+00	0.00000e+00
Worst	3.98658e+00	3.89289e+00	2.11560e+00	0.00000e+00
FR	100	72	28	100
$\overline{vio}$	0.00000e+00	7.93954e-02	2.86476e-02	0.00000e+00

	C17	C18	C19	C20
Best	1.08556e-02	3.65977e+01	0.00000e+00	3.65636e-01
Median	9.85734e-01	3.65977e+01	0.00000e+00	1.31802e+00
c	(1,0,0)	(0,0,0)	(1,0,0)	(0,0,0)
$\overline{\nu}$	5.50000e+00	0.00000e+00	6.63359e+03	0.00000e+00
Mean	7.35195e-01	3.65977e+01	1.12033e+00	1.16962e+00
Std	3.22377e-01	1.65017e-05	2.34063e+00	3.92899e-01
Worst	9.14715e-01	3.65978e+01	6.91572e+00	1.61705e+00
FR	0	100	0	100
$\overline{vio}$	5.84982e+00	0.00000e+00	6.63487e+03	0.00000e+00

	C21	C22	C23	C24
Best	3.98790e+00	3.46248e-27	2.37633e+00	2.35616e+00
Median	3.98790e+00	3.95606e-27	2.37633e+00	5.49779e+00
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	4.41294e+00	6.37853e-01	2.49894e+00	6.12609e+00
Std	2.12289e+00	1.49164e+00	3.27363e-01	2.22143e+00
Worst	1.46028e+01	3.98658e+00	3.94272e+00	8.63938e+00
FR	100	100	96	100
$\overline{vio}$	0.00000e+00	0.00000e+00	9.44651e-04	0.00000e+00

	C25	C26	C27	C28
Best	0.00000e+00	9.98017e-01	3.65977e+01	0.00000e+00
Median	0.00000e+00	8.02797e-01	3.65977e+01	6.80280e+00
c	(0,0,0)	(1,0,0)	(0,0,0)	(1,0,0)
$\overline{\nu}$	0.00000e+00	5.50000e+00	0.00000e+00	6.63925e+03
Mean	0.00000e+00	7.54398e-01	7.59099e+01	8.68373e+00
Std	0.00000e+00	3.24742e-01	1.37003e+02	8.52235e+00
Worst	0.00000e+00	7.37173e-01	5.83500e+02	2.35855e+01
FR	100	0	100	0
$\overline{vio}$	0.00000e+00	5.41743e+00	0.00000e+00	6.64044e+03

Table III

RESULTS OF 25 INDEPENDENT ALGORITHM RUNS AFTER 6×10^5 FUNCTION EVALUATIONS IN DIMENSION $N = 30$.

	C01	C02	C03	C04
Best	2.11415e-28	1.74597e-28	3.16333e-28	2.78596e+01
Median	3.75498e-28	3.77532e-28	7.06819e-28	6.36771e+01
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	3.74522e-28	3.75972e-28	6.72867e-28	7.02885e+01
Std	7.20265e-29	7.26202e-29	1.07072e-28	3.11293e+01
Worst	4.85347e-28	5.51611e-28	8.34812e-28	1.51233e+02
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00

	C05	C06	C07	C08
Best	0.00000e+00	6.60639e+01	-1.14275e+03	-2.83981e-04
Median	0.00000e+00	1.39392e+02	-6.56915e+02	-2.83981e-04
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	0.00000e+00	1.80493e+02	-7.00796e+02	-2.83981e-04
Std	0.00000e+00	9.95983e+01	2.32472e+02	3.94013e-16
Worst	0.00000e+00	4.52818e+02	-3.76968e+02	-2.83981e-04
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00

	C09	C10	C11	C12
Best	-2.66551e-03	-1.02842e-04	-9.24932e-01	3.98253e+00
Median	-2.66551e-03	-1.02842e-04	-9.24932e-01	5.87889e+01
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	-2.66551e-03	-1.02842e-04	-9.24932e-01	4.60940e+01
Std	0.00000e+00	0.00000e+00	7.02631e-15	2.97043e+01
Worst	-2.66551e-03	-1.02842e-04	-9.24932e-01	9.97942e+01
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00

	C13	C14	C15	C16
Best	6.17530e-30	1.45277e+00	2.35616e+00	0.00000e+00
Median	2.72256e-27	1.61165e+00	2.36424e+00	0.00000e+00
c	(0,0,0)	(0,0,0)	(0,1,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	5.68794e-03	0.00000e+00
Mean	2.89231e-27	1.62885e+00	6.75132e+00	0.00000e+00
Std	1.88720e-27	9.17065e-02	5.93960e+00	0.00000e+00
Worst	6.90002e-27	1.76746e+00	2.49031e+00	0.00000e+00
FR	100	100	28	100
$\overline{vio}$	0.00000e+00	0.00000e+00	1.32767e-02	0.00000e+00

	C17	C18	C19	C20
Best	9.72432e-01	3.65203e+01	0.00000e+00	1.99567e+00
Median	9.74542e-01	3.65203e+01	8.14833e+00	7.82205e+00
c	(1,0,0)	(0,0,0)	(1,0,0)	(0,0,0)
$\overline{\nu}$	1.55000e+01	0.00000e+00	2.13803e+04	0.00000e+00
Mean	9.71519e-01	3.65482e+01	7.59850e+00	7.66346e+00
Std	1.74694e-02	1.39410e-01	9.07569e+00	1.23984e+00
Worst	1.02744e+00	3.72173e+01	3.30986e+01	8.69217e+00
FR	0	100	0	100
$\overline{vio}$	1.55160e+01	0.00000e+00	2.13833e+04	0.00000e+00

	C21	C22	C23	C24
Best	9.77517e+00	1.95451e-25	1.46426e+00	2.35619e+00
Median	5.87889e+01	2.45512e-25	1.64756e+00	1.17809e+01
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	4.84465e+01	2.47154e-25	1.65017e+00	9.14202e+00
Std	1.58282e+01	2.96830e-26	8.73119e-02	3.91965e+00
Worst	5.87889e+01	3.06252e-25	1.82715e+00	1.17810e+01
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00

	C25	C26	C27	C28
Best	0.00000e+00	9.80793e-01	3.65203e+01	8.57150e+00
Median	0.00000e+00	9.83960e-01	3.65203e+01	5.35723e+01
c	(0,0,0)	(1,0,0)	(0,0,0)	(1,0,0)
$\overline{\nu}$	0.00000e+00	1.55000e+01	0.00000e+00	2.14284e+04
Mean	0.00000e+00	9.77687e-01	3.65761e+01	5.84051e+01
Std	0.00000e+00	1.77218e-02	1.93004e-01	2.32881e+01
Worst	0.00000e+00	1.02891e+00	3.72173e+01	8.15130e+01
FR	100	0	100	0
$\overline{vio}$	0.00000e+00	1.55351e+01	0.00000e+00	2.14273e+04

Table IV
RESULTS OF 25 INDEPENDENT ALGORITHM RUNS AFTER 10×10^5 FUNCTION EVALUATIONS IN DIMENSION $N = 50$.

	C01	**C02**	**C03**	**C04**
Best	1.79260e-27	2.38675e-27	3.33151e-27	7.16366e+01
Median	2.85451e-27	2.76832e-27	3.94976e-27	1.13425e+02
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	2.86649e-27	2.89158e-27	4.05510e-27	1.18904e+02
Std	3.95654e-28	3.62945e-28	3.58307e-28	2.80329e+01
Worst	3.49875e-27	3.69957e-27	5.05699e-27	1.94016e+02
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
	C05	**C06**	**C07**	**C08**
Best	0.00000e+00	8.96798e+01	-1.78555e+03	-1.34535e-04
Median	0.00000e+00	2.44139e+02	-1.43726e+03	-1.34535e-04
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	0.00000e+00	2.87077e+02	-1.37452e+03	-1.34535e-04
Std	0.00000e+00	1.31115e+02	3.40376e+02	1.22963e-16
Worst	0.00000e+00	6.00815e+02	-1.52561e+02	-1.34535e-04
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
	C09	**C10**	**C11**	**C12**
Best	-2.03709e-03	-4.82664e-05	-2.01060e+00	3.98145e+00
Median	-2.03441e-03	-4.82664e-05	-2.00941e+00	6.21667e+01
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	6.65930e-01	-4.82658e-05	-3.70107e+00	5.05710e+01
Std	1.87113e+00	1.82543e-09	8.28813e+00	2.04558e+01
Worst	6.76956e+00	-4.82589e-05	-3.69191e+01	6.21667e+01
FR	100	100	76	100
$\overline{vio}$	0.00000e+00	0.00000e+00	2.25814e-06	0.00000e+00
	C13	**C14**	**C15**	**C16**
Best	9.13945e-28	1.25625e+00	8.63931e+00	0.00000e+00
Median	1.34380e-25	1.34727e+00	1.49225e+01	0.00000e+00
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	2.94609e+02	1.34236e+00	1.45443e+01	0.00000e+00
Std	4.43651e+02	3.73854e-02	1.00708e+01	0.00000e+00
Worst	1.18276e+03	1.40323e+00	2.33669e+00	0.00000e+00
FR	100	100	68	100
$\overline{vio}$	0.00000e+00	0.00000e+00	2.04615e-03	0.00000e+00
	C17	**C18**	**C19**	**C20**
Best	1.03777e+00	3.64693e+01	1.07504e+00	1.38740e+01
Median	1.03025e+00	3.64693e+01	7.12232e+00	1.52590e+01
c	(1,0,0)	(0,0,0)	(1,0,0)	(0,0,0)
$\overline{\nu}$	2.55000e+01	0.00000e+00	3.61220e+04	0.00000e+00
Mean	1.03467e+00	3.65879e+01	1.25226e+01	1.51521e+01
Std	6.11782e-03	5.93037e-01	9.73061e+00	5.50861e-01
Worst	1.04324e+00	3.94345e+01	4.07398e+01	1.59346e+01
FR	0	100	0	100
$\overline{vio}$	2.55000e+01	0.00000e+00	3.61280e+04	0.00000e+00
	C21	**C22**	**C23**	**C24**
Best	3.98317e+00	9.99546e-25	1.25578e+00	8.63938e+00
Median	6.21667e+01	1.14227e+03	1.33507e+00	1.17810e+01
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	5.52513e+01	9.76091e+02	1.34299e+00	1.21579e+01
Std	1.68432e+01	6.05679e+02	4.51597e-02	1.88494e+00
Worst	6.21667e+01	2.21972e+03	1.41518e+00	1.49226e+01
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
	C25	**C26**	**C27**	**C28**
Best	0.00000e+00	1.02647e+00	3.64693e+01	2.65416e+01
Median	0.00000e+00	1.03637e+00	3.64693e+01	9.10592e+01
c	(0,0,0)	(1,0,0)	(0,0,0)	(1,0,0)
$\overline{\nu}$	0.00000e+00	2.55000e+01	0.00000e+00	3.62007e+04
Mean	0.00000e+00	1.03259e+00	3.64693e+01	9.52777e+01
Std	0.00000e+00	7.26148e-03	5.15651e-06	4.42570e+01
Worst	0.00000e+00	1.02529e+00	3.64693e+01	2.08776e+02
FR	100	0	100	0
$\overline{vio}$	0.00000e+00	2.55000e+01	0.00000e+00	3.62026e+04

Table V
RESULTS OF 25 INDEPENDENT ALGORITHM RUNS AFTER 20×10^5 FUNCTION EVALUATIONS IN DIMENSION $N = 100$.

	C01	**C02**	**C03**	**C04**
Best	3.23317e-26	3.10296e-26	3.68397e-26	2.16900e+02
Median	4.10676e-26	4.01605e-26	4.41296e-26	2.44759e+02
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	3.97567e-26	3.98324e-26	4.42639e-26	2.50768e+02
Std	4.38169e-27	3.90338e-27	4.76039e-27	2.27963e+01
Worst	4.68313e-26	4.78019e-26	5.39501e-26	3.09431e+02
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
	C05	**C06**	**C07**	**C08**
Best	2.57032e+01	2.76009e+02	-3.39138e+03	-4.73405e-05
Median	2.68480e+01	9.78040e+02	-2.64807e+03	-4.60526e-05
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	2.68528e+01	8.05511e+02	-2.48247e+03	-4.55131e-05
Std	5.17808e-01	3.99321e+02	7.89462e+02	1.32142e-06
Worst	2.82252e+01	1.29612e+03	-2.80746e+02	-4.20729e-05
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
	C09	**C10**	**C11**	**C12**
Best	3.46265e-05	-5.39999e-06	-5.73016e+00	1.88577e+01
Median	2.16310e+00	-1.66908e-06	-5.72868e+00	3.15768e+01
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	3.03527e+00	-1.15932e-06	-6.67039e+00	3.00505e+01
Std	3.81979e+00	2.34224e-06	6.10239e+00	4.21847e+00
Worst	1.63674e+01	3.17053e-06	-3.57517e+01	3.15768e+01
FR	92	100	96	100
$\overline{vio}$	1.28972e-02	0.00000e+00	2.47649e+01	0.00000e+00
	C13	**C14**	**C15**	**C16**
Best	4.02130e+01	9.03602e-01	2.35612e+00	0.00000e+00
Median	4.08579e+01	9.57591e-01	1.49225e+01	0.00000e+00
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	4.14351e+01	9.58110e-01	8.63968e+00	0.00000e+00
Std	1.39174e+00	2.68118e-02	6.01542e+00	0.00000e+00
Worst	4.62473e+01	1.00418e+00	2.36038e+00	0.00000e+00
FR	100	100	88	100
$\overline{vio}$	0.00000e+00	0.00000e+00	2.32818e-04	0.00000e+00
	C17	**C18**	**C19**	**C20**
Best	1.09505e+00	3.63770e+01	3.37320e+01	3.39085e+01
Median	1.09592e+00	3.63770e+01	5.18906e+01	3.55781e+01
c	(1,0,0)	(0,0,0)	(1,0,0)	(0,0,0)
$\overline{\nu}$	5.05000e+01	0.00000e+00	7.30388e+04	0.00000e+00
Mean	1.09382e+00	3.63770e+01	8.98566e+01	3.54396e+01
Std	1.97660e-03	3.50005e-06	6.75222e+01	7.24673e-01
Worst	1.09284e+00	3.63770e+01	2.34440e+02	3.64781e+01
FR	0	100	0	100
$\overline{vio}$	5.05000e+01	0.00000e+00	7.30676e+04	0.00000e+00
	C21	**C22**	**C23**	**C24**
Best	3.15768e+01	2.79549e+03	9.36578e-01	2.35612e+00
Median	3.15768e+01	3.99373e+03	9.63368e-01	8.63941e+00
c	(0,0,0)	(0,0,0)	(0,0,0)	(0,0,0)
$\overline{\nu}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
Mean	3.15768e+01	4.23137e+03	9.67335e-01	9.39332e+00
Std	5.60796e-14	1.04096e+03	1.78701e-02	4.99044e+00
Worst	3.15768e+01	6.53606e+03	1.00702e+00	1.49226e+01
FR	100	100	100	100
$\overline{vio}$	0.00000e+00	0.00000e+00	0.00000e+00	0.00000e+00
	C25	**C26**	**C27**	**C28**
Best	0.00000e+00	1.09087e+00	3.63770e+01	1.27127e+02
Median	0.00000e+00	1.09636e+00	3.63770e+01	1.73367e+02
c	(0,0,0)	(1,0,0)	(0,0,0)	(1,0,0)
$\overline{\nu}$	0.00000e+00	5.05000e+01	0.00000e+00	7.31194e+04
Mean	4.23913e-14	1.09461e+00	7.67704e+01	1.71016e+02
Std	1.30290e-13	1.89628e-03	2.01967e+02	2.03571e+01
Worst	4.80650e-13	1.09292e+00	1.04621e+03	2.07738e+02
FR	100	0	96	0
$\overline{vio}$	0.00000e+00	5.05000e+01	8.64869e-01	7.31208e+04