Particle Swarm Optimization for Minimax Problems

E.C. Laskari, K.E. Parsopoulos and M.N. Vrahatis
Department of Mathematics,
University of Patras Artificial Intelligence Research Center (UPAIRC),
GR-26110 Patras, Greece
{elena, kostasp, vrahatis}@math.upatras.gr

Abstract - This paper investigates the ability of the Particle Swarm Optimization (PSO) method to cope with minimax problems through experiments on well–known test functions. Experimental results indicate that PSO tackles minimax problems effectively. Moreover, PSO alleviates difficulties that might be encountered by gradient–based methods, due to the nature of the minimax objective function, and potentially lead to failure. The performance of PSO is compared with that of other established approaches, such as the Sequential Quadratic Programming (SQP) method and a recently proposed Smoothing Technique; conclusions are derived.

I. INTRODUCTION

In general, the minimax problem can be defined as

$$\min_x f(x), \tag{1}$$

where

$$f(x) = \max_{i=1,\ldots,m} f_i(x), \tag{2}$$

with $f_i(x) : S \subset \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$. Such problems are encountered in numerous optimal control, engineering design, discrete optimization, Chebyshev approximation and game theory applications [7], [8], [34].

Specifically, in Chebyshev approximation, given a function $g : Y^{(0)} \subset \mathbb{R}^m \to \mathbb{R}$, the Chebyshev approximate p_z of g in p_n solves the following minimax problem [34]:

$$\min_z \max_{y \in Y^{(0)}} (g(y) - p_z(y))^2.$$

In game theory, a game is defined as a triple (Y, Z, k) where Y, Z, denotes the spaces of strategies for player I and II, respectively, and k is a real–valued pay–off function of $y \in Y$ and $z \in Z$. Under natural conditions, the optimal strategies for both players solve the saddle point problem [34]:

$$\min_{z \in Z} \max_{y \in Y} k(y, z) = \max_{y \in Y} \min_{z \in Z} k(y, z).$$

In numerous engineering design problems, one is interested in minimizing the largest eigenvalue of an $n \times n$ symmetric matrix–valued function $A(y)$ of a variable y in $\mathbb{R}^n$. Thus, if $\lambda_i(y)$, $i = 1, \ldots, n$, is the i–th eigenvalue of $A(y)$ and by setting $f(i, n) = \lambda_i(y)$, then the following minimax problem is obtained [34]:

$$\min_{y \in Y^{(0)}} \max_{i=1,\ldots,n} f(i, y).$$

Another example is error minimization in the manufacturing of electronic parts, with a prespecified tolerance. Specifically, suppose that when a state z is specified, the process actually produces the state $y+z$ for some y in the tolerance set Z and let $\theta(y+z)$ measure the resulting distortion. Since y is not known in advance, the worst–case distortion should be minimized, leading to the minimax problem [34]:

$$\min_{z \in Z} \max_{y \in Y} \theta(y + z).$$

Moreover, a nonlinear programming problem, with inequality constraints, of the form

$$\begin{aligned} &\min F(x), \\ &\text{subject to } g_i(x) \geqslant 0, \quad i = 2, \ldots, m, \end{aligned} \tag{3}$$

can be transformed into the following minimax problem

$$\begin{aligned} \min_x \max_{1 \leqslant i \leqslant m} f_i(x), \\ f_1(x) &= F(x), \\ f_i(x) &= F(x) - \alpha_i g_i(x), \\ \alpha_i &> 0, \end{aligned} \tag{4}$$

for $2 \leqslant i \leqslant m$. It has been proved that for sufficiently large α_i, the optimum point of the minimax problem, coincides with the optimum point of the nonlinear programming problem [2].

In addition to the above, numerous other applications involve solving minimax problems, justifying the ongoing interest for the development of techniques that can

0-7803-7282-4/02/$10.00 ©2002 IEEE

cope efficiently with it. However, the nature of the minimax objective function $f(x)$ of Eq. (1), may pose difficulties in the process of solving minimax problems. Specifically, at points where $f_j(x) = f(x)$ for two or more values of $j \in \{1, \ldots, m\}$, the first partial derivatives of $f(x)$ are discontinuous, even if all the functions $f_i(x)$, $i = 1, \ldots, m$, have continuous first partial derivatives. This difficulty cannot be addressed directly by the well–known and widely used gradient–based methods, and several techniques have been proposed to cope with it [6], [21], [22]. Moreover, globally optimal solutions are frequently not only desirable but also indispensable.

Sequential Quadratic Programming (SQP) is a common gradient–based approach for solving minimax problems. Starting from an initial approximation of the solution, a Quadratic Programming (QP) problem is solved at each iteration of the SQP method, yielding a direction in the search space. To this direction, a vector is obtained through line search, in order to produce a sufficient decrease of a merit function. This point is considered the new approximation of the solution. Smoothing Techniques work in a very similar manner. Following this approach, a smoothing function, sometimes called the *Exponential Penalty Function* or *Aggregate Function*, is used to approximate the objective function $f(x)$ of Eq. (1), [4], [5], [18], [19]. The smoothing function is minimized through a gradient–based technique with line search. Under strict conditions, line search ensures the global convergence of the algorithm. Recently, a new smoothing function has been proposed in [33], and a quadratic approximation of this function is solved using a gradient–based method with line search.

Gradient–based methods, such as SQP and Smoothing, perform, in general, local minimization. Thus, the quality of the obtained minimizer is heavily dependent on the initial approximation (starting point) of the solution, unless the objective function is convex, twice differentiable, and line search is used. Moreover, derivatives information for the objective function is required, and, thus, if the derivatives are not analytically available, they need to be approximated using finite differences. Unfortunately, in most applications, the only available information regarding the objective function, is its value. Besides, the objective function might be discontinuous. In such cases, gradient–based methods encounter grave difficulties in the process of obtaining satisfactory solutions.

Evolutionary and Swarm Intelligence algorithms are stochastic optimization methods that exploit algorithmic mechanisms similar to natural evolution and social behavior respectively. They can cope with problems that involve discontinuous objective functions and disjoint search spaces. In contrast to gradient–based methods they do not require derivatives information for the objective function, but only its value; and the search is performed simultaneously by many search points [17], [30]. Thus, they are considered as a good and efficient alternative for general global optimization problems.

In this contribution, the PSO's ability to tackle minimax problems is investigated and its performance is compared with that of the SQP algorithm. The rest of the paper is organized as follows: in Sections II and III the workings of the PSO method, the SQP, and Smoothing Techniques are briefly exposed. In Section IV experimental results for well–known test problems are exhibited and Section V is devoted to conclusions.

II. PARTICLE SWARM OPTIMIZATION

PSO is a Swarm Intelligence method for global optimization. It differs from other well–known Evolutionary Algorithms (EA) [3], [9], [13], [17], [30], in that no operators, inspired by evolutionary procedures, are applied on the population to generate new promising solutions. Instead, in PSO, each individual, named *particle*, of the population, called *swarm*, adjusts its trajectory toward its own previous best position, and toward the previous best position attained by any member of its topological neighborhood [15]. In the global variant of PSO, the whole swarm is considered as the neighborhood. Thus, global sharing of information takes place and the particles profit from the discoveries and previous experience of all other companions during the search for promising regions of the landscape. For example, in the single–objective minimization case, such regions possess lower function values than other, visited previously.

Several variants of the PSO technique have been proposed so far, following Eberhart and Kennedy's pioneering work [9], [10], [16], [17]. In our experiments, three global versions of PSO were investigated. All three versions are defined using the same equations, described in the following paragraph [17].

First, let us define the notation adopted in this paper: assuming that the search space is D–dimensional, the i-th particle of the swarm is represented by the D–dimensional vector $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$ and the best particle of the swarm, i.e. the particle with the lowest function value, is denoted by index g. The best previous position (i.e. the position giving the best function value) of the i-th particle is recorded and represented by $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$, and the position change (velocity) of the i-th particle is $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$.

The particles are manipulated according to the following equations (the superscripts denote the iteration number):

$$v_{id}^{n+1} = w v_{id}^{n} + c_1 r_{i1}^{n} (p_{id}^{n} - x_{id}^{n}) + c_2 r_{i2}^{n} (p_{gd}^{n} - x_{id}^{n}), \quad (5)$$

0-7803-7282-4/02/$10.00 ©2002 IEEE

$$x_{id}^{n+1} = x_{id}^{n} + \chi\, v_{id}^{n+1}, \quad (6)$$

where $d = 1, 2, \ldots, D$; N is the swarm's size; $i = 1, 2, \ldots, N$; χ is a *constriction factor* used to control and constrict velocities; w is the *inertia weight*; c_1 and c_2 are two positive constants, called the *cognitive* and *social* parameter respectively; r_{i1}^n and r_{i2}^n are two random numbers uniformly distributed within the range $[0, 1]$.

Eq. (5) is used to calculate the i-th particle's new velocity, at each iteration. Three terms are taken into consideration. The first term, wv_{id}^n, is the particle's previous velocity weighted by the inertia weight w. The second term, $(p_{id}^n - x_{id}^n)$, is the distance between the particle's best previous position, and its current position. Finally, the third term, $(p_{gd}^n - x_{id}^n)$, is the distance between the swarm's best experience, and the i-th particle's current position. The parameters $c_1 r_{i1}^n$, $c_2 r_{i2}^n$, provide randomness that makes the technique less predictable yet more flexible [15]. Eq. (6) provides the new position of the i-th particle, adding its new velocity, to its current position. In general, the performance of each particle is measured according to a fitness function, which is problem–dependent. In optimization problems, the fitness function is usually the objective function under consideration.

The role of the inertia weight w is crucial for the PSO's convergence behavior. The inertia weight is employed to control the impact of the previous history of velocities on the current velocity. Thus, the parameter w regulates the trade–off between the global (wide–ranging) and the local (nearby) exploration abilities of the swarm. A large inertia weight facilitates global exploration (searching new areas), while a small one tends to facilitate local exploration, i.e. fine–tuning the current search area. A proper value for the inertia weight w provides balance between the global and local exploration ability of the swarm, and, thus contributes to improved convergence rates. Experimental results suggest that it is better to initially set the inertia to a large value, in order to promote the global exploration of the search space, and gradually decrease it to obtain refined solutions [31]. Our approach incorporates a time–decreasing inertia weight. The initial population, as well as the velocities, can be generated either randomly or using a Sobol sequence generator [29], which ensures that the D-dimensional vectors will be uniformly distributed within the search space. Some variants of PSO impose a maximum allowed velocity V_{max} to prevent the swarm from explosion. Thus, if $v_{id}^{n+1} > V_{max}$ in Eq. (5), then $v_{id}^{n+1} = V_{max}$ [17].

PSO resembles, to some extent, EA. Although it does not rely on a direct recombination operator, the recombination concept is represented by the stochastic movement of each particle toward its own previous position, as well as toward the global best position of the entire swarm or its neighborhood's best position, depending on the variant of the PSO used [11]. Moreover, PSO's mutation–like behavior is directional, due to the velocity of each particle, with a kind of momentum built in. In other words, PSO is considered as performing mutation with a "conscience", as pointed out by Eberhart and Shi [11].

The PSO technique has proved to be very effective in solving global optimization problems, in static, noisy and, continuously changing environments, [23]–[26], exhibiting competitive results with Evolutionary Algorithms [1]. Moreover, it copes efficiently with Multiobjective Optimization problems [27].

III. THE SQP ALGORITHM AND THE SMOOTHING TECHNIQUE

The SQP algorithm starts from an initial point x_0 and an initial approximation of the Hessian matrix of the objective function, and it consists of three main stages: updating of the Hessian; solving of a QP problem; and line search for obtaining a new potential solution.

Thus, at each iteration, a positive definite quasi–Newton approximation of the Hessian is calculated. In our experiments, the BFGS method was used for that purpose. Thus, the Hessian update is defined as

$$H_{n+1} = H_n + \frac{q_n q_n^\top}{q_n^\top s_n} - \frac{H_n^\top H_n}{s_n^\top H_n s_n}, \quad (7)$$

where $s_n = x_{n+1} - x_n$, and $q_n = \nabla f(x_{n+1})$. After the calculation of the new approximation of the Hessian, the following QP problem is solved in z:

$$\min_z \; q(z) = \frac{1}{2} z^\top H z + c^\top z. \quad (8)$$

The solution, z_n, of the QP problem, is used to find a new potential solution,

$$x_{n+1} = x_n + \alpha_n z_n, \quad (9)$$

where the step length α_n is determined through line search.

Recently, an interesting Smoothing Technique has been proposed in [33] for solving minimax problems. It uses the smoothing function

$$f(x, \mu) = \mu \; \ln \sum_{i=1}^{m} \exp\left(\frac{f_i(x)}{\mu}\right), \quad (10)$$

to approximate the objective function $f(x)$. This function is considered a good approximation of $f(x)$ in the sense that $f(x) \leqslant f(x, \mu) \leqslant f(x) + \mu \ln m$, for $\mu > 0$, as it is mentioned in [33].

The proposed method solves a quadratic approximation of $f(x, \mu)$ for decreasing values of μ. The global

0-7803-7282-4/02/$10.00 ©2002 IEEE

convergence of the algorithm is ensured, under conditions, using Armijo's line–search procedure [28], [32].

A point x^* is a stationary point to the minimax problem defined in Eq. (1), if there exists a vector $y^* = (y_1^*, \ldots, y_m^*)$ such that

$$\sum_{j=1}^{m} y_j^* \nabla f_j(x^*) = 0, \quad (11)$$

$$y_j^* \geqslant 0,\ j = 1, \ldots, m, \quad \sum_{j=1}^{m} y_j^* = 1, \quad (12)$$

$$y_j^* = 0,\ \text{if } f_j(x^*) < \max\{f_1(x^*), \ldots, f_m(x^*)\}. \quad (13)$$

Related to the above, the following theorem has been proved:

Theorem 1 ([7], [33]) If x^* is a local minimum to the problem defined in Eq. (1), then it is a stationary point that satisfies Eqs. (11)–(13). Conversely, assume that $f(x)$ is convex, then if x^* is a stationary point, x^* is a global minimum to the minimax problem.

For an extended theoretical presentation of the aforementioned aspects, as well as the convergence properties of the Smoothing Technique, refer to [33]. Theoretical aspects of the SQP algorithm are reported in [12], [14].

IV. EXPERIMENTAL RESULTS

Three variants of PSO were used in the experiments: one with inertia weight and without constriction factor, denoted as PSO-In; one with constriction factor and without inertia weight, denoted as PSO-Co; and one with both constriction factor and inertia weight, denoted as PSO-Bo.

The performance of the three variants of PSO was investigated on several test problems defined in [6], [20], [30] and [33]. For all experiments, the maximum number of allowed function evaluations was set to 20000; the desired accuracy was 10^{-4}; the constriction factor χ was set equal to 0.729; the inertia weight w gradually decreased from 1 towards 0.1; $c_1 = c_2 = 2$; and $V_{max} = 4$. The aforementioned values for all PSO's parameters are considered default values, and they are widely used in the relevant literature [17]. There was no preprocessing stage that might yield more suitable values for the parameters. For each test problem, 30 experiments were performed, starting with a swarm and velocities uniformly distributed within the range $[-50, 50]^D$, where D is the dimension of the corresponding search space.

For the experiments using the SQP approach, the advanced algorithms implemented in the Optimization Toolbox, Ver. 2, of Matlab© were used with the same accuracy and maximum number of function evaluations as for PSO. For each test problem, 30 experiments starting from a random initial point within the range $[-50, 50]^D$ were performed.

For both approaches, the number of successes, as well as the mean value, the median and the standard deviation of the required number of function evaluations, were recorded. Failure to find the solution, implies that the algorithm was not capable of finding the global minimum and the corresponding minimizer, with the desired accuracy, in the maximum allowed number of iterations.

Four test functions defined in [33] were considered. The first two (denoted as F_1 and F_2), are both 2–dimensional, involving 3 functions $f_i(x)$, and they are defined as follows:

$$\begin{aligned} &\min_x F_1(x), \\ F_1(x) &= \max\{f_i(x)\},\ i = 1, 2, 3, \\ f_1(x) &= x_1^2 + x_2^4, \\ f_2(x) &= (2 - x_1)^2 + (2 - x_2)^2, \\ f_3(x) &= 2\exp(-x_1 + x_2), \end{aligned}$$

and

$$\begin{aligned} &\min_x F_2(x), \\ F_2(x) &= \max\{f_i(x)\},\ i = 1, 2, 3, \\ f_1(x) &= x_1^4 + x_2^2, \\ f_2(x) &= (2 - x_1)^2 + (2 - x_2)^2, \\ f_3(x) &= 2\exp(-x_1 + x_2). \end{aligned}$$

The swarm's size was 20 in both cases. The other 2 test problems (denoted as F_3 and F_4) taken from [33], have the general form of Eq. (3) and they are solved after transforming them to minimax problems, following Eq. (4). They are defined as follows:

$$\begin{aligned} F_3(x) &= x_1^2 + x_2^2 + 2x_3^2 + x_4^2 - 5x_1 - 5x_2 - 21x_3 + 7x_4, \\ g_2(x) &= -x_1^2 - x_2^2 - x_3^3 - x_4^2 - x_1 + x_2 - x_3 + x_4 + 8 \\ g_3(x) &= -x_1^2 - 2x_2^2 - x_3^2 - 2x_4^2 + x_1 + x_4 + 10, \\ g_4(x) &= -x_1^2 - x_2^2 - x_3^2 - 2x_1 + x_2 + x_4 + 5, \end{aligned}$$

and,

$$\begin{aligned} F_4(x) &= (x_1 - 10)^2 + 5(x_2 - 12)^2 + 3(x_4 - 11)^2 + x_3^4 + \\ &\quad +10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7, \\ g_2(x) &= -2x_1^2 - 3x_3^4 - x_3 - 4x_4^2 - 5x_5 + 127, \\ g_3(x) &= -7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 + 282, \\ g_4(x) &= -23x_1 - x_2^2 - 6x_6^2 + 8x_7 + 196, \\ g_5(x) &= -4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7. \end{aligned}$$

These problems are four and seven dimensional with $m = 4$, and $m = 5$, respectively. The swarm's size was 20 and 50 respectively. Note that, although the swarm's size was increased in the latter case, the maximum number of function evaluations remained equal to 20000.

0-7803-7282-4/02/$10.00 ©2002 IEEE

Furthermore, two minimax problems defined in [30] were considered. The first is 2–dimensional (we will refer it as F_5), and the second is 10–dimensional (referred as F_6), and they are defined as follows:

$$\min \max\{|x_1 + 2x_2 - 7|, |2x_1 + x_2 - 5|\},$$

and

$$\min \max\{|x_i|\},\ 1 \leqslant i \leqslant 10.$$

The swarm's size for F_5 was 20, while for F_6 it was 50, while the maximum number of function evaluations remained fixed, at 20000.

Four other test problems were selected from [20] to further investigate the performance of PSO. The name of each problem, its dimension, and the number of $f_i(x)$ functions involved, are reported in Table I. The swarm's

TABLE I
DIMENSION AND NUMBER OF FUNCTIONS $f_i(x)$ FOR THE TEST PROBLEMS F_7–F_{10}.

Function	Dimension	# $f_i(x)$
F_7 (SPIRAL)	2	2
F_8 (POLAK6)	4	4
F_9 (WONG1)	7	5
F_{10} (OET6)	4	21

size for the test problems F_7, F_8, F_{10} was 20, while for F_9 was 50, with maximum number of function evaluations equal to 20000 in all cases.

The results for all test problems are reported in Table II. At this point it is important to note that the solutions obtained by PSO for the test problems F_3 and F_4, are better than the solutions obtained through the Smoothing Technique, reported in [33], in the sense that they satisfy all the constraints of the corresponding nonlinear programming problem, while the solutions obtained using the Smoothing Technique do not satisfy the constraints $g_2(x)$ for F_3, and $g_5(x)$ for F_4, respectively.

V. CONCLUSIONS

The ability of PSO to tackle minimax problems was investigated. Experimental results indicate that PSO is effective in solving minimax problems. Although in less complex problems it was outperformed by the much faster SQP method, in most cases, it exhibited higher success rates. The fact that in cases where SQP failed in all experiments, PSO had success rates higher than 90% (test problems F_4, F_8 and F_9), is impressive. Comparison of the results obtained in some test problems, with the results obtained using the Smoothing Technique, reported in [33], indicate that the quality of the solutions given by PSO was in many cases superior than that of smoothing.

TABLE II
SUCCESS RATE, MEAN NUMBER, STANDARD DEVIATION, AND MEDIAN OF FUNCTION EVALUATIONS, FOR ALL THE TEST PROBLEMS.

Function	Method	Succ.	Mean	St.D.	Median
F_1	PSO-In	30/30	6012.0	1186.9	5780
	PSO-Co	30/30	2348.0	1542.8	1850
	PSO-Bo	29/30	2296.6	3449.8	1370
	SQP	24/30	4044.5	8116.6	56
F_2	PSO-In	30/30	5612.0	409.9	5740
	PSO-Co	30/30	1693.3	282.7	1700
	PSO-Bo	30/30	1534.0	166.7	1510
	SQP	18/30	8035.7	9939.9	61
F_3	PSO-In	30/30	5124.0	545.8	5120
	PSO-Co	30/30	1142.6	260.4	1080
	PSO-Bo	30/30	1022.0	220.9	990
	SQP	30/30	135.5	21.1	132
F_4	PSO-In	29/30	10526.6	2649.7	7750
	PSO-Co	30/30	5150.0	1509.7	3000
	PSO-Bo	28/30	5161.6	4286.9	2500
	SQP	0/30	20000.0	0.0	20000
F_5	PSO-In	30/30	5588.6	349.0	5620
	PSO-Co	30/30	1673.3	225.5	1680
	PSO-Bo	30/30	1432.0	108.3	1440
	SQP	30/30	140.6	38.5	133
F_6	PSO-In	30/30	15398.3	1152.6	15250
	PSO-Co	30/30	10511.6	634.7	10600
	PSO-Bo	28/30	7016.6	3563.7	6100
	SQP	30/30	611.6	200.6	549
F_7	PSO-In	30/30	2534.0	814.8	2600
	PSO-Co	29/30	1790.0	3484.5	1060
	PSO-Bo	30/30	1244.6	1154.0	950
	SQP	10/30	15684.0	7302.4	20000
F_8	PSO-In	30/30	3422.0	2250.8	3170
	PSO-Co	30/30	1026.6	1003.4	780
	PSO-Bo	29/30	1428.0	3523.8	750
	SQP	0/30	20000.0	0.0	20000
F_9	PSO-In	29/30	10306.6	2826.4	9775
	PSO-Co	30/30	5660.0	1522.8	5325
	PSO-Bo	27/30	5371.6	4999.2	3700
	SQP	0/30	20000.0	0.0	20000
F_{10}	PSO-In	24/30	7336.0	6616.4	4220
	PSO-Co	22/30	7882.0	8484.1	1730
	PSO-Bo	17/30	9366.6	9488.2	1970
	SQP	22/30	4886.5	8488.4	229

Regarding the different variants of PSO, the one which utilized only constriction factor (PSO-Co) performed better in terms of the cases where it achieved success rate 100% (8 problems, instead of 7 for PSO-In) and it was always faster than PSO-In. The variant in which both inertia weight and constriction factor were used, was faster than the other two variants, but had the worst success rates among all of them.

Moreover, PSO is very easily implemented and does not require gradient information. Thus, it is unaffected by discontinuities of the objective function, that cannot

0-7803-7282-4/02/$10.00 ©2002 IEEE

be addressed by the gradient–based methods.

Thus, PSO can be considered as a good alternative for solving minimax problems, in cases where the gradient–based techniques fail. If the problem under consideration is a "black–box", and only function values are provided, then using PSO for tackling it, or finding a good approximation of the solution through PSO, and then continue with a faster gradient–based method, such as SQP or Smoothing, may be the proper choice.

VI. ACKNOWLEDGEMENT

Part of this work was done while the authors (K.E.P. and M.N.V.) were at the Department of Computer Science, University of Dortmund, D–44221 Dortmund, Germany. This material was partially supported by the Deutsche Forschungsgemeinschaft–DFG (German National Research Foundation) as a part of the collaborative research center "Computational Intelligence" (SFB 531).

References

[1] P.J. Angeline, "Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and Performance Differences", Evolutionary Programming VII, pp. 601–610, 1998.

[2] J.W. Bandler and C. Charalambous, "Nonlinear Programming Using Minimax Techniques", *J. Optim. Th. Appl.*, Vol. 13, pp. 607–619, 1974.

[3] W. Banzhaf, P. Nordin, R.E. Keller and F.D. Francone, Genetic Programming–An Introduction, Morgan Kaufmann: San Francisco, 1998.

[4] D.P. Bertsekas, "Minimax Methods Based on Approximations", Proc. 1976 John Hopkins Conf. Inform. Sciences and Systems, 1976.

[5] D.P. Bertsekas, "A New Algorithm for Solution of Nonlinear Resistive Networks Involving Diodes", *IEEE Trans. Circ. Th.*, Vol. 23, pp. 599–608, 1976.

[6] C. Charalambous and A.R. Conn, "An Efficient Method to Solve the Minimax Problem Directly", *SIAM J. Numer. Anal.*, Vol. 15, pp. 162–187, 1978.

[7] V.F. Demyanov and V.N. Molozemov, Introduction to Minimax, Wiley: New York, 1974.

[8] D.Z. Du and P.M. Pardalos, Minimax and Applications, Kluwer: Dordrecht, 1995.

[9] R.C. Eberhart, P.K. Simpson and R.W. Dobbins, Computational Intelligence PC Tools, Academic Press Professional: Boston, 1996.

[10] R.C. Eberhart and Y.H. Shi, "Evolving Artificial Neural Networks", Proc. Int. Conf. on Neural Networks and Brain, Beijing, P.R. China, 1998.

[11] R.C. Eberhart and Y.H. Shi, "Comparison Between Genetic Algorithms and Particle Swarm Optimization", Evolutionary Programming VII, pp. 611–615, 1998.

[12] R. Fletcher, Practical Method of Optimization, Vol. 1 & 2, John Wiley and Sons., 1980.

[13] D.B. Fogel, Evolutionary Computation: Toward a New Philosophy of Machine Intelligence, IEEE Press: New York, 1995.

[14] P.E. Gill, W. Murray and M.H. Wright, Practical Optimization, Academic Press, London, 1981.

[15] J. Kennedy, "The Behavior of Particles", Evolutionary Programming VII, pp. 581–587, 1998.

[16] J. Kennedy and R.C. Eberhart, "Particle Swarm Optimization", Proc. of the IEEE International Conference on Neural Networks, Piscataway, NJ, USA, pp. 1942–1948, 1995.

[17] J. Kennedy and R.C. Eberhart, Swarm Intelligence, Morgan Kaufmann Publishers, 2001.

[18] B.W. Kort and D.P. Bertsekas, "A New Penalty Function Algorithm for Constrained Minimization", Proc. 1972 IEEE Conf. Decision and Control, New Orlean, Louisiana, 1972.

[19] X.S. Li, "An Aggregate Function Method for Nonlinear Programming", *Science in China (A)*, Vol. 34, pp. 1467–1473, 1991.

[20] L. Lukšan and J. Vlček, "Test Problems for Nonsmooth Unconstrained and Linearly Constrained Optimization", Technical Report No. 798, Institut of Computer Science, Academy of Sciences of the Czech Republic, 2000.

[21] W. Murray and M.L. Overton, "A Projected Lagrangian Algorithm for Nonlinear Minimax Optimization", *SIAM J. Scient. Stat. Comp.*, Vol. 1, pp. 345–370, 1980.

[22] M.R. Osborne and G.A. Watson, "An Algorithm for Minimax Approximation in the Non–linear Case", *Comput. J.*, Vol. 12, pp. 63–68, 1969.

[23] K.E. Parsopoulos, V.P. Plagianakos, G.D. Magoulas and M.N. Vrahatis, "Objective Function "Stretching" to Alleviate Convergence to Local Minima", *Nonlinear Analysis TMA*, Vol. 47(5), pp. 3419–3424, 2001.

[24] K.E. Parsopoulos, V.P. Plagianakos, G.D. Magoulas and M.N. Vrahatis, "Stretching Technique for Obtaining Global Minimizers Through Particle Swarm Optimization", Proc. of the Particle Swarm Optimization Workshop, Indianapolis (IN), USA, pp. 22–29, 2001.

[25] K.E. Parsopoulos and M.N. Vrahatis, "Modification of the Particle Swarm Optimizer for Locating All the Global Minima", V. Kurkova, N. Steele, R. Neruda, M. Karny (Eds.), Artificial Neural Networks and Genetic Algorithms, Springer: Wien (Computer Science Series), pp. 324–327, 2001.

[26] K.E. Parsopoulos and M.N. Vrahatis, "Particle Swarm Optimizer in Noisy and Continuously Changing Environments", M.H. Hamza (Ed.), Artificial Intelligence and Soft Computing, IASTED/ACTA Press, pp. 289–294, 2001.

[27] K.E. Parsopoulos and M.N. Vrahatis, "Particle Swarm Optimization Method in Multiobjective Problems", ACM SAC 2002 Conference, Madrid, Spain, in press.

[28] E. Polak, Optimization: Algorithms and Consistent Approximations, Springer–Verlag: New York, 1997.

[29] W.H. Press, W.T. Vetterling, S.A. Teukolsky and B.P. Flannery, Numerical Recipes in Fortran 77, Cambridge University Press: Cambridge, 1992.

[30] H.-P. Schwefel, Evolution and Optimum Seeking, Wiley, 1995.

[31] Y. Shi and R.C. Eberhart, "Parameter Selection in Particle Swarm Optimization", EvolutionaryProgramming VII, pp. 591–600, 1998.

[32] M.N. Vrahatis, G.S. Androulakis, J.N. Lambrinos and G.D. Magoulas, "A Class of Gradient Unconstrained Minimization Algorithms with Adaptive Step Size", *J. Comp. Appl. Math.*, Vol. 114, 367–386, 2000.

[33] S. Xu, "Smoothing Method for Minimax Problems", *Comp. Optim. Appl.*, Vol. 20, pp. 267–279, 2001.

[34] S. Zuhe, A. Neumaier and M.C. Eiermann, "Solving Minimax Problems by Interval Methods", *BIT*, Vol. 30, pp. 742–751, 1990.