Modified Constrained Differential Evolution for Solving Nonlinear Global Optimization Problems

Md. Abul Kalam Azad and Edite M.G.P. Fernandes

Algoritmi R&D Centre
University of Minho, 4710-057 Braga, Portugal
{akazad,emgpf}@dps.uminho.pt

Abstract. Nonlinear optimization problems introduce the possibility of multiple local optima. The task of global optimization is to find a point where the objective function obtains its most extreme value while satisfying the constraints. Some methods try to make the solution feasible by using penalty function methods, but the performance is not always satisfactory since the selection of the penalty parameters for the problem at hand is not a straightforward issue. Differential evolution has shown to be very efficient when solving global optimization problems with simple bounds. In this paper, we propose a modified constrained differential evolution based on different constraints handling techniques, namely, feasibility and dominance rules, stochastic ranking and global competitive ranking and compare their performances on a benchmark set of problems. A comparison with other solution methods available in literature is also provided. The convergence behavior of the algorithm to handle discrete and integer variables is analyzed using four well-known mixed-integer engineering design problems. It is shown that our method is rather effective when solving nonlinear optimization problems.

Keywords: Nonlinear programming, Global optimization, Constraints handling, Differential evolution

1 Introduction

Problems involving global optimization over continuous spaces are ubiquitous throughout the scientific community. Many real world problems are formulated as mathematical programming problems involving continuous variables with linear/nonlinear objective function and constraints. The constraints can be of inequality and/or equality type. Generally, the constrained nonlinear optimization problems are formulated as follows:

$$\begin{array}{lll} \text{minimize} & f(\boldsymbol{x}) & \\ \text{subject to} & g_k(\boldsymbol{x}) \leq 0 & k = 1, 2, \ldots, m_1 \\ & h_l(\boldsymbol{x}) = 0 & l = 1, 2, \ldots, m_2 \\ & lb_j \leq x_j \leq ub_j & j = 1, 2, \ldots, n, \end{array} \tag{1}$$

where $f, g_k, h_l : \mathbb{R}^n \longrightarrow \mathbb{R}$ with feasible set $\mathcal{F} = \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{g}(\boldsymbol{x}) \leq 0, \boldsymbol{h}(\boldsymbol{x}) = 0$ and $\boldsymbol{lb} \leq \boldsymbol{x} \leq \boldsymbol{ub}\}$. f, g_k, h_l may be differentiable and the information about derivatives may or may not be provided.

Problem (1) involving global optimization (here a minimization problem) of a multivariate function with constraints is widespread in the mathematical modeling of real world systems. Many problems can be described only by nonlinear relationships, which introduce the possibility of multiple local minima. The task of global optimization is to find a point where the objective function obtains its most extreme value, the global minimum, while satisfying the constraints.

Several deterministic and stochastic solution methods with different constraints handling techniques have been proposed to solve (1). Unlike the stochastic methods, the outcome of a deterministic algorithm does not depend on pseudo random variables. In general, its performance depends heavily on the structure of the problem since the design relies on the mathematical attributes of the optimization problem. Compared with deterministic methods, the implementation of stochastic algorithms is often easier. To handle the constraints of the problem, some methods try to make the solution feasible by repairing the infeasible one or penalizing an infeasible solution with a penalty function. However, to find the appropriate penalty parameter is not an easy task. Deb [6] proposed an efficient constraints handling technique for genetic algorithm based on the feasibility and dominance rules. The author used a penalty function that does not require any penalty parameter. Barbosa and Lemonge [2] proposed a parameterless adaptive penalty scheme for genetic algorithm. In the very recent paper the authors proposed this adaptive penalty scheme for differential evolution [23]. Hedar and Fukushima [12] proposed a simulated annealing method that uses the filter method [10] rather than the penalty function method to handle the constraints. Runarsson and Yao proposed a stochastic ranking [21] and a global competitive ranking [22] techniques for constrained nonlinear optimization problems based on evolution strategy. The authors presented a new view on the usual penalty function methods in terms of the dominance of penalty and objective functions. Dong et al. [8] proposed a swarm optimization based on the constraint fitness priority-based ranking technique. Zahara and Hu [28] proposed a hybrid of Nelder-Mead simplex method and a particle swarm optimization based on this technique [8]. Rocha and Fernandes proposed a electromagnetism-like algorithm based on the feasibility and dominance rules [18] and the self-adaptive penalties [19]. Rocha et al. [20] used an augmented Lagrangian method coupled with an artificial fish swarm algorithm for global optimization. Coello Coello [4] proposed constraints handling using an evolutionary multiobjective optimization technique. Coello Coello and Cortés [5] proposed hybridizing of a genetic algorithm with an artificial immune system that uses genotypic-based distances to move from infeasible solution to feasible one. Another constraints handling technique is the multilevel Pareto ranking based on the constraints matrix [16, 17]. Ray and Tai [16] proposed an evolutionary algorithm with a multilevel pairing strategy and Ray and Liew [17] proposed a society and civilization algorithm based on the simulation of social behavior.

Differential evolution (DE) proposed by Storn and Price [24] is a population-based heuristic approach that is very efficient to solve global optimization problems with simple bounds. DE performance depends on the amplification factor

of differential variation and crossover control parameter. Hence adaptive control parameters have been implemented in DE in order to obtain a competitive algorithm. Further, to improve solution accuracy, techniques that are able to exploit locally certain regions, detected in the search space as promising, are also required. When the solutions ought to be restricted to a set of inequality and equality constraints, an efficient constraints handling technique is also required in the solution method. In this paper, we propose a modified constrained differential evolution algorithm (herein denoted as mCDE) that uses the self-adaptive control parameters [3], a mixture of modified mutations, and also includes the inversion operation, a modified selection and the elitism to be able to progress efficiently towards global solutions of problems (1).

The organization of this paper is as follows. Firstly, we describe the constraints handling techniques in Section 2 and then the modified constrained differential evolution is outlined in Section 3. Section 4 describes the experimental results and finally we draw the conclusions of this study in Section 5.

2 Constraints Handling Techniques

Stochastic methods have been primary developed for the global optimization of unconstrained problems. Extensions to the constrained problems then appear with the modification of some solution procedures. To deal with a constrained problem, a widely used approach is based on penalty functions where a penalty term is added to the objective function in order to penalize the constraint violation. This enable us to transform a constrained problem into a sequence of unconstrained subproblems. The penalty function method can be applied to any type of constraints, but the performance of penalty-type method is not always satisfactory because of choosing an appropriate penalty parameter. For this reason alternative constraints handling techniques have been proposed in the last decades. Three different techniques, usually used in population-based methods have been implemented and extensively tested in our proposed mCDE algorithm: a) the *feasibility and dominance rules*, b) the *stochastic ranking*, and c) the *global competitive ranking*. They are briefly described below. In a population-based solution method with N candidate solutions $\boldsymbol{x}_i,\ i = 1, 2, \ldots, N$ at each generation, a common measure of infeasibility of an individual point $\boldsymbol{x}_i$ is the average measure of constraint violation given by

$$\zeta(\boldsymbol{x}_i) = \frac{1}{m}\left(\sum_{k=1}^{m_1} \max\{0, g_k(\boldsymbol{x}_i)\} + \sum_{l=1}^{m_2} |h_l(\boldsymbol{x}_i)|\right),$$

where $m = m_1 + m_2$ and $\zeta(\boldsymbol{x}_i)$ is a non-negative real-valued function, with $\zeta(\boldsymbol{x}_i) = 0$ if the point $\boldsymbol{x}_i$ is feasible.

2.1 Feasibility and Dominance Rules

Deb [6] proposed a constraints handling technique for population-based solution methods based on a set of rules that uses feasibility and dominance (FD)

principles, as follows. First, the constraint violation ζ is calculated for all the individuals in a population. Then the objective function f is evaluated only for feasible individuals. Two individual points are compared at a time, and the following criteria are always enforced:

a) any feasible point is preferred to any infeasible point;
b) between two feasible points, one having better objective function is preferred;
c) between two infeasible points, one having smaller constraint violation is preferred.

In this case, the fitness of each individual point $\boldsymbol{x}_i$ is calculated as follows

$$\Phi_{\mathrm{FD}}(\boldsymbol{x}_i) = \begin{cases} f(\boldsymbol{x}_i) & \text{if } \boldsymbol{x}_i \text{ is feasible} \\ f_{\max,f} + \zeta(\boldsymbol{x}_i) & \text{otherwise,} \end{cases} \tag{2}$$

where $f_{\max,f}$ is the objective function of the worst feasible solution in the population. When all individuals are infeasible then its value is set to zero. This fitness function is used to choose the best individual point in a population.

2.2 Stochastic Ranking

Runarsson and Yao [21] first proposed the stochastic ranking (SR) for the constrained nonlinear optimization problems. This is a bubble-sort-like algorithm to give ranks to individuals in a population stochastically. In this ranking method, two adjacent individual points are compared and given ranks and swapped. The algorithm is halt if there is no swap. Individuals are ranked primarily based on their constraint violations. The objective function values are then considered if: i) individuals are feasible, or ii) a uniform random number between 0 and 1 is less than or equal to P_f. The probability P_f is used only for comparisons of the objective function in the infeasible region of the search space. Such ranking ensures that good feasible solutions as well as promising infeasible ones are ranked in the top of the population.

In our implementation of the stochastic ranking method in the modified constrained differential evolution, each individual point $\boldsymbol{x}_i$ is evaluated according to the fitness function

$$\Phi_{\mathrm{SR}}(\boldsymbol{x}_i) = \frac{I_i - 1}{N - 1}, \tag{3}$$

where I_i represents the rank of point $\boldsymbol{x}_i$. From (3), the fitness of an individual point having the highest rank will be 0 and that with the lowest rank will be 1. The best individual point in a population has the lowest fitness value.

2.3 Global Competitive Ranking

Runarsson and Yao [22] proposed another constraints handling technique in order to strike the right balance between the objective function and the constraint violation. This method is called global competitive ranking (GR), where

an individual point is ranked by comparing it against all other members in the population.

In this ranking process, after calculating f and ζ for all the individuals, f and ζ are sorted separately in ascending order (since we consider the minimization problem) and given ranks. Special consideration is given to the *tied individuals*. In case of tied individuals the same higher rank will be given. For example, in these eight individuals, already in ascending order, $\langle 6, (5, 8), 1, (2, 4, 7), 3\rangle$ (individuals in parentheses have same value) the corresponding ranks are $I(6) = 1, I(5) = I(8) = 2, I(1) = 4, I(2) = I(4) = I(7) = 5, I(3) = 8$. After giving ranks to all the individuals based on the objective function f and the constraint violation ζ, separately, the fitness function of each individual point $\boldsymbol{x}_i$ is calculated by

$$\Phi_{\mathrm{GR}}(\boldsymbol{x}_i) = P_f \frac{I_{i,f} - 1}{N - 1} + (1 - P_f)\frac{I_{i,\zeta} - 1}{N - 1}, \tag{4}$$

where $I_{i,f}$ and $I_{i,\zeta}$ are the ranks of point $\boldsymbol{x}_i$ based on the objective function and the constraint violation, respectively. P_f indicates the probability that the fitness is calculated based on the rank of objective function. It is clear from the above that P_f can be used easily to bias the calculation of fitness according to the objective function or the constraint violation. The probability should take a value $0.0 < P_f < 0.5$ in order to guarantee that a feasible solution may be found. From (4), the fitness of an individual point is a value between 0 and 1, and the best individual point in a population has the lowest fitness value.

3 Modified Constrained Differential Evolution

The population-based differential evolution algorithm [24] has become popular and has been used in many practical cases, mainly because it has demonstrated good convergence properties and is easy to understand. DE is a floating point encoding that creates a new candidate point by adding the weighted difference between two individuals to a third one in the population. This operation is called mutation. The mutant point's components are then mixed with the components of target point to yield the trial point. This mixing of components is referred to as crossover. In selection, a trial point replaces a target point for the next generation only if it is considered an equal or better point. In unconstrained optimization, the selection operation relies on the objective function. DE has three control parameters: amplification factor of differential variation F, crossover control parameter Cr, and population size N.

It is not an easy task to set the appropriate control parameters since these depend on the nature and size of the optimization problems. Hence, the adaptive control parameters ought to be implemented. Brest et al. [3] proposed the self-adaptive control parameters for DE when solving global optimization problems with simple bounds. In most original DE, three points are chosen randomly for mutation and the base point is then chosen at random within the three. This has an exploratory effect but it slows down the convergence of DE. Kaelo and Ali [14] proposed a modified mutation for differential evolution.

The herein presented modified constrained differential evolution algorithm - mCDE - for constrained nonlinear optimization problems (1) includes:

1) the self-adaptive control parameters F and Cr, as proposed by Brest et al.;
2) a modified mutation that mixes the modification proposed by Kaelo and Ali with the cyclical use of the overall best point as the base point;
3) the inversion operation;
4) a modified selection that is based on the fitness of individuals;
5) the elitism.

The modification in mutation allows mCDE to keep the exploration as well as enhance the exploitation around the overall best point. In modified selection of mCDE, we implement and test the three different techniques described so far for calculating the fitness of individuals that are capable to handle the constraints of problems (1). The modified constrained differential evolution is outlined below.

The target point of mCDE, at iteration/generation z, is defined by $\boldsymbol{x}_{i,z} = (x_{i1,z}, x_{i2,z}, \ldots, x_{in,z})$, where n is the number of variables of the optimization problem and $i = 1, 2, \ldots, N$. The initial population is chosen randomly and should cover the entire component spaces.

Self-adaptive control parameters: In mCDE, we use the self-adaptive control parameters for F and Cr, as proposed by Brest et al. [3] by generating a different set (F_i, Cr_i) for each point $\boldsymbol{x}_i$ in the population. The new control parameters for the next generation $F_{i,z+1}$ and $Cr_{i,z+1}$ are calculated by

$$F_{i,z+1} = \begin{cases} F_l + \lambda_1 \times F_u & \text{if } \lambda_2 < \tau_1 \\ F_{i,z} & \text{otherwise} \end{cases}$$
$$Cr_{i,z+1} = \begin{cases} \lambda_3 & \text{if } \lambda_4 < \tau_2 \\ Cr_{i,z} & \text{otherwise,} \end{cases}$$

where $\lambda_k \sim \mathrm{U}[0,1], k = 1, \ldots, 4$ and $0 < \tau_1, \tau_2 < 1$ represent the probabilities to adjust parameters F_i and Cr_i, respectively, and $0 < F_l < F_u < 1$, so the new $F_{i,z+1}$ takes a value from $(0,1)$ in a random manner. The new $Cr_{i,z+1}$ takes a value from $[0,1]$. $F_{i,z+1}$ and $Cr_{i,z+1}$ are obtained before the mutation is performed. So, they influence the mutation, crossover and selection operations of the new point $\boldsymbol{x}_{i,z+1}$.

Modified mutation: In mCDE, this is a mixture of two different types of mutation operations. We use the mutation proposed in [14]. After choosing three points randomly, the best point among three based on the fitness function is selected for the base point and the remaining two points are used as differential variation, i.e., for each target point $\boldsymbol{x}_{i,z}$, a mutant point is created according to

$$\boldsymbol{v}_{i,z+1} = \boldsymbol{x}_{r_3,z} + F_{i,z+1}(\boldsymbol{x}_{r_1,z} - \boldsymbol{x}_{r_2,z}), \tag{5}$$

where r_1, r_2, r_3 are randomly chosen from the set $\{1, 2, \ldots, N\}$, mutually different and different from the running index i and r_3 is the index with the best fitness (among the three points). This modification has a local effect when the points in the population form a cluster around the global minimizer.

Furthermore, at every B generations, the best point found so far is used as the base point and two randomly chosen points are used as differential variation, i.e.,

$$\boldsymbol{v}_{i,z+1} = \boldsymbol{x}_{\text{best}} + F_{i,z+1}(\boldsymbol{x}_{r_1,z} - \boldsymbol{x}_{r_2,z}). \tag{6}$$

This modified mutation allows mCDE to maintain its exploratory feature as well as at the same time to exploit the region around the best individual point in the population expediting the convergence.

Crossover: In order to increase the diversity of the mutant points' components, crossover is introduced. To this end, the crossover point $\boldsymbol{u}_{i,z+1}$ is formed, where

$$u_{ij,z+1} = \begin{cases} v_{ij,z+1} & \text{if } (r_j \leq Cr_{i,z+1}) \text{ or } j = s_i \\ x_{ij,z} & \text{if } (r_j > Cr_{i,z+1}) \text{ and } j \neq s_i. \end{cases} \tag{7}$$

In (7), $r_j \sim \text{U}[0,1]$ performs the mixing of jth component of points, s_i is randomly chosen from the set $\{1, 2, \ldots, n\}$ and ensures that $\boldsymbol{u}_{i,z+1}$ gets at least one component from $\boldsymbol{v}_{i,z+1}$.

Inversion: Since in mCDE, a point has n-dimensional real components, inversion [13] can easily be applicable. With the inversion probability ($p_{\text{inv}} \in [0,1]$), two positions are chosen on the point $\boldsymbol{u}_i$, the point is cut at those positions, and the cut segment is reversed and reinserted back into the point to create the trial point $\boldsymbol{u}'_i$. In practice, mCDE with the inversion has been shown to give better results than those obtained without the inversion. An illustrative example of inversion is shown in Figure 1.

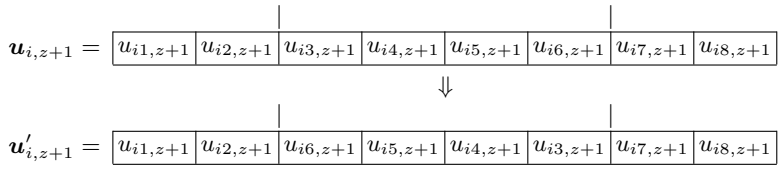

Fig. 1. Inversion used in mCDE

Bounds check: When creating the mutant point and when the inversion operation is performed, some components can be created outside the bound constraints. So, in mCDE after inversion the bounds of each component should be checked with the following projection of bounds:

$$u'_{ij,z+1} = \begin{cases} l_j & \text{if } u'_{ij,z+1} < l_j \\ u_j & \text{if } u'_{ij,z+1} > u_j \\ u'_{ij,z+1} & \text{otherwise.} \end{cases}$$

Modified selection: In original DE, the target and the trial points are compared based on their corresponding objective function value to decide which point becomes a member of next generation, that is if the trial point's objective

function is less than or equal to the that of target point, then the trial point will be the target point for the next generation.

In this paper, for constrained nonlinear optimization problems, we propose a modified selection based on one of the fitness functions of individuals discussed so far (Section 2). When using the stochastic ranking technique, all the target points at generation z and trial points at generation $z+1$ are ranked together and their corresponding fitness Φ_{SR} are calculated. Then the modified selection is performed, i.e., the trial and the target points are compared to decide which will be the new target points for the next generation based on their calculated fitness by the following way

$$\boldsymbol{x}_{i,z+1} = \begin{cases} \boldsymbol{u}'_{i,z+1} & \text{if } \Phi_{\mathrm{SR}}(\boldsymbol{u}'_{i,z+1}) \leq \Phi_{\mathrm{SR}}(\boldsymbol{x}_{i,z}) \\ \boldsymbol{x}_{i,z} & \text{otherwise.} \end{cases}$$

A similar procedure is performed when the global competitive ranking technique is implemented. After performing selection in mCDE, the best point is chosen in the current generation based on the lowest fitness of the target points.

On the other hand, when using the feasibility and dominance rules, the trial and the target points are compared based on the three feasibility and dominance principles to decide which will be the new target points for the next generation. After performing selection, the fitness function Φ_{FD} for all the target points are calculated, and the best point based on the lowest fitness function in the current generation is chosen. We remark that this point is the overall best point in the entire generations so far.

Elitism: The elitism is also performed to keep the best point found so far in the entire generations. The elitism aims at preserving in the entire generations the individual point that, with the constraint violation 0 or smaller than others, has the smallest objective function. This is required when either the stochastic ranking or the global competitive ranking is used to calculate fitness of individuals. We remark that in these two techniques, fitness values of individuals are calculated at every generation based on their corresponding ranks. Thus, the fitness of best individual point (based on the objective function and the constraint violation) may not be the lowest one.

Termination criterion: Let $G_{\max}$ be the maximum number of generations. If f_{best} is the best objective function value found so far and f_{opt} is the known optimal value, then our proposed mCDE algorithm terminates if $z > G_{\max}$ or $|f_{\mathrm{best}} - f_{\mathrm{opt}}| \leq \eta$, for a small positive number η.

mCDE algorithm: The algorithm of the herein proposed modified constrained differential evolution for constrained nonlinear optimization problems is described in Algorithm 1.

4 Experimental Results

We code mCDE in C with AMPL [11] interfacing and compile with Microsoft Visual Studio 9.0 compiler in a PC having 2.5 GHz Intel Core 2 Duo processor

Algorithm 1 mCDE algorithm

Require: N, $G_{\max}$, B, P_f, F_l, F_u, τ_1, τ_2, p_{inv}, and η.

Set $z = 1$. Randomly initialize $F_{i,1}$, $Cr_{i,1}$ and the population $\boldsymbol{x}_{i,1}\ \forall i = 1, \ldots, N$.
Calculate the fitness $\Phi(\boldsymbol{x}_{i,1})$, for all i, and perform elitism to choose f_{best} and $\boldsymbol{x}_{\text{best}}$.
while the termination criterion is not met **do**
  for $i = 1$ to N **do**
    Compute the control parameters $F_{i,z+1}$ and $Cr_{i,z+1}$.
    if MOD$(z + 1, B) = 0$ **then**
      Compute the mutant point $\boldsymbol{v}_{i,z+1}$ using (6).
    else
      Compute the mutant point $\boldsymbol{v}_{i,z+1}$ using (5).
    end if
    Perform the crossover to make point $\boldsymbol{u}_{i,z+1}$.
    if $\gamma \sim \text{U}[0, 1] \leq p_{\text{inv}}$ **then**
      Perform inversion to make the trial point $\boldsymbol{u}'_{i,z+1}$.
    end if
    Check the bounds of the trial point.
  end for
  Calculate the fitness $\Phi(\boldsymbol{x}_{i,z})$, $\Phi(\boldsymbol{u}'_{i,z+1})$, for all i.
  Perform modified selection.
  Perform elitism to choose f_{best} and $\boldsymbol{x}_{\text{best}}$. Set $z = z + 1$.
end while

and 4 GB RAM. We set $N = \min(100, 10n)$, $B = 10$, $P_f = 0.45$, $\tau_1 = \tau_2 = 0.1$, $F_l = 0.1$, $F_u = 0.9$, $p_{\text{inv}} = 0.05$ and $\eta = 10^{-6}$. We consider 13 benchmark constrained nonlinear optimization problems [21]. Their characteristics are outlined in Table 1. For these problems, we consider an individual point as a feasible one if $\zeta(\boldsymbol{x}) \leq \delta$, where δ is a very small positive number. Here we set $\delta = 10^{-8}$.

Table 1. Characteristics of the test problems

Prob.	Type of f	f_{opt}	n	m_1	m_2	m
g01	quadratic	-15.0000	13	9	0	9
g02	general	-0.8036	20	2	0	2
g03	polynomial	-1.0005	10	0	1	1
g04	quadratic	-30665.5387	5	6	0	6
g05	cubic	5126.4967	4	2	3	5
g06	cubic	-6961.8139	2	2	0	2
g07	quadratic	24.3062	10	8	0	8
g08	general	-0.0958	2	2	0	2
g09	general	680.6301	7	4	0	4
g10	linear	7049.2480	8	6	0	6
g11	quadratic	0.7499	2	0	1	1
g12	quadratic	-1.0000	3	1	0	1
g13	general	0.0539	5	0	3	3

At first, we compare the three different variants of mCDE: a) mCDE_FD (based on *feasibility and dominance rules*), b) mCDE_SR (based on *stochastic ranking*) and c) mCDE_GR (based on *global competitive ranking*) using the performance profiles as described in [7]. A comparison with other solution methods available in literature is also included. To be able to fairly compare the variants mCDE_SR and mCDE_GR with the variant mCDE_FD, after the modified selection step of the algorithm, the fitness function was recalculated using (2) so that the best and the worst target points in the population are identified according to the objective function and constraint violation.

4.1 Comparison by Performance Profiles

We ran the three variants of mCDE for 30 times and recorded the results. We used different $G_{\max}$ for the 13 problems, but used the same value for all the variants in comparison. The performance profiles proposed by Dolan and Moré [7] are the graphical representation of the performance ratio of different solvers/variants when solving a set of test problems. The profiles plot the cumulative distribution function of the performance ratio obtained from an appropriate performance metric.

Let $\mathcal{P}$ be the set of test problems and $\mathcal{S}$ be the set of all variants of mCDE in comparison. In our comparative study, the metric, $m_{(p,s)}$, found by variant $s \in \mathcal{S}$ on problem $p \in \mathcal{P}$, measures the average improvement of the objective function values, based on a relative scaled distance to the optimal objective function value f_{opt} [1], defined by

$$m_{(p,s)} = \frac{f_{\text{avg}(p,s)} - f_{\text{opt}}}{f_w - f_{\text{opt}}}, \tag{8}$$

where $f_{\text{avg}(p,s)}$ is the average of the best solutions obtained by the variant s on problem p after 30 runs and f_w is the worst objective function value of problem p after 30 runs among all variants. The performance ratio is thus defined by

$$r_{(p,s)} = \begin{cases} 1 + m_{(p,s)} - q & \text{if } q \leq 10^{-5} \\ \dfrac{m_{(p,s)}}{q} & \text{otherwise,} \end{cases}$$

where $q = \min\{m_{(p,s)} : s \in \mathcal{S}\}$.

The fraction of problems for which variant s has a performance ratio $r_{(p,s)}$ within a factor $\tau \in \mathbb{R}$, is given by $\rho_s(\tau) = (n_{P_\tau})/(n_P)$, where n_{P_τ} is the number of problems in $\mathcal{P}$ with $r_{(p,s)} \leq \tau$ and n_P is the total number of problems in $\mathcal{P}$. $\rho_s(\tau)$ is the probability (for $s \in \mathcal{S}$) that the performance ratio $r_{(p,s)}$ is within a factor τ of the best possible ratio.

Figure 2 shows the profiles of the performance metric in (8). If we are only interested in knowing which variant is the most efficient, in the sense that it reaches the best solutions mostly, we compare the values of $\rho_s(1)$, for $s \in \mathcal{S}$, and find the highest value which is the probability that the variant will win over the remaining ones. However, to assess the robustness of variants, we compare the

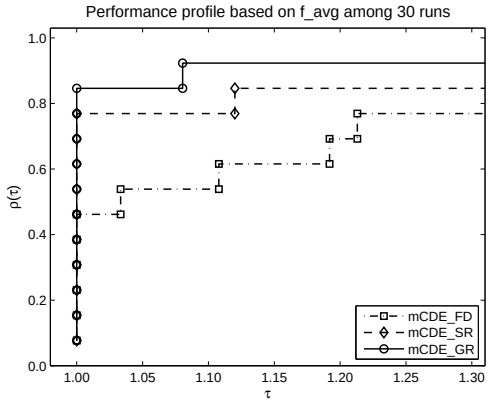

Fig. 2. Performance profile based on the average improvement of function values

values of $\rho_s(\tau)$ for large values of τ. The variant with the largest probability is the most robust one. In this figure it is shown that the variant mCDE_GR wins over the other two variants. Hence, the comparison with other solution methods available in literature uses the variant mCDE_GR, hereafter denoted by mCDE.

4.2 Comparison With Other Methods

We also compare mCDE with the stochastic ranking, SRES, presented in [21] and the global competitive ranking, GRES, presented in [22]. The authors proposed these techniques based on a $(30, 200)$ evolution strategy. An adaptive penalty scheme for constraints handling with dynamic use of variants of differential evolution (DUVDE) [23] is also used in this comparison. According to [21, 22], we set $G_{\max} = 1750$ for all problems except problem g12, where $G_{\max} = 175$. Here, we aim to get a solution within 0.001% of the known optimal solution f_{opt}.

Tables 2 and 3 show the experimental results of the 13 problems, where 'f_{best}' is the best of the objective function values obtained among 30 runs, 'f_{avg}' is the average of the best objective function values and 'std_f' means the standard deviation of objective function values among 30 runs. The results from SRES, GRES and DUVDE are taken from their corresponding literatures. In mCDE, we use the population size N dependent on the dimension of the test problem and in DUVDE the authors used the population size 50 and the maximum number of generations 3684 for all the test problems. Problems g12 and g13 were not considered with DUVDE. From Tables 2 and 3 we may conclude that for most of the problems, and with respect to all measures of comparison, mCDE performs rather well when compare with SRES, GRES and DUVDE.

4.3 Solving Mixed-Integer Design Problems

We now consider four engineering design problems to show the effectiveness of our proposed method when solving problems with discrete, integer and con-

Table 2. Experimental results from SRES and GRES

Prob.	SRES			GRES		
	f_{best}	f_{avg}	std_f	f_{best}	f_{avg}	std_f
g01	-15.0000	-15.0000	0.00E+00	-15.0000	–	0.00E+00
g02	-0.8035	-0.7820	2.00E-02	-0.8035	–	1.70E-02
g03	-1.0000	-1.0000	1.90E-04	-1.0000	–	2.60E-05
g04	-30665.5390	-30665.5390	2.00E-05	-30665.5390	–	5.40E-01
g05	5126.4970	5128.8810	3.50E+00	5126.4970	–	1.10E+00
g06	-6961.8140	-6875.9400	1.60E+02	-6943.5600	–	2.90E+02
g07	24.3070	24.3740	6.60E-02	24.3080	–	1.10E-01
g08	-0.0958	-0.0958	2.60E-17	-0.0958	–	2.60E-17
g09	680.6300	680.6560	3.40E-02	680.6310	–	5.80E-02
g10	7054.3160	7559.1920	5.30E+02	*	–	*
g11	0.7500	0.7500	8.00E-05	0.7500	–	7.20E-05
g12	-1.0000	-1.0000	0.00E+00	-1.0000	–	0.00E+00
g13	0.0539	0.0675	3.10E-02	0.0539	–	1.30E-04

(–) not available; (*) not solved

Table 3. Experimental results from DUVDE and mCDE

Prob.	DUVDE			mCDE		
	f_{best}	f_{avg}	std_f	f_{best}	f_{avg}	std_f
g01	-15.0000	-12.5000	2.37E+00	-15.0000	-15.0000	1.16E-06
g02	-0.8036	-0.7688	3.57E-02	-0.8036	-0.8007	4.95E-03
g03	-1.0000	-0.2015	3.45E-01	-1.0000	-1.0000	3.90E-05
g04	-30665.5000	-30665.5000	0.00E+00	-30665.5387	-30665.5387	2.38E-05
g05	5126.4965	5126.4965	0.00E+00	5126.4978	5126.4979	1.83E-04
g06	-6961.8000	-6961.8000	0.00E+00	-6961.8161	-6950.5609	6.16E+01
g07	24.3060	30.4040	2.16E+01	24.2316	24.2317	7.44E-05
g08	-0.0958	-0.0958	0.00E+00	-0.0958	-0.0958	2.71E-06
g09	680.6300	680.6300	3.00E-05	680.6301	680.6301	1.38E-06
g10	7049.2500	7351.1700	5.26E+02	7049.2533	7053.3441	6.99E+00
g11	0.7500	0.9875	5.59E-02	0.7500	0.7506	3.11E-03
g12	†	†	†	-1.0000	-1.0000	2.33E-06
g13	†	†	†	0.0539	0.0539	3.53E-17

(†) not considered

tinuous variables. Engineering problems with mixed-integer design variables are quite common. Therefore, the convergence behavior of our proposed mCDE when handling discrete and integer variables is to be provided.

For discrete variables, we randomly generate values from an appropriate discrete set in the two procedures: initialization and mutation.

For integer variables, a simple heuristic that relies on the rounding off to the nearest integer at evaluation stages is implemented.

We considered four well-known engineering design problems. Since the optimal solutions of the considered problems are unknown, we used only $G_{\max}$ for

the termination criterion and $\delta = 0$. For each problem 30 independent runs were carried out.

Spring Design

This is a real world optimization problem involving discrete, integer and continuous design variables. The objective is to minimize the volume of a compression spring under static loading. The design problem has three variables and eight inequality constraints [15], where x_1, the wire diameter, is taken from a set of discrete values and x_3, the number of coils, is integer. We set $G_{\max} = 500$. We compare the obtained results from our mCDE with DE [15] and ranking selection-based particle swarm optimization, RPSO [26]. The comparative results are shown in Table 4.

Table 4. Comparative results of spring design problem

Method	x_1	x_2	x_3	f_{best}	$G_{\max}$
DE	0.283	1.223	9	2.65856	650
RPSO	0.283	1.223	9	2.65856	750
mCDE	0.283	1.223	9	2.65856	500

Pressure Vessel Design

The design of a cylindrical pressure vessel with both ends capped with a hemispherical head is to minimize the total cost of fabrication [5, 25]. The problem has four design variables and four inequality constraints. This is a mixed variables problem where x_1, the shell thickness, and x_2, the head thickness, are discrete of integer multiples of 0.0625 inch., and other two are continuous. We set $G_{\max} = 1000$. The comparative results from mCDE with hybrid genetic algorithm, HGA [5] and cost-effective particle swarm optimization, CPSO [25] are shown in Table 5.

Table 5. Comparative results of pressure vessel design problem

Method	x_1	x_2	x_3	x_4	f_{best}	$G_{\max}$
HGA	0.8125	0.4375	42.0870	176.7791	6061.123	5000
CPSO	0.8125	0.4375	42.0984	176.6366	6059.714	10000
mCDE	0.8125	0.4375	42.0984	176.6366	6059.714	1000

Speed Reducer Design

The weight of the speed reducer is to be minimized subject to the constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts and stress in the shafts as described in [5, 25]. There are seven variables and 11 inequality constraints. This is a mixed variables problem, where x_3 is integer (number of teeth) and the others are continuous. We set $G_{\max} = 500$. The comparative results among mCDE, HGA and CPSO are shown in Table 6.

Table 6. Comparative results of speed reducer design problem

Method	x_1	x_2	x_3	x_4	x_5	x_6	x_7	f_{best}	$G_{\max}$
HGA	3.5	0.7	17	7.3	7.7153	3.3502	5.2867	2994.342	5000
CPSO	3.5	0.7	17	7.3	7.8000	3.3502	5.2867	2996.348	10000
mCDE	3.5	0.7	17	7.3	7.7153	3.3502	5.2867	2994.342	500

Stepped Cantilever Beam Design
The design variables of a stepped cantilever beam are the widths and depths of rectangular cross-sections. The objective of this problem is to minimize the volume of the beam under static loading [27]. This is a mixed-integer design problem having 10 variables and 11 constraints, where x_1 and x_2 are integer, x_3 to x_6 are discrete and the remaining are continuous. We set $G_{\max} = 1000$. The comparative results from our mCDE with genetic algorithm, GA [9], and linearization techniques method [27] are shown in Table 7.

Table 7. Comparative results of stepped cantilever beam design problem

Method	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	f_{best}	$G_{\max}$
GA	3	60	3.1	55	2.6	50	2.2700	45.2500	1.7500	35.0000	64447.00	–
in [27]	3	60	3.1	55	2.6	50	2.2045	44.0907	1.7498	34.9960	63892.56	–
mCDE	3	60	3.1	55	2.6	50	2.2055	44.0855	1.7502	34.9924	63897.45	1000

(–) not available

From the Tables 4 - 7, it is found that mCDE is competitive with other solution methods when solving engineering design problems.

From the above discussion it is clear that the herein presented modified constrained differential evolution algorithm, based on global competitive ranking for constraints handling, is rather effective when converging to global solutions.

5 Conclusions

In this paper, to make the DE methodology more efficient to handle the constraints, a modified constrained differential evolution algorithm (mCDE) is proposed. The modifications focus on the self-adaptive control parameters, a modified mutation, a modified selection and the elitism. Inversion has also been implemented in the proposed mCDE.

The modifications that mostly influence the efficiency of the algorithm are the following: a) the mixed modified mutation, aiming at exploring both the entire search space (when using the mutation as in [14]) and the neighborhood of the best point found so far (when using the best point as the base point cyclically); b) the modified selection, to handle the constraints effectively, that uses a fitness function based on the global competitive ranking technique. In this technique, fitness of all target and trial points are calculated all together after

giving them ranks based on the objective function and the constraint violation separately, for competing in modified selection to decide which points win for the next generation. This technique seems to have stricken the right balance between the objective function and the constraint violation for obtaining a global solution while satisfying the constraints.

To test the effectiveness of our mCDE, 13 benchmark constrained nonlinear optimization problems have been considered. These problems have also been solved with the stochastic ranking and the feasibility and dominance rules techniques and a comparison has been carried out based on their performance profiles. We could observe that the performance of the mCDE with the global competitive ranking is relatively better than the other two in comparison. The numerical experiments also show that mCDE is rather competitive when compared with the other solution methods available in literature. Further, it is also found that the mCDE is competitive with other known heuristics when solving mixed-integer engineering design problems.

Acknowledgments. The first author acknowledges Ciência 2007 of FCT, Fundação para a Ciência e a Tecnologia (Foundation for Science and Technology), Portugal for the financial support under fellowship grant: C2007-UMINHO-ALGORITMI-04. The second author acknowledges FEDER COMPETE, Programa Operacional Fatores de Competitividade (Operational Programme Thematic Factors of Competitiveness) and FCT for the financial support under project grant: FCOMP-01-0124-FEDER-022674.

References

1. Ali, M.M., Khompatraporn, C., Zabinsky, Z.B.: A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. J. Glob. Optim. 31, 635–672 (2005)
2. Barbosa, H.J.C., Lemonge, A.C.C.: A new adaptive penalty scheme for genetic algorithms. Inf. Sci. 156, 215–251 (2003)
3. Brest, J., Greiner, S., Bošković, B., Mernik, M., Žumer, V.: Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans. Evol. Comput. 10, 646–657 (2006)
4. Coello Coello, C.A.: Constraint-handling using an evolutionary multiobjective optimization technique. Civ. Eng. Environ. Syst. 17, 319–346 (2000)
5. Coello Coello, C.A., Cortés, N.C.: Hybridizing a genetic algorithm with an artificial immune system for global optimization. Eng. Optim. 36, 607–634 (2004)
6. Deb, K.: An efficient constraint handling method for genetic algorithms. Comput. Methods Appl. Mech. Eng. 186, 311–338 (2000)
7. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. 91, 201–213 (2002)
8. Dong, Y., Tang, J., Xu, B., Wang, D.: An application of swarm optimization to nonlinear programming. Comput. Math. Appl. 49, 1655–1668 (2005)
9. Erbatur, F., Hasançebi, O., Tütüncü, İ., Kılıç, H.: Optimal design of planar and space structures with genetic algorithms. Comput. Struct. 75, 209–224, (2000)
10. Fletcher, R., Leyffer, S.: Nonlinear programming without a penalty function. Math. Program. 91, 239–269 (2002)

11. Fourer, R., Gay, D.M., Kernighan, B.W.: AMPL: A modeling language for mathematical programming. Boyd & Fraser Publishing Co., Massachusetts (1993)
12. Hedar, A.R., Fukushima, M.: Derivative-free filter simulated annealing method for constrained continuous global optimization. J. Glob. Optim. 35, 521–549 (2006)
13. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor (1975)
14. Kaelo, P., Ali, M.M.: A numerical study of some modified differential evolution algorithms. Eur. J. Oper. Res. 169, 1176–1184 (2006)
15. Lampinen, J., Zelinka, I.: Mixed integer-discrete-continuous optimization by differential evolution. In: Proceedings of the 5th International Conference on Soft Computing, pp. 71–76 (1999)
16. Ray, T., Tai, K.: An evolutionary algorithm with a multilevel pairing strategy for single and multiobjective optimization. Found. Comput. Decis. Sci. 26, 75–98 (2001)
17. Ray, T., Liew, K.M.: Society and civilization: An optimization algorithm based on the simulation of social behavior. IEEE Trans. Evol. Comput. 7, 386–396 (2003)
18. Rocha, A.M.A.C., Fernandes, E.M.G.P.: Feasibility and dominance rules in the electromagnetism-like algorithm for constrained global optimization. In: Gervasi et al. (eds.), Computational Science and Its Applications–ICCSA 2008, Part II, LNCS, vol 5073, pp. 768–783. Springer-Verlag, Heidelberg (2008)
19. Rocha, A.M.A.C., Fernandes, E.M.G.P.: Self adaptive penalties in the electromagnetism-like algorithm for constrained global optimization problems. In: Proceedings of the 8th World Congress on Structural and Multidisciplinary Optimization, pp. 1–10 (2009)
20. Rocha, A.M.A.C., Martins, T.F.M.C., Fernandes, E.M.G.P.: An augmented Lagrangian fish swarm based method for global optimization, J. Comput. Appl. Math., 235, 4611–4620 (2011)
21. Runarsson, T.P., Yao, X.: Stochastic ranking for constrained evolutionary optimization. IEEE Trans. Evol. Comput. 4, 284–294 (2000)
22. Runarsson, T.P., Yao, X.: Constrained evolutionary optimization – the penalty function approach. In: Sarker et al. (eds.), Evolutionary Optimization: International Series in Operations Research and Management Science, vol. 48, pp. 87–113, Springer, New York (2003)
23. Silva, E.K., Barbosa, H.J.C., Lemonge, A.C.C.: An adaptive constraint handling technique for differential evolution with dynamic use of variants in engineering optimization. Optim. Eng. 12, 31–54 (2011)
24. Storn, R., Price, K.: Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. 11, 341–359 (1997)
25. Tomassetti, G.: A cost-effective algorithm for the solution of engineering problems with particle swarm optimization. Eng. Optim. 42(5), 471–495 (2010)
26. Wang, J., Yin, Z.: A ranking selection-based particle swarm optimizer for engineering design optimization problems. Struct. Multidisc. Optim. 37(2), 131–147 (2008)
27. Wang, P.-C., Tsai, J.-F.: Global optimization of mixed-integer nonlinear programming for engineering design problems. In: Proceedings of the International Conference on System Science and engineering, pp. 255-259 (2011)
28. Zahara, E., Hu, C.-H.. Solving constrained optimization problems with hybrid particle swarm optimization. Eng. Optim. 40, 1031–1049 (2008)