Optimization
Vol. 00, No. 00, January 2009, 1–17

RESEARCH ARTICLE

Numerical Study of Augmented Lagrangian Algorithms for Constrained Global Optimization

Ana Maria A. C. Rocha* and Edite M. G. P. Fernandes

Department of Production and Systems, University of Minho, 4710-057 Braga, Portugal

(*v1.0 released February 2011*)

This paper presents a numerical study of two augmented Lagrangian algorithms to solve continuous constrained global optimization problems. The algorithms approximately solve a sequence of bound constrained subproblems whose objective function penalizes equality and inequality constraints violation and depends on the Lagrange multiplier vectors and a penalty parameter. Each subproblem is solved by a population-based method that uses an electromagnetism-like mechanism to move points towards optimality. Three local search procedures are tested to enhance the EM algorithm. Benchmark problems are solved in a performance evaluation of the proposed augmented Lagrangian methodologies. A comparison with other techniques presented in the literature is also reported.

Keywords: global optimization; augmented Lagrangian; electromagnetism-like mechanism; heuristics;

AMS Subject Classification: 90C30; 90C26; 90C56; 90C59

1. Introduction

This paper aims at analyzing the practical behavior of two augmented Lagrangian methodologies for continuous constrained global optimization, where the subproblems have bound constraints only and are solved by the electromagnetism-like mechanism, a stochastic population-based algorithm. The problem to be addressed has the form:

$$\min f(x) \text{ subject to } g(x) \leq 0\,, h(x) = 0\,, x \in \Omega, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^p$ and $h : \mathbb{R}^n \to \mathbb{R}^m$ are nonlinear continuous functions and $\Omega = \{x \in \mathbb{R}^n : lb \leq x \leq ub\}$. We do not assume that the objective function f is convex. There may be many local minima in the feasible region. This class of global optimization problems arises frequently in engineering applications. Specially for large scale problems, derivative-free and stochastic methods are the most well-known and used methods. When equality and inequality constraints are present in the optimization problem, one of the following categories of methods can be used. In the methods based on penalty functions, the constraints violation is combined with the objective function to define a penalty function. This function aims at penalizing infeasible solutions by increasing their fitness values proportionally to their level of constraints violation. Penalty functions require the use

*Corresponding author. Email: arocha@dps.uminho.pt

ISSN: 0233-1934 print/ISSN 1029-4945 online

DOI: 10.1080/0233193YYxxxxxxxx
http://www.informaworld.com

of a positive penalty parameter that aims to balance function and constraint violation values. The most popular penalty functions use static, dynamic, annealing or adaptive penalty updating schemes [2, 4, 8, 17, 19, 22]. An augmented Lagrangian function is a penalty function that depends on a penalty parameter, as well as on the Lagrange multiplier vectors associated with the constraints of the problem. Augmented Lagrangians are common in deterministic type methods for global optimization [6, 7, 18], but rare when combined with heuristics that rely on a population of points to converge to the solution [1, 25, 27, 28]. The other category defines methods based on biasing feasible over infeasible solutions. They seem to be nowadays an interesting alternative to penalty methods for handling constraints. In this type of methods, constraints violation and the objective function are used separately and optimized by some sort of order, being the constraints violation the most important. See, for example, [9, 13, 20, 21, 23, 24, 29].

In this paper, we are interested in a penalty-type method that uses augmented Lagrangian methodologies to handle the equality and inequality constraints of the problem (1), where the subproblems are approximately solved by a stochastic global population-based algorithm. Due to its simplicity, the electromagnetism-like (EM) algorithm proposed in [4, 5] is used to obtain the solution of each subproblem. The EM algorithm simulates the electromagnetism theory of physics by considering each point in the population as an electrical charge. The method uses an attraction-repulsion mechanism to move a population of points towards optimality. Since the EM algorithm has been designed to find a minimizer which satisfies $x \in \Omega$, our subproblem is defined as a bound constrained optimization problem.

The herein proposed implementation of an augmented Lagrangian methodology follows two paradigms. First, we apply the augmented Lagrangian function of Powell-Hestenes-Rockafellar (PHR) directly to the problem (1), and use the EM algorithm to solve the bound constrained subproblems. The EM algorithm has been used within a classical penalty technique [4], but has not been used with an augmented Lagrangian function so far. Second, we reformulate problem (1) converting each equality constraint into an inequality as herein shown: $|h_j(x)| \leq \varepsilon$, where ε is a positive relaxation parameter. This is an usual procedure in stochastic based methods [9, 14, 21, 22]. In general, the relaxation parameter is fixed over the entire iterative process. Typically, 10^{-3}, 10^{-4} and 10^{-5} are common values in the literature. Our proposal defines a sequence $\{\varepsilon^k\}$ of decreasing nonnegative numbers such that $\lim_{k\to\infty} \varepsilon^k = \varepsilon^* > 0$. The idea is to tighten the equality constraints relaxation scheme as iterations proceed. Further, a different updating scheme for the penalty parameter is also proposed. When the level of constraints violation is under a specified tolerance, even if the infeasibility did not improve, the penalty is allowed to decrease instead of increasing (see Algorithm 2.2). In both cases, the bound constrained subproblems are approximately solved by the EM algorithm. This algorithm has been enhanced by a random local search procedure [5]. However, in practical terms, the therein proposed local search may require a large number of function evaluations since the local search is carried out along the coordinates. Attempting to reduce computational requirements while improving accuracy, two new local search procedures are herein proposed and tested to enhance the EM algorithm. One is based on the computation of a descent direction and the other relies on a unit-length randomly generated direction. We remark that there is no theoretical analysis yet for this algorithm.

The paper is organized as follows. Section 2 describes the two augmented Lagrangian paradigms and Section 3 reviews the EM algorithm and presents the three local search procedures in comparison. Section 4 contains the results of all the numerical experiments, performance assessments based on Dolan and Moré's profiles

[10], and a comparison with other methods in the literature. Finally, the remarks are included in Section 5.

2. Augmented Lagrangian methodologies

Most stochastic methods for global optimization are developed primarily for unconstrained or simple bound constrained problems. Then they are extended to constrained optimization problems using, for example, a penalty technique. This type of technique transforms the constrained problem into a sequence of unconstrained subproblems by penalizing the objective function when constraints are violated. The objective penalty function, in the unconstrained subproblem, consists of the objective function $f(x)$ plus a positive penalty parameter times a measure of the aggregate constraint violation. The choice of the penalty parameter may be problematic. In general, the penalty parameter is updated throughout the iterative process. With most penalty functions, the solution of the constrained problem is reached for an infinite value of the penalty parameter. An augmented Lagrangian is a more sophisticated penalty function for which a finite penalty parameter value is sufficient to yield convergence to the solution of the constrained problem [3].

Two augmented Lagrangian functions for solving constrained global optimization problems are now presented. Practical and theoretical issues from the augmented Lagrangian methodology are used with a stochastic population based algorithm, the EM algorithm [5], to compute approximate solutions of the sequence of bound constrained subproblems.

2.1. *Handling equalities and inequalities separately*

Our first proposal uses the original formulation (1) and makes use of the augmented Lagrangian function of Powell-Hestenes-Rockafellar (PHR):

$$\mathcal{L}_{\rho}^{EI}(x,\lambda,\mu) = f(x) + \frac{\rho}{2}\left\{\sum_{i=1}^{m}\left[h_i(x)+\frac{\lambda_i}{\rho}\right]^2 + \sum_{j=1}^{p}\left[\max\left(0,\, g_j(x)+\frac{\mu_j}{\rho}\right)\right]^2\right\} \quad (2)$$

where $\lambda \in \mathbb{R}^m$, $\mu \in \mathbb{R}^p$ are the vectors of Lagrange multipliers associated with $h(x) = 0$ and $g(x) \leq 0$ respectively, and ρ is a positive penalty parameter. For the sake of completeness we present in the Algorithm 2.1 the ideas presented in [7]. In this cited paper, the subproblems, at each iteration k,

$$\min_{x} \mathcal{L}_{\rho^k}^{EI}(x,\lambda^k,\mu^k) \text{ subject to } x \in \Omega \quad (3)$$

are approximately solved using a deterministic global optimization method known as αBB method.

According to recent works with the function (2) [6, 7], the penalty parameter is increased whenever the infeasibility is not reduced; otherwise it is not changed (see lines 7-11 in Algorithm 2.1). The initial value for the parameter is

$$\rho^1 = \max\left\{10^{-6}, \min\left\{10,\, 2|f(x^0)|/(\|\max(0,g(x^0))\|^2 + \|h(x^0)\|^2)\right\}\right\}$$

for an arbitrary initial approximation x^0. The algorithm also updates the Lagrange multipliers using first order estimates and safeguarded schemes (lines 12-13 in Algorithm 2.1). This is a crucial issue to maintain the sequences $\{\lambda^k\}, \{\mu^k\}$ bounded.

Algorithm 2.1 (Augmented Lagrangian algorithm)

1: **Given:** $\mu^+ > 0$, $\lambda^- < \lambda^+$, $\epsilon^* > 0$, $0 < \tau_c < 1$, $\gamma > 1$, $k_{\max}$, $\mu^1 \in [0, \mu^+]$, $\lambda^1 \in [\lambda^-, \lambda^+]$
2: choose arbitrary x^0 in Ω; compute ρ^1; set $k = 1$
3: **while** $\max\{\|v^{k-1}\|, \|h(x^{k-1})\|\} > \epsilon^*$ and $k \leq k_{\max}$ **do**
4: $\epsilon^k = \max\left\{\epsilon^*, 10^{-k}\right\}$;
5: compute x^k, an ϵ^k-global solution of $\min_x \mathcal{L}_{\rho^k}^{EI}(x, \lambda^k, \mu^k)$ subject to $x \in \Omega$
6: compute $v_j^k = \max\left\{g_j(x^k), -\dfrac{\mu_j^k}{\rho^k}\right\}, j = 1, \ldots, p$
7: **if** $k = 1$ or $\max\{\|v^k\|, \|h(x^k)\|\} \leq \tau_c \max\{\|v^{k-1}\|, \|h(x^{k-1})\|\}$ **then**
8: $\rho^{k+1} = \rho^k$
9: **else**
10: $\rho^{k+1} = \gamma\rho^k$
11: **end if**
12: update $\mu_j^{k+1} = \min\left\{\max\left(0,\, \mu_j^k + \rho^k g_j(x^k)\right),\, \mu^+\right\}, j = 1, \ldots, p$
13: update $\lambda_i^{k+1} = \min\left\{\max\left(\lambda^-,\, \lambda_i^k + \rho^k h_i(x^k)\right),\, \lambda^+\right\}, i = 1, \ldots, m$
14: $k = k + 1$
15: **end while**

This paper aims at providing a different augmented Lagrangian algorithm that can be also implemented with the augmented Lagrangian function $\mathcal{L}_\rho^{EI}$. The main differences can be summarized as follows:

i) the initial approximation x^0 is a randomly generated point;
ii) the subproblems (3) are solved by the EM algorithm, a stochastic algorithm based on a population of points, which uses the best solution found so far as the initial approximation to the subproblem of the next iteration;
iii) the penalty parameter ρ, besides being increased, is also reduced whenever the constraints violation is under a specified tolerance ϵ^k, even if the level of infeasibility has increased.

Further, the penalty updating scheme herein used integrates a safeguarded scheme. This is motivated by the need to keep the penalty parameters bounded and the subproblems well conditioned. This procedure is reported in the lines 12–20 of the new Algorithm 2.2. With this algorithm, we aim to show the above mentioned differences, as well as the differences between using the formulation based on the Lagrangian (2) (translated in the Algorithm 2.1) and that based only on inequality constraints, as shown in (6). Issues related with the equality constraint relaxation parameter, ε^k, and details concerning the solving of subproblem (3) using the EM algorithm are described in the next subsection.

2.2. *Formulation based on inequality constraints*

Since equality constraints are the most difficult to be satisfied, the other augmented Lagrangian methodology considers problems only with inequality constraints, using a common procedure in stochastic methods for global optimization to convert the equality constraints of the problem into inequality constraints, as follows: $|h_j| \leq \varepsilon$, $j = 1, \ldots, m$ for a fixed $\varepsilon > 0$. For simplicity, the problem (1) is rewritten as

$$\min f(x) \text{ subject to } G(x) \leq 0\,,\, x \in \Omega, \tag{4}$$

where the vector of the inequality constraints is now defined by $G(x) = (g_1(x), \ldots, g_p(x), |h_1(x)| - \varepsilon, \ldots, |h_m(x)| - \varepsilon)$. We now define $t = p + m$. Our proposal concerning the relaxed equality constraints aims at tightening the relaxation

scheme as iterations proceed, using variable relaxation parameter values. Thus, a sequence of decreasing nonnegative values bounded by $\varepsilon^* > 0$ is defined as:

$$\varepsilon^{k+1} = \max\left\{\varepsilon^*, \frac{1}{\gamma}\varepsilon^k\right\}, \ \gamma > 1. \tag{5}$$

The PHR formula that corresponds to the inequality constraints in the converted problem (4) yields the augmented Lagrangian:

$$\mathcal{L}^I_\rho(x,\mu) = f(x) + \frac{\rho}{2}\sum_{i=1}^{t}\left[\max\left(0,\, G_i(x) + \frac{\mu_i}{\rho}\right)\right]^2 \tag{6}$$

where the Lagrange multiplier vector associated with the constraints $G(x) \leq 0$, μ, has now t elements.

Algorithm 2.2 (Proposed augmented Lagrangian algorithm)

1: **Given:** $\mu^+ > 0$, $\epsilon^* > 0$, $0 < \tau_c < 1$, $\gamma > 1$, $k_{\max}$, $l_{\max}$, $\varepsilon^* > 0$, $0 < \rho^- < \rho^+$, $\mu^1 \in [0, \mu^+]$
2: randomly generate x^0 in Ω; compute ρ^1; set $k = 1$
3: **while** $\|v^{k-1}\| > \epsilon^*$ and $k \leq k_{\max}$ **do**
4: $\epsilon^k = \max\left\{\epsilon^*, 10^{-k}\right\}$; update ε^k using (5); set $l = 1$
5: **while** $\left(\mathcal{L}^I_{\text{avg}} - \mathcal{L}^I_{\rho^k}(x(\text{best}), \mu^k)\right) > \epsilon^k$ and $l \leq l_{\max}$ **do**
6: use x^{k-1} and randomly initialize a population of $p_{\text{size}} - 1$ points in Ω
7: run EM to compute a population of solutions to $\min_x \mathcal{L}^I_{\rho^k}(x, \mu^k)$ subject to $x \in \Omega$
8: $l = l + 1$
9: **end while**
10: $x^k = x(\text{best})$
11: compute $v_i^k = \max\left\{G_i(x^k), -\frac{\mu_i^k}{\rho^k}\right\}, i = 1, \ldots, t$
12: **if** $k = 1$ or $\|v^k\| \leq \tau_c\|v^{k-1}\|$ **then**
13: $\rho^{k+1} = \rho^k$
14: **else**
15: **if** $\|v^k\| \leq \epsilon^k$ **then**
16: $\rho^{k+1} = \max\{\rho^-, \frac{1}{\gamma}\rho^k\}$
17: **else**
18: $\rho^{k+1} = \min\{\rho^+, \gamma\rho^k\}$
19: **end if**
20: **end if**
21: update $\mu_i^{k+1} = \min\left\{\max\left(0,\, \mu_i^k + \rho^k G_i(x^k)\right),\, \mu^+\right\}, i = 1, \ldots, t$
22: $k = k + 1$
23: **end while**

The herein proposed augmented Lagrangian algorithm adapted to the reformulation (4), of the original problem (1), and based on the Lagrangian (6) is presented in Algorithm 2.2. Lines 5-9 of the algorithm show details of the inner iterative process to compute an approximation to the solution of subproblem (3). Since the EM algorithm is based on a population of points, with size p_{size}, the point which yields the least objective function value, denoted by the best point of the population, $x(\text{best})$, after stopping, is taken as the next approximation to the problem (1). We also note that the stochastic EM algorithm uses the approximation x^{k-1} as one of the points of the population to initialize the EM algorithm. The remaining $p_{\text{size}} - 1$ points are randomly generated.

The inner iteration counter is represented by l. This process terminates when the difference between the function value at the best point, $\mathcal{L}^I_{\rho^k}(x(\text{best}), \mu^k)$, and the

average of the function values of the population, $\mathcal{L}^I_{\text{avg}}$, is under a specified tolerance ϵ^k. This tolerance decreases as outer iterations proceed. A limit of $l_{\max}$ iterations is also imposed.

3. The electromagnetism-like mechanism

This section reviews the electromagnetism-like mechanism, proposed in [5], for solving the subproblems in the Algorithm 2.2. In this algorithm context, an approximate minimizer of the augmented Lagrangian function, $\mathcal{L}_{\rho^k}(x, \mu^k)$, for fixed values of the parameters ρ^k and μ^k is required. To simplify the notation, $\mathcal{L}^k(x) = \mathcal{L}_{\rho^k}(x, \mu^k)$ will be used throughout the remainder of the paper. Because EM is a population-based algorithm, the inner iterative process begins with a population of p_{size} solutions (line 6 in Algorithm 2.2). The best solution found so far, denoted by $x(\text{best})$, and the average of the objective function values, are defined by

$$x(\text{best}) = \arg\min \left\{ \mathcal{L}^k(x(s)) : s = 1, \ldots, p_{\text{size}} \right\} \text{ and } \mathcal{L}^k_{\text{avg}} = \sum_{s=1}^{p_{\text{size}}} \mathcal{L}^k(x(s))/p_{\text{size}}, \tag{7}$$

respectively, where $x(s), s = 1, \ldots, p_{\text{size}}$ represent the points of the population. The main steps of the EM mechanism are shown in Algorithm 3.1. Details of each step follow.

Algorithm 3.1 (EM algorithm)

1: **Given:** $x(s), s = 1, \ldots, p_{\text{size}}$
2: evaluate the population and select $x(\text{best})$
3: compute the charges $c(s)$, $s = 1, \ldots, p_{\text{size}}$
4: compute the total forces $F(s)$, $s = 1, \ldots, p_{\text{size}}$
5: move the points except $x(\text{best})$
6: evaluate the new population and select $x(\text{best})$
7: apply a local search to $x(\text{best})$
8: compute $\mathcal{L}^k(x(\text{best}))$ and $\mathcal{L}^k_{\text{avg}}$.

The EM mechanism starts by identifying the best point, $x(\text{best})$, of the population using the augmented Lagrangian $\mathcal{L}^k$ for point assessment, see (7). According to the electromagnetism theory, the total force exerted on each point $x(s)$ by the other $p_{\text{size}} - 1$ points is inversely proportional to the square of the distance between the points and directly proportional to the product of their charges:

$$F(s) = \sum_{r \neq s}^{p_{\text{size}}} F^s_r \equiv \begin{cases} (x(r) - x(s)) \dfrac{c(s)c(r)}{\|x(r) - x(s)\|^2}, \text{ if } \mathcal{L}^k(x(r)) < \mathcal{L}^k(x(s)) \\ (x(s) - x(r)) \dfrac{c(s)c(r)}{\|x(r) - x(s)\|^2}, \text{ otherwise} \end{cases},$$

for $s = 1, \ldots, p_{\text{size}}$, where the charge $c(s)$ of point $x(s)$ determines the magnitude of attraction of that point over the others through

$$c(s) = \exp\left(\frac{-n\left(\mathcal{L}^k(x(s)) - \mathcal{L}^k(x(\text{best}))\right)}{\sum_{r=1}^{p_{\text{size}}} \left(\mathcal{L}^k(x(r)) - \mathcal{L}^k(x(\text{best}))\right)} \right).$$

Then, the normalized total force vector exerted on each point $x(s)$ is used to move the point in the direction of the force by a random step size $\iota \sim U[0,1]$, maintaining the point inside the set Ω. Thus for $s = 1, \ldots, p_{\text{size}}$ ($s \neq \text{best}$) and for each component $i = 1, \ldots, n$

$$x_i(s) = \begin{cases} x_i(s) + \iota \dfrac{F_i(s)}{\|F(s)\|}(ub_i - x_i(s)), \text{ if } F_i(s) > 0 \\ x_i(s) + \iota \dfrac{F_i(s)}{\|F(s)\|}(x_i(s) - lb_i), \text{ otherwise} \end{cases}.$$

3.1. *A random local search*

Step 7 of Algorithm 3.1 aims at refining the search around the best point of the population only. A simple local search procedure proposed in [5], in the context of the EM algorithm, is described in Algorithm 3.2. This is a coordinatewise search applied to $x(\text{best})$. For each component i, $x(\text{best})$ is assigned to a temporary point y. Then a random movement of maximum length $\Delta = \delta \max_j(ub_j - lb_j)$, $\delta > 0$, is carried out and if a better position is obtained within $\max_{local}$ iterations, $x(\text{best})$ is replaced by y, the search ends for that component and proceeds to another one. When $y \notin \Omega$, the trial point is rejected and another random movement is tried for that component.

Although this local search is based on a simple random procedure, it has shown that improves accuracy of the EM algorithm although at a cost of function evaluations. See the numerical study presented in Subsection 4.3 and the results in Table 1. To avoid the search along the coordinates, two other local search procedures to enhance the EM algorithm in the augmented Lagrangian context are presented in the next subsections. One uses a descent direction for the augmented Lagrangian function and the other is based on a random direction.

Algorithm 3.2 (Local search algorithm)

1: **Given:** $x(\text{best})$, $\max_{local}$, δ
2: $\Delta = \delta \max_j(ub_j - lb_j)$
3: **for** $i = 1$ to n **do**
4: set $it = 1$
5: **while** $it < \max_{local}$ **do**
6: $y \leftarrow x(\text{best})$
7: $y_i = y_i + \iota\,\Delta$, $\iota \sim U[-1,1]$ (reject if not feasible)
8: **if** $\mathcal{L}^k(y) < \mathcal{L}^k(x(\text{best}))$ **then**
9: $x(\text{best}) \leftarrow y$, $it = \max_{local} - 1$
10: $it = it + 1$
11: **end while**
12: **end for**

3.2. *A descent local search*

Here, a detailed description of a derivative-free heuristic method that produces an approximate descent direction and aims to generate a new trial point around the best point of the population is presented. In [12], a descent direction is proposed in a point-to-point search context, a simulated annealing method. It is shown that for a set of l exploring points close to $x(\text{best})$, an approximate descent direction may be produced if:

i) the points are randomly generated in a small neighborhood of $x(\text{best})$ and $l = 2$, or
ii) the points are in equal distance to $x(\text{best})$, define with $x(\text{best})$ a set of orthogonal directions, and $l = n$.

For simplicity, case i) is implemented, and to produce a descent direction, two points in a neighborhood of ray $\delta > 0$, of the best point, $x(\text{best})$, are randomly generated as follow:

$$x_i^j(\text{rand}) = x_i(\text{best}) + \iota\,\delta \ (i = 1, \ldots, n), \ \text{ for } \ j = 1, 2 \tag{8}$$

where $\iota \sim U[-1, 1]$ and δ is a sufficiently small positive value. The approximate descent direction d for the augmented Lagrangian function $\mathcal{L}^k$, at $x(\text{best})$, is defined by

$$d = -\frac{1}{\sum_{l=1}^{2} |\Delta_l|} \sum_{j=1}^{2} \Delta_j \frac{x(\text{best}) - x^j(\text{rand})}{\|x(\text{best}) - x^j(\text{rand})\|}, \tag{9}$$

where $\Delta_j = \mathcal{L}^k(x(\text{best})) - \mathcal{L}^k(x^j(\text{rand}))$. A trial point is generated along the descent direction with a prescribed step size,

$$y = x(\text{best}) + \alpha\, d, \tag{10}$$

where $\alpha \in (0, 1]$ represents the step size. We remark that if $y \notin \Omega$, the point y is projected onto the set Ω. When selecting a step size to detect a trial point y that leads to an improvement in $\mathcal{L}^k$, when compared with the best point, the herein proposed algorithm uses a classical backtracking strategy. Algorithm 3.3 presents a formal description of the descent local search. First, two random exploring points and a descent direction are generated. These two steps (lines 5-6) in the Algorithm 3.3 are executed whenever *flag* is set to 1. Then, a trial point y is calculated and, according to the augmented Lagrangian function values, either y or $x(\text{best})$ is selected. If $x(\text{best})$ still is the best point, then y is discarded, the step size is halved (i.e., $\alpha \leftarrow \alpha/2$) and a new point is evaluated along that descent direction (*flag* is set to 0 in the Algorithm 3.3). However, when y is the best, another approximate descent direction is computed (*flag* is set to 1, and α is reset to 1) and the process is repeated. The search for a better point is implemented for at most $\max_{local}$ iterations. Practical performance of this descent search, when compared with the other two local search procedures, is shown in Subsection 4.4.

3.3. *A random walk*

The random walk with direction exploitation method can be used as a local search procedure to refine the search around a particular point in the population. It has been applied as a local search operator to enhance a particle swarm optimization algorithm [19] and recently a differential evolution algorithm [16]. This random walk generates a random vector, as a search direction, and when applied to the best point $x(\text{best})$ gives

$$y = x(\text{best}) + \alpha\, z, \tag{11}$$

where $\alpha \in (0, 1]$ represents the step size and z is a unit-length random vector. The components of z are randomly generated in the interval $[-1, 1]$. The algorithm

Algorithm 3.3 (Descent local search algorithm)

Given: $x(\text{best})$, $\max_{local}$, δ
set $flag = 1$, $\alpha = 1$, $it = 0$
while $it \leq \max_{local}$ **do**
  if $flag = 1$ **then**
    generate two random points using (8)
    compute descent direction d using (9)
  end if
  compute trial point y using (10) (project if not feasible)
  if $\mathcal{L}^k(y) < \mathcal{L}^k(x(\text{best}))$ **then**
    $x(\text{best}) \leftarrow y$, $\alpha = 1$, $flag = 1$
  else
    $\alpha = \alpha/2$, $flag = 0$
  end if
  $it = it + 1$
end while

herein implemented projects the point y onto the set Ω, when the point falls outside the bounds. Experiments have shown that this projection scheme is more efficient than the feasibility repair proposed in [16].

The random walk exploitation search can be summarized as the Algorithm 3.4 below. A backtracking strategy is also implemented. If y does not improve over $x(\text{best})$, the step size is halved, and the random walk is tried again; otherwise, y replaces $x(\text{best})$, α is reset to 1, and a new random walk is tried. Random walks can be tried for at most $\max_{local}$ iterations.

Algorithm 3.4 (Random walk algorithm)

Given: $x(\text{best})$, $\max_{local}$
set $\alpha = 1$, $it = 0$
while $it \leq \max_{local}$ **do**
  generate the random vector and compute point y using (11) (project if not feasible)
  if $\mathcal{L}^k(y) < \mathcal{L}^k(x(\text{best}))$ **then**
    $x(\text{best}) \leftarrow y$, $\alpha = 1$
  else
    $\alpha = \alpha/2$
  end if
  $it = it + 1$
end while

4. Numerical experiments

In this section, we report the results of our numerical study, after running a set of 24 benchmark constrained global problems, described in full detail in [15]. The problems are known as g01-g24 (the 'g' suit, where six problems only have equality constraints, thirteen have inequality constraints, five have both equalities and inequalities and all have simple bounds). Not all problems have multi-modal objective functions, although some are difficult to solve. The best known solution for problem g20 is slightly infeasible. We remark that g02, g03, g08 and g12 are maximization problems that were transformed and solved as minimization ones. The C programming language is used in this real-coded algorithm that contains an interface to connect to AMPL and read the problems coded in AMPL [11]. The computational tests were performed on a PC with a 3GHz Pentium IV microprocessor and 1Gb of memory.

Since the algorithm relies on some random parameters and variables, we solve each problem 30 times and take average of the obtained solutions, herein denoted by f_{avg}. The best of the solutions found after all runs is denoted by f_{best}. The size of the population depends on n, and since some problems have large dimension, $n > 20$, we choose $p_{\text{size}} = \min\{200, 10n\}$. The fixed parameters are set in this study as follows: $\lambda^+ = \mu^+ = \rho^+ = 10^{12}$, $\epsilon^* = 10^{-6}$, $\tau_c = 0.5$, $\gamma = 2$, $\lambda^- = -10^{12}$, $\varepsilon^* = \rho^- = 10^{-12}$, $\varepsilon^1 = 10^{-3}$. We define $k_{\max} = 50$ and $l_{\max} = 30$ so that a maximum of 1500 iterations are allowed. We remark that the other conditions in the stopping criteria of the Algorithm 2.2 (in the outer and inner iterative processes) may cause the termination of the algorithm before reaching the 1500 iterations. The initial multiplier vectors are set to the null vectors.

The values for the two parameters in the local search procedures are set as proposed in [5]: $\max_{local} = 10$, $\delta = 0.001$.

Overall, the algorithm with $\mathcal{L}^I$ has nine parameters and with $\mathcal{L}^{EI}$ has eleven (including the two from the local search algorithm). The population size and the maximum number of allowed iterations are not counted as parameters since they are common to most population based techniques that we may use for comparison. Several tests were performed and some comparisons were carried out to choose appropriate values for some of the listed parameters.

4.1. *Comparisons based on performance profiles*

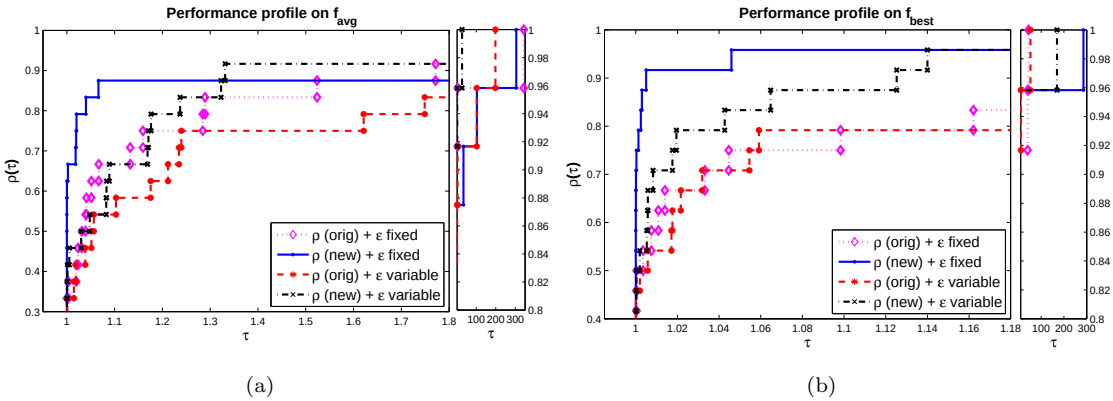
To compare the performance of the two augmented Lagrangian methodologies, and to analyze the effect of some parameters in the algorithm, we use performance profiles as described in Dolan and Moré's paper [10]. Our profiles are based on the metrics: f_{avg}, the average of the solutions obtained at the end of each one of the 30 runs, and f_{best}, the best solution found in the 30 runs. Based on the chosen metric, these profiles compare the performance of a set of solvers, denoted by $\mathcal{S}$, when solving a set of problems, here denoted by $\mathcal{P}$. Let $m_{p,s}$ be the value of the metric when solving problem $p \in \mathcal{P}$ by solver $s \in \mathcal{S}$. The comparison is based on the performance ratios defined by

$$r_{p,s} = \begin{cases} 1 + m_{p,s} - \min\{m_{p,s} : s \in \mathcal{S}\}, \text{ if } \min\{m_{p,s} : s \in \mathcal{S}\} < \beta \\ \frac{m_{p,s}}{\min\{m_{p,s} : s \in \mathcal{S}\}}, \text{ otherwise} \end{cases},$$

where β is a positive small parameter [26]. We use $\beta = 0.00001$. The overall assessment of the performance of the solver s is given by $\rho_s(\tau) =$ (no. of problems where $r_{p,s} \leq \tau$)/(total no. of problems). Thus, $\rho_s(\tau)$ gives the probability (for $s \in \mathcal{S}$) that $r_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio. The value of $\rho_s(1)$ gives the probability that a particular solver, s, will win over the others in comparison. Thus, to just see which solver is the best, i.e., which solver has the least value of the metric mostly, then $\rho_s(1)$ should be compared for all the solvers. The higher the ρ_s the better the solver is. On the other hand, $\rho_s(\tau)$ for large values of τ measures the solver robustness.

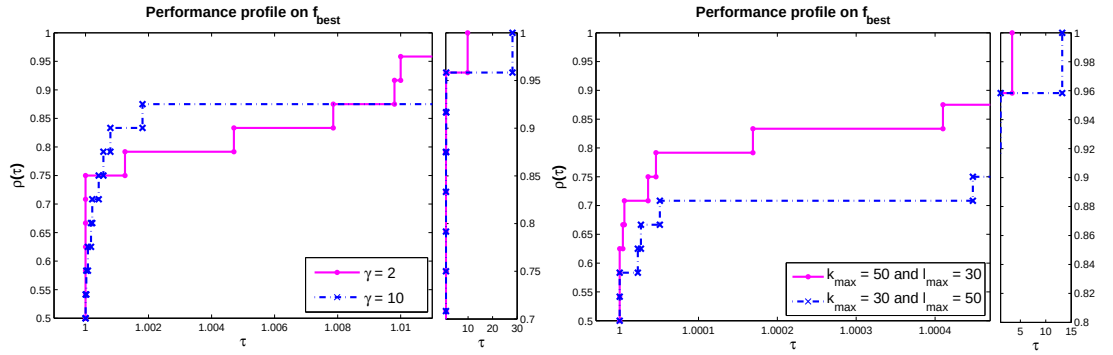
First, we carried out some tests to analyze the effect of the relaxation parameter choices in the context of the augmented Lagrangian $\mathcal{L}^I_\rho(x, \mu)$. Besides the usual setting of a fixed value, for example $\varepsilon = 10^{-5}$, we implemented a variable update, as previously described in (5). The two penalty parameter updating schemes are also compared. The Figure 1 contains two plots. The plot (a) shows the performance profiles on the average performance, f_{avg}, of the four cases in comparison, herein denoted for convenience as:

- ρ (orig) + ε fixed – ρ update according to [7] and $\varepsilon = 10^{-5}$;



(a) (b)

Figure 1. Comparison of penalty and relaxation parameter updates based on: (a) f_{avg} and (b) f_{best}.



(a) (b)

Figure 2. Sensitivity analysis for: (a) the parameter γ; (b) different combinations of $k_{\max}$ and $l_{\max}$.

- ρ (new) + ε fixed – ρ update according to Algorithm 2.2 and $\varepsilon = 10^{-5}$;
- ρ (orig) + ε variable – ρ update according to [7] and ε update according to (5);
- ρ (new) + ε variable – ρ update according to Algorithm 2.2 and ε update according to (5).

Plot (b) shows the profiles on the best obtained solution over the 30 runs, f_{best}. We can conclude that the new proposed ρ updating scheme, combined with fixed ε, outperforms the other combinations. Thus, this is the combination used in the remaining numerical tests.

To analyze the effect of parameter γ, from the penalty parameter updating, as well as the effect of using less outer iterations and more inner iterations, while maintaining a maximum of 1500 iterations, on the performance of the algorithm, we use the augmented Lagrangian $\mathcal{L}^I$, and solve the 'g' suit. For the first set of experiments, we test $\gamma = 2$ and $\gamma = 10$. From the plot on the left of Figure 2 we may conclude that the choice $\gamma = 2$ is slightly preferable. The profiles are based on the best performance of the algorithm, although a similar conclusion could be drawn from the average performance. The other plot, on the right, displays the profiles of the two cases in comparison: $k_{\max} = 50$ and $l_{\max} = 30$ *versus* $k_{\max} = 30$ and $l_{\max} = 50$. Based on the best performance, the algorithm with the former choice is more effective in reaching the solution. (The same is true for the average performance.)

Table 1. EM without and with random local search defined by the pair $(\delta, \max_{local})$.

Prob.	n	function	p	m	f^*	variant	f_{best}	f_{avg}	Nfe_{avg}
g01	13	quadratic	9	0	-15.0000	without	-9.2716	-7.9919	81181
						(0.001, 10)	-14.9993	**-14.9981**	140384
						(0.00001, 5)	**-14.9999**	-12.9932	149216
g05	4	polynomial	2	3	5126.497	without	5138.835	5270.772	60451
						(0.001, 10)	**5126.962**	**5139.517**	67352
						(0.00001, 5)	5130.838	5203.950	70953
g08	2	nonlinear	2	0	-0.09583	without	-0.09581	-0.08355	16670
						(0.001, 10)	**-0.09583**	**-0.09582**	19988
						(0.00001, 5)	-0.09575	-0.07794	20689
g09	7	polynomial	4	0	680.630	without	694.277	723.444	35865
						(0.001, 10)	**681.325**	**682.973**	90063
						(0.00001, 5)	682.737	710.918	57616
g15	3	quadratic	0	2	961.715	without	961.720	964.791	44951
						(0.001, 10)	**961.715**	961.930	31717
						(0.00001, 5)	**961.715**	**961.845**	34844
g16	5	nonlinear	38	0	-1.90516	without	-1.82091	-1.65362	43647
						(0.001, 10)	**-1.90511**	**-1.85746**	71832
						(0.00001, 5)	-1.84184	-1.66073	63246
g18	9	quadratic	13	0	-0.86603	without	-0.58326	-0.04607	137951
						(0.001, 10)	**-0.86598**	**-0.85909**	150500
						(0.00001, 5)	-0.62815	-0.20088	176436

4.2. *Algorithm complexity*

For completeness, the algorithm complexity, according to [15] is reported, using

$$T_1 = \frac{1}{N}\sum_{i=1}^{N} cp_i, \;\; T_2 = \frac{1}{N}\sum_{i=1}^{N} ccp_i, \tag{12}$$

where cp_i and ccp_i represent the computing time (in seconds) of 10000 evaluations of the basic functions (f, g and h) for problem i, and the complete computing time for the algorithm when 10000 evaluations of the functions are allowed, for problem i, respectively, and N is the number of problems used in this computation. Using the 'g' suit ($N = 24$) and the augmented Lagrangian $\mathcal{L}^I$, we obtain:

$$T_1 = 0.0723, \;\; T_2 = 0.8239 \;\; \text{and} \;\; (T_2 - T_1)/T_1 = 10.3956.$$

We remark that the displayed values are the average values over three runs and that the code was not yet optimized. We may conclude that the computational effort of the operations involved in the algorithm is about ten times the effort of evaluating the basic functions.

4.3. *Random local search effect*

Here, the effect of the random local search procedure (Algorithm 3.2) in the EM algorithm, is analyzed. Seven problems with different dimensions are selected: g01 with $n = 13$, g05 with $n = 4$, g08 with $n = 2$, g09 with $n = 7$, g15 with $n = 3$, g16 with $n = 5$ and g18 with $n = 9$. Each problem was solved 30 times. We run Algorithm 2.2 and selected the augmented Lagrangian $\mathcal{L}^I$ for this comparison. Table 1 displays the problem (Prob.), the number of variables (n), the type of objective function (function), the number of inequality constraints (p), the number of equality constraints (m), the best known solution as reported in [15] (f^*), the variant, f_{best}, f_{avg} and Nfe_{avg} (this represents the average number of function

evaluations after the 30 runs). The three variants in comparison include one without local search and two with local search, each defined by the pair of parameters $(\delta, \max_{local})$. The accuracy of the obtained solutions is improved when the local search is incorporated into the EM algorithm, although, as expected, at a cost of function evaluations. The choice (0.001, 10) for the pair of parameters is also better than the other in comparison. In all tables of the paper, the best results for f_{best} and f_{avg}, in each table, are in boldface.

4.4. *Comparing local search procedures*

The random local search of Subsection 3.1 has an important limitation. Using it as described in Step 7 of Algorithm 3.1 can be time-consuming. It may require $n \max_{local}$ extra function evaluations, at each iteration. Its use seems impracticable for moderate dimension problems. The proposals in Subsections 3.2 and 3.3 are more attractive since they look less expensive to implement and computational requirements do not depend on the problem dimension, as opposed to the case of random local search.

We run Algorithm 2.2 with both augmented Lagrangian functions, $\mathcal{L}^{EI}$ and $\mathcal{L}^{I}$, and tested the three local search procedures. In this study, the algorithm is terminated when 100000 function evaluations are reached. Table 2 contains the results for the problems of the 'g' suit. The local search procedures are identified in the table by: 'coordinate' (Algorithm 3.2), 'descent' (Algorithm 3.3) and 'rand. walk' (Algorithm 3.4). The character '–' means that the solution is infeasible. From the table we may conclude that Algorithm 3.2 slightly outperforms the other two in comparison, and the augmented Lagrangian $\mathcal{L}^{I}$ attains, in general, the most accurate results.

4.5. *Comparison with other algorithms*

To compare the performance of the herein proposed augmented Lagrangian algorithm with other penalty techniques in the literature [2, 17, 19], we report in Tables 3, 4 and 5 our results and those of the cited papers. In [2], a genetic algorithm (GA) combined with an adaptive penalty function (APF) is implemented. Five variants are tested. These are mainly concerned with the frequency of penalty parameters updating and constraints violation computation. The authors in [17] propose a momentum-type particle swarm optimization (PSO) algorithm combined with a dynamic penalty function (DPF) for solving constrained problems.

The method in [19] is a memetic particle swarm optimization (MPSO) algorithm, with a local search based on a random walk with direction exploitation (RW), and a dynamic penalty function. Both local and global variants are therein tested. To compare our results with the three chosen techniques, the problems were solved using the conditions described in the paper in comparison. These conditions are displayed in each table. They differ from one case to another and are concerned with p_{size}, number of runs, and maximum number of iterations/generations or function evaluations allowed. To identify the problem ('Prob.') we use the notation of the paper, except when the problem is of the 'g' suit or it has been used in a previous table. We report the results obtained with both augmented Lagrangians. The random local search procedure was chosen for these tests since it performed well in previous experiments.

Table 2. Comparison of local search procedures and augmented Lagrangians.

Prob.	n		$\mathcal{L}^{EI}$ coordinate	$\mathcal{L}^{EI}$ descent	$\mathcal{L}^{EI}$ rand. walk	$\mathcal{L}^{I}$ coordinate	$\mathcal{L}^{I}$ descent	$\mathcal{L}^{I}$ rand. walk
g01	13	f_{best}	-14.9999	-14.9999	-14.9905	-14.9994	**-15.0000**	-14.9919
		f_{avg}	-14.9991	**-14.9998**	-14.9709	-14.9988	-14.9994	-14.9707
g02	20	f_{best}	-0.46412	-0.53511	-0.54800	-0.55100	-0.52706	**-0.61182**
		f_{avg}	-0.43477	-0.43894	-0.44931	-0.44935	-0.44155	**-0.45506**
g03	10	f_{best}	-0.99889	-0.99801	**-0.99985**	-0.99628	-0.99034	-0.99274
		f_{avg}	-0.99649	-0.99488	**-0.99926**	-0.98333	-0.97917	-0.97583
g04	5	f_{best}	-30665.53	-30665.52	-30665.53	**-30665.54**	**-30665.54**	-30665.53
		f_{avg}	-30665.52	-30665.50	-30665.52	**-30665.53**	-30665.52	-30665.52
g05	4	f_{best}	5126.600	5128.881	5133.819	**5126.517**	5130.504	5126.884
		f_{avg}	5129.494	5247.374	5255.988	**5128.023**	5268.96	5169.594
g06	2	f_{best}	-6958.508	-6961.699	-6961.759	-6961.002	-6961.717	**-6961.801**
		f_{avg}	-6951.732	-6961.472	-6961.599	-6953.515	-6961.511	**-6961.642**
g07	10	f_{best}	24.3079	24.3115	24.3191	**24.3076**	24.3141	24.3162
		f_{avg}	24.3137	24.3246	24.3279	**24.3112**	24.3243	24.3289
g08	2	f_{best}	**-0.09583**	**-0.09583**	**-0.09583**	**-0.09583**	**-0.09583**	**-0.09583**
		f_{avg}	**-0.09583**	**-0.09583**	**-0.09583**	-0.09582	**-0.09583**	-0.09582
g09	7	f_{best}	**680.630**	680.633	680.631	**680.630**	680.632	680.631
		f_{avg}	**680.630**	680.641	680.634	685.195	680.638	680.633
g10	8	f_{best}	7062.69	7059.47	7246.59	7070.14	**7057.88**	7152.41
		f_{avg}	**7156.16**	7212.84	8078.80	7256.98	7867.54	7639.34
g11	2	f_{best}	0.75088	0.81034	0.75082	**0.74999**	**0.74999**	**0.74999**
		f_{avg}	0.87873	0.97309	0.96779	**0.74999**	0.75000	**0.74999**
g12	3	f_{best}	**-1.00000**	**-1.00000**	**-1.00000**	**-1.00000**	**-1.00000**	**-1.00000**
		f_{avg}	**-1.00000**	**-1.00000**	**-1.00000**	**-1.00000**	**-1.00000**	**-1.00000**
g13	5	f_{best}	**0.05394**	**0.05394**	**0.05394**	0.05395	0.05396	**0.05394**
		f_{avg}	**0.05396**	0.05397	0.05401	0.05632	0.06563	0.05398
g14	10	f_{best}	**-47.7649**	-47.7648	-47.7647	-47.7570	-47.7603	-47.7647
		f_{avg}	**-47.7637**	-47.7466	-47.7625	-47.4253	-47.7457	-47.7565
g15	3	f_{best}	**961.715**	**961.715**	**961.715**	**961.715**	**961.715**	**961.715**
		f_{avg}	**961.715**	961.716	961.716	961.719	961.779	961.728
g16	5	f_{best}	**-1.90514**	-1.90489	-1.90496	**-1.90514**	-1.90490	-1.90507
		f_{avg}	-1.90506	-1.84890	-1.87424	**-1.90512**	-1.85303	-1.86282
g17	6	f_{best}	8856.65	8874.67	8967.28	**8855.57**	9011.92	9011.90
		f_{avg}	8890.55	8962.38	9005.49	**8868.11**	9018.28	9049.12
g18	9	f_{best}	-0.86598	-0.86597	-0.86551	**-0.86600**	-0.86595	-0.86568
		f_{avg}	-0.86579	-0.86561	-0.86507	**-0.86591**	-0.86544	-0.86511
g19	15	f_{best}	34.2133	45.1205	48.0194	**33.7640**	44.0871	47.8128
		f_{avg}	38.5755	54.2540	63.3142	**35.5095**	52.9676	61.1339
g20	24	f_{best}	(0.328)†	–	–	(0.433)‡	–	–
		f_{avg}	–	–	–	–	–	–
g21	7	f_{best}	199.803	251.604	**196.864**	318.207	229.648	201.318
		f_{avg}	302.791	515.192	264.104	320.119	**234.261**	506.068
g22	22	f_{best}	**236.816**	263.643	260.929	241.089	544.381	378.238
		f_{avg}	**283.712**	1322.5	1602.1	316.427	4756.6	1906.0
g23	9	f_{best}	-396.830	**-399.851**	-396.040	-398.955	-395.357	-399.810
		f_{avg}	**-377.815**	-375.862	-370.603	-375.397	-356.822	-360.810
g24	2	f_{best}	**-5.50801**	-5.50799	-5.50800	**-5.50801**	**-5.50801**	-5.50786
		f_{avg}	-5.50800	-5.50790	-5.50793	**-5.50801**	-5.50799	-5.50710

† least constraints violation obtained with $\mathcal{L}^{EI}$.
‡ least constraints violation obtained with $\mathcal{L}^{I}$.

Table 3 shows f_{best} and f_{avg} obtained by our study and those of [2] for the eleven problems therein reported (g01-g11). In [2] each variable was encoded with

Table 3. Comparison of our results with the best of 5 variants in [2].

Prob.	f^*	our study			[2]	
		Aug. Lagrangian	f_{best}	f_{avg}	f_{best}	f_{avg}
g01	-15.0000	$\mathcal{L}^{EI}$	**-14.9994**	-14.9983	-14.9998	**-14.9989**
		$\mathcal{L}^{I}$	-14.9993	-14.9985		
g02	-0.80362	$\mathcal{L}^{EI}$	-0.57604	-0.43129	**-0.79252**	**-0.72555**
		$\mathcal{L}^{I}$	-0.49211	-0.33695		
g03	-1.00050	$\mathcal{L}^{EI}$	-0.99684	**-0.99524**	**-0.99725**	-0.77797
		$\mathcal{L}^{I}$	-0.99470	-0.97080		
g04	-30665.54	$\mathcal{L}^{EI}$	-30665.52	**-30665.44**	-30665.32	-30578.55
		$\mathcal{L}^{I}$	**-30665.54**	-30665.26		
g05	5126.497	$\mathcal{L}^{EI}$	5128.380	5135.457	5126.779	5323.866
		$\mathcal{L}^{I}$	**5126.738**	**5130.937**		
g06	-6961.814	$\mathcal{L}^{EI}$	-6950.783	-6896.591	**-6961.448**	-6805.229
		$\mathcal{L}^{I}$	-6954.896	**-6910.745**		
g07	24.3062	$\mathcal{L}^{EI}$	24.3078	24.4817	24.5450	27.8486
		$\mathcal{L}^{I}$	**24.3070**	**24.3579**		
g08	-0.09583	$\mathcal{L}^{EI}$	**-0.09583**	**-0.09583**	**-0.09583**	-0.08769
		$\mathcal{L}^{I}$	**-0.09583**	**-0.09583**		
g09	680.630	$\mathcal{L}^{EI}$	**680.630**	**680.645**	680.681	681.470
		$\mathcal{L}^{I}$	**680.630**	680.736		
g10	7049.25	$\mathcal{L}^{EI}$	7098.94	8844.95	7070.56	8063.29
		$\mathcal{L}^{I}$	**7058.56**	**7147.76**		
g11	0.74990	$\mathcal{L}^{EI}$	**0.74993**	**0.74998**	0.75217	0.88793
		$\mathcal{L}^{I}$	0.74999	0.75002		

Conditions in [2]: $p_{\text{size}} = 100$, runs = 25, maximum number of generations = 1000, leading to 100000 fitness function evaluations.

25 bits in a binary-coded GA. From the description of the algorithm in the paper, it is possible to identify three parameters in GA plus two in the procedure related with APF. We have better performance (both in f_{best} and f_{avg}) than the adaptive penalty algorithm of [2] in six problems.

Table 4 contains the results of our study to compare with the results reported in [17]. In the technique therein presented it is possible to identify four parameters in the PSO plus five in DPF. Since the algorithm was allowed to run for 5000 iterations, the solutions presented in this table may be better than those of the other tables. We have better performance (both in f_{best} and f_{avg}) than [17] in two problems.

Finally, to compare with the dynamic penalty algorithm of [19], we register in Table 5 our results of f_{avg} and the corresponding standard deviation (Stand. Dev.), for the six problems listed in [19]. From the paper it is possible to identify three parameters in MPSO, plus two in RW and at least two in DPF. We obtain better f_{avg} values in two problems and the other results are competitive.

A comparison based on the algorithms' complexity cannot be carried out since the computing times T_1 and T_2, see (12), are not provided in the papers [2, 17, 19].

5. Final remarks

From our preliminary numerical tests, we may conclude that the proposed augmented Lagrangian algorithm is able to effectively solve constrained problems till optimality. In particular, the augmented Lagrangian paradigm that uses relaxed equality constraints produces solutions with good accuracy. The augmented La-

Table 4. Comparison of our results with the results in [17].

Prob.	f^*		our study		[17]	
		Aug. Lagrangian	f_{best}	f_{avg}	f_{best}	f_{avg}
g02	-0.80362	$\mathcal{L}^{EI}$	-0.46286	-0.42938	**-0.80360**	**-0.746**
		$\mathcal{L}^{I}$	-0.58645	-0.43610		
g06	-6961.814	$\mathcal{L}^{EI}$	-6959.716	-6948.447	**-6961.814**	**-6961.781**
		$\mathcal{L}^{I}$	-6957.870	-6933.025		
g14	-47.7649	$\mathcal{L}^{EI}$	**-47.7616**	**-47.7600**	-47.562	-46.604
		$\mathcal{L}^{I}$	-47.7544	-47.5938		
P2	-31026.44	$\mathcal{L}^{EI}$	**-31026.42**	**-31026.40**	-31025.56	-31025.56
		$\mathcal{L}^{I}$	**-31026.42**	-31026.38		
P3	-11.0000	$\mathcal{L}^{EI}$	**-11.0000**	-10.9535	**-11.0000**	**-11.0000**
		$\mathcal{L}^{I}$	-10.9999	-10.9998		
P4	-213.000	$\mathcal{L}^{EI}$	**-213.000**	-212.997	**-213.000**	**-213.000**
		$\mathcal{L}^{I}$	**-213.000**	-212.998		

Conditions in [17]: $p_{\text{size}} = 50$, runs = 20, maximum number of iterations = 5000.

Table 5. Comparison of our results with the best of the 2 variants in [19].

Prob.	f^*		our study		[19]	
		Aug. Lagrangian	f_{avg}	Stand. Dev.	f_{avg}	Stand. Dev.
g04	-30665.54	$\mathcal{L}^{EI}$	-30665.41	0.091	**-30665.55**	0.000
		$\mathcal{L}^{I}$	-30665.39	0.153		
g06	-6961.814	$\mathcal{L}^{EI}$	-6926.984	18.75	**-6961.283**	0.380
		$\mathcal{L}^{I}$	-6936.535	6.25		
g09	680.630	$\mathcal{L}^{EI}$	**680.631**	0.0008	680.784	0.062
		$\mathcal{L}^{I}$	**680.631**	0.0007		
TP10	1.39347	$\mathcal{L}^{EI}$	1.40640	0.011	1.427	0.061
		$\mathcal{L}^{I}$	**1.39982**	0.004		
P2 †	-31026.44	$\mathcal{L}^{EI}$	-31026.30	0.144	**-31026.44**	0.000
		$\mathcal{L}^{I}$	-31026.36	0.067		
P4 ‡	-213.000	$\mathcal{L}^{EI}$	-212.995	0.003	**-213.047**	0.002
		$\mathcal{L}^{I}$	-212.995	0.002		

Conditions in [19]: $p_{\text{size}} = 100$, runs = 30, maximum number of function evaluations = 100000.
† TP14 in [19]; ‡ TP15 in [19].

grangian framework, coupled with the random local search procedure to enhance the EM algorithm, has shown to be competitive with other penalty based algorithms. The other two tested local search procedures did not improve significantly the final results. The convergence of the proposed algorithm will be carried out in the future. Other important issues, like the conditions for stopping the algorithm, will be analyzed.

Practical engineering problems, for example, those reported in [2], will be solved in the near future. We also aim to test our algorithms with a point-to-point search yet stochastic method, when solving the bound constrained subproblems.

Acknowledgments

The authors wish to thank two anonymous referees for their careful reading of the manuscript and their fruitful comments and suggestions.

References

[1] H.J.C. Barbosa, *A coevolutionary genetic algorithm for constrained optimization*, in *Proceedings of the 1999 Congress on Evolutionary Computation*, DOI:10.1109/CEC.1999.785466, Vol. 3. (1999) pp. 1605–611.

[2] H.J.C. Barbosa and A.C.C. Lemonge, *An adaptive penalty method for genetic algorithms in constrained optimization problems*, in *Frontiers in Evolutionary Robotics*, H. Iba (ed.) 34 pages. 2008 (ISBN: 978-3-902613-19-6) I-Tech Education Publ., Austria.

[3] D.P. Bertsekas, *Nonlinear Programming*, 2nd edn. Athena Scientific, Belmont, 1999.

[4] S.I. Birbil, *Stochastic Global Optimization Techniques*, Ph.D. diss., North Carolina State University, 2002.

[5] S.I. Birbil and S.-C. Fang, *An electromagnetism-like mechanism for global optimization*, Journal of Global Optimization, 25 (2003), pp. 263–282.

[6] E.G. Birgin, R. Castillo, and J.M. Martinez, *Numerical comparison of Augmented Lagrangian algorithms for nonconvex problems*. Computational Optimization and Applications, 31 (2004), pp. 31-56.

[7] E.G. Birgin, C.A. Floudas and J.M. Martinez, *Global minimization using an Augmented Lagrangian method with variable lower-level constraints*, Mathematical Programming, Ser. A, DOI:10.1007/s10107-009-0264-y.

[8] C.A. Coello Coello, *Use of a self-adaptive penalty approach for engineering optimization problems*, Computers in Industry 41 (2000), pp. 113–127.

[9] K. Deb, *An efficient constraint handling method for genetic algorithms*, Computer Methods in Applied Mechanics and Engineering, 186 (1998), pp. 311–338.

[10] E.D. Dolan and J.J. Moré, *Benchmarking optimization software with performance profiles*, Mathematical Programming, 91 (2002), pp. 201–213.

[11] R. Fourer, D.M. Gay, and B.W. Kernighan, *A modeling language for mathematical programming*, Management Science, 36 (1990), pp. 519–554.

[12] A.R. Hedar and M. Fukushima, *Heuristic pattern search and its hybridization with simulated annealing for nonlinear global optimization*, Optimization Methods and Software, 19 (2004) 291–308.

[13] A.R. Hedar and M. Fukushima, *Derivative-free filter simulated annealing method for constrained continuous global optimization*, Journal of Global Optimization, 35 (2006) 521–549.

[14] D. Karaboga and B. Basturk, *Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems*, *Lecture Notes In Artificial Intelligence*, (P. Melin et al. (eds.)), Vol. 4529 (2007), pp. 789–798.

[15] J.J. Liang, T.P. Runarsson, E. Mezura-Montes, M. Clerc, P.N. Suganthan, C.A.C. Coello, and K. Deb, *Problem definitions and evaluation criteria for the CEC2006 special session on constrained real-parameter optimization* 2006. (http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC-06/CEC06.htm)

[16] T.W. Liao, *Two hybrid differential evolution algorithms for engineering design optimization*, Applied Soft Computing 10 (2010). pp. 1188–1199.

[17] J.-L. Liu and J.-H. Lin, *Evolutionary computation of unconstrained and constrained problems using a novel momentum-type particle swarm optimization*, Engineering Optimization 39 (2007), pp. 287–305.

[18] H. Luo, X. Sun, and H. Wu, *Convergence properties of augmented Lagrangian methods for constrained global optimization*, Optimization Methods and Software, 23 (2008), pp. 763–778.

[19] Y.G. Petalas, K.E. Parsopoulos, and M.N. Vrahatis, *Memetic particle swarm optimization*, Annals of Operations Research, 156 (2007), pp. 99–127.

[20] A.M.A.C. Rocha and E.M.G.P. Fernandes, *Feasibility and dominance rules in the electromagnetism-like algorithm for constrained global optimization problems*, *Lecture Notes in Computer Science*, Computational Science and Its Applications (O. Gervasi et al. (eds.)), Vol. 5073 (2008), pp. 768–783.

[21] A.M.A.C. Rocha and E.M.G.P. Fernandes, *Implementation of the electromagnetism-like algorithm with a constraint-handling technique for engineering optimization problems*, in *2008 Eighth International Conference on Hybrid Intelligent Systems*, ISBN: 978-0-7695-3326-1, IEEE Computer Society, (2008), pp. 690–695.

[22] A.M.A.C. Rocha and E.M.G.P. Fernandes, *Self-adaptive penalties in the electromagnestism-like algorithm for constrained global optimization problems*, in *Proceedings of the 8th World Congress on Structural and Multidisciplinary Optimization*, CD 10 pages, Lisbon 2009.

[23] A.M.A.C. Rocha and E.M.G.P. Fernandes, *Hybridizing the electromagnetism-like algorithm with descent search for solving engineering design problems*, International Journal of Computer Mathematics, 86 (2009), pp. 1932–1946.

[24] T.P. Runarsson and X. Yao, *Stochastic ranking for constrained evolutionary optimization*, IEEE Transactions on Evolutionary Computation, 4 (2000), pp. 284–294.

[25] K. Sedlaczek and P. Eberhard, *Augmented Lagrangian particle swarm optimization in mechanism design*, Journal of System Design and Dynamics, 1 (2007), pp. 410–421.

[26] A.I.F. Vaz and L.N. Vicente, *A particle swarm pattern search method for bound constrained global optimization*, Journal of Global Optimization, 39 (2007), pp. 197–219.

[27] B.W. Wah and T. Wang, *Efficient and Adaptive Lagrange-Multipliers*, Journal of Global Optimization, 14 (1999), pp. 1–25.

[28] T. Wang and B.W. Wah, *Handling inequality constraints in continuous nonlinear global optimization*, in *Integrated Design and Process Science* (1996), pp. 267–274.

[29] A.E.M. Zavala, A.H. Aguirre, and E.R.V. Diharce, *Constrained optimization via particle evolutionary swarm optimization algorithm (PESO)*, in *GECCO'05*, (2005), pp. 209–216.