I Congresso de Estatística e Investigação Operacional da Galiza e Norte de Portugal
VII Congreso Galego de Estatística e Investigación de Operacións
Guimarães 26, 27 e 28 de Outubro de 2005

Particle swarm algorithms for multi-local optimization

A. Ismael F. Vaz[1], Edite M.G.P. Fernandes[1]

[1]Departamento de Produção e Sistemas
Escola de Engenharia, Universidade do Minho, Campus de Gualtar
4710-057 Braga, Portugal
Email: {aivaz,emgpf}@dps.uminho.pt.
Web: *http://www.norg.uminho.pt.*

ABSTRACT

Traditional particle swarm optimization (PSO) algorithms proved to be valid in finding a global optimum. In this paper we propose a modification to the PSO algorithm by introducing the gradient information or an approximate descent direction to enable the computation of all the global and local optima of a single multi-modal objective function. The numerical experiments carried out with a set of well-known test problems show the effectiveness of the proposed algorithm.

Keywords: Particle swarm optimization, descent directions, multi-local optimization.

1. INTRODUCTION

In this paper we address the problem of finding multiple optimal solutions of

$$\min_{x \in X} f(x) \tag{1}$$

where $f(x)$ is a given multi-modal objective function and X is a compact set defined by $X = \{x \in R^n : a_j \leq x_j \leq b_j, j = 1, \ldots, n\}$. We assume that the problem (1) has a finite number of optimal solutions and the single function $f(x)$ is continuously differentiable.

There are some applications where all the global and local solutions have to be computed in a fast and reliable manner. This is the case with reduction type methods for solving semi-infinite programming problems (Hettich and Kortanek (1993), León et al. (1998)). Due to the existence of multiple local and global optima, these problems cannot be efficiently solved by classical optimization techniques. Eberhart and Kennedy (1995) and Kennedy and Eberhart (1995) proposed the particle swarm optimization (PSO) algorithm which is a simple population based algorithm motivated from the simulation of social behavior. Although this is an effective algorithm, when compared with other evolutionary methods, for computing a global solution, some problems can arise when the objective function has more than one global minimum, since the algorithm oscillates between the solutions. Recently, there have been a number of attempts to combine the use of PSO algorithms with other techniques in order to locate multiple solutions of the problem (1). Parsopoulos and Vrahatis (2002) proposed a modification of the PSO algorithm that relies on a stretching function technique to isolate some particles in the swarm and generating small populations around them for a finer search in their local neighborhood while the big swarm continues searching for better

solutions. Brits et al. (2002) presented the NichePSO algorithm, a variation of the traditional PSO algorithm that locates niches through the growing of subswarms from an initial swarm of particles and the use of a Guaranteed Convergence Particle Swarm Optimization algorithm (van den Bergh (2002)) to "train" the subswarm particles. An adaptive swarm algorithm that is capable of identifying potentially good leaders during the evolution and maintaining diversity of leaders across the search space in presented in Meng et al. (2004).

In this paper, we propose another modification of the PSO algorithm which uses descent directions in order to drive each particle to a neighbor local minimum, thus locating multiple solutions.

This paper is organized as follows. In Section 2 we present an overview of the PSO algorithm. Section 3 describes the ideas behind the new multi-local swarm optimizer, Section 4 presents some implementation details and Section 5 reports on some numerical results using a set of benchmark mathematical problems. The conclusions make up Section 6.

2. PARTICLE SWARM OPTIMIZATION

The particle swarm algorithm mimics a swarm behavior in looking for a certain objective. The PSO algorithm simulates the social behavior concept to compute the global optima of problem (1). This algorithm uses a population (swarm) of individuals (particles). To each particle i, at time instant (or iteration) t, is assigned a position $x^i(t) \in X$ and a velocity $v^i(t)$ that provides information to where the particle is travelling to. The information related to the best position $y^i(t)$ that the particle has visited so far is also maintained. The velocity at time instant $t+1$ is updated as follows:

$$v_j^i(t+1) = \iota(t)v_j^i(t) + \mu\omega_{1j}(t)\left(y_j^i(t) - x_j^i(t)\right) + \nu\omega_{2j}(t)\left(\hat{y}_j(t) - x_j^i(t)\right), \quad j = 1, \ldots, n \quad (2)$$

where $\iota(t)$ is the inertial parameter, μ is the cognitive parameter, ν is the social parameter, $\omega_{1j}(t)$ and $\omega_{2j}(t)$ are uniformly random numbers drawn from $(0, 1)$, and $\hat{y}$ represents the best particle in the entire swarm. Thus, $y_j^i(t) - x_j^i(t)$ is the direction to the particle previous best ever position (cognitive direction) and $\hat{y}_j(t) - x_j^i(t)$ is the direction to the swarm best ever position (social direction). The new particle position is then defined by

$$x_j^i(t+1) = x_j^i(t) + v_j^i(t+1), \quad j = 1, \ldots, n.$$

A particle therefore uses the best position encountered by itself and that of the entire swarm to position itself toward an optimal solution. In the traditional versions, the PSO algorithm should only be applied to problems with at most a global minimum. To address the problem of computing all the global and local optima we describe in the next section the multi-local PSO algorithm which is able to compute multiple solutions by making use of descent directions evaluated at each best particle position.

3. MULTI-LOCAL PARTICLE SWARM OPTIMIZER

The multi-local particle swarm optimization (MLPSO) algorithm that is capable of locating multiple solutions of problem (1) is presented in this section. We will make use of two new ideas to avoid the concentration of all the particles in the swarm around the best swarm position. First, we replace the direction to the swarm best ever position, in the velocity equation (2), by the steepest descent direction evaluated at the best ever particle position. Alternatively, we also use an heuristic method to evaluate an approximate descent direction at each particle best position.

To be able to maintain diversity, we avoid the social propagation of information related to a global solution and use each particle personal experience and the steepest descent information to drive a particle to a nearby minimum. The new particle swarm optimization algorithm differs from the original one, in such a way that the social direction is dropped out from equation (2) and the gradient information is used instead. The new equation for the velocity is then

$$v_j^i(t+1) = \iota(t)v_j^i(t) + \mu\omega_{1j}(t)\left(y_j^i(t) - x_j^i(t)\right) + \nu\omega_{2j}(t)\left(-\nabla f_j(y^i(t))\right), \quad j = 1, \ldots, n. \quad (3)$$

The inclusion of the steepest descent direction in the velocity equation (3) aims to drive each particle to a neighbor local minimum and since we have a population of particles, each one will be driven to a local minimum. Global minima are also detected, since they are local minima as well. The inclusion of the gradient into the direction can pose some difficulties to the algorithm, since the computed velocity can make particles to get out of the feasible region. To prevent this behavior, the velocity is scaled to fit the maximum velocity allowed and whenever a particle gets out of the feasible region its position is projected onto it.

The use of the gradient, however, may not be appropriate in the case where it is computationally expensive. So, a derivative-free strategy becomes a method of choice. In what follows we briefly describe a derivative-free heuristic method that is able to produce approximate descent directions at each particle best position.

We use a recent strategy in which a set of m exploring points are used to generate an approximate descent direction at a particular point $y \in R^n$ (Hedar and Fukushima (2004)). Let y^i be the best position of particle i. Based on a set of m points $\{p_k^i\}_{k=1}^m$ close to y^i, the direction w^i defined by

$$w^i = \frac{-1}{\sum_{k=1}^m |f(p_k^i) - f(y^i)|} \sum_{k=1}^m (f(p_k^i) - f(y^i)) \frac{(p_k^i - y^i)}{\|p_k^i - y^i\|}$$

is expected to be a descent direction at y^i when f is a differentiable nonlinear function. Moreover, under certain conditions, the direction w^i simulates the steepest descent direction. We refer to Hedar and Fukushima (2004) for details.

There are two methods to generate the points p_k^i, $k = 1, \ldots, m$. The orthogonal method and the random method. Since the orthogonal method is computationally more expensive, we choose to implement the random method in which $m = 2$ points are randomly generated from a small neighborhood of each y^i, $N(y^i, \epsilon) = \{x \in R^n : \|y^i - x\| \leq \epsilon\}$, for some small $\epsilon > 0$.

4. IMPLEMENTATION DETAILS

In this section we describe some implementation details regarding the implemented algorithm, namely the stopping criteria and the used environment.

The used stopping criterion must account for optimality for all the particles. Using the first order necessary optimality conditions for problem (1) we have that

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = \nabla f(x^*) - \lambda_a^* + \lambda_b^* = 0 \quad (4)$$

where $\mathcal{L}$ is the Lagrangian function and $\lambda_a^* \geq 0$, $\lambda_b^* \geq 0$ are the Lagrange multipliers vectors, at x^*, associated with the lower and upper bound constraints, respectively.

A given velocity for particle i at iteration t, $v^i(t)$, is descent to the objective function f if

$$v^i(t)^T \nabla f(x^i(t)) < 0. \quad (5)$$

By using the strict complementarity condition we have that $(\lambda_b^*)_j > 0$ when $x_j^* = b_j$ and $(\lambda_b^*)_j = 0$ otherwise. The same scenario is valid for the lower bound limits.

Deriving an expression for $\nabla f(x)$ from (4) and using (5) we propose the stopping criterion

$$\max_i [v^i(t)]_{opt} \leq \epsilon_p \tag{6}$$

where

$$[v^i(t)]_{opt} = \left(\sum_{j=1}^{n} \begin{cases} 0 & \text{if } x_j^i(t) = b_j \text{ and } v_j^i(t) \geq 0 \\ 0 & \text{if } x_j^i(t) = a_j \text{ and } v_j^i(t) \leq 0 \\ \left(v_j^i(t)\right)^2 & \text{otherwise} \end{cases} \right)^{1/2}.$$

MLPSO algorithm was coded in the C programming language and connected to AMPL (Fourer et al. (1990), *http://www.ampl.com*) to provide the coded problems. The MLOCP-SOA solver, publicly available in *http://www.norg.uminho.pt/aivaz/*, already includes the gradient version and we plan to include the approximate descent direction version on its next release.

The interface with AMPL allows a great flexibility to users, as parameters can be easily changed and problems can be coded in the AMPL language for a quick run of the solver.

5. NUMERICAL EXPERIMENTS

Below we elaborate on the implementation of our MLPSO algorithm. The algorithm terminates if criterion (6) is met with $\epsilon_p = 0.01$ or the number of iterations exceeds $N_t^{max} = 100000$. Coefficients μ and ν were both set to 1.2. The inertial parameter $\iota(t)$ was linearly scaled from 0.7 to 0.2 over a maximum of N_t^{max} iterations.

The neighborhood radius ε, which is used to generate the exploring points p_k^i is set equal to 10^{-3}.

A set of uni and multi-modal well-known problems were used in the experiments. The test set also includes a nondifferentiable problem in order to check the approximate descent direction version. A list of the test problems is reported in Table 1. Column "Problems" shows the problem name, n is the problem dimension, N_{x^*} is the number of known global optima and f^* is the optimal objective function value. The AMPL models can be requested from the first author.

For each problem, the optimizer was run 5 times with different initial particle positions and velocities (randomly chosen from the search domains). The average number of function evaluations, N_{afe}, the average number of gradient evaluations, N_{age}, and the average best function values (f_a^*) over the 5 runs are reported in Table 2. The term f_{best} indicates the best function value obtained in all runs. *F.O.* represents the percentage of frequency of occurrence, which is given by the ratio between the number of detected solutions and the number of known solutions. The swarm size is given by $\min(6^n, 100)$, where n is the problem dimension.

It is noteworthy that the approximate descent direction version requires more function evaluations than the gradient version but yields higher success rates. Since, in some cases, the norm of the gradient vector is large, the repeated use of the projection procedure in the gradient version of the algorithm maintains the particles on the boundary of the feasible region, thus preventing them to converge to a solution.

6. CONCLUSIONS AND FUTURE WORK

This paper introduces a new multi-local optimization algorithm that evaluates multiple optimal solutions for multi-modal optimization problems. Our MLPSO algorithm adapts

	Problems	n	N_{x^*}	f^*		Problems	n	N_{x^*}	f^*
1	b2	2	1	0.000E+00	17	rosenbrock5	5	1	0.000E+00
2	bohachevsky	2	1	0.000E+00	18	shekel10	4	1	-1.054E+01
3	branin	2	3	3.979E-01	19	shekel5	4	1	-1.015E+01
4	dejoung	3	1	0.000E+00	20	shekel7	4	1	-1.040E+01
5	easom	2	1	-1.000E+00	21	shubert	2	18	-1.867E+02
6	f1	30	1	-1.257E+04	22	storn1	2	2	-4.075E-01
7	goldprice	2	1	3.000E+00	23	storn2	2	2	-1.806E+01
8	griewank	6	1	0.000E+00	24	storn3	2	2	-2.278E+02
9	hartmann3	3	1	-3.863E+00	25	storn4	2	2	-2.429E+03
10	hartmann6	6	1	-3.322E+00	26	storn5	2	2	-2.478E+04
11	hump	2	2	0.000E+00	27	storn6	2	2	-2.493E+05
12	hump_camel	2	2	-1.032E+00	28	zakharov10	10	1	0.000E+00
13	levy3	2	18	-1.765E+02	29	zakharov2	2	1	0.000E+00
14	parsopoulos	2	12	0.000E+00	30	zakharov20	20	1	0.000E+00
15	rosenbrock10	10	1	0.000E+00	31	zakharov4	4	1	0.000E+00
16	rosenbrock2	2	1	0.000E+00	32	zakharov5	5	1	0.000E+00

Table 1: Test functions and characteristics

	Gradient version					Approximate descent direction version			
	F.O.	N_{afe}	N_{age}	f_a^*	f_{best}	*F.O.*	N_{afe}	f_a^*	f_{best}
1	100	3444343	873	0,000E+00	0,000E+00	100	3602386	0,000E+00	0,000E+00
2	100	2782058	545	0,000E+00	0,000E+00	100	3600983	0,000E+00	0,000E+00
3	100	1740823	1397	3,979E-01	3,979E-01	100	3601171	3,979E-01	3,979E-01
4	100	1647820	4420	2,618E-23	0,000E+00	100	10003223	0,000E+00	0,000E+00
5	100	283500	70615	-1,000E+00	-1,000E+00	100	3601354	-1,000E+00	-1,000E+00
6	Not differentiable					100	10104250	-1,448E+04	-1,468E+04
7	20	3600000	59	2,431E+01	4,583E+00	100	3600967	3,000E+00	3,000E+00
8	20	10000000	7754	1,084E-02	0,000E+00	0	10004487	2,257E-02	1,503E-02
9	100	10000000	483	-3,850E+00	-3,861E+00	100	10002098	-3,862E+00	-3,863E+00
10	40	10000000	525	-2,937E+00	-3,185E+00	100	10002652	-3,202E+00	-3,242E+00
11	100	963259	1082	-1,032E+00	-1,032E+00	100	3600946	-1,032E+00	-1,032E+00
12	100	1171181	1329	4,651E-08	4,651E-08	100	3601098	2,362E-06	6,756E-07
13	0	3600000	439	-1,276E+02	-1,592E+02	49	3601052	-1,765E+02	-1,765E+02
14	85	2952979	2295	4,922E-23	3,749E-33	75	3600819	2,607E-07	9,685E-08
15	0	10000000	154	8,051E+04	3,387E+04	0	10009292	8,726E+00	7,386E+00
16	0	3600000	91	3,046E+00	1,190E+00	100	3601268	1,437E-06	5,698E-07
17	0	10000000	177	4,652E+03	2,393E+03	40	10005589	2,203E-01	1,327E-01
18	100	10000000	1850	-9,160E+00	-1,026E+01	100	10004066	-1,052E+01	-1,052E+01
19	100	10000000	2126	-7,801E+00	-8,760E+00	100	10003906	-1,012E+01	-1,014E+01
20	100	10000000	1909	-9,401E+00	-9,997E+00	100	10004069	-1,037E+01	-1,039E+01
21	0	3600000	335	-1,024E+02	-1,648E+02	60	3600999	-1,867E+02	-1,867E+02
22	100	1366222	973	-4,075E-01	-4,075E-01	100	3600804	-4,075E-01	-4,075E-01
23	100	3600000	570	-1,806E+01	-1,806E+01	100	3600902	-1,806E+01	-1,806E+01
24	100	3600000	194	-2,278E+02	-2,278E+02	100	3601003	-2,278E+02	-2,278E+02
25	100	3600000	167	-2,429E+03	-2,429E+03	100	3601160	-2,429E+03	-2,429E+03
26	90	3600000	81	-2,477E+04	-2,478E+04	100	3601278	-2,478E+04	-2,478E+04
27	10	3600000	58	1,607E+05	-2,436E+05	100	3601418	-2,493E+05	-2,493E+05
28	0	10000000	141	4,470E+02	3,102E+01	60	10009759	3,977E-02	2,506E-02
29	0	10000000	135	1,289E+05	7,935E+02	0	10016905	3,633E-01	2,404E-01
30	100	1433664	16314	8,325E-112	0,000E+00	100	3601264	4,987E-07	4,464E-08
31	100	10000000	313	1,997E-13	2,780E-21	100	10005221	2,231E-04	6,612E-05
32	40	10000000	160	8,338E+00	3,031E-04	100	10006065	2,005E-03	1,186E-03

Table 2: Numerical results

the unimodal particle swarm optimizer using descent directions information to maintain diversity and to drive the particles to neighbor local minima so avoiding the concentration of the swarm around a unique global solution. Descent directions are obtained through the gradient vector or an heuristic method to produce an approximate descent direction.

Experimental results indicate that the proposed algorithm is able to evaluate multiple optimal solutions with reasonably success rates.

The use of a properly scaled gradient vector and the optimizer performance analysis on high-dimensional problems are issues under investigation.

A inclusion of the proposed algorithm is planned to help a reduction type method for semi-infinite programming.

7. ACKNOWLEDGMENTS

Work partially supported by FEDER and FCT under grant POCI/MAT/58957/2004, and Algoritmi research center.

8. REFERENCES

Brits, R., Engelbrecht, A., and van den Bergh, F. (2002). A niching particle swarm optimizer. In *4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL2002)*, pages 692–696.

Eberhart, R. and Kennedy, J. (1995). New optimizers using particle swarm theory. In *Proceedings of the 1995 6th International Sysmposium on Micro Machine and Human Science*, pages 39–43.

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

Hedar, A.-R. and Fukushima, M. (2004). Heuristic pattern search and its hibridization with simulated annealing for nonlinear global optimization. *Optimization Methods and Software*, 19(3-4):291–308.

Hettich, R. and Kortanek, K. (1993). Semi-infinite programming: Theory, methods, and applications. *SIAM Review*, 35(3):380–429.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of the 1995 IEEE International Conference on Neural Networks*, pages 1942–1948, Perth, Australia. IEEE Service Center, Piscataway, NJ. http://engr.iupui.edu/∼shi/Coference/psopap4.html.

León, T., Sanmatías, S., and Vercher, E. (1998). A multi-local optimization algorithm. *Top*, 6(1):1–18.

Meng, T., Ray, T., and Dhar, P. (2004). Supplementary material on parameter estimation using particle swarm. *Preprint submitted to Elsevier Science*.

Parsopoulos, K. and Vrahatis, M. (2002). Recent appoaches to global optimization. *Natural Computing*, 1:235–306.

van den Bergh, F. (2002). *An Analysis of Particle Swarm Optimizers*. PhD thesis, Department of Computer Science, University of Pretoria, South Africa.