AFSFilter: Artificial Fish Swarm Filter-Based Algorithm for Global Optimization*

Ana Maria A.C. Rocha,[1] M. Fernanda P. Costa,[2] and Edite M.G.P. Fernandes[1]

[1]*Algoritmi R&D Centre, U Minho, 4710-057 Braga, Portugal,* {arocha,emgpf}@dps.uminho.pt

[2]*Mathematics R&D Centre, U Minho, Braga, Portugal,* mfc@math.uminho.pt

Abstract A fish swarm intelligence algorithm based on the filter set concept to accept, at each iteration, a population of trial solutions whenever they improve constraint violation or objective function, relative to the current solutions, is proposed for constrained global continuous optimization problems. Preliminary numerical results are provided.

Keywords: Global optimization, Fish Swarm Intelligence, Filter method

1. Introduction

The problem to be addressed in this paper has the form

$$\min_{x\in\Omega} f(x)\,,\ \text{subject to}\ g_j(x)\leq 0\,, j=1,\ldots,m \tag{1}$$

where at least one of the functions $f, g_j : \mathrm{R}^n \to \mathrm{R}$, is nonlinear and $\Omega = \{x \in \mathrm{R}^n : l_k \leq x_k \leq u_k, k = 1,\ldots,n\}$. Problems with equality constraints can be reformulated in the above form using a small tolerance. When convexity is not assumed, problem (1) may have multiple optimal solutions in Ω. This paper aims at proposing a stochastic method to compute a global solution of (1). From the class of stochastic methods, swarm intelligence algorithms have shown to be effective in reaching a global solution. Recent studies involving the artificial fish swarm (AFS) algorithm show that highly accurate solutions may be obtained with reduced computational costs [6, 7]. Although penalty function methods are probably the most known constraint handling techniques, a penalty function depends, in general, on a penalty parameter. Unfortunately, the performance of these algorithms depends strongly on the values set to the penalty parameter throughout the iterative procedure. Adaptive penalties [9] and augmented Lagrangian methodologies [2, 8] are just recent strategies to overcome partially this issue. The separate use of objective function and constraint violation with the nondominance concept from multiobjective programming, for example in [1], avoids the use of penalty parameters. Fletcher and Leyffer [4] proposed a filter method as an alternative to penalty functions to guarantee convergence to optimizers in nonlinear constrained optimization. This technique incorporates the concept of nondominance to build a filter set that is able to accept solutions if they improve either the objective function or the constraint violation, instead of a linear combination of those two measures.

*The financial support from FEDER COMPETE (Programa Operacional Fatores de Competitividade / Operational Programme Thematic Factors of Competitiveness) and FCT (Fundação para a Ciência e a Tecnologia / Portuguese Foundation for Science and Technology) Project FCOMP-01-0124-FEDER-022674 is gratefully acknowledged.

In this paper, an artificial fish swarm filter-based algorithm, hereafter denoted by AFSFilter, for nonlinear constrained global optimization problems is proposed. Results from preliminary numerical experiments are provided.

2. Artificial Fish Swarm Algorithm

Here is some notation used in the paper. Constraint violation of a point x is measured by the function

$$\theta(x) = \sum_{j=1}^{m} \max\{0, g_j(x)\} + \sum_{k=1}^{n} \left(\max\{0, x_k - u_k\} + \max\{0, l_k - x_k\}\right), \tag{2}$$

$x^i \in \mathrm{R}^n$ represents the ith point of a population of size p, and x^{best} is the best point in the population. Pairwise comparisons in the population use the following concept: between two points x^i and x^j, x^i is better than x^j if the following condition holds:

$$\theta(x^i) < \theta(x^j) \ \text{ or } \ \left(\theta(x^i) = \theta(x^j) \text{ and } f(x^i) < f(x^j)\right). \tag{3}$$

In the AFS algorithm, the initial population of p points is randomly generated inside the set Ω. A crucial quantity of the algorithm is the 'visual scope' of a point, say x^i. This is defined as the closed neighborhood with center x^i and radius equal to a positive quantity $v = \varsigma \max_{k\in\{1,\dots,n\}}(u_k - l_k)$, where ς is a positive visual parameter. Let n^i be the number of points in its 'visual scope' ($n^i < p$). If the condition $n^i/p \le \kappa$ holds, where $\kappa \in (0,1]$ is the crowd parameter, the 'visual scope' of x^i is said to be not crowded. Depending on the relative position of the points in the population, one of the following three situations occurs.

1. When $n^i = 0$, the 'visual scope' is empty, and the point x^i, with no other points in its neighborhood to follow, has a random behavior. Here, a point is randomly generated in the search space, x^r, and a movement is tried along the direction defined by $d = x^r - x^i$.

2. When the 'visual scope' is crowded, the point has some difficulty in following any particular point, and starts by following a searching behavior. A point inside the 'visual scope' is randomly generated, x^s, and a movement towards it is carried out if x^s is better than x^i (see condition (3)); otherwise, x^i moves according to a random behavior.

3. When the 'visual scope' is not crowded, the point firstly tries the chasing behavior moving towards the best point inside the 'visual scope', denoted by $x^{\min}$, if this is better than x^i, thus being the direction of movement $d = x^{\min} - x^i$. Otherwise, the point tries to follow the swarming behavior moving towards the central point, c, of the 'visual scope'. However, if c is not better than x^i, the point tries to follow a searching behavior; and if that randomly generated point x^s is not better than x^i, the point follows a random behavior.

The algorithm also implements an elitism procedure in the sense that the best point of the population is not moved and is maintained throughout the iterative process. For each current point x^i of the population, the trial point y^i is generated according to a direction d and a step size $\alpha \in (0,1]$

$$y^i = x^i + \alpha\, d, \ i = 1, \dots, p \text{ and } i \ne best. \tag{4}$$

The procedure that decides if the trial solution is to be accepted and replaces the current point is a filter method combined with a backtracking line search, as described in the next section. The algorithm terminates with a successful run if the stopping conditions

$$\left|f(x^{best}) - f^*\right| \le \epsilon_1 |f^*| + \epsilon_2 \ \text{ and } \ \theta(x^{best}) \le \epsilon_2$$

are satisfied, for small positive tolerances ϵ_1, ϵ_2; otherwise $x^i \leftarrow y^i$ for all $i \neq best$ and the procedure is repeated until a maximum number of iterations is reached, where f^* is the best global solution of the problem available in the literature.

3. The Implemented Filter Methodology

This section briefly describes the filter methodology that aims at deciding which trial solution is to be accepted in the sequence of Eq. (4). The herein proposed AFSFilter method uses the filter set concept, as outlined in [3, 4], with the ability of exploring both feasible and infeasible regions, and building a filter set that is able to accept a trial point if it improves either the objective function or the constraint violation, relative to the current point. Filter-based algorithms treat the optimization problem as a biobjective problem aiming to minimize both the objective function and the nonnegative constraint violation function (2).

After a search direction d has been computed, A decreasing sequence of step sizes $\{\alpha_j\}$ with $\lim_j \alpha_j = 0$ is tried, until a set of acceptance conditions are satisfied. This j denotes the iteration counter for the inner loop. A trial step size α_j might be accepted if the corresponding trial point $y^i = x^i + \alpha_j\, d$ is acceptable by the filter. We only require an improvement in θ or in f, relative to the current point x^i, to consider the trial point y^i, in Eq. (4), to be acceptable, as shown:

$$\theta(y^i) < \theta(x^i) \text{ or } f(y^i) < f(x^i). \tag{5}$$

However, when x^i is (almost) feasible, the trial point y^i has to satisfy only the condition of simple reduction on f:

$$f(y^i) < f(x^i) \tag{6}$$

to be acceptable. To prevent cycling between points that improve either θ or f, at each iteration, the algorithm maintains a filter $\mathcal{F}$ which is defined as a finite set of entries $(\theta(x^j), f(x^j))$ that correspond to a collection of infeasible solutions x^j such that no filter entry is dominated by any of the others in the filter. During the backtracking line search procedure, the y^i is acceptable only if $(\theta(y^i), f(y^i)) \notin \mathcal{F}$. (Only solutions that are not dominated by any entry in the filter might be accepted.)

The filter is initialized with entries (θ, f) that satisfy $\theta \geq \theta_{\max}$, where $\theta_{\max} > 0$ is the upper bound on θ. Furthermore, the filter is augmented whenever y^i is accepted because condition (5) is satisfied. When it is not possible to find a point y^i with a step size $\alpha_j > \alpha_{\min} > 0$ that satisfies one of the conditions (5) or (6), a restoration phase is invoked. In this phase, the algorithm performs a coordinate random local search around the best point, with length $10^{-3}\max_k\{u_k - l_k\}$, to find a point inside $[l, u]$ that is acceptable to the filter. If no such point is found, the algorithm maintains the current point to the next iteration.

4. Preliminary Results

Table 1 contains the numerical results of our preliminary experiments with the AFSFilter method. Three well-known engineering design problems are used in the comparison with the results obtained by the Filter Simulated Annealing (SA) method proposed in [5]. The welded 'beam' design problem has four design variables and seven inequality constraints, the tension/compression 'spring' design problem has three continuous variables and four inequality constraints and the cylindrical 'vessel' design problem has four design variables (two of them are multiples of 0.0625) and four inequality constraints [5]. The size of the population is set to $p = 5n$ and the algorithm was allowed to run for a maximum of 200 iterations. A comparison with the pattern search hybrid GA from MatLab$^{\text{TM}}$ (with the tournament selection option to handle constraints) is also provided. A set of four small but difficult problems, with $n = 2$, selected from [2] and a technical report from the same authors[1], is also tested and the results are reported in the table for comparison with GA. The size of the population is set to $p = 20$ and a maximum of 50 iterations is allowed.

Each problem was run 30 times and the results reported in the table are: 'f^{best}', the best solution obtained during all the runs and 'avg.n.f.e.', the average number of function evaluations (from the 30 runs). Other parameter values are set as follows: $\epsilon_1 = 10^{-6}, \epsilon_2 = 10^{-8}$, $\theta_{\max} = 10^4$, $\alpha_{\min} = 10^{-3}$ and ς is set to one and is reduced every p iterations until it reaches 0.1. We may conclude that the proposed AFSFilter method is effective in reaching a global optimal solution with reasonable computational costs. New developments with further experimentation will follow.

Table 1. Comparison of AFSFilter with Filter SA in [5] and Hybrid GA in MatLab.

Prob.		AFSFilter		Filter SA in [5]		Hybrid GA from MatLab	
	f^*	f^{best}	avg.n.f.e.	f^{best}	avg.n.f.e.	f^{best}	avg.n.f.e.
beam	2.38081	2.3866641	65 687	2.381065	56 243	2.5526753	168 119
spring	0.012664	0.0126653	35 929	0.0126653	49 531	0.0126663	3 480
vessel	5854.738	5868.974	45 283	5868.765	108 883	5859.977	21 289
Example 1 in [2]	-1.0000000	-0.9983634	7 906	n.a.	n.a.	-0.9999910	4 027
Example 3 in [2]	1.0000000	1.0000013	10 849	n.a.	n.a.	0.9999987	5 366
Example 5 in [2]	-2.0000000	-2.0000330	8 180	n.a.	n.a.	-2.0000043	51 067
Problem 6†	-9.4772944	-9.4772842	9 458	n.a.	n.a.	-3.2883714‡	5 108

† Technical report MCDO-051015 (2005) in http://www.ime.usp.br/~egbirgin/, pre-print to [2]
‡ This is a local optimal solution

References

[1] Ali, M.M., Golalikhani, M. (2010). An electromagnetism-like method for nonlinearly constrained global optimization. Computers and Mathematics with Applications, 60, 2279–2285

[2] Andreani, R., Birgin, E.G., Martinez, J.M., Schuverdt, M.L. (2007). On augmented Lagrangian methods with general lower-level constraints. SIAM Journal on Optimization, 18(4), 1286–1309

[3] Audet, C., Dennis, Jr., J.E. (2004). A pattern search filter method for nonlinear programming without derivatives. SIAM Journal on Optimization, 14(4) 980–1010

[4] Fletcher, R., Leyffer, S. (2002). Nonlinear programming without a penalty function. Mathematical Programming, 91, 239–269

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

[6] Gao, X.Z., Wu, Y., Zenger, K., Huang, X. (2010). A knowledge-based artificial fish-swarm algorithm. 13th IEEE International Conference on Computational Science and Engineering, 327–332

[7] Rocha, A.M.A.C., Fernandes, E.M.G.P., Martins, T.F.M.C. (2011). Novel fish swarm heuristics for bound constrained global optimzation problems. ICCSA 2011, Part III, Lecture Notes in Computer Science 6784, B. Murgante et al. (Eds.) 185–199

[8] Rocha, A.M.A.C., Martins, T.F.M.C., Fernandes, E.M.G.P. (2011). An augmented Lagrangian fish swarm based method for global optimization. Journal of Computational and Applied Mathematics, 235(16) 4611–4620

[9] Silva, E.K., Barbosa, H.J.C., Lemonge, A.C.C. (2011). An adaptive constraint handling technique for differential evolution with dynamic use of variants in engineering optimization. Optimization and Engineering, 12, 31–54

[1]Technical report MCDO-051015 (2005) in http://www.ime.usp.br/~egbirgin/.