

A CMA-ES-based 2-Stage Memetic Framework for Solving Constrained Optimization Problems

Vinícius Veloso de Melo

Institute of Science and Technology,
Federal University of São Paulo, UNIFESP,
São José dos Campos, São Paulo, Brazil, 12231-280
Email: vinicius.melo@unifesp.br

Giovanni Iacca

INCAS³,
P.O. Box 797, 9400 AT Assen, The Netherlands
Email: giovanniiacca@incas3.eu

Abstract—Constraint optimization problems play a crucial role in many application domains, ranging from engineering design to finance and logistics. Specific techniques are therefore needed to handle complex fitness landscapes characterized by multiple constraints. In the last decades, a number of novel meta-heuristics have been applied to constraint optimization. Among these, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) has been attracting lately the most attention of researchers. Recent variants of CMA-ES showed promising results on several benchmarks and practical problems. In this paper, we attempt to improve the performance of an adaptive penalty CMA-ES recently proposed in the literature. We build upon it a 2-stage memetic framework, coupling the CMA-ES scheme with a local optimizer, so that the best solution found by CMA-ES is used as starting point for the local search. We test, separately, the use of three classic local search algorithms (Simplex, BOBYQA, and L-BFGS-B), and we compare the baseline scheme (without local search) and its three memetic variants with some of the state-of-the-art methods for constrained optimization.

I. INTRODUCTION

Several real-world optimization problems are characterized by the presence of one or more constraints which limit the search space where the feasible solutions lie. Without loss of generality, we define a constrained optimization problem as the search for $\mathbf{x}$ that minimizes $f(\mathbf{x})$, subject to the constraints $h_i(\mathbf{x}) = 0, i = 1, 2, \dots, m$ and $g_j(\mathbf{x}) \leq 0, j = 1, 2, \dots, p$, where $f(\mathbf{x})$ is the objective (or fitness) function to be optimized, and $\mathbf{x} \in \mathbb{R}^n$ is an n -dimensional vector of design variables, $\mathbf{x} = [x_1, x_2, \dots, x_n]$. Generally speaking, variables can be integer, discrete and continuous. Each $x_k, k = 1, 2, \dots, n$ is usually bounded by lower and upper limits $L_k \leq x_k \leq U_k$ (box constraints); $h_i(\mathbf{x})$ and $g_j(\mathbf{x})$ are called equality and inequality constraints, being their number m and p , respectively. Both kinds of constraint can be linear or nonlinear.

In the last decades, several meta-heuristics, such as Genetic Algorithms (GA) [1], [2], Particle Swarm Optimization (PSO) [3], [4], and Differential Evolution (DE) [5], [6], have been proposed to solve constrained optimization problems. Lately, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [7], which has proven particularly successful in unconstrained optimization, has been attracting the attention also in the constrained optimization research community, with seminal works represented by [8]–[13]. In [8], box constraints are handled allowing the offspring's genome to violate the constraints, while its fitness is penalized by a quadratic function of the

distance from the evaluation point. In [10], a surrogate model of the constraint functions is built during the evolutionary process; such a model is used to rotate the covariance matrix in the proximity of the constraint boundaries. Similarly, the (1+1)-CMA-ES scheme proposed in [11], and further extended in [13], learns a constraint model online and uses it to reduce the variances of the matrix along the directions where a constraint violation is observed: the resulting strategy is then able to approach the boundaries of the feasible domain and search in directions tangential to them, without violating constraints.

Among the few studies on CMA-ES, in our previous work [12] we proposed a modified CMA-ES with an adaptive penalty function that aggregates the constraint violation information over the entire population. Since our method has shown a fairly good performance on a broad set of benchmark functions and engineering problems, we deemed interesting to further investigate its scheme. In this paper, we attempt to improve it by combining CMA-ES with a local optimizer (in the following, referred to as “LO”), aiming at refining the best solution found by the Evolution Strategy. Thus we devise a 2-stage memetic framework¹ where CMA-ES is coupled (separately) with three classic local optimizers, namely Simplex [15], BOBYQA [16], and L-BFGS-B [17].

The paper is organized as follows: the next Section summarizes the main elements of our previous modified CMA-ES. Section III describes the proposed variants obtained combining it with local search. Section IV details our experiments, where we compare the baseline scheme (without local search) with the novel memetic variants and with several methods from the literature, on a set of nine well-known constrained optimization problems. Finally, Section V concludes this work.

II. CMA-ES WITH ADAPTIVE PENALTY FUNCTION

The original CMA-ES consists of the following. At the beginning of the optimization, a mean vector $\mathbf{m} \in \mathbb{R}^n$ is randomly initialized inside the problem bounds $L_k \leq m_k \leq U_k$, for $k = 1, 2, \dots, n$, where n is the number of variables of the problem; additionally, a covariance matrix $\text{cov} = \sigma^2 C$ is defined, where $C \in \mathbb{R}^{n \times n}$ is initially set to I , the identity matrix, and σ is the initial step size, a parameter of the

¹We must remark that our method is not a memetic algorithm in the classic sense. Rather, it can be seen as an instance of memetic computing frameworks according to the much broader definition proposed in [14], that are “computing structures composed of interacting modules (memes)”, i.e. “simple strategies whose harmonic coordination allows the solution of various problems.”

algorithm. After the initialization, each step of the algorithm first samples λ new solutions from the multivariate normal distribution $\mathcal{N}(\mathbf{m}, \text{cov})$; then, $\mathbf{m}$, σ and C are adaptively updated from a weighted sum of the best μ solutions in the population. The standard settings of λ and μ are, respectively, $4 + \lfloor 3 \ln(n) \rfloor$ and $\lfloor \lambda/2 \rfloor$, while the initial value of σ is set, in general, to 0.5 [7]. Once the distribution is updated, the loop is repeated until a stop condition is met. The standard stopping criteria are based on: (1) a maximum number of iterations; (2) a tolerance on the fitness value to reach (if known); (3) a negative definition of the matrix C ; (4) a tolerance (1E-12) on the smallest standard deviation. In our previous work [12], we introduced simple modifications in the CMA-ES sampling process and stopping criteria, as well as an adaptive penalty function that aggregates the violation information on the entire population. In what follows, we summarize these elements.

A. Modifications of the CMA-ES scheme

The sampling scheme is modified in such a way that it includes an element proportional to the range $[L, U]$, so that $\mathbf{x}_i = \mathbf{m} + \text{domain} \times \sigma \times \mathcal{N}(0, C)$, $i = 1, 2, \dots, \lambda$, where *domain* is *percentageDomain* of the range for each variable. For instance, for a problem with two variables whose domains are $[-10, 10]$ and $[0, 100]$, so that their range width is respectively 20 and 100, a *percentageDomain* = 50% gives *domain* = $[10, 50]$. In this way the effect of σ is proportional to the width of each variable's domain, thus avoiding that along some dimensions newly sampled solutions are too far from (or too close to) the current mean $\mathbf{m}$.

Moreover, in addition to the original stopping criteria, we stop the CMA-ES when: (1) the number of consecutive iterations where the best solution is unfeasible reaches a given limit *infeasMaxTrials*; (2) σ becomes greater than a given threshold *maxSigma*; C is numerically not symmetrical and positive definite (based on a threshold, *Machine_double_precision*); (4) any eigenvalue of C becomes negative². When at least one of the criteria is met, the CMA-ES search is restarted from a new initial random point, setting $\lambda = \lambda \cdot 1.5$ (as suggested in [18]) and $\sigma = 0.1$, while all other parameters are reset to their default values.

B. Adaptive penalty function

The main idea of the adaptive penalty function is to update the penalty value based on the constraints violation of *all* the solutions in the population (or, in case of LO, the current solution to evaluate). An unfeasible solution that presents small constraints violations can have a small penalty and yet present a better fitness value than a feasible solution (with no violations). However, an unfeasible solution cannot have a better value than the best feasible solution in the population. Therefore, the larger the violations, the larger the penalty.

With reference to Algorithm 1, whenever the penalty mechanism is activated (i.e., at the end of each CMA-ES generation or, in the memetic variants described below, every time the local optimizer generates a new solution), the penalty function calculates the objective value and the constraint violations of all input solutions in *pop* ($N = \lambda$ in case of CMA-ES, $N = 1$ in case of local optimizer). In the pseudo-code, we assume that

Algorithm 1 Adaptive penalty function

```

global worstValue, fBestFeas
function PENALTY(pop = [pop1, ..., popN])
    penalty = 0, bestfeas = worstValue
    for all  $i \in [1, \dots, N]$  do
         $\mathbf{x} = \text{pop}_i$ 
        if any element of  $\mathbf{x}$  is outside the box bounds then
             $f(\mathbf{x}) = \text{worstValue}$ 
             $\mathbf{g}(\mathbf{x}) = [\text{worstValue}, \dots, \text{worstValue}]$ 
        else
             $[f(\mathbf{x}), \mathbf{g}(\mathbf{x})] = \text{objfun}(\mathbf{x})$ 
        end if
         $\text{violation}_i = \text{sum}(\mathbf{g}(\mathbf{x})[\mathbf{g}(\mathbf{x}) > 0])$ 
         $\text{penalty} = \max(\text{penalty}, \text{violation}_i)$ 
        if  $\text{violation}_i \leq 0$  and  $f(\mathbf{x}) < \text{bestfeas}$  then
             $\text{bestfeas} = f(\mathbf{x})$ 
            if  $\text{bestfeas} < f_{\text{BestFeas}}$  then
                 $f_{\text{BestFeas}} = \text{bestfeas}$ 
            end if
        end if
    end for
    for all  $i \in [1, \dots, N]$  do
        if  $\text{violation}_i > 0$  then
             $f(\mathbf{x}) = \text{bestfeas} + |\text{bestfeas} \cdot 0.01| +$ 
                 $|f(\mathbf{x}) - \text{bestfeas}| \cdot |\text{violation}_i / \text{penalty}|$ 
        end if
    end for
    return  $f\mathbf{x}$ 
end function

```

function *objfun*($\mathbf{x}$) returns both the objective value $f(\mathbf{x})$ and a vector $\mathbf{g}(\mathbf{x})$, containing for each constraint the corresponding violation (assumed positive for violated constraints, otherwise negative or zero) of the focal solution $\mathbf{x}$, which is the i^{th} solution in *pop*. The objective value of the best feasible solution in *pop*, *bestfeas*, is also calculated. If there are no feasible solutions among the input solutions, *bestfeas* is set to a predefined constant *worstValue*. Moreover, the maximum sum of constraint violations (violation_i) among all solutions in *pop* is stored into an adaptive penalty value (in the pseudo-code, *penalty*). Also, if a feasible solution better than the current best is found, the global variable f_{BestFeas} , i.e. the fitness of the best feasible solution, is updated.

After all input solutions have been evaluated, the penalty function updates the objective value of each unfeasible solution by creating a composed penalty that takes into account: the objective value of the best feasible solution in *pop*, *bestfeas*; the difference between that value and the objective value of the current solution; the sum of its constraint violations, violation_i , scaled by *penalty*. It is clear that when the violations get closer to zero, or when the fitness of the current individual is close to *bestfeas*, the penalty decreases.

III. CMA-ES-BASED 2-STAGE MEMETIC FRAMEWORK

In the present work, we build upon the CMA-ES scheme described in the previous Section a 2-stage memetic framework where the modified CMA-ES is coupled with a local optimizer. In such a framework, the best solution $\mathbf{x}_{\text{best}}$ found by CMA-ES (that plays the role of global searcher), is used as starting point for the LO, an approach similar to “super-fit” schemes [19]–[21], where a high-quality solution is passed from an algorithm to another (although those schemes follow the opposite logic: a locally improved solution is fed into the initial population of an evolutionary algorithm). Both the components (CMA-ES and LO) make use of the adaptive penalty function seen

²We calculate eigenvalues in R, using the LAPACK routines *DSYEV*, *DGEEV*, *ZHEEV* and *ZGEEV*, highly optimized for efficient computations.

in Algorithm 1. Because some local optimizers are not able to search on the bounds of the domain, if necessary we correct the solution by adding/subtracting $1E - 10$ to each violated variable. We show the pseudo-code of the proposed CMA-ES-based memetic framework in Algorithm 2.

Algorithm 2 CMA-ES-based 2-Stage Memetic Framework

```

// CMA-ES
initialize covariance matrix  $C = I$ , and step-size  $\sigma$ 
initialize the mean vector  $\mathbf{m}$  with a random solution in  $[\mathbf{L}, \mathbf{U}]$ 
while CMA-ES budget is not exhausted do
    sample  $\lambda$  new individuals from distribution  $\mathcal{N}(\mathbf{m}, \sigma^2 C)$ 
    evaluate individuals (apply penalty using Algorithm 1)
    update  $\mathbf{m}$  based on a weighted sum of the best  $\mu$  individuals
    update covariance matrix  $C$  and step-size  $\sigma$ 
    restart if at least one stop condition is met
end while
// local optimizer
set  $\mathbf{x}_{trial}$  as the current  $\mathbf{x}_{best}$  returned by CMA-ES
 $\mathbf{x}_{trial} = \text{LOCALOPTIMIZATION}(\mathbf{x}_{trial})$  (apply penalty using Algorithm 1)
return the best individual between  $\mathbf{x}_{trial}$  and  $\mathbf{x}_{best}$ 

```

To investigate the effect of diverse local search logics, we test separately three different LOs, namely Simplex [15], BOBYQA [16], and L-BFGS-B [17]. Although our analysis can be extended to other LOs, we chose these methods since they are representative of three different ways of performing local search, that is by recombining multiple solutions (Simplex), using a quadratic approximation of the fitness function (BOBYQA), or building a numerical approximation of its gradient (L-BFGS-B). Moreover, these are classic methods that have been studied extensively and whose properties are well-known. We highlight the key points of the three methods below.

Nelder-Mead Simplex - In the Nelder-Mead Simplex [15], a set of $n + 1$ points, $\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_n$, form an n -dimensional polytope, or simplex, in the search space. At each iteration of the algorithm, these points are sorted according to their objective function's value, so that $\mathbf{x}_0$ has the best value and $\mathbf{x}_n$ presents the worst value. The procedure then consists in constructing a candidate replacement point $\mathbf{x}_r$ for $\mathbf{x}_n$ by reflection of $\mathbf{x}_n$ with respect to the barycenter $\mathbf{x}_m$ of the other $\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_{n-1}$ points. Depending on the fitness of $\mathbf{x}_r$ compared to $\mathbf{x}_0$ and $\mathbf{x}_{n-1}$, an extension point may be created in an optimistic attempt to explore further in the same direction, or on the contrary a contraction point may be computed closer to $\mathbf{x}_m$. If none of the above attempts leads to a better solution, the simplex is contracted around its best point, to reduce the exploration range in the next iteration. This procedure is repeated until a stopping criterion is met.

BOBYQA - The Powell's Bounded Optimization BY Quadratic Approximation [16] minimizes a function of n variables by a trust region method that forms quadratic models by interpolation. In particular, each iteration employs a quadratic approximation Q of the objective function $f(\mathbf{x})$ based on m interpolation points chosen and adjusted automatically, being $m = 2n + 1$ the typical value. The approximation is updated by minimizing the Frobenius norm of the change to the second derivative matrix of Q , thus no derivatives of the objective function are required. The BOBYQA method handles by definition box constraints on the parameters.

L-BFGS-B - The Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Box constraints [17] belongs

to the family of quasi-Newton optimization methods, which approximate and update the Hessian matrix by means of (approximated) gradient evaluations. Unlike its basic version (BFGS), which uses a complete $n \times n$ Hessian approximation, the L-BFGS-B algorithm stores only a history of the past m updates of the solution $\mathbf{x}$ and its gradient $\nabla f(\mathbf{x})$ (being m small, typically < 10), and builds a limited-memory representation of the Hessian upon such updates. This feature makes the L-BFGS-B method particularly suitable for large-scale optimization problems. In addition to that, the L-BFGS-B algorithm handles by definition box constraints.

IV. NUMERICAL RESULTS

To evaluate the performance of the three proposed CMA-ES + LO memetic frameworks, we considered nine well-known constrained optimization problems, namely (1) g06, (2) g07, (3) g09, and (4) g10, taken from the constrained optimization benchmark [22], and five engineering design problems taken from [23], namely (5) Design of a Welded Beam, (6) Design of a Speed Reducer, (7) Design of a Three-Bar Truss, (8) Design of a Pressure Vessel and (9) Minimization of the weight of a Tension/Compression Spring. Below, we detail the algorithmic setup and we analyze the numerical results.

A. Algorithm and Experiment Setup

All our experiments were implemented in R, using various packages specific for optimization³. As for CMA-ES, we used the original implementation available in the R package `cmaES`, and modified its code as explained in Section II. In the following, we refer to the modified CMA-ES with adaptive penalty and restart, but without local optimization, as “standalone CMA-ES” or simply “CMA-ES”, our baseline method. All the CMA-ES parameters were set to the package default values, except the initial population size. The latter was set to the value $3 \cdot \lceil 4 + 3 \ln(n) \rceil$ suggested in [7], being n the dimension of the problem, thus resulting in 18, 30, 27, 30, 24, 27, 18, 24, and 21 for problems 1 to 9, respectively. CMA-ES was restarted from a new random solution whenever one of the stop conditions was met. The number of function evaluations (NFEs) for CMA-ES was set to 30000, 75000, 30000, 120000, 21000, 30000, 3000, 45000, and 30000 respectively for Problems 1 to 9. These values were chosen to have a fair comparison with the best algorithms in the literature. The other parameters, introduced in Section II, were set as in our previous work [12], namely $maxSigma = 2$, $infeasMaxTrials = 25$, $\lambda = 1.5$ and $Machine_double_precision = 2.220446E-16$, $percentageDomain = 0.5$.

As for the local optimizers, for the Nelder-Mead Simplex we used the “bounded” (box constrained) implementation `nmkb` available in the R package `dfoptim`, where bounds are enforced by means of a parameter transformation; for BOBYQA, we used the implementation `bobyqa` included in the R package `minqa`; finally, for L-BFGS-B we used the implementation `optim` from the R package `stats`. For each local optimizer, we maintained the default parameter configuration in the R package. Each local search method was run using the best solution found by CMA-ES as starting point and continuing the search for a maximum of 5000 evaluations.

³R packages are available at <http://cran.r-project.org/web/packages>.

For each problem we executed 100 runs and obtained results for CMA-ES, CMA-ES+Simplex, CMA-ES+BOBYQA, CMA-ES+L-BFGS-B. We then considered the best objective values obtained in all runs of each of the four algorithms, and performed the Bartlett's test [24] to test the homogeneity of variances among them. If there were significant difference in the variances, we selected the algorithm with the lowest median for comparison against the state-of-the-art methods. Otherwise, we performed the Friedman's test [25] to test if the samples came from populations having identical properties. When the null hypothesis was rejected, suggesting that at least one sample was significantly different from the others, we performed the pairwise Wilcoxon signed-rank test [26] to rank the results and select the algorithm showing the lowest median of the objective function. In all tests we set $\alpha = 0.05$.

B. Discussion

The analysis of the numerical results is presented next, separately for each of the nine problems we considered. In the following, for the problems where we detected a significant difference between the medians of a memetic variant (i.e., CMA-ES + LO) and the standalone CMA-ES, we show the statistical comparison of the results obtained with the four methods, namely CMA-ES and the three variants CMA-ES + LO, see Tables X-XIV. It is important to note that all the best solutions found in this work are feasible. Also, since we chose not to use any tolerance for constraint violations, in our results all constraints are active. As a final remark, we note that for all our experiments we adopted a π value 3.14159265358979.

Problem 1: g06 - We detected a significant difference in the variances of the results, meaning that at least one LO was capable of improving the solution from CMA-ES. Therefore, we selected the approach with the lowest median, in this case CMA-ES+Simplex, which was the only method that improved the outliers and escaped from local optima. The results of the four methods are presented in Table X, where for simplicity of notation we omitted "CMA-ES" from the names of the three memetic variants. Compared to the best results from the literature, see Table I, CMA-ES+Simplex is able to obtain competitive results, reaching the best fitness value known so far, namely $f(x) = -6961.8138755801664956$.

Problem 2: g07 - The Friedman's test resulted in significant differences in the samples, and based on the pairwise Wilcoxon's test it was possible to rank CMA-ES+Simplex as the best approach (see Table XI). The best fitness we obtained is $f(x) = 24.306209068179839505$, on par with the best results from the literature, but with fewer evaluations and with a smaller standard-deviation (referred to as "SD"), see Table II.

Problem 3: g09 - In this case the four algorithms are statistically equivalent, therefore we report only the results obtained with the standalone CMA-ES. The best result we obtained is $f(x) = 680.63005737440187204$. From the comparison with the best results from the literature, see Table III, it can be seen that CMA-ES is particularly competitive, as it reaches the best known result using a much smaller computational budget (1/10 of the budget needed by other methods, or less). Also the robustness of the algorithm, indicated by a SD of $1.42E-13$, seems stronger than other algorithms from the literature.

Problem 4: g10 - For this problem, CMA-ES+Simplex was the best method (statistically different from the others),

see Table XII. The best solution found has a fitness value $f(x) = 7049.2480205286810815$. From Table IV, it can be seen that CMA-ES+Simplex obtains results on par with [6], although with much smaller SD and NFEs.

Problem 5: Welded Beam - For this problem, the standalone CMA-ES presented results sufficiently close to the known global optimum without the need of a LO. Thus, again the comparison is made using CMA-ES. The best solution found by our approach has a fitness value $f(x) = 1.7248523085973674895$. When compared to results from the literature, see Table V, the modified CMA-ES achieved results equivalent to the best ones reported by [6] and [27] with similar NFEs but slightly smaller SD. The smallest SD was achieved in [28], but requiring twice NFEs. On the other hand, the work [29] used 25% fewer evaluations, but found worse results.

Problem 6: Speed Reducer - For this problem the best solution we found has a fitness $f(x) = 2994.4710661468225226$. Medians were considered equivalent by the statistical test. Because none of the local optimizers improved upon the solutions found by CMA-ES, we present the results produced by the standalone CMA-ES. Table VI shows the comparison with other methods. It can be seen that the standalone CMA-ES presents smaller SD than MVDE [29] but bigger than DELC [27] and DEDS [30]. MBA [28] used only 6300 evaluations but the mean fitness was much bigger.

Problem 7: Three-Bar Truss - This problem was solved with small effort by CMA-ES, without using any local optimization method. The best fitness we obtained was $f(x) = 263.89584337654889623$. In Table VII, one can see that the worst solution found by CMA-ES in 3000 evaluations is very close to the ones found by PSO-DE [31] and DELC [27], respectively in 17600 and 10000 evaluations.

Problem 8: Pressure Vessel - After running the statistical tests, it was identified that the improvement performed by Simplex was significant. The results of the four methods are presented in Table XIII. Increasing the maximum NFEs in our algorithm could possibly improve the final results. As shown in Table VIII, however, the budget we used to obtain the best value (45300 evaluations) is bigger than the budget required by the state-of-the-art methods to obtain similar results. On the other hand, this NFEs is smaller than the budget used by other methods, although the final mean fitness obtained with our algorithm is much better.

Problem 9: Tension/Compression Spring - In this last problem, CMA-ES + Simplex was selected for comparison against the state-of-the-art algorithm for the same previous reason. The comparisons against CMA-ES and the other two memetic variants are presented in Table XIV. It can be seen that, given a budget of 30300 evaluations, the best fitness found with CMA-ES + Simplex was $f(x) = 0.012665232788560226024$. An analysis of the fitness trend revealed that the improvement rate was not very big and in the last iterations there were also short periods of stagnation from which our modified CMA-ES escaped thanks to the restart mechanism. The comparisons with state-of-the-art methods are shown in Table IX. The best results in terms of SD are obtained by HEAA [32], but CMA-ES + Simplex has also competitive performance, outperforming most of the considered methods.

V. CONCLUSIONS

In this paper we proposed a CMA-ES-based 2-stage memetic framework for solving constrained optimization problems. Such framework was obtained combining a previously proposed adaptive penalty CMA-ES with three different local optimizers, namely Simplex, BOBYQA, and L-BFGS-B (considered separately), in such a way that the best solution obtained by CMA-ES (acting as global search) was used as initial point to the local search, aiming at improving upon it.

The performance of the proposed memetic framework was assessed on four functions taken from the CEC 2006 benchmark, and five well-known constrained optimization engineering problems. We compared the standalone (i.e., without local optimization) adaptive penalty CMA-ES with the three memetic variants including local optimization. From the analysis of the results, we observed that the four methods resulted statistically different in five problems, for which the combination CMA-ES + Simplex showed the best results. In all the other cases, none of the three local optimizers was able to improve upon the solutions found by the stand-alone CMA-ES. To complete our analysis, we compared the best results obtained with the four aforementioned methods with the best results from the literature. Such a comparison showed that the proposed methods are competitive, in terms of solution quality and number of evaluations, with the state-of-the-art.

The main result of this study is that local search methods are not always effective in continuous constrained optimization, even when the starting solution is of high quality. A possible explanation is that following the gradient in the presence of constraints might actually mislead the search and make it stuck on the boundaries of the feasible region. However, this result is not surprising as the local methods we tested are not specifically designed for handling constraints. Therefore, our conclusions are limited only to the empirical evidence gathered in our experiments, but we cannot consider them valid in general. As well-established in the literature, the performances of memetic structures strictly depend, in fact, on the specific local search operators used (and how they are coordinated), and may change dramatically from one set of problems to another. Promising results have been obtained, for instance, by memetic algorithms including nonlinear programming techniques [33], ad-hoc operators [34], or alternative local search methods [35], [36]. Going beyond the scope of this paper, memetic computing has been successful also in combinatorial constrained optimization, such as in vehicle routing problems [37].

Nevertheless, local search should not be considered beneficial *per se* and therefore it is not always a necessary component in an algorithmic structure (as suggested by the Ockham's Razor [38]). Rather, one should analyze, whenever possible, the properties of the problems at hand, test empirically which local search method performs better, and determine if the use of local search is really needed. Interestingly enough, this reasoning applies to every component of a memetic structure.

ACKNOWLEDGMENTS

This study was financially supported by CNPq (Projeto Universal no. 486950/2013-1), and CAPES (CsF no. 12180-13-0). INCAS³ is co-funded by the Province of Drenthe, the Municipality of Assen, the European Fund for Regional Development and the Ministry of Economic Affairs, Peaks in the Delta.

REFERENCES

- [1] Z. Michalewicz, D. Dasgupta, R. L. Riche, and M. Schoenauer, "Evolutionary algorithms for constrained engineering problems," *Computers & Industrial Engineering Journal*, vol. 30, no. 4, pp. 851–870, September 1996.
- [2] C. A. Coello Coello and E. Mezura-Montes, "Constraint-handling in genetic algorithms through the use of dominance-based tournament selection," *Advanced Engineering Informatics*, vol. 16, no. 3, pp. 193–203, 2002.
- [3] M. Pant, R. Thangaraj, and A. Abraha, "Low Discrepancy Initialized Particle Swarm Optimization for Solving Constrained Optimization Problems," *Fundam. Inf.*, vol. 95, pp. 511–531, December 2009.
- [4] L. d. S. Coelho, "Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems," *Expert Systems with Applications*, vol. 37, no. 2, pp. 1676–1683, 2010.
- [5] V. V. de Melo and G. L. C. Carosio, "Evaluating differential evolution with penalty function to solve constrained engineering problems," *Expert Systems with Applications*, vol. 39, no. 9, pp. 7860–7863, 2012.
- [6] A. W. Mohamed and H. Z. Sabry, "Constrained optimization based on modified differential evolution algorithm," *Inf. Sci.*, vol. 194, pp. 171–208, Jul. 2012.
- [7] 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, 2003.
- [8] N. Hansen, A. S. P. Niederberger, L. Guzzella, and P. Koumoutsakos, "A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 1, pp. 180–197, Feb. 2009.
- [9] G. Collange, N. Delattre, N. Hansen, I. Quinquis, and M. Schoenauer, "Multidisciplinary Optimisation in the Design of Future Space Launchers," in *Multidisciplinary Design Optimization in Computational Mechanics*, P. Breitkopf and R. F. Coelho, Eds. Wiley, 2010, ch. 12, pp. 487–496.
- [10] O. Kramer, A. Barthelmes, and G. Rudolph, "Surrogate constraint functions for CMA evolution strategies," in *32nd annual German conference on Advances in artificial intelligence*, ser. KI'09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 169–176.
- [11] D. V. Arnold and N. Hansen, "A (1+1)-CMA-ES for Constrained Optimisation," in *2012 Genetic and Evolutionary Computation Conference (GECCO 2012)*. Philadelphia, USA: ACM Press, July 2012, pp. 297–304, ISBN: 978-1-4503-1177-9.
- [12] V. V. de Melo and G. Iacca, "A modified Covariance Matrix Adaptation Evolution Strategy with adaptive penalty function and restart for constrained optimization," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7077 – 7094, 2014.
- [13] A. Maesani and D. Floreano, "Viability Principles for Constrained Optimization Using a (1+1)-CMA-ES," in *Parallel Problem Solving from Nature-PPSN XII*, 2014, to appear.
- [14] F. Neri, C. Cotta, and P. Moscato, Eds., *Handbook of Memetic Algorithms*, ser. Studies in Computational Intelligence. Springer, 2011, vol. 379.
- [15] J. A. Nelder and R. Mead, "A Simplex Method for Function Minimization," *The Computer Journal*, vol. 7, no. 4, pp. 308–313, January 1965.
- [16] M. J. D. Powell, "The BOBYQA algorithm for bound constrained optimization without derivatives," Aug. 2009.
- [17] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, Sep. 1995.
- [18] A. Auger and N. Hansen, "A restart CMA evolution strategy with increasing population size," in *2005 IEEE Congress on Evolutionary Computation*, 2005, vol. 2, Sept 2005, pp. 1769–1776 Vol. 2.
- [19] G. Iacca, R. Mallipeddi, E. Mininno, F. Neri, and P. N. Suganthan, "Super-fit and population size reduction in compact Differential Evolution," in *Memetic Computing*, 2011, pp. 21–29.
- [20] F. Caraffini, G. Iacca, F. Neri, L. Picinali, and E. Mininno, "A CMA-

- ES super-fit scheme for the re-sampled inheritance search,” in *IEEE Congress on Evolutionary Computation*, 2013, pp. 1123–1130.
- [21] F. Caraffini, F. Neri, J. Cheng, G. Zhang, L. Picinali, G. Iacca, and E. Mininno, “Super-fit Multicriteria Adaptive Differential Evolution,” in *IEEE Congress on Evolutionary Computation*, 2013, pp. 1678–1685.
 - [22] J. Liang, T. Runarsson, E. Mezura-Montes, M. Clerc, P. Suganthan, C. Coello, and K. Deb, “Problem Definitions and Evaluation Criteria for the CEC 2006 Special Session on Constrained Real-Parameter Optimization.” Singapore, Technical Report, 2006.
 - [23] T. Ray and K. Liew, “Society and Civilization: An Optimization Algorithm Based on the Simulation of Social Behavior,” *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 4, pp. 386–396, August 2003.
 - [24] M. S. Bartlett, “Properties of sufficiency and statistical tests,” *Proceedings of the Royal Society of London. Series A - Mathematical and Physical Sciences*, vol. 160, no. 901, pp. 268–282, 1937.
 - [25] M. Friedman, “A comparison of alternative tests of significance for the problem of m rankings,” *The Annals of Mathematical Statistics*, vol. 11, no. 1, pp. 86–92, 03 1940.
 - [26] F. Wilcoxon, “Individual comparisons by ranking methods,” *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.
 - [27] L. Wang and L.-p. Li, “An effective differential evolution with level comparison for constrained engineering design,” *Structural and Multidisciplinary Optimization*, vol. 41, no. 6, pp. 947–963, June 2010.
 - [28] A. Sadollah, A. Bahreininejad, H. Eskandar, and M. Hamdi, “Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems,” *Applied Soft Computing*, vol. 13, no. 5, pp. 2592–2612, 2013.
 - [29] V. V. de Melo and G. L. C. Carosio, “Investigating Multi-View Differential Evolution for solving constrained engineering design problems,” *Expert Systems with Applications*, vol. 40, no. 9, pp. 3370–3377, Jul. 2013.
 - [30] M. Zhang, W. Luo, and X. Wang, “Differential evolution with dynamic stochastic selection for constrained optimization,” *Information Sciences*, vol. 178, no. 15, pp. 3043–3074, August 1 2008.
 - [31] H. Liu, Z. Cai, and Y. Wang, “Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization,” *Applied Soft Computing*, vol. 10, no. 2, pp. 629–640, March 2010.
 - [32] Y. Wang, Z. Cai, Y. Zhou, and Z. Fan, “Constrained optimization based on hybrid evolutionary algorithm and adaptive constraint-handling technique,” *Structural and Multidisciplinary Optimization*, vol. 37, no. 4, pp. 395–413, January 2009.
 - [33] J. Sun and J. M. Garibaldi, “A Novel Memetic Algorithm for Constrained Optimization,” in *2010 IEEE Congress on Evolutionary Computation (CEC 2010)*, Barcelona, Spain, July 18–23 2010, pp. 549–556.
 - [34] M. Pescador Rojas and C. A. Coello Coello, “A memetic algorithm with simplex crossover for solving constrained optimization problems,” in *World Automation Congress (WAC)*, 2012, 2012, pp. 1–6.
 - [35] F. Kang, J. Li, and H. Li, “Artificial bee colony algorithm and pattern search hybridized for global optimization,” *Applied Soft Computing*, vol. 13, no. 4, pp. 1781 – 1791, 2013.
 - [36] F. Kang, J. Li, and Z. Ma, “Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions,” *Information Sciences*, vol. 181, no. 16, pp. 3508 – 3531, 2011.
 - [37] Y. Nagata, O. Br aysy, and W. Dullaert, “A penalty-based edge assembly memetic algorithm for the vehicle routing problem with time windows,” *Computers & Operations Research*, vol. 37, no. 4, pp. 724 – 737, 2010.
 - [38] G. Iacca, F. Neri, E. Mininno, Y. S. Ong, and M. H. Lim, “Ockham’s Razor in Memetic Computing: Three Stage Optimal Memetic Exploration,” *Information Sciences*, vol. 188, pp. 17–43, 2012.
 - [39] A. Amirjanov, “The development of a changing range genetic algorithm,” *Computer Methods in Applied Mechanics and Engineering*, vol. 195, no. 19–22, pp. 2495–2508, 2006.
 - [40] I. C. Trelea, “The particle swarm optimization algorithm: convergence analysis and parameter selection,” *Information Processing Letters*, vol. 85, no. 6, pp. 317–325, 2003.
 - [41] D. Karaboga and B. Basturk, “Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems,” in *Foundations of Fuzzy Logic and Soft Computing, 12th International Fuzzy Systems Association, World Congress, IFSA 2007*, P. Melin, O. Castillo, L. T. Aguilar, J. Kacprzyk, and W. Pedrycz, Eds. Cancun, Mexico: Springer, Lecture Notes in Artificial Intelligence Vol. 4529, June 2007, pp. 789–798.
 - [42] S. Koziel and Z. Michalewicz, “Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization,” *Evolutionary Computation*, vol. 7, no. 1, pp. 19–44, 1999.
 - [43] A. Aguirre, A. Muñoz Zavala, E. Villa Diharce, and S. Botello Rionda, “COPSO: Constrained Optimization via PSO algorithm,” Center of Research in Mathematics (CIMAT), Guanajuato, México, Technical Report I-07-04, 2007.
 - [44] A. S. S. M. B. Ullah, R. Sarker, D. Cornforth, and C. Lokan, “AMA: a new approach for solving constrained real-valued optimization problems,” *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 13, no. 8–9, pp. 741–762, July 2009.
 - [45] E. Z. Elfekey, R. A. Sarker, and D. L. Essam, “A simple ranking and selection for constrained evolutionary optimization,” in *Simulated Evolution and Learning, Proceedings*. Springer, Lecture Notes in Computer Science Vol. 4247, 2006, pp. 537–544.
 - [46] B. Tessema and G. G. Yen, “A Self Adaptive Penalty Function Based Algorithm for Constrained Optimization,” in *2006 IEEE Congress on Evolutionary Computation (CEC 2006)*, Vancouver, BC, Canada, July 2006, pp. 950–957.
 - [47] P. Chootinan and A. Chen, “Constraint Handling In Genetic Algorithms Using A Gradient-Based Repair Method,” *Computers and Operations Research*, vol. 33, no. 8, pp. 2263–2281, August 2006.
 - [48] R. L. Becerra and C. A. C. Coello, “Cultured differential evolution for constrained optimization,” *Computer Methods in Applied Mechanics and Engineering*, vol. 195, no. 33–36, pp. 4303–4322, 2006.
 - [49] K.-Z. Tang, T.-K. Sun, and J.-Y. Yang, “An improved genetic algorithm based on a novel selection strategy for nonlinear programming problems,” *Computers & Chemical Engineering*, vol. 35, no. 4, pp. 615–621, April 2011.
 - [50] A. E. Muñoz Zavala, A. H. Aguirre, and E. R. Villa Diharce, “Constrained optimization via particle evolutionary swarm optimization algorithm (PESO),” in *Proceedings of the 2005 conference on Genetic and evolutionary computation*, ser. GECCO ’05. New York, NY, USA: ACM, 2005, pp. 209–216.
 - [51] R. V. Rao, V. J. Savsani, and D. P. Vakharia, “Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems,” *Computer-Aided Design*, vol. 43, no. 3, pp. 303–315, March 2011.
 - [52] G. Zhang, J. Cheng, M. Gheorghe, and Q. Meng, “A hybrid approach based on differential evolution and tissue membrane systems for solving constrained manufacturing parameter optimization problems,” *Applied Soft Computing*, vol. 13, no. 3, pp. 1528–1542, 2013.
 - [53] E. Mezura-Montes and C. A. Coello Coello, “A simple multimembered evolution strategy to solve constrained optimization problems,” *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 1, pp. 1–17, 2005.
 - [54] C. A. C. Coello and E. Mezura-Montes, “Constraint-handling in genetic algorithms through the use of dominance-based tournament selection,” *Advanced Engineering Informatics*, vol. 16, no. 3, pp. 193–203, July 2002.
 - [55] C. A. Coello Coello and R. L. Becerra, “Efficient evolutionary optimization through the use of a cultural algorithm,” *Engineering Optimization*, vol. 36, no. 2, pp. 219–236, 2004.
 - [56] Q. He and L. Wang, “An effective co-evolutionary particle swarm optimization for constrained engineering design problems,” *Engineering Applications of Artificial Intelligence*, vol. 20, no. 1, pp. 89–99, February 2007.
 - [57] —, “A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization,” *Applied Mathematics and Computation*, vol. 186, no. 2, pp. 1407–1422, March 15th 2007.
 - [58] E. Zahara and Y.-T. Kao, “Hybrid Nelder-Mead simplex search and particle swarm optimization for constrained engineering design problems,” *Expert Systems with Applications*, vol. 36, no. 2, pp. 3880–3886, March 2009.
 - [59] E. Mezura-Montes and C. A. C. Coello, “Useful Infeasible Solutions in Engineering Optimization with Evolutionary Algorithms.” in

MICAI, ser. Lecture Notes in Computer Science, A. F. Gelbukh, A. de Alborno, and H. Terashima-Marín, Eds., vol. 3789. Springer, 2005, pp. 652–662.

- [60] E. Mezura-Montes, C. A. Coello Coello, and J. Velázquez-Reyes, “Increasing Successful Offspring and Diversity in Differential Evolution for Engineering Design,” in *7th International Conference on Adaptive Computing in Design and Manufacture*, I. Parmee, Ed., Bristol, UK, April 2006, pp. 131–139.
- [61] E. Mezura-Montes, J. Velazquez-Reyes, and C. Coello Coello, “Modified Differential Evolution for Constrained Optimization,” in *2006 IEEE Congress on Evolutionary Computation (CEC 2006)*, 2006, pp. 25–32.
- [62] L. dos Santos Coelho, “Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems,” *Expert Systems with Applications*, vol. 37, no. 2, pp. 1676–1683, March 2010.

TABLE I. COMPARISON OF RESULTS (CMA-ES + SIMPLEX) FOR PROBLEM 1 (G06). “NA” MEANS NOT AVAILABLE, AND SD IS THE STANDARD-DEVIATION.

Method	Worst	Mean	Best	SD	NFEs
CMA-ES+Simplex (this work)	-6961.81308	-6961.813867	-6961.813875	7.87E-05	30233
COMDE [6]	-6961.813875	-6961.813875	-6961.813875	NA	12000
CRGA [39]	-6077.123	-6740.288	-6956.251	270	3700
PSO [40]	-6961.81381	-6961.81387	-6961.81388	6.5E-06	140100
DELIC [27]	-6961.814	-6961.814	-6961.814	7.3E-10	20000
DEDS [30]	-6961.814	-6961.814	-6961.814	0	225000
HEAA [32]	-6961.814	-6961.814	-6961.814	4.6E-12	200000
ABC [41]	-6961.805	-6961.813	-6961.814	2E-03	240000
HM [42]	-5473.9	-6342.6	-6952.1	NA	1400000
MBA [28]	-6961.813875	-6961.813875	-6961.813875	NA	2835

TABLE II. COMPARISON OF RESULTS (CMA-ES + SIMPLEX) FOR PROBLEM 2 (G07). “NA” MEANS NOT AVAILABLE.

Method	Worst	Mean	Best	SD	NFEs
CMA-ES+Simplex (this work)	24.306209	24.306209	24.306209	2.62E-10	75000
COMDE [6]	24.306211	24.306209	24.306209	4.7E-07	200000
COPSO [43]	24.306219	24.306212	24.306209	NA	225000
PSO-DE [31]	24.3062172	24.3062100	24.3062091	1.3E-06	140100
AMA [44]	24.491	24.392	24.325	5.18E-02	350000
TC [45]	NA	25.057	24.505	2.38E-01	350000
SAPF [46]	33.095	27.328	24.838	2.17E+00	500000
CC [47]	24.8352	24.4719	24.3294	1.29E-01	350000
HM [42]	25.069	24.826	24.62	NA	1400000
CRGA [39]	27.381	25.746	24.882	7.0E-01	45800
CULDE [48]	24.306212	24.306210	24.306209	1.0E-6	100000

TABLE III. COMPARISON OF RESULTS (CMA-ES) FOR PROBLEM 3 (G09). “NA” MEANS NOT AVAILABLE.

Method	Worst	Mean	Best	SD	NFEs
CMA-ES (this work)	680.630057	680.630057	680.630057	1.42E-13	30000
HM [42]	683.18	681.16	680.91	4.11E-02	1400000
IGA [49]	680.6304037	680.6302536	680.6301361	NA	500000
CRGA [39]	682.965	681.347	680.726	5.70E-01	50000
PSO [40]	684.5289146	680.9710606	680.6345517	5.1E-01	140100
DELIC [27]	680.63	680.63	680.63	3.2E-12	80000
DEDS [30]	680.63	680.63	680.63	2.9E-13	225000
HEAA [32]	680.63	680.63	680.63	5.8E-13	200000
PESO [50]	680.630058	680.630057	680.630057	NA	350000
COMDE [6]	680.630057	680.630057	680.630057	4.07E-13	70000
CULDE [48]	680.630057	680.630057	680.630057	NA	100100
ABC [41]	680.638	680.64	680.634	4E-03	240000
TLBO [51]	680.638	680.633	680.63	NA	100000
MBA [28]	680.7882	680.662	680.6322	3.30E-02	71750

TABLE IV. COMPARISON OF RESULTS (CMA-ES + SIMPLEX) FOR PROBLEM 4 (G10). “NA” MEANS NOT AVAILABLE.

Method	Worst	Mean	Best	SD	NFEs
CMA-ES+Simplex (this work)	7049.248020	7049.248020	7049.248020	2.68E-11	120030
COMDE [6]	7049.248615	7049.248077	7049.248020	1.5E-04	200000
PSO-DE [31]	7049.249223	7049.248038	7049.248021	3.0E-05	140100
CULDE [48]	7049.24848	7049.24826	7049.24805	1.67E-04	100100
ABC [41]	7604.132	7224.407	7053.904	NA	240000
PESO [50]	7251.396	7099.101	7049.459	NA	350000
DETPS [52]	7063.406	7050.834	7049.257	NA	100000
TLBO [51]	7224.497	7083.673	7049.248	NA	100000
M-ES [53]	7638.366	7253.047	7051.903	NA	240000

TABLE V. COMPARISON OF RESULTS (CMA-ES) FOR PROBLEM 5 (WELDED BEAM). “NA” MEANS NOT AVAILABLE.

Method	Worst	Mean	Best	SD	NFEs
CMA-ES (this work)	1.7248523	1.7248523	1.7248523	2.02294E-14	21000
MVDE [29]	1.7249215	1.7248621	1.7248527	7.88359E-06	15000
GA [54]	1.993408	1.792654	1.728226	7.47E-02	80000
CAEP [55]	3.179709	1.971809	1.724852	4.43E-01	50020
CPSO [56]	1.782143	1.748831	1.728024	1.29E-02	240000
HPSO [57]	1.814295	1.749040	1.724852	4.01E-02	81000
NM-PSO [58]	1.733393	1.726373	1.724717	3.50E-03	80000
DELIC [27]	1.724852	1.724852	1.724852	4.1E-13	20000
COMDE [6]	1.724852	1.724852	1.724852	1.60E-12	20000
($\mu + \lambda$)-ES [59]	NA	1.777692	1.724852	8.8E-02	30000
ABC [41]	NA	1.741913	1.724852	3.1E-02	30000
TLBO [51]	NA	1.7284676	1.724852	NA	10000
MBA [28]	1.724853	1.724853	1.724853	6.94E-19	47340

TABLE VI. COMPARISON OF RESULTS (CMA-ES) FOR PROBLEM 6 (SPEED REDUCER). “NA” MEANS NOT AVAILABLE.

Method	Worst	Mean	Best	SD	NFEs
CMA-ES (this work)	2994.471066	2994.471066	2994.471066	1.729541E-08	30000
MVDE [29]	2994.471069	2994.471066	2994.471066	2.819316E-07	30000
SC [23]	3009.964736	3001.758264	2994.744241	4.0	54456
PSO-DE [31]	2996.348204	2996.348174	2996.348167	6.4E-06	54350
DELIC [27]	2994.471066	2994.471066	2994.471066	1.9E-12	30000
DEDS [30]	2994.471066	2994.471066	2994.471066	3.6E-12	30000
HEAA [32]	2994.752311	2994.613368	2994.499107	7.0E-02	40000
MDE [60], [61]	NA	2996.367220	2996.356689	8.2E-03	24000
($\mu + \lambda$)-ES [59]	NA	2996.348	2996.348	NA	30000
ABC [41]	NA	2997.058	2997.058	NA	30000
TLBO [51]	NA	2996.34817	2996.34817	NA	10000
MBA [28]	2999.652444	2996.769019	2994.482453	1.56	6300

TABLE VII. COMPARISON OF RESULTS (CMA-ES) FOR PROBLEM 7 (THREE-BAR TRUSS).

Method	Worst	Mean	Best	SD	NFEs
CMA-ES (this work)	263.8958435	263.89584338	263.89584337	1.935435E-08	3000
MVDE [29]	263.8958548	263.89584338	263.89584337	2.576062E-07	7000
SC [23]	263.969756	263.903356	263.895846	1.3E-02	17610
PSO-DE [31]	263.895843	263.895843	263.895843	4.5E-10	17600
DELIC [27]	263.8958434	263.8958434	263.8958434	4.3E-14	10000
DEDS [30]	263.895849	263.895843	263.895843	9.7E-07	15000
HEAA [32]	263.896099	263.895865	263.895843	4.9E-05	15000
MBA [28]	263.915983	263.897996	263.895852	3.93E-03	13280

TABLE VIII. COMPARISON OF RESULTS (CMA-ES + SIMPLEX) FOR PROBLEM 8 (PRESSURE VESSEL). “NA” MEANS NOT AVAILABLE.

Method	Worst	Mean	Best	SD	NFEs
CMA-ES+Simplex (this work)	6370.7797127519	6081.37210303	6059.714335	53.310642847	45300
MVDE [29]	6090.5335277869	6059.99723569	6059.714387	2.9102896082	15000
GA [54]	6469.322	6177.2533	6059.9463	130.9297	80000
CPSO [56]	6363.8041	6147.1332	6061.0777	86.45	240000
HPSO [57]	6288.677	6099.9323	6059.7143	86.2	81000
G-QPSO [62]	7544.4925	6440.3786	6059.7208	448.4711	8000
QPSO [62]	8017.2816	6440.3786	6059.7209	479.2671	8000
PSO [40]	14076.324	8756.6803	6693.7212	1492.567	8000
DELIC [27]	6059.7143	6059.7143	6059.7143	2.1E-11	30000
PSO-DE [31]	NA	6059.714	6059.714	NA	42100
ABC [41]	NA	6245.308	6059.714	205	30000
TLBO [51]	NA	6059.71434	6059.714335	NA	10000

TABLE IX. COMPARISON OF RESULTS (CMA-ES + SIMPLEX) FOR PROBLEM 9 (TENSION/COMPRESSION SPRING). “NA” MEANS NOT AVAILABLE.

Method	Worst	Mean	Best	SD	NFEs
CMA-ES+Simplex (this work)	0.012729907	0.012667379	0.01266523278	7.632233E-06	30300
MVDE [29]	0.012719055	0.012667324	0.01266527172	2.451838E-06	10000
GA [54]	0.012973	0.012742	0.012681	5.90E-05	80000
CAEP [55]	0.015116	0.0135681	0.012721	8.42E-04	50020
CPSO [56]	0.012924	0.01273	0.0126747	5.20E-04	240000
HPSO [57]	0.012719	0.0127072	0.0126652	1.58E-05	81000
NM-PSO [58]	0.012633	0.0126314	0.0126302	8.47E-07	80000
G-QPSO [62]	0.017759	0.013524	0.012665	0.001268	2000
QPSO [62]	0.018127	0.013854	0.012669	0.001341	2000
PSO [40]	0.071802	0.019555	0.012857	0.011662	2000
DELIC [27]	0.012665575	0.012665267	0.012665233	1.3E-07	20000
DEDS [30]	0.012738262	0.012669366	0.012665233	1.3E-05	24000
HEAA [32]	0.01266524	0.012665234	0.012665233	1.4E-09	24000
PSO-DE [31]	0.012665304	0.012665244	0.012665233	1.2E-08	24950
ABC [41]	NA	0.012709	0.012665	0.012813	30000
TLBO [51]	NA	0.01266576	0.012665	NA	10000
MBA [28]	0.0129	0.012713	0.012665	6.30E-05	7650

TABLE X. COMPARISON OF CMA-ES VERSUS CMA-ES + LO FOR PROBLEM 1 (G06)

	CMA-ES	Simplex	BOBYQA	L-BFGS-B
Min.	-6961.8138755801664956	-6961.8138755801664956	-6961.8138755801664956	-6961.8138755801664956
1st Qu.	-6961.8138755801664956	-6961.8138755801664956	-6961.8138755801664956	-6961.8138755801664956
Median	-6961.8138755801592197	-6961.8138755801592197	-6961.8138755801592197	-6961.8138755801592197
Mean	-6939.1267706483859001	-6961.8138672270788447	-6939.796301801890877	-6955.6607990839702325
3rd Qu.	-6961.8138755800828221	-6961.8138755800828221	-6961.8138755800828221	-6961.8138755800828221
Max.	-5668.5737580408704162	-6961.8130894739879295	-5703.9712129372683194	-6503.3839232603259006
Std.Dev	161.1994805535746309	7.870910703561306062E-05	156.49783928081558315	48.302769622558891172
Mean.Evals	30017.430000000000291	30233.169999999998254	30023.18999999999869	30050.759999999998399
Std.Dev.Evals	39.125866563482453842	87.246203934670418789	39.458889231628646144	52.995162729510255417

TABLE XI. COMPARISON OF CMA-ES VERSUS CMA-ES + LO FOR PROBLEM 2 (G07)

	CMA-ES	Simplex	BOBYQA	L-BFGS-B
Min.	24.306209068179839505	24.306209068179839505	24.306209068179839505	24.306209068179839505
1st Qu.	24.306209068179882138	24.306209068179875032	24.306209068179882138	24.306209068179882138
Median	24.306209068179899901	24.306209068179889243	24.306209068179899901	24.306209068179899901
Mean	24.306209068211675373	24.30620906821167182	24.306209068211675373	24.306209068211675373
3rd Qu.	24.30620906817991056	24.306209068179907007	24.30620906817991056	24.30620906817991056
Max.	24.306209070730250943	24.306209070730236732	24.306209070730250943	24.306209070730250943
Std.Dev	2.6202654222435227442E-10	2.6202589036577830195E-10	2.6202654222435227442E-10	2.6202654222435227442E-10
Mean.Evals	74987.050000000000291	75872.9299999999993015	75008.050000000000291	75097.529999999998836
Std.Dev.Evals	33.553026026806520576	42.514942114864780365	33.553026026806520576	34.163142464215773941

TABLE XII. COMPARISON OF CMA-ES VERSUS CMA-ES + LO FOR PROBLEM 4 (G10)

	CMA-ES	Simplex	BOBYQA	L-BFGS-B
Min.	7049.2480205286810815	7049.2480205286810815	7049.2480205286810815	7049.2480205286810815
1st Qu.	7049.2480205287129138	7049.2480205287120043	7049.2480205287129138	7049.2480205287129138
Median	7049.2480205287247372	7049.2480205287229182	7049.2480205287247372	7049.2480205287247372
Mean	7049.2480205287301942	7049.2480205287292847	7049.2480205287301942	7049.2480205287301942
3rd Qu.	7049.2480205287420176	7049.2480205287411081	7049.2480205287420176	7049.2480205287420176
Max.	7049.2480205288429715	7049.2480205288429715	7049.2480205288429715	7049.2480205288429715
Std.Dev	2.69483662893174617E-11	2.6859389876579132064E-11	2.69483662893174617E-11	2.69483662893174617E-11
Mean.Evals	120032.690000000000233	120577.910000000000349	120053.690000000000233	120120.350000000000582
Std.Dev.Evals	58.787873242736381485	60.471396527447573988	58.787873242736381485	59.059698439781925572

TABLE XIII. COMPARISON OF CMA-ES VERSUS CMA-ES + LO FOR PROBLEM 8 (PRESSURE VESSEL)

	CMA-ES	Simplex	BOBYQA	L-BFGS-B
Min.	6059.7143350488204305	6059.7143350488204305	6059.7143350488204305	6059.7143350488204305
1st Qu.	6059.7143350632013608	6059.7143350632013608	6059.7143350632013608	6059.7143350632013608
Median	6059.8273283257212825	6059.7143957105799927	6059.8273283257212825	6059.7963942990772921
Mean	6081.7816912614825924	6081.3721030397409777	6081.6736933771126132	6081.6668786048312541
3rd Qu.	6090.5262016912429317	6090.5262016912201943	6090.5262016912429317	6090.5262016912429317
Max.	6370.7797127519461355	6370.7797127519461355	6370.7797127519461355	6370.7797127519461355
Std.Dev	53.198980006654885244	53.310642847031445513	53.217958703371785134	53.2203792755922791
Mean.Evals	45002.040000000000873	45303.000	45024.459999999999127	45052.580000000001746
Std.Dev.Evals	46.45404376174567318	56.64990839544333312	47.436383768690724594	47.432499620007590124

TABLE XIV. COMPARISON OF CMA-ES VERSUS CMA-ES + LO FOR PROBLEM 9 (TENSION/COMPRESSION SPRING)

	CMA-ES	Simplex	BOBYQA	L-BFGS-B
Min.	0.012665232788560226024	0.012665232788560226024	0.012665232788560226024	0.012665232788560226024
1st Qu.	0.01266523927311774024	0.012665236796817995121	0.01266523927311774024	0.01266523927311774024
Median	0.012665366304410587833	0.012665305783512634946	0.012665366304410587833	0.012665366304410587833
Mean	0.012667990695468618959	0.012667379911721326352	0.012667990695468618959	0.012667990695468618959
3rd Qu.	0.012666781091689416697	0.012666560829270551852	0.012666781091689416697	0.012666781091689416697
Max.	0.012731588254956927378	0.012729907807229856251	0.012731588254956927378	0.012731588254956927378
Std.Dev	8.4599963983957441243E-06	7.6322339831501574356E-06	8.4599963983957441243E-06	8.4599963983957441243E-06
Mean.Evals	30037.9099999999999854	30274.069999999999709	30057.3699999999998981	30069.380000000001019
Std.Dev.Evals	54.935057158889947004	61.674979783645909492	54.871135537288772355	54.914953530091608513