Extensions of Firefly Algorithm for Nonsmooth Nonconvex Constrained Optimization Problems

Rogério B. Francisco[1], M. Fernanda P. Costa[1], Ana Maria A. C. Rocha[2]

[1] Centre of Mathematics, University of Minho, Portugal

rbf@estgf.ipp.pt; mfc@math.uminho.pt

[2] Algoritmi Research Centre, University of Minho, Portugal

arocha@dps.uminho.pt

Abstract. Firefly Algorithm (FA) is a stochastic population-based algorithm based on the flashing patterns and behavior of fireflies. Original FA was created and successfully applied to solve bound constrained optimization problems. In this paper we present extensions of FA for solving nonsmooth nonconvex constrained global optimization problems. To handle the constraints of the problem, feasibility and dominance rules and a fitness function based on the global competitive ranking, are proposed. To enhance the speed of convergence, the proposed extensions of FA invoke a stochastic local search procedure. Numerical experiments to validate the proposed approaches using a set of well know test problems are presented. The results show that the proposed extensions of FA compares favorably with other stochastic population-based methods.

Keywords: Firefly algorithm, Constrained Global Optimization, Stochastic Ranking.

1 Introduction

In the last decades, different methods have been developed in order to solve a wide range of different kind of optimization problems. Metaheuristics are an important class of contemporary global optimization algorithms, computational intelligence and soft computing. The observation and study of nature and behavior of some living species have been served as inspiration for the development of new methods. A subset of metaheuristics, often referred to as swarm intelligence based algorithms, have been developed by mimicking the so-called swarm intelligence characteristics of biological agents such as birds, fish, humans among others. Swarm Intelligence belongs to an artificial intelligence subject that became increasingly popular over the last decade [1]. The three main purposes of metaheuristics are: to solve problems with low computational time, to solve large dimensional problems, and to obtain robust algorithms. In fact, metaheuristics are the most used stochastic optimization algorithms. In recent years, metaheuristic algorithms have emerged as global search approaches

used for solving complex optimization problems. The most popular metaheuristic methods are Genetic Algorithm (GA) [2], Ant Colony Optimization [3], Particle Swarm Optimization (PSO) [4], Harmony Search [5] and Firefly Algorithm (FA) [6]. All of them are metaheuristic population-based methods. The FA, initially proposed by Yang, is one of the new metaheuristic techniques inspired by the flashing behavior of fireflies and was designed for solving bound constrained optimization (BCO) problems. This algorithm is inspired by the nocturnal luminous of the fireflies, mating and social behavior. The FA algorithm takes into account what each firefly notes in its line of sight in an attempt to move to a new location, which is brighter than its prior. Simulation results indicate that FA is superior over GA and PSO [7,8]. Although the original version of FA was designed to solve BCO problems, many variants of this algorithm has been developed and applied to solve constrained problems from different areas. FA has become popular and widely used in many applications like economic dispatch problems [9,10], mixed variable optimization problems [11,12,13] and multiobjective continuous optimization problems [14,15]. A recent review and advances of the firefly algorithms are available in [16,17].

In this paper, we aim to extend the FA for solving nonsmooth nonconvex constrained global optimization (CGO) problems. The mathematical formulation of the problem to be addressed has the form:

$$\begin{aligned} \underset{x\in\Omega}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g_k(x) \le 0, \quad k = 1,\dots,p \\ & h_j(x) = 0, \quad j = 1,\dots,m \end{aligned} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$, $\mathrm{g} : \mathbb{R}^n \to \mathbb{R}^p$ and $h : \mathbb{R}^n \to \mathbb{R}^m$ are nonlinear continuous functions, possibly non differentiable, and $\Omega = \{x \in \mathbb{R}^n : -\infty < l_b \le x \le u_b < \infty\}$, with l_b and u_b the vectors of lower and upper bounds on the variables, respectively. In (1), *f*, *g* and *h* may be nonconvex functions and many local minima may exist in the feasible region $\Omega_F = \{x \in \Omega : g(x) \le 0, h(\boldsymbol{x}) = 0\}$. In order to solve (1) two constraint-handling techniques based on feasibility and dominance rules and a global competitive ranking, are proposed.

The paper is organized as follows. Section 2 briefly presents some common constraint-handling techniques and the main ideas that motivated this work. Section 3 describes the original FA and in Section 4 we propose three extensions of FA for solving nonsmooth nonconvex CGO problems. The preliminary numerical experiments are reported in Section 5 and the paper is concluded in Section 6.

2 Constraint-Handling Techniques

In population-based methods, the widely used approach to deal with constrained optimization problems is based on exterior penalty methods [18,19,20,21]. In this type of approach, the constrained problem is replaced by a sequence of unconstrained subproblems, defined by penalty functions. A penalty function consists of the objective function of the constrained problem combined with one additional term for each

constraint (which is positive when the point is infeasible for that constraint and zero otherwise) multiplied by some positive penalty parameter. Making the penalty parameter larger along the iterative process, the constraints violation is more severely penalized, forcing in this way the minimizer of the penalty function to be closer to the feasible region of the original problem.

A well-known penalty function is the ℓ_1 exact penalty function, in which the terms that measure the constraints violation of a point x_i, are given by

$$\zeta(x_i) = \sum_{k=1}^{p} max\{0, g_k(x_i)\} + \sum_{j=1}^{m} |h_j(x_i)|.$$

Assuming that the bound constraints on the variables are guaranteed by the population stochastic method, at each iteration, the problem (1) is transformed into a BCO problem as follows:

$$\underset{x\in\Omega}{minimize}\ f(x) + \lambda\left(\sum_{k=1}^{p} max\{0, g_k(x_i)\} + \sum_{j=1}^{m} |h_j(x_i)|\right) \quad (2)$$

where $\lambda > 0$ is the penalty parameter. For a sufficiently large, positive value of λ, one minimization of exact penalty function (2) will produce the solution of problem (1). However, in practice it is hard to determine a priori the λ values, being necessary to use rules for adjusting this parameter along the iterative process.

Despite the popularity of penalty methods regarding its simplicity and easy implementation, they have several drawbacks. The most difficult issue lies in finding the appropriate penalty parameter values λ, since they require a suitable fine tuning to estimate the degree of penalization to be applied. New penalty approaches in this field are constantly under research.

2.1 Global Competitive Ranking

Runarsson and Yao [22] proposed a constraint-handling technique called global competitive ranking, where an individual point x_i is ranked by comparing it against all other members in the population, for $i = 1, \ldots, N$ being N the population size. In this technique, first the objective function value, $f(x_i)$, and the constraints violation value $\zeta(x_i)$, are calculated, for all points of the population. Then, considering a minimization problem, these values are ranked separately in ascending order. In case of tied individuals, the same higher rank will be given. After giving ranks to all points, based on f and ζ, separately, the fitness function of each individual point x_i is computed by:

$$\Phi(x_i) = P_f \frac{I_{i,f}-1}{N-1} + (1 - P_f)\frac{I_{i,\zeta}-1}{N-1} \quad (3)$$

where $I_{i,f}$ and $I_{i,\zeta}$ are the ranks of point x_i based on the objective function f and the constraints violation ζ, respectively. P_f is the probability that the fitness is calculated based on the rank of the objective function. According to the authors of [22], the probability should take a value on $0 < P_f < 0.5$ in order to guarantee that a feasible solution may be found. The main goal of this technique is to strike the right balance between the objective function and the constraints violation. From (3), the best point of

the population is the point that has the lowest fitness value. One drawback detected by the authors associated to this constraint handling technique was the need to use different values of P_f to solve different optimization problems. To prevent this drawback, using the same ranking process, we propose a new fitness function that does not depend on the probability value P_f.

2.2 Feasibility and Dominance Rules

Deb [23] proposed another constraint-handling technique that is based on biasing feasible over infeasible points. The constraints violation and the objective function values are used separately and optimized by some sort of order, where feasible points are always preferable to infeasible ones. This technique is based on three simple feasibility and dominance rules proposed for binary tournaments:

(i) Any feasible point is preferred to any infeasible one.
(ii) Between two feasible points, the one having better objective function is preferred.
(iii) Between two infeasible points, the one having smaller constraint violation is preferred.

In this work, we propose a ranking scheme based on rules (i)-(iii) with the additional following new rule, that takes into account the number of violated constraints (*nc*):

(iv) Between two infeasible points, the one having smaller number of violated constraints is preferred.

Hence, when two points of the population are compared to see which one improves over the other, the rules (i)-(iv) are used. These rules can be mathematically stated in the following definition.

Definition 1 (Point y *improves over* point x)
Let x and y be two points in Ω. The point y improves over point x if the following condition holds:

$$\big(\zeta(x) > \zeta(y) \text{ or } nc(x) > nc(y)\big) \text{ or } (\zeta(x) = \zeta(y) = 0 \text{ and } f(x) > f(y))$$

3 Firefly Algorithm

3.1 Standard Firefly Algorithm

FA is a stochastic population-based algorithm for solving BCO problems. In order to develop FA, some of the flashing characteristics of fireflies were idealized. Yang formulated FA by assuming three simple rules [6].

- All fireflies are unisex, meaning that any firefly will be attracted to other fireflies regardless of their sex.
- The brightness of a firefly is determined by the objective function value.
- Attractiveness between fireflies is proportional to their brightness but decreases with distance. For any two fireflies, the firefly with less bright will move towards the brighter.

In the description of the algorithm, the position of the firefly j will be represented by $x_j \in \mathbb{R}^n$and firefly j is brighter than firefly i if $f(x_j) < f(x_i)$. Most of metaheuristics optimization methods are based on the generation of random initial population of feasible points. All points of the population are placed in the search space to guide the search to the best location. Thus, the FA applies a similar strategy and the random initial population of Ω is generated as follows:

$$x_{i_s} = l_{b_s} + rand_s(u_{b_s} - l_{b_s}), \quad s = 1, \dots, n.$$

where $rand_s$~U(0,1) is a uniformly distributed random number in [0,1]. After generating the initial population, the objective function values $f(x_i)$ for all points x_i, $i = 1, \dots, N$; are calculated and ranked from lowest to largest value of f, and the iteration counter k is set to 1. In each iteration k, for each point x_i, the FA examines every point x_j, $j = 1, 2, \dots, N$. If point x_i has higher objective function value than x_j (firefly *j* is brighter than firefly *i*), the firefly *i* moves towards the firefly *j* according to following movement equation:

$$x_i = x_i + \beta(x_j - x_i) + \alpha(rand_i - 0.5)\, S \tag{4}$$

where $rand_i$ is a vector of random numbers generated from a uniform distribution in [0,1], α is a randomization parameter defined by the user, usually a number in the range [0,1] and S (scale of the problem) is a problem dependent vector scaling parameter defined componentwise by $S = |l_b - u_b|$. The parameter β of (4) is the attractiveness between fireflies i and j, and is defined in terms of the monotonically decreasing negative exponential function as follows:

$$\beta(r) = \beta_0 \mathrm{e}^{-\gamma \|x_i - x_j\|} \tag{5}$$

where $\|.\|$ is the Cartesian distance between the fireflies *i* and *j*, and β_0 is the attraction parameter when the distance between themselves is zero. The variation of the attractiveness is defined by the control parameter γ. The value of parameter γ is crucial to determine the speed of the convergence and how the FA behaves. In theory, γ could take any value in the set $[0, \infty[$. When $\gamma \to 0$, the value of $\beta \approx \beta_0$, meaning that a flashing firefly can be seen anywhere in the search space and, when $\gamma \to \infty$, the attractiveness is almost zero in the sight of other fireflies and each firefly moves in a random way.

Finally, whenever a position of a point x_i is updated, the FA controls the bound constraints, i.e., the point x_i is projected onto the search space as follows:

$$x_{i_s} = \begin{cases} l_{i_s} & if \quad x_{i_s} < l_{i_s} \\ u_{i_s} & if \quad x_{i_s} > u_{i_s} \end{cases}$$

The pseudo-code of the standard FA is presented in the Algorithm 1.

Algorithm 1: Standard Firefly Algorithm

Data: k_{max}, α, β_0, γ
Set $k = 1$
Randomly generate a population of *N* fireflies, $x_i^k \in \Omega$, $i = 1, \dots, N$
Based on $\{x_1^k, \dots, x_N^k\}$ evaluate $f(x_i^k), = 1, \dots, N$
Rank the fireflies using the objective function values (from lowest to largest of f)
Set $x_{best}^k = x_1^k$ and $f_{best}^k = f(x_1^k)$
Compute the scaler parameter S as $|l_b - u_b|$
while $k \leq k_{max}$ **do**
  for $i = 1$ *to N*
    for $j = 1$ *to N*
      if $f(x_i^k) > f(x_j^k)$ **then**
        Compute the attractiveness β using (5)
        Move firefly i towards firefly j using (4)
      end if
    end for j
  end for i
  Project x_i^k onto Ω, for all $i = 1, \dots, N$
  Evaluate $f(x_i^k)$, $i = 1, \dots, N$
  Rank the fireflies using the objective function values (from lowest to largest of f)
  Set $k = k + 1$
  Set $x_{best}^k = x_1^k$ and $f_{best}^k = f(x_1^k)$
end while

3.2 Dynamic Updates of the Parameters α, γ and *S*

The parameters α and γ affects the performance of FA. In the version of FA proposed in [11] to solve mixed variable structural optimization problems, the authors improved the solution quality by reducing the value of the parameter α with a geometric progression reduction scheme defined by $\alpha = \alpha_0 \theta^k$, where α_0 is the initial randomness scaling factor, $0 < \theta < 1$ is the reduction factor of randomization and k is the current iteration. In [12] the authors improved the quality of the solutions by reducing the randomness of the parameters α and γ. The computational experiments shown that they must take large values at the beginning of the iterative process and decrease gradually as the optimum solution is approached, to enforce the algorithm to increase the diversity and the convergence of the algorithm. In order to improve convergence speed and

solution accuracy, dynamic updates of these parameters, which depend on the iteration counter of the algorithm, were defined. The parameter α is defined at each iteration k as follows:

$$\alpha^{(k)} = \alpha_{max} - k\frac{\alpha_{max}-\alpha_{min}}{k_{max}} \tag{6}$$

where α_{max} and α_{min} are the limits to an upper and lower level for α, k is the number of current iteration and k_{max} is the maximum number of iterations allowed. The parameter γ, used for increasing the attractiveness with k, is defined at each iteration k by the following dynamic update formula:

$$\gamma^{(k)} = \gamma_{max} e^{\frac{klog(\frac{\gamma_{min}}{\gamma_{max}})}{k_{max}}} \tag{7}$$

where γ_{min} and γ_{max} are the minimum variation and maximum variation of the attractiveness, respectively.

In this paper we propose a dynamic update formula to compute the vector of scaling parameters with *k*, in order enhance the convergence of the proposed FA extensions. Thus, the vector S is dynamically updated in order to decrease with *k* as follows:

$$S^{(k)} = \frac{\left|(l_b-u_b)-(x_N^k-x_1^k)\right|}{k} \tag{8}$$

where $x_N^k - x_1^k$ is the vector of the ranges given by the positions between the best and the worst fireflies.

4 Constrained Firefly Algorithm

In this section, we present extensions of FA for solving nonsmooth nonconvex CGO problems. We propose two constraint-handling techniques based on feasibility and dominance rules and the global competitive ranking that are able to explore both feasible and infeasible regions.

4.1 Ranking Scheme Proposals

In the global competitive ranking (GR) proposed algorithm, after calculating $f(x_i)$, $\zeta(x_i)$ and $nc(x_i)$, for all points x_i of the population, the points are ranked considering separately the ascending order of $f(x_i)$ and $\zeta(x_i)$, $i = 1, \dots, N$. Then, taking into account the ranking of all points , the fitness function of each point x_i, *i=1,…,N;* is computed by:

$$v(x_i) = \frac{I_{i,f}-1}{N(N-1)} + nc(x_i)\frac{I_{i,\zeta}-1}{N(N-1)} \tag{9}$$

where $I_{i,f}$ and $I_{i,\zeta}$ are the ranks of point x_i based on the objective function f and the constraints violation ζ respectively. Finally, using the fitness function values $v(x_i)$, $i = 1, \dots, N$, the *N* points of the population are ranked by comparing all pairs of points

in at least *N* sweeps. The description of the proposed GR scheme based on fitness function (9) is presented in Algorithm 2.

Algorithm 2. GR

Compute I_f and I_ζ
for $i = 1$ *to* $N - 1$
  for $j = i + 1$ *to* N
    if $v(x_i) > v(x_j)$
      switch rank of firefly x_i with firefly x_j
    end if
  end for j
end for i

In the ranking scheme based on feasibility and dominance (FD) rules, first the objective function value, $f(x_i)$, the constraint violation value, $\zeta(x_i)$, and the number of constraints violated, $nc(x_i)$, are calculated for all points x_i of the population; $i=1,\ldots,N$. Then, using the rules (i)-(iv) the *N* points of the population are ranked by comparing all pairs of points in at least *N* sweeps.

A formal description of the proposed ranking scheme based on the FD rules (i)-(iv) (Definition 1) is presented in Algorithm 3.

Algorithm 3. FD rules

for $i = 1$ *to* $N - 1$
  for $j = i + 1$ *to* N
    if x_j *improves over* x_i
      switch rank of firefly x_i with firefly x_j
    end if
  end for j
end for i

Both ranking schemes, the GR and FD rules, ensure that good feasible solutions as well as promising infeasible ones are ranked in the top of the population.

4.2 Local Search

In order to reach high quality solutions the proposed extensions of FA are designed to invoke, at the end of each iteration, a stochastic local intensification search procedure aiming to exploit the search region around the best firefly, x_{best}. This local search, presented in [24], is a random line search algorithm that is applied coordinate by coordinate to the best point of the population. The procedure can be described as follows. First, for a fixed parameter δ the procedure computes the maximum feasible step length

$$\Delta = \delta\left(\max_{1\le s\le n}\left(u_{b_s} - l_{b_s}\right)\right).$$

Then, for each coordinate s ($s = 1, 2, \ldots, n$), a random number $\mu \sim U[0,1]$ (uniformly distributed between 0 and 1) is selected as a step length and a trial point y is componentwise moved along that direction and a new position is obtained as follows

$$y_s = x_{best_s} + \mu\Delta.$$

When $y \notin \Omega$, the trial point is rejected and the search along that coordinate ends. If y improves over the best point x_{best} according to Definition 1, within $LSit_{\max}$ iterations, the best point x_{best} is replaced by the trial point y and the search along that coordinate s ends. A description of the local search procedure is presented in Algorithm 4.

Algorithm 4: Local Search

Data: x_{best} (the best point of the population at iteration *k*), $LSit_{\max}$, δ

$\Delta = \delta \max_{1\le s\le n}\left(u_{b_s} - l_{b_s}\right)$

for $s = 1$ *to* n do

  Set $it = 1$

  while $it < LSit_{\max}$ do

    Set $y = x_{best}$

    $y_s = x_{best_s} + \mu\Delta, \ \ \mu \sim U[0,1]$

    if y *improves over* x_{best} and $y \in \Omega$ **then**

      Set $x_{best} = y$

    else

      Set $it = LSit_{\max} - 1$

    end if

    Set $it = it + 1$

  end while

end for

4.3 Extensions of FA

The proposed extensions of FA (herein denoted by exts-FA) use a population of points/fireflies to compute, at each iteration k, an approximate solution, x_{best}^k, to the problem (1). Along the iterative process, the exts-FA generate approximate solutions, x_{best}^k , that satisfy the bound constraints, with increasingly better accuracy.

In the standard FA, each firefly *i* moves towards the brighter fireflies. However, when a firefly *i*, located at x_i, moves as in standard FA, its brightness may decrease. To prevent this, after moving each firefly *i* in the direction of a brighter firefly *j*, the selection rule given by Definition 1 is applied. We remark that if the trial position lies outside the search space Ω, the point is projected onto Ω. Denoting by t_i the trial position, if t_i *improves over* x_i, t_i will be the current position of the firefly *i* for the next movement; otherwise, the position x_i will be maintained as current for the next

movement. To further improve exts-FA, all fireflies will move according to (4) except the less bright firefly, x_N. The position of firefly *N* is replaced by a random movement position of the brightest firefly. The pseudo-code of exts-FA is given in Algorithm 5.

Algorithm 5: exts- FA

Data: k_{max}, α_{max}, α_{min}, γ_{max}, γ_{min}
Set *k*=1
Randomly generate a population of *N* fireflies, $x_i^k \in \Omega,\ i = 1, \dots, N$
Based on $\{x_1^k, \dots, x_N^k\}$ evaluate $f(x_i^k), \zeta(x_i^k), nc(x_i^k),\ i = 1, \dots, N$
Rank the fireflies using GR (Algorithm 2) or FD rules (Algorithm 3)
Set $x_{best}^k = x_1^k$ and $f_{best}^k = f(x_1^k)$
While (stopping criteria is not met)
  Compute the randomization parameter $\alpha^{(k)}$ using (6)
  Compute the scale parameter $S^{(k)}$ using (8)
  for $i = 2$ to $N - 1$
    for $j = 1$ to $i - 1$
      Compute the attractiveness β using (5) and (7)
      Move firefly *i* towards firefly *j* using (4), and project onto Ω the trial position t_i
      Evaluate $f(t_i), \zeta(t_i), nc(t_i)$
      if t_i *improves over* x_i^k **then**
        Set $x_i^k = t_i$
      end if
    end for j
  end for i
  Set $t_N = x_1^k + \varepsilon$, where $\varepsilon \sim$ U(0,1) (vector of random numbers) and project onto Ω
  Evaluate $f(t_N), \zeta(t_N), nc(t_N)$
  if t_N *improves over* x_N^k **then**
    Set $x_N^k = t_N$
  end if
  Set $k = k + 1$
  Evaluate $f(x_i^k), \zeta(x_i^k), nc(x_i^k),\ i = 1, \dots, N$
  Rank the fireflies using GR (Algorithm 2) or FD rules (Algorithm 3)
  Set $x_{best}^k = x_1^k$ and $f_{best}^k = f(x_1^k)$
  Invoke the Local Search (Algorithm 4)
end while

We will denote by FA1 the exts-FA with FD rules, and by FA2 the exts-FA with GR in the ranking of the points. In the context of the implementation of FA1, we also propose a new movement equation (instead of (4)) in which all fireflies will move towards the best one. We will denoted this implementation by FA1#.

4.4 Stopping Criteria

The algorithm stops when the following condition is reached:

$$(\left|f_{opt} - f_{best}\right| \leq 10^{-6} \text{ and } \zeta(f_{best}) \leq 10^{-6}) \quad \text{or} \quad k > k_{\max} \qquad (10)$$

where f_{opt} represents the known global optimal solution, f_{best} is the objective function value of the best point of the population, k denotes the iteration counter and $k_{\max}$ is the maximum number of iterations allowed.

5 Experimental Results

In this section, we aim to investigate the performance of FA1, FA1# and FA2 when solving a set of nonlinear optimization problems. Thirteen benchmark global optimization test problems, with dimensions ranging from 2 to 20, chosen from [25] containing characteristics that are representative of what can be considered difficult when solving global optimization problems. Their characteristics are outlined in Table 1.

Table 1. Summary of main properties of the benchmark problems.

Prob.	f_{opt}	n	function	LI	NI	LE	NE	ρ (%)
G01	-15.000000	13	Quadratic	9	0	0	0	0.011
G02	-0.803619	20	Nonlinear	1	1	0	0	99.99
G03	-1.000500	10	Nonlinear	0	0	0	1	0.002
G04	-30665.538672	5	Quadratic	0	6	0	0	52.123
G05	5126.496714	4	Nonlinear	2	0	0	3	0.000
G06	-6961.813876	2	Nonlinear	0	2	0	0	0.006
G07	24.306209	10	Quadratic	3	5	0	0	0.000
G08	-0.095825	2	Nonlinear	0	2	0	0	0.856
G09	680.630057	7	Nonlinear	0	4	0	0	0.521
G10	7049.248021	8	Linear	3	3	0	0	0.001
G11	0.749900	2	Quadratic	0	0	0	1	0.000
G12	-1.000000	3	Quadratic	0	9	0	0	4.779
G13	0.053942	5	Nonlinear	0	0	1	3	0.000

The two first columns display the name of the problem (Prob.) and the best known solution (f_{opt}), followed by the number of variables (n), the type of objective function (function), the number of inequality constraints (LI and NI, for linear and nonlinear inequality constraints, respectively), the number of equality constraints (LE and NE, for linear and nonlinear equality constraints, respectively), as reported in [25]. The feasibility ratio ρ, in the last column, is an estimate of the size of the feasible search space Ω_F to the size of the whole search space. In practice, ρ represents the degree of difficulty of each problem.

The numerical experiments were carried out on a MacBook Pro (13-inch, Mid 2012) with processor 2.5 GHz and 4 Gb of memory. The algorithms were coded in Matlab® programming language, version 8.01 (R2013a).

Since the FA is a stochastic method, each problem was solved 20 times. The size of the population used was $N = 40$ fireflies and in the stopping criteria, defined in (10), the maximum number of iterations allowed was $k_{\max} = 5000$ iterations. The initial parameters used to dynamically compute α and γ are: $\alpha_{max} = 0.9$, $\alpha_{min} = 0.01$, $\gamma_{max} = 100$ and $\gamma_{min} = 0.001$. All equality constraints $h_j(x)$ have been converted into inequality constraints using $|h_j(x)| - \varepsilon \leq 0$, where $\varepsilon > 0$ is a very small violation tolerance. In our numerical experiments $\varepsilon = 10^{-4}$ is used for the problems G05, G11 and G13 and $\varepsilon = 10^{-6}$ for the remaining problems. The step length in the local search procedure is set to $\delta = 10^{-5}$, except those marked with (*) that are set to $\delta = 10^{-2}$. The maximum number of local search iterations allowed is $LSit_{\max} = 10$.

Table 2 summarizes the numerical results produced by the exts-FA, namely the proposed FA1#, FA1 and FA2. The first column shows the name of the problem, followed by the acronym of the exts-FA implementation. The remaining columns present: the best (f_{best}), the mean (f_{mean}), the median (f_{med}), the standard deviation (SD) and the worst (f_{worst}) solution values obtained over the 20 runs.

We remark that our proposed algorithms were able to find feasible solutions for all the runs of all of thirteen benchmark tested problems. This is due to the fact that the proposed algorithms prioritize the search of feasible solutions before proceeding to the search of the global optimum value.

The proposed exts-FA were also able to achieve very good results in almost of the problems. In the G01, G03, G08, G11 and G12 problems, the FA1#, FA1 and FA2 implementations reached the known global optimal solution in all runs. Consequently the measures of mean, median and the worst of the objective function values are equal to the global optimum and the standard deviation is zero. For G05 and G13 problems, the optimal solutions obtained by FA1 are lower than the known optimum values. This is related to the fact that the equality constraints of these problems were relaxed by a threshold value of $\varepsilon = 10^{-4}$. For the G04, G06, G07 and G09 problems the proposed extensions of FA produced very competitive results since they were able to obtain optimum values very close to the known optimum ones. On the other hand, for the problems G02 and G10 they were not able to reach the known optimum solution.

In general, the best performance was obtained with FA1 and FA2 implementations. Then, we analyze the performance of these two extensions when compared with five stochastic population-based global methods. In [26] the method incorporates a homomorphous mapping between an *n*-dimensional cube and the feasible search space. Runarsson and Yao in [27] present results of the original stochastic ranking method for constrained evolutionary optimization. In [28] a self-adaptive fitness formulation is used. The results reported in [29] were obtained with an adaptive penalty method with dynamic use of DE variants, while in [30] a self-adaptive penalty based genetic algorithm is used. Table 3 reports the best results found by these methods and by our proposed best implementations FA1 and FA2.

Table 2. Results produced by the FA1#, FA1 and FA2.

Prob.	exts-FA	f_{best}	f_{mean}	f_{med}	SD.	f_{worst}
	FA1#	-15.0000	-15.0000	-15.0000	0.0000	-15.0000
G01	FA1	-15.0000	-15.0000	-15.0000	0.0000	-15.0000
	FA2	-15.0000	-15.0000	-15.0000	0.0000	-15.0000
	FA1#	-0.4373	-0.2968	-0.2841	0.0557	-0.2405
G02	FA1	-0.4048	-0.2941	-0.2911	0.0398	-0.2414
	FA2	-0.4799	-0.3458	-0.3330	0.0631	-0.2488
	FA1#	-1.0005	-1.0005	-1.0005	0.0000	-1.0005
G03	FA1	-1.0005	-1.0005	-1.0005	0.0000	-1.0005
	FA2	-1.0005	-1.0005	-1.0005	0.0000	-1.0004
	FA1#	-30665.5385	-30538.1527	-30546.2560	96.7170	-30372.8244
G04	FA1	-30665.5386	-30663.8601	-30665.5382	7.5011	-30631.9925
	FA2	-30665.5385	-30660.8451	-30665.5382	14.4310	-30611.1510
	FA1#*	5126.7259	5402.6462	5274.4115	279.0658	5986.442138
G05	FA1	5126.3617	5157.5428	5128.489	66.0467	5401.545758
	FA2	5126.5175	5128.7370	5128.4245	2.0577	5133.3343
	FA1#*	-6961.8101	-6961.7905	-6961.7905	0.0130	-6961.7553
G06	FA1	-6961.8016	-6961.7747	-6961.7747	0.0172	-6961.7347
	FA2*	-6961.5405	-6959.8675	-6960.2069	1.2835	-6956.3551
	FA1#*	24.6203	26.0814	26.1657	0.7935	27.1457
G07	FA1	24.3571	24.5827	24.5456	0.1967	25.2044
	FA2	24.3273	24.3772	24.3663	0.0311	24.4368
	FA1#	-0.095825	-0.095825	-0.095825	0.0000	-0.095825
G08	FA1	-0.095825	-0.095825	-0.095825	0.0000	-0.095824
	FA2	-0.095825	-0.095825	-0.095825	0.0000	-0.095825
	FA1#	680.6445	680.7663	680.7867	0.0676	680.8671
G09	FA1	680.6348	680.6895	680.6659	0.0468	680.7948
	FA2	680.6328	680.6869	680.6821	0.0414	680.7852
	FA1#	7069.4263	8183.2192	7785.5967	1053.5456	10276.0255
G10	FA1*	7125.2110	7552.1114	7364.5715	486.5234	8872.8741
	FA2*	7073.7810	7171.9798	7145.6712	107.9335	7542.8818
	FA1#	0.7499	0.7499	0.7499	0.0000	0.749910
G11	FA1	0.7499	0.7499	0.7499	0.0000	0.749900
	FA2*	0.7499	0.7499	0.7499	0.0000	0.749900
	FA1#	-1.0000	-1.0000	-1.0000	0.0000	-1.0000
G12	FA1	-1.0000	-1.0000	-1.0000	0.0000	-1.0000
	FA2	-1.0000	-1.0000	-1.0000	0.0000	-1.0000
	FA1#*	0.054106	0.325382	0.435195	0.2509	0.817849
G13	FA1	0.053556	0.283400	0.434445	0.1924	0.444500
	FA2	0.053944	0.053960	0.053951	0.0000	0.054017

* means the step length of $\delta = 10^{-2}$ in the local search procedure.

From Table 3 the competitiveness of FA1 and FA2 with the reported approaches is shown. The stochastic ranking in [27] produced very good results. However this algorithm was able to obtain feasible solutions only in 6 out of the 30 runs performed for the test problem G10. In [28] only 17 out of 20 runs produced feasible solutions for the problem G10 and 9 out of 20 runs for G05 problem, while the exts-FA obtained

feasible solutions for all problems in all runs. In general, the proposed exts-FA is competitive as the reported algorithms in the related field.

Table 3. Comparison of our study with others stochastic population-based methods

Prob.	FA1	FA2	[26]	[27]	[28]	[29]	[30]
G01	-15.0000	-15.0000	-14.7082	-15.0000	-15.0000	-15	-15.000
G02	-0.4048	-0.4799	-0.79671	-0.803515	-0.802970	-0.8036	-0.803202
G03	-1.0005	-1.0005	-0.9989	-1.0000	-1.0000	-1.0	-1.000
G04	-30665.539	-30665.539	-30655.3	-30665.539	-30665.500	-30665.5	-30665.401
G05	5126.3617	5126,5175	n.a.	5126.497	5126.989	5126.4981	5126.907
G06	-6961.8016	-6961,5405	-6342.6	-6961.814	-6961.800	-6961.8	-6961.046
G07	24.3571	24,3273	24.826	24.307	24.480	24.306	24.838
G08	-0.095825	-0.095825	-0.089157	-0.095825	-0.095825	-0.09582	-0.095825
G09	680.6348	680.6328	681.16	680.630	680.640	680.63	680.773
G10	7125.2110	7073.7810	8163.6	7054.316	7061.340	7049.25	7069.981
G11	0.749900	0.749900	0.75	0.750	0.75	0.75	0.749
G12	-1.000000	-1.000000	-0.999135	-1.000000	n.a.	n.a.	-1.000000
G13	0.053556	0.053944	0.557	0.053957	n.a.	n.a.	0.053941

n.a. means not available.

6 Conclusions

The FA is a stochastic global optimization algorithm, inspired by the social behavior of fireflies and based on their flashing and attraction, which was originally designed to solve bound constrained optimization problems. In this paper we extend the FA to solve nonsmooth nonconvex constrained global optimization problems. The extensions of FA denoted by FA1 and FA1# incorporate the constraint-handling technique based on the feasibility and dominance rules, while FA2 uses a global competitive ranking combined with a new fitness function. Moreover, FA1# uses a movement equation where all fireflies move towards the best one.

Thirteen well known benchmark problems were used in order to test the performance of the implementations of the exts-FA. The numerical experiments show that the proposed exts-FA are competitive when compared with other stochastic methods. Further research will be directed to improve the results through testing other fitness functions combined with the GR scheme. Future developments may include solving problems with large dimensions.

Acknowledgements. This work has been supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT – Fundação para a Ciência e Tecnologia within the projects UID/CEC/00319/2013 and UID/MAT/00013/2013.

References

1. Blum, C., Li, X.: Swarm intelligence in optimization. In: Blum, C., Merkle, D. (eds.), Swarm Intelligence: Introduction and Applications, pp. 43–86, Springer Verlag, Berlin (2008)
2. Tuba, M.: Swarm Intelligence Algorithms Parameter Tuning. In: Proceedings of the American Conference on Applied Mathematics (AMERICAN-MATH'12), pp. 389-394, Harvard, Cambridge, USA (2012)
3. Holland, J.H.: Adaptation in Natural and Artificial Systems, Ann Arbor: University of Michigan Press (1975)
4. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: IEEE International Conference on Neural Networks (Perth, Australia), pp. 1942–1948, IEEE Service Center, Piscataway, NJ (1995)
5. Geem, Z.W., Kim , J.H., Loganathan, G.V.: A new heuristic optimization algorithm: harmony search. Simulations 76, 60-68 (2001)
6. Yang, X. S.: Nature-Inspired Metaheuristic Algorithms. Luniver Press (2008)
7. Dorigo, M.: Optimization, learning and natural algorithms, PhD Thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy (1992)
8. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science. Nagoya: IEEE Press, pp. 39-43 (1995)
9. Horng, M.H., Liou, R.J.:Multilevel minimum cross entropy threshold selection based on the firefly algorithm., Expert Systems with Applications 38 (12), pp. 14805-14811 (2011)
10. Yang, X. S., Hosseini, S. S., Gandomi, A. H.: Firefly algorithm for solving non-convex economic dispatch problems with valve loading effect. Applied Soft Computing 12(3), pp. 1180-1186 (2012)
11. Gandomi, A. H., Yang, X. S., Alavi, A. H.: Mixed variable structural optimization using Firefly Algorithm. Computers & Structures, 89 (23-24), pp. 2325–2336 (2011)
12. Costa, M.F.P., Rocha, A.M.A.C., Francisco, R.B., Fernandes, E.M.G.P.: Heuristic-Based Firefly Algorithm for Bound Constrained Nonlinear Binary Optimization. Advances in Operations Research, 2014, Article ID 215182, 12 pages (2014)
13. Costa, M.F.P., Rocha, A.M.A.C., Francisco, R.B., Fernandes, E.M.G.P.: Firefly penalty-based algorithm for bound constrained mixed-integer nonlinear programming, Optimization (2016) (in press) DOI: 10.1080/02331934.2015.1135920.
14. Yang, X.S.: Firefly algorithms for multimodal optimization. In: Watanabe, O., Zeugmann, T. (eds.) Stochastic Algorithms: Foundations and Applications (SAGA 2009), LNCS, vol. 5792, pp. 169-78, Springer-Verlag, Berlin (2009)
15. Yang, X. S.: Multiobjective firefly algorithm for continuous optimization. Engineering with Computers 29(2), 175-184 (2013)
16. Fister, I., Fister Jr., I., Yang, X.-S., Brest, J.: A comprehensive review of firefly algorithms. Swarm and Evolutionary Computation 13, 34-46 (2013)
17. Yang, X.-S., He, X.: Firefly algorithm: recent advances and applications. International Journal of Swarm Intelligence 1 (1), 36–50 (2013)
18. Ali, M., Zhu, W. X.: A Penalty Function-Based Differential Evolution Algorithm for Constrained Global Optimization. Computational Optimization and Applications 54 (3), 707-739 (2013)
19. Barbosa, H. J. C., Lemonge, A. C. C.: An Adaptive Penalty Method for Genetic Algorithms in Constrained Optimization Problems. In: H. Iba (eds.) Frontiers in Evolutionary Robotics, pp. 9-34. Vienna: I-Tech Education Publications (2008)

20. Mezura-Montes, E., Coello Coello. C. A. C.: Constraint-Handling in Nature-Inspired Numerical Optimization: Past, Present and Future. Swarm and Evolutionary Computation 1 (4), 173–194 (2011)
21. Lemonge, A. C. C., Barbosa, H. J. C., Bernardino, H. S.: Variants of an Adaptive Penalty Scheme for Steady-State Genetic Algorithms in Engineering Optimization. Engineering Computations: International Journal for Computer-Aided Engineering and Software 32(8), 2182–2215 (2015)
22. Runarsson, T.P., Yao, X.: Constrained evolutionary optimization – the penalty function approach. In: Sarker et al. (eds.), Evolutionary Optimization: International Series in Operations Research and Management Science, vol. 48, pp. 87–113, Springer, New York (2003)
23. Deb, K.: An Efficient Constraint-handling Method for Genetic Algorithms. Computer Methods in Applied Mechanics and Engineering 186, No. 0045-7825, pp. 311-338.(2000)
24. Birbil, S. I., Fang, S.-C.:An electromagnetism-like mechanism for global optimization. Journal of Global Optimization 25, 263-282 (2003)
25. Liang, J.J., Runarsson, T.P., Mezura-Montes, E., Clerc, M., Suganthan, P.N., Coello, C.A.C., Deb, K.: Problem Definition and Evolution Criteria for the CEC 2006 Special Session on Constrained Real-Parameter Optimization. In: IEEE Congress on Evolutionary Computation, Vancouver, Canada, 17–21 July (2006)
26. Koziel, S., Michalewicz, Z.: Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. Evolutionary Computations 7(1), 19-44 (1999)
27. Runarsson, T. P., Yao, X.: Stochastic ranking for constrained evolutionary optimization. IEEE Transactions on Evolutionary Computation 4 (3), 284–294 (2000)
28. Farmani, R., Wright, J.: Self-adaptive fitness formulation for constrained optimization. IEEE Transaction on Evolutionary Computation 7 (5), 445-455 (2003)
29. Silva, E.K., Barbosa, H.J.C., Lemonge, A.C.C.: An adaptive constraint handling technique for differential evolution with dynamic use of variants in engineering optimization. Optimization and Engineering 12 (1–2), 31–54 (2011)
30. Tessema, R, Yen, G.G.: A self adaptive penalty function based algorithm for constrained optimization. In: IEEE Congress on Evolutionary Computation (CEC 2006), pp. 246-253, Vancouver, Canada (2006)