Feasibility and Dominance Rules in the Electromagnetism-like Algorithm 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
{arocha;emgpf}@dps.uminho.pt

Abstract

This paper presents the use of the feasibility and dominance (FAD) rules in a electromagnetism-like mechanism (EM) for the constraints handling in constrained global optimization. The FAD rules are easily incorporated in the EM algorithm: when two points are compared at a time and a selection is required, and when the direction of individual forces between two points has to be decided. Numerical experiments are presented, including a comparison with other methods recently reported in the literature.

1. Introduction

The problem that is addressed in the paper considers finding a global solution of a nonlinear optimization problem in the following form:

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in \Omega, \end{array} \tag{1}$$

where $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $g : \mathbb{R}^n \rightarrow \mathbb{R}^p$ and $h : \mathbb{R}^n \rightarrow \mathbb{R}^m$ are nonlinear continuous functions and the closed set Ω is defined by $\Omega = \{x \in \mathbb{R}^n : l \leq x \leq u\}$. We assume that the objective function f is nonconvex and possesses many local minima in the feasible region. This class of global optimization problems is very important and frequently encountered in engineering applications.

In the last decades, many algorithms have been proposed to solve problem (1). Probably the most extensively used in practice are stochastic-type algorithms.

In this paper, we are interested in the electromagnetism-like (EM) algorithm proposed in [2]. This is a population-based algorithm that 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. The EM algorithm was specifically designed for solving optimization problems with bound constraints [1, 2, 3]

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \Omega. \end{array} \tag{2}$$

We have been incorporating different strategies into the EM algorithm in order to improve the accuracy of the results as well as the efficiency of the overall algorithm. A pattern search method was introduced as a local search procedure and compared with the original EM algorithm [13]. A modification in the total force vector, that is used to move each point in the population, was tested and compared with other well-known heuristics such as the genetic algorithm, the particle swarm optimization (PSO) and an hybrid evolutionary algorithm (EA) [15]. The effect of different charge calculations has been analyzed and compared with PSO, EA and differential evolution methods [14].

In this work, the EM algorithm is extended for solving constrained optimization problems like (1). A previous extension of the EM algorithm to constrained problems relies on penalty and barrier methods [1]. In these methods, constrained problems like (1) can be transformed into the bound constrained form (2) so that the EM algorithm can be directly applied. Tests therein reported involve a set of 5 problems selected from [7] and do not include a comparison with other heuristics.

The extension herein presented for handling the constraints $g(x) \leq 0$ and $h(x) = 0$ considers a very simple heuristic consisting of three rules, designated by feasibility and dominance (FAD). The reader is referred to [10] and [21] for details. In [10], FAD is used in an artificial bee colony (ABC) optimization algorithm context, whereas in [21], the authors incorporate the heuristic FAD into a particle evolutionary swarm optimization algorithm, therein denoted by PESO.

The paper is organized as follows. In Section 2 we describe the EM algorithm. Section 3 resumes the FAD rules for constraints handling and Section 4 presents the modifications that were introduced in the EM algorithm. Section 5 contains the results of all the numerical experiments, including comparisons with other methods, and we conclude the paper in Section 6.

2. The electromagnetism-like mechanism for bound constrained problems

In this section we describe the EM algorithm for solving problem (2). The algorithm starts with a population of randomly generated points from the feasible set Ω. Analogous to electromagnetism, each point is a charged particle that is released to the space. The charge of each point is related to the objective function value and determines the magnitude of attraction of the point over the others in the population. The better the objective function value, the higher the magnitude of attraction. The charges are used to find a direction for each point to move in subsequent iterations. The regions that have higher attraction will signal other points to move towards them. In addition, a repulsion mechanism is also introduced to explore new regions for even better solutions. A more detailed explanation follows.

First, a population of p_{size} points is randomly generated from the feasible region Ω. Let x^i be the ith. point of the population. Then each coordinate of a point is denoted as x^i_k ($k = 1, \ldots, n$) and computed by

$$x^i_k = l_k + \lambda(u_k - l_k) \tag{3}$$

where $\lambda \sim U(0,1)$. All points are evaluated (the corresponding objective function values are computed) and compared in order to identify the best point, x^{best}.

Then the total force vector F^i exerted on each point x^i by the other $p_{size} - 1$ points is calculated by adding the individual component forces, F^i_j, between any pair of points x^i and x^j of the population. According to the electromagnetism theory, each individual force is inversely proportional to the square of the distance between the two points and directly proportional to the product of their charges.

Since the charge q^i of point x^i determines the power of attraction or repulsion for that point, the charge is computed according to the objective function value by

$$q^i = \exp\left(\frac{-n(f(x^i) - f(x^{best}))}{\sum_{j=1}^{p_{size}}(f(x^j) - f(x^{best}))}\right), i = 1, \ldots, p_{size}. \tag{4}$$

In this way the points that have better objective function values possess higher charges. This is a scaled distance of the function value at x^i to the function value of the best point in the population. Different approaches for charge evaluations may be adopted, see for example [5, 9, 14].

Since the charges (4) are all positive, the direction of a force F^i_j depends on the objective function values at x^i and x^j. If $f(x^j) < f(x^i)$ then the point x^j attracts x^i and the direction of the force should be $\overrightarrow{x^i x^j}$, whereas if $f(x^j) \geq f(x^i)$ then x^j repels x^i and the direction of the force is $\overrightarrow{x^j x^i}$. Thus,

$$F^i_j = \begin{cases} (x^j - x^i)\frac{q^i q^j}{\|x^j - x^i\|^3} & \text{if } f(x^j) < f(x^i) \\ (x^i - x^j)\frac{q^i q^j}{\|x^j - x^i\|^3} & \text{otherwise} \end{cases} \tag{5}$$

for $j \neq i$. The total force vector F^i exerted on each point x^i is then calculated by adding the individual component forces, F^i_j,

$$F^i = \sum_{j \neq i}^{p_{size}} F^i_j, \quad i = 1, \ldots, p_{size}.$$

Finally, this total force vector, F^i, is used to move the point x^i in the direction of the force by a random step length $\lambda \sim U(0,1)$ as follows

$$x^i_k = \begin{cases} x^i_k + \lambda\frac{F^i_k}{\|F^i\|}(u_k - x^i_k) & \text{if } F^i_k > 0 \\ x^i_k + \lambda\frac{F^i_k}{\|F^i\|}(x^i_k - l_k) & \text{otherwise} \end{cases} \tag{6}$$

for each k ($k = 1, 2, \ldots, n$) and for $i = 1, \ldots, p_{size}$ and $i \neq best$. The best point, x^{best}, is not moved and is carried out to the subsequent iteration.

We would like to remark that, as shown in (6), feasibility is maintained by using the normalized total force vector and scaling it with the allowed range of movement towards the lower bound l_k, or the upper bound u_k, of the set Ω.

This EM algorithm was indeed designed to generate and iteratively move points in the population that are feasible in a bound constrained problem context.

Like other hybrid population-based algorithms, the EM algorithm incorporates a local search procedure in order to improve the accuracy of the solution. Although this local refinement could be applied to all points in the population, it has been shown that when the local procedure is applied only to the best point, the accuracy of the results still improves and the number of function evaluations is not extremely high [2].

Thus, the local refinement is applied coordinate by coordinate to the best point in the population as described below. Based on a parameter δ, the maximum feasible step length $s_{max} = \delta(\max_k(u_k - l_k))$ is computed. Then, for each coordinate k, a random number λ between 0 and 1 is selected as a step length and a new point is calculated along that direction. The new point is compared with the old one. If an improvement is observed, within nit_{max} iterations, the new point replaces the best point and the search continues with the next coordinate.

The EM algorithm for bound constrained optimization can be described as shown below.

Algorithm 1 *(EM algorithm)*
Given l, u, p_{size}, nfe_{max}, nit_{max}, δ and set $k = 1$

`step 1` *Randomly generate a population of p_{size} points in the set Ω, using (3)*

`step 2` *Evaluate the points and compare them to identify the best point x^{best}*

`step 3` *If termination criterion is satisfied, stop*

`step 4` *Compute the charges of all points by using (4)*

`step 5` *Compute the forces and decide their directions based on function values using (5)*

`step 6` *Compute the total force vector for each point in the population*

`step 7` *Move all points, except the best point, in the set Ω according to (6). Evaluate the points and compare them to identity the best point.*

`step 8` *Perform a local search about the best point to look for an improvement*

`step 9` *set $k = k + 1$ and go to step 3.*

At this moment there are two issues in the Algorithm 1 that need to be further explained. One involves the evaluation of the points, computing the corresponding objective function values, and their comparison, and the other is concerned with the termination criterion.

We remark here that in `step 2`, `step 5`, `step 7` and `step 8` of the algorithm, points in the population are compared solely based on function values, in order to select the best point. Thus, in the context of solving problem (2), the rule that is operated is summarized as follows:

- when comparing two points at a time, select the one with lowest function value.

Finally, the condition that is used to terminate the algorithm considers a limit on the number of function evaluations, nfe_{max}.

3. Heuristic for handling constraints

Most stochastic methods for global optimization are developed primarily for unconstrained problems. Then they are extended to constrained optimization problems by modifying the original procedures or by using penalty functions. The most common approach to solving constrained problems is based on penalty functions. Penalty terms are added to the objective function to penalize constraints violation. The penalty techniques transform the constrained problem into an unconstrained problem by penalizing f when constraints are violated and then minimizing the penalty function using methods for unconstrained problems. There are two main classes of penalty terms: i) a stationary penalty term that uses fixed penalty parameters throughout the iterative process and depends on constraints violation [1, 4, 12]; ii) non-stationary penalty term that changes dynamically and depends on the iteration counter [11, 17]. Barrier methods use a similar framework although they require initial feasible points and maintain feasibility along the iterative process [1]. Recently, filled function methods have also been used for solving constrained global optimization [20]. Other constraint handling techniques treat constrained optimization as multi-objective optimization, where constraints are regarded as objective functions. Adapted from the multi-objective optimization, a relaxed dominance concept is used in [19] to assess progress towards feasibility and optimality. Another approach that is able to balance dominance between objective and penalty functions in a stochastic manner, known as stochastic ranking, is proposed and tested in [16].

The approach herein adopted for handling the inequality and equality constraints of problem (1) relies on a simple heuristic consisting of three rules denoted by feasibility and dominance (FAD) rules [10, 21].

We assume that all equality constraints are converted into inequality constraints, $|h_j| \leq \epsilon$, $j = 1, \ldots, m$. (For example, $\epsilon = 0.001$.) For simplicity the problem (1) is rewritten as

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & c(x) \leq 0 \\ & x \in \Omega, \end{array} \tag{7}$$

where the vector of inequality constraints of the problem is defined by:

$$c(x) = (g_1(x), \ldots, g_p(x), |h_1(x)| - \epsilon, \ldots, |h_m(x)| - \epsilon)\,,$$

and the constraints violation is measured by the l_2 norm of a vector, $\|\pi(x)\|_2$, where

$$\pi_j(x) = \max\{0, c_j(x)\},\ j = 1, \ldots, p + m\,.$$

In the sequel, a point x^i with $\|\pi(x^i)\|_2 = 0$ is feasible, whereas if $\|\pi(x^i)\|_2 > 0$ then the point is infeasible. When using the FAD rules, two points are compared at a time, and if

FAD_1 both points are feasible, select the one with lowest function value;

FAD_2 one point is feasible and the other is infeasible, select the feasible point;

FAD_3 both points are infeasible, select the one with lowest constraints violation.

4. FAD-EM algorithm for constrained problems

The modifications that were introduced in the EM algorithm for handling the inequality constraints $c(x) \leq 0$ in problem (7) are justified and presented below.

4.1. Evaluation of the points

The first modification is concerned with the evaluation of the points in the population. In the constrained problem (7) context, when the evaluation of the points is required, the objective function value and the constraints violation are computed for each point in the population. This pair of information $(f(x^i), \|\pi(x^i)\|_2)$ defines the point fitness.

4.2. The direction of forces

To decide the direction of the individual forces F_j^i, the points x^j and x^i are compared. We say that x^j is better that x^i if x^j is the selected point according to the FAD rules and consequently x^i is attracted to x^j. Conversely, x^i is repelled by x^j if x^i is the selected point according to the FAD rules. Hence, the new conditions are summarized as follows:

$$F_j^i = \begin{cases} (x^j - x^i)\frac{q^i q^j}{\|x^j - x^i\|^3} & \text{if } x^j \text{ is better than } x^i \\ (x^i - x^j)\frac{q^i q^j}{\|x^j - x^i\|^3} & \text{otherwise} \end{cases} \quad (8)$$

for $j \neq i$.

4.3. Charges calculation

Another important issue is related with the charges calculation. Maintaining a similar approach to that of the original EM algorithm, where only objective function values are compared, then

$$q^i = \exp\left(\frac{-n|f(x^i) - f(x^{best})|}{\sum_{j=1}^{p_{size}} |f(x^j) - f(x^{best})|}\right), i = 1, \ldots, p_{size}. \quad (9)$$

We now have to use absolute values because there is no guarantee that the best point in the population has the smallest function value. With (9), the charges q^i ($i = 1, \ldots, p_{size}$) remain positive and the direction of each individual force F_j^i relies on x^i and x^j fitness as described above.

However, when constraints are present in the problem like (7), two objectives have to be achieved in the way to the solution: one is to minimize the objective function f and the other is to minimize constraints violation $\|\pi\|_2$. Since a solution must be a feasible point, the minimization of $\|\pi\|_2$ is indeed the most important. Thus, the charge of a point could be related to its constraints violation in the sense that feasible points are the only candidates to solutions. The better the constraints violation, the higher the magnitude of attraction. Our proposal here considers the charge calculation as a scaled measure of feasibility

$$q^i = \exp\left(\frac{-n\|\pi(x^i)\|_2^2}{\sum_{j=1}^{p_{size}} |f(x^j) - f(x^{best})|}\right), i = 1, \ldots, p_{size}. \quad (10)$$

In this way, the feasible points in the population have unit charges, whereas the infeasible points have smaller charges and the larger the violation the smaller the charge. This means that feasible points have the ability to attract infeasible points.

4.4. The local search

When a local refinement is applied coordinate by coordinate to the best point, two points are to be compared to see if an improvement is observed. The fitness of the points are compared according to the FAD rules.

4.5. The algorithm

The algorithm for solving constrained global optimization problems that uses the electromagnetism-like mechanism and the feasibility and dominance rules for constraints handling is as follows:

Algorithm 2 *(FAD-EM algorithm)*
Given l, u, p_{size}, nfe_{max}, nit_{max}, δ and set $k = 1$

`step 1` *Randomly generate a population of p_{size} points in the set Ω, using (3)*

`step 2` *Evaluate the points and apply the* `FAD rules` *to identify the best point x^{best}*

`step 3` *If termination criterion is satisfied, stop*

`step 4` *Compute the charges of all points by using (9) or (10)*

`step 5` *Compute the forces and apply the* `FAD rules` *to decide the direction of the force using (8);*

`step 6` *Compute the total force vector for each point in the population*

`step 7` *Move all points, except the best point, in the set Ω according to (6). Evaluate the points and apply the* `FAD rules` *to identify the best point*

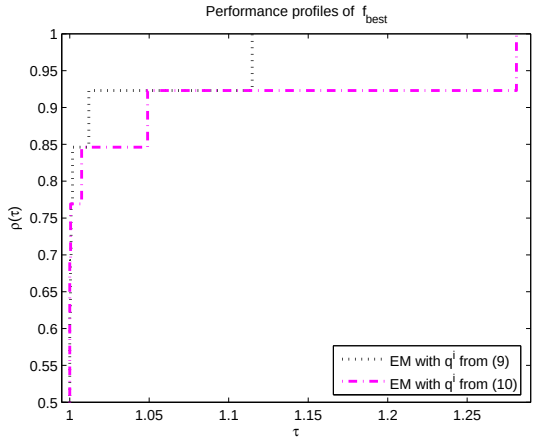
`step 8` *Perform a local search about the best point and apply the* `FAD rules` *to look for an improvement*

`step 9` *set $k = k + 1$ and go to step 3.*

Table 1. Details of the 13 constrained problems.

P	Type of f	n	p	m	n_{active}
g01	quadratic	13	9	0	6
g02	general	20	2	0	1
g03	polynomial	10	0	1	1
g04	quadratic	5	6	0	2
g05	cubic	4	2	3	3
g06	cubic	2	2	0	2
g07	quadratic	10	8	0	6
g08	general	2	2	0	0
g09	general	7	4	0	2
g10	linear	8	6	0	6
g11	quadratic	2	0	1	1
g12	quadratic	3	1	0	0
g13	general	5	0	3	3

Figure 1. Performance profiles of f_{best} over 30 runs.



5. Numerical experiments

To evaluate the performance of the herein proposed EM algorithm for constrained problems a set of 13 benchmark problems, described in full detail in the Appendix of [21], is used. The problems are known as g01, g02, ..., g13 [10, 16, 21]. We remark that g02, g03, g08 and g12 are maximization problems while the others are minimization ones.

The algorithm is coded in the C programming language and it contains an interface to connect to AMPL so that the problems coded in AMPL could be easily solved [8]. AMPL is a mathematical programming language that allows the codification of optimization problems in a powerful and easy to learn language. The set of coded problems may be obtained from the first author upon request.

The fixed values for the parameters in the EM algorithm are as follows: $\delta = 0.001$ and $nit_{max} = 10$.

Details of the selected problems are listed in Table 1, where n represents the number of variables, p and m are the number of inequality and equality constraints respectively and n_{active} is the number of active constraints at the solution.

Initially, we study the two versions of the EM algorithm for constrained optimization based on the FAD rules, which differ in the computation of the charges, see (9) and (10) respectively. We select the best version and report the results obtained with the selected 13 benchmark problems, using the best, the average and the worst objective function values obtained after 30 independent runs each starting from a random population with different seeds. Here, we consider a population size of 50 points and use a limit of 350000 function evaluations to terminate the algorithm.

We then compare our results with other methods recently reported in the literature. For a set of comparisons, we repeat the experiments with different conditions: $p_{size} = 40$ and a limit of 240000 function evaluations.

5.1. Comparison of the two versions

The comparison of the two versions of the FAD-EM algorithm which differ in the computation of the charges, herein designated by FAD-EM-(9) and FAD-EM-(10), corresponding to (9) and (10) respectively, is based on the Dolan and Moré's performance profile approach as outline in [6]. The performance profiles give, for every $\tau \geq 1$, the proportion $\rho(\tau)$ of test problems on which each algorithm under comparison has a performance within a factor τ of the best. An explanation of our implementation of this approach follows.

Let $\mathcal{P}$ be the set of all problems and $\mathcal{S}$ the set of solvers used in the comparative study. Let $m_{(p,s)}$ be the performance metric found by solver $s \in \mathcal{S}$ on problem $p \in \mathcal{P}$ after a fixed number of function evaluations. The metric that is used when the performance assessment is on the best result obtained by the solvers is

$$m_{(p,s)} = f_{best\,(p,s)}, \tag{11}$$

where $f_{best\,(p,s)}$ is the best function value found by solver s on problem p, after $nfe_{max} = 350000$ function evaluations, and over 30 runs. However, if the performance assessment is on the average result obtained by the solvers then $f_{avg\,(p,s)}$ should de used in the metric (11).

The performance ratio adopted for this comparative

Figure 2. Performance profiles of f_{best} for $1 \leq \tau \leq 1.002$.

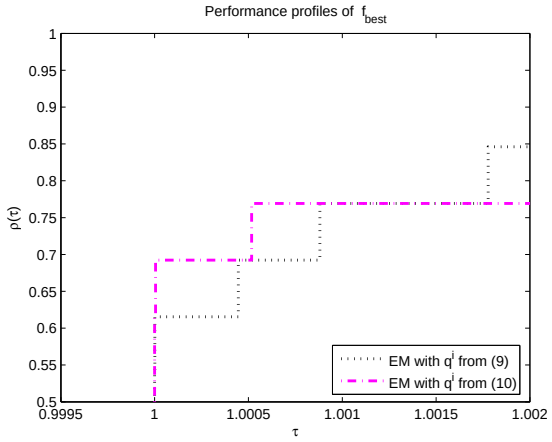


Figure 3. Performance profiles of f_{avg} over 30 runs.

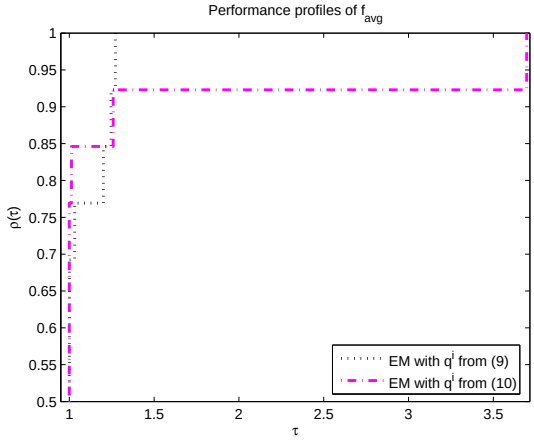


Table 2. Average number of function evaluations for the fixed accuracy (12).

P	FAD-EM-(9)	FAD-EM-(10)
g03	323221	303962
g11	270250	271220
g12	529	443

Figure 4. Performance profiles of f_{avg} for $1 \leq \tau \leq 1.02$.

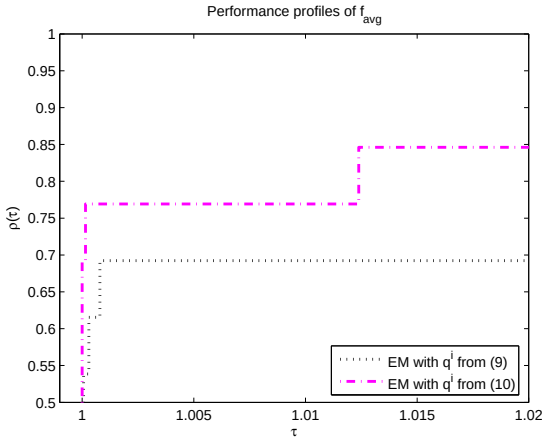


Table 3. Best, average, worst and standard deviation of the results.

P	f_{global}	f_{best}	f_{avg}	f_{worst}
g01	-15.000000	-14.999805	-14.590470	-12.500515
		sd:	0.770339	
g02	0.803619	0.444939	0.427705	0.377406
		sd:	0.012574	
g03	1.000500	1.002983	0.999661	0.996438
		sd:	0.001663	
g04	-30665.539	-30642.600	-30591.897	-30521.434
		sd:	29.849753	
g05	5126.497	5135.818	5338.583	6117.660
		sd:	273.027370	
g06	-6961.814	-6953.411	-6942.915	-6933.264
		sd:	4.250238	
g07	24.30621	27.46692	53.08965	103.5507
		sd:	19.236274	
g08	0.095825	0.095825	0.095825	0.095825
		sd:	0.000000	
g09	680.630	681.777	689.896	701.344
		sd:	5.104498	
g10	7049.248	7187.181	9492.735	16395.51
		sd:	1671.593930	
g11	0.749900	0.749001	0.749079	0.749346
		sd:	0.000086	
g12	1.000000	1.000000	1.000000	1.000000
		sd:	0.000000	
g13	0.053942	0.063196	2.083421	20.784428
		sd:	5.010076	

Table 4. Settings used in the comparisons.

p_{size}	nfe_{max}	version	methods	references
50	350 000	FAD-EM$_{50}$	PSO, PESO	[21]
			TCPSO, SR	[16, 21]
40	240 000	FAD-EM$_{40}$	DE, ABC	[10]

study is

$$r_{(p,s)} = \frac{m_{(p,s)}}{\min\left\{m_{(p,s)} : s \in \mathcal{S}\right\}}$$

and the overall assessment of the performance of a particular solver s is defined by

$$\rho_s(\tau) = \frac{n_{P_\tau}}{n_P}$$

where n_P is the number of problems in the set $\mathcal{P}$ and n_{P_τ} is the number of problems in the set such that the performance ratio $r_{(p,s)}$ is less than or equal to τ for solver s. Hence, $\rho_s(\tau)$ is the probability (for solver $s \in \mathcal{S}$) that the performance ratio $r_{(p,s)}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio.

Using the performance profile plot one can compare how well a solver can estimate the optimum relative to the others. The value of $\rho_s(1)$ gives the probability that the solver s will win over the others in the set. However, for large values of τ, the $\rho_s(\tau)$ measures the solver robustness.

Figure 1 contains the performance profiles of f_{best} for the two versions in comparison. Figure 2 contains details of the plots for values of τ near 1. They show that the version based on the charge calculation (10) is superior on the best performance. For $\tau = 1$, this version wins in 70% of the problems over the other, whereas the version based on charge (9) is better for values of τ greater than 1.0018. When the average performance is considered, Figures 3 and 4 show that the version FAD-EM-(10) outperforms the other version for small values of τ and is exceeded by FAD-EM-(9) for $\tau > 1.3$.

To compare further the two versions of the FAD-EM algorithm, we record in Table 2 the average number of function evaluations needed for finding a feasible solution satisfying a fixed accuracy level, herein defined by

$$|f(x^{best}) - f_{global}| \leq 0.0001 \qquad (12)$$

over the 30 runs. The table reports only the problems where condition (12) was verified before the algorithm has terminated due to the 350000 function evaluations. The performances are competitive.

5.2. Detailed results of FAD-EM algorithm

We now report detailed results of the best version - FAD-EM-(10). The experiments are based on a population size of 50 points and the algorithm is terminated after 350000 function evaluations. Table 3 contains the best, average and worst function values obtained over the 30 runs. The table also shows the standard deviation (sd) of the obtained function values in a row below the other results. The column headed with f_{global} lists the most recent known global solutions (see http://www.ntu.edu.sg/home/EPNSugan/).

Table 5. Comparison of the best results, over the 30 runs, with PSO and PESO.

P	f_{global}	FAD-EM$_{50}$	PSO	PESO
g01	-15.000000	-14.999886	-15.000000	-15.000000
g02	0.803619	0.439669	0.669158	0.792608
g03	1.000500	1.002100	0.993930	1.005010
g04	-30665.539	-30628.932	-30665.539	-30665.539
g05	5126.497	5126.703	5126.484	5126.484
g06	-6961.814	-6957.004	-6961.814	-6961.814
g07	24.30621	35.18561	24.37015	24.30692
g08	0.095825	0.095825	0.095825	0.095825
g09	680.630	686.858	680.630	680.630
g10	7049.248	7539.071	7049.381	7049.459
g11	0.749900	0.749000	0.749000	0.749000
g12	1.000000	1.000000	1.000000	1.000000
g13	0.053942	0.056688	0.085655	0.081498

5.3. Comparison with other heuristics

To demonstrate the performance of the new EM type algorithm for general constrained global optimization problems, we carried out a comparison with six very well-known and recent methods: the PSO (standard particle swarm optimization), PESO (particle evolutionary swarm optimization), TCPSO (the Toscano and Coello's PSO), SR (stochastic ranking), DE (differential evolution) and ABC (artificial bee colony).

To be able to compare our results with those of recent publications, we use the same set of 13 problems. The results obtained by PSO, PESO and TCPSO are reported in [21] and the ones obtained by SR can be seen in [16, 21]. For a fair comparison, we use the population size and the maximum number of objective function evaluations similar to the therein reported. Table 4 contains these settings. To compare with the DE and ABC algorithms, we change the settings and the DE and ABC results are reported in [10].

Table 5 reports the best function value obtained after 30 runs. We denote our version as FAD-EM$_{50}$, since it is based on a population of 50 points and is to be compared with two specific versions of the particle swarm optimization algorithm: PSO and PESO. We remark that the table summa-

Table 6. Comparison of the average results, over the 30 runs, with PSO and PESO.

P	f_{global}	FAD-EM$_{50}$	PSO	PESO
g01	-15.000000	-12.151068	-14.710417	-15.000000
g02	0.803619	0.414594	0.419960	0.721749
g03	1.000500	0.998865	0.764813	1.005006
g04	-30665.539	-30589.882	-30665.539	-30665.539
g05	5126.497	5336.962	5135.973	5129.178
g06	-6961.814	-6943.954	-6961.814	-6961.814
g07	24.30621	196.253	32.40727	24.37125
g08	0.095825	0.075382	0.095825	0.095825
g09	680.630	698.440	680.630	680.630
g10	7049.248	11947.504	7205.500	7099.101
g11	0.749900	0.749056	0.749000	0.749000
g12	1.000000	1.000000	0.998875	1.000000
g13	0.053942	1.671718	0.569358	0.626881

Table 7. Comparison of the average results, over the 30 runs, with TCPSO and SR.

P	f_{global}	FAD-EM$_{50}$	TCPSO	SR
g01	-15.000000	-12.151068	-15.000000	-15.000000
g02	0.803619	0.414594	0.790406	0.781975
g03	1.000500	0.998865	1.003814	1.000000
g04	-30665.539	-30589.882	-30665.500	-30665.539
g05	5126.497	5336.962	5461.081	5128.881
g06	-6961.814	-6943.954	-6961.810	-6875.940
g07	24.306201	196.253	25.35577	24.374
g08	0.095825	0.075382	0.095825	0.095825
g09	680.630	698.440	680.852	680.656
g10	7049.248	11947.504	7560.048	7559.192
g11	0.749900	0.749056	0.750107	0.750000
g12	1.000000	1.000000	1.000000	1.000000
g13	0.053942	1.671718	1.716426	0.057006

rizes the results of the version FAD-EM-(9), that are slightly different from those listed in Table 3. The results obtained for the mean best function values are reported in Table 6.

From Table 5 we may conclude that FAD-EM is better than PSO in problems g03 and g13, is better than PESO only in the problem g13, while in the problems g08, g11 and g12 is competitive with both PSO and PESO.

Table 6 shows that FAD-EM is clearly better than PSO in problems g03 and g12, but PSO is better in all the remaining problems of the set. PESO is also better than FAD-EM in all problems, except in problem g12 that is competitive. We note that FAD-EM worst values are really far from the best values, giving a poor performance on the average values.

Table 7 compares our average results with those of other algorithms: TCPSO and SR. TCPSO refers to the Toscano and Coello's PSO that incorporates a constraints handling mechanism into the standard PSO algorithm [18], and SR denotes stochastic ranking [16]. This is an evolution strategy enhanced with a stochastic constraints handling mechanism. FAD-EM average results are better than TCPSO results in problems g05, g11 and g13, and are competitive in g12. FAD-EM is better than SR only in the problem g11 and is competitive in problem g12.

In this part of the paper we compare the FAD-EM algorithm with DE and ABC algorithms. The best function value and the mean best function values are listed in Tables 8 and 9 respectively. The version under comparison, denoted by FAD-EM$_{40}$, considers the formula (9) to evaluate the charges and is based on a population of 40 points. The algorithm terminates when the number of function evaluations exceeds 240000.

From Table 8, we note that FAD-EM best results are better than DE results in problems g02, g03, g11 and g13, and are competitive in problems g08 and g12. When comparing with ABC results, FAD-EM wins in the problems g03, g11 and g13. Both algorithms are competitive in problems g08 and g12.

Table 9 contains the average results of FAD-EM, DE and ABC obtained when solving the 13 problems. In the table, "-" means that no feasible solutions were found. Although the average results are competitive in problems g08 and g12, FAD-EM is clearly better only in the problem g11.

Table 8. Comparison of the best results, over the 30 runs, with DE and ABC.

P	f_{global}	FAD-EM$_{40}$	DE	ABC
g01	-15.000000	-14.999830	-15.000	-15.000
g02	0.803619	0.491136	0.472	0.803598
g03	1.000500	1.003318	1.000	1.000
g04	-30665.539	-30605.032	-30665.539	-30665.539
g05	5126.497	5126.672	5126.484	5126.484
g06	-6961.814	-6953.069	-6954.434	-6961.814
g07	24.30621	25.81694	24.306	24.330
g08	0.095825	0.095825	0.095825	0.095825
g09	680.630	681.792	680.630	680.634
g10	7049.248	7739.708	7049.248	7053.904
g11	0.749900	0.749003	0.752	0.750
g12	1.000000	1.000000	1.000	1.000
g13	0.053942	0.095896	0.385	0.760

Table 9. Comparison of the average results, over the 30 runs, with DE and ABC.

P	f_{global}	FAD-EM$_{40}$	DE	ABC
g01	-15.000000	-14.503678	-14.555	-15.000
g02	0.803619	0.427662	0.665	0.792412
g03	1.000500	0.999822	1.000	1.000
g04	-30665.539	-30573.439	-30665.539	-30665.539
g05	5126.497	5303.154	5264.270	5185.714
g06	-6961.814	-6936.598	-	-6961.813
g07	24.30621	66.57473	24.310	24.473
g08	0.095825	0.095825	0.095825	0.095825
g09	680.630	689.423	680.630	680.640
g10	7049.248	9476.079	7147.334	7224.407
g11	0.749900	0.749074	0.901	0.750
g12	1.000000	1.000000	1.000	1.000
g13	0.053942	1.312976	0.872	0.968

5.4. Comparison with results in Birbil's thesis [1]

To further examine the performance of the proposed feasibility and dominance EM algorithm, we carried out one more comparison. The results reported in [1], where penalty and barrier methods are incorporated into the EM algorithm for the constraints handling, are compared with FAD-EM results. In these comparisons the termination criterion in Algorithm 2 limits the number of iterations, herein denoted by nEM_{max}, instead of the number of function evaluations. The Table 10 contains the details of the 5 tested problems. For a fair comparison we use the conditions reported in [1]: $\delta = 0.01$ in the local procedure of the EM algorithm; 10 runs were made for each problem; the size of the population and the maximum number of FAD-EM iterations are as listed below

- TP1 – $p_{size} = 30$; $nEM_{max} = 75$
- TP2 – $p_{size} = 40$; $nEM_{max} = 100$
- TP3 – $p_{size} = 20$; $nEM_{max} = 50$
- TP4 – $p_{size} = 30$; $nEM_{max} = 50$
- TP5 – $p_{size} = 20$; $nEM_{max} = 75$.

The results are reported in Table 11. We use TP to identify the problem; ALG to identify the algorithm; P means penalty method, B barrier method, and EM refers to the FAD-EM algorithm that uses the formula (9) for the charges.

Table 10. Details of the 5 constrained problems in [1].

P	Type of f	n	p	m
TP1	quadratic	5	6	0
TP2	quadratic	6	6	0
TP3	linear	2	2	0
TP4	general	3	1	0
TP5	general	4	2	0

Table 11. Comparison with the results in [1].

TP	ALG	f_{global}	f_{best}	f_{avg}	nfe
1	P [1]	-30665.539	-30563.285	-30374.467	2534
	B [1]		-30596.784	-30447.682	2264
	EM		-30399.961	-30168.944	2632
2	P [1]	-310	-297.7095	-293.9035	4311
	B [1]		-307.2018	-297.8254	3969
	EM		-297.2020	-295.190	4577
3	P [1]	-5.508013	-5.5036	-5.4256	886
	B [1]		-5.4756	-5.4245	885
	EM		-5.500585	-5.451045	1089
4	P [1]	-83.254	-83.096	-82.5450	1597
	B [1]		-83.1574	-81.9877	1347
	EM		-83.063359	-82.481219	1684
5	P [1]	-5.7398	-5.6450	-5.4857	1092
	B [1]		-5.6346	-5.0523	1251
	EM		-5.687241	-5.645319	1812

6. Final remarks

This paper presents a new EM type algorithm for solving constrained global optimization problems. The technique used for the constraints handling relies on a very simple heuristic known as the feasibility and dominance rules. These rules are easily incorporated into the EM algorithm whenever

- the best point in the population has to be identified;
- the direction of individual forces between pair of points in the population has to be defined;
- a local refinement is applied to the best point and an improvement has to be assessed.

The preliminary results reported in the paper seem to show that this approach is competitive with other algorithms, although some modifications are in progress to improve accuracy. Further testing with a different set of constrained prob-

lems [4] and engineering problems like the ones reported in [12, 19] are required.

Since the charges calculation seems a crucial issue in the electromagnetism-like mechanism, future developments will focus on a new approach so that a balance between the objective function value and the constraints violation is achieved.

References

[1] S. I. Birbil. *Stochastic global optimization techniques*. PhD thesis, North Carolina State University, 2002.

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

[3] S. I. Birbil, S.-C. Fang, and R.-L. Sheu. On the convergence of a population-based global optimization algorithm. *Journal of Global Optimization*, 30:301–318, 2004.

[4] E. G. Birgin, C. A. Floudas, and J. M. Martínez. Global minimization using an augmented Lagrangian method with variable lower-level constraints. November 2006.

[5] D. Debels, B. DeReyck, R. Leus, and M. Vanhoucke. A hybrid scatter search electromagnetism meta-heuristic for project scheduling. *European Journal of Operational Research*, 169:638–653, 2006.

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

[7] C. A. Floudas, P. M. Pardalos, C. S. Adjiman, W. R. Esposito, Z. H. Gumus, S. T. Harding, J. L. Klepeis, C. A. Meyer, and C. A. Schweiger. *Handbook of test problems in local and global optimization*. Kluwer Academic Publishers, Vol. 33, 1999.

[8] R. Fourer, D. M. Gay, and B. W. Kernighan. A modeling language for mathematical programming. *Management Science*, 36(5):519–554, 1990.

[9] P. Kaelo and M. M. Ali. Differential evolution algorithms using hybrid mutation. *Computational Optimization and Applications*, 37:231–246, 2007.

[10] D. Karaboga and B. Basturk. *Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems*, volume 4529 of *LNAI*, pages 789–798. Springer-Verlag, IFSA 2007 edition, 2007.

[11] 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(3):287–305, 2007.

[12] K. E. Parsopoulos and M. N. Vrahatis. *Unified particle swarm optimization for solving constrained engineering optimization problems*, volume 3612 of *LNCS*, pages 582–591. Springer-Verlag, ICNC 2005 edition, 2005.

[13] A. M. A. C. Rocha and E. M. G. P. Fernandes. A modified electromagnetism-like algorithm based on a pattern search method. ECC 2007 (to appear), 2008.

[14] A. M. A. C. Rocha and E. M. G. P. Fernandes. On charge effects to the electromagnetism-like algorithm. (paper submitted to EurOPT 2008), January 2008.

[15] A. M. A. C. Rocha and E. M. G. P. Fernandes. Performance profile assessment of electromagnetism-like algorithms for global optimization. IeCCS 2007 (to appear), 2008.

[16] T. P. Runarsson and X. Yao. Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on Evolutionary Computation*, 4(3):284–294, 2000.

[17] M.-J. Tahk, H.-W. Woo, and M.-S. Park. A hybrid optimization method of evolutionary and gradient search. *Engineering Optimization*, 39(1):87–104, 2007.

[18] G. Toscano and C. Coello. A constraint-handling mechanism for particle swarm optimization. In *Proceedings of the 2004 Congress on Evolutionary Computation*, 2004.

[19] A. I. F. Vaz and E. M. G. P. Fernandes. Optimization of nonlinear constrained particle swarm. *Technological and Economic Development of Economy*, 12(1):30–36, 2006.

[20] Z. Y. Wu, F. S. Bai, H. W. J. Leee, and Y. J. Yang. A filled function method for constrained global optimization. *Journal of Global Optimization*, 39:495–507, 2007.

[21] 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.