Journal of Engineering Science and Technology
Vol. 3, No. 1 (2008) 11 - 29
© School of Engineering, Taylor's University College

APPLICATION OF A PRIMAL-DUAL INTERIOR POINT ALGORITHM USING EXACT SECOND ORDER INFORMATION WITH A NOVEL NON-MONOTONE LINE SEARCH METHOD TO GENERALLY CONSTRAINED MINIMAX OPTIMISATION PROBLEMS

INTAN S. AHMAD*[1], VASSILIOS S. VASSILIDIS[2]

[1]Department of Chemical and Environmental Engineering, Faculty of Engineering
University Putra Malaysia, 43400 Serdang, Selangor, MALAYSIA
[2]Department of Chemical Engineering, University of Cambridge
Pembroke Street, Cambridge CB2 3RA, UK
*Corresponding Author: intan@eng.upm.edu.my

Abstract

This work presents the application of a primal-dual interior point method to minimax optimisation problems. The algorithm differs significantly from previous approaches as it involves a novel non-monotone line search procedure, which is based on the use of standard penalty methods as the merit function used for line search. The crucial novel concept is the discretisation of the penalty parameter used over a finite range of orders of magnitude and the provision of a memory list for each such order. An implementation within a logarithmic barrier algorithm for bounds handling is presented with capabilities for large scale application. Case studies presented demonstrate the capabilities of the proposed methodology, which relies on the reformulation of minimax models into standard nonlinear optimisation models. Some previously reported case studies from the open literature have been solved, and with significantly better optimal solutions identified. We believe that the nature of the non-monotone line search scheme allows the search procedure to escape from local minima, hence the encouraging results obtained.

Keywords: Mathematical Programming, Multi-Objective, Non-Linear Programming, Optimisation.

Nomenclatures

F	Objective function
H	Hessian matrix, history list at each penalty parameter level
H_L	Hessian matrix of the Langrangian function
h	Equality constraints as defined in the text
i	Standard index
$\hat{i}$	Operating level of the “Wildcat” line search penalty parameter
J	Jacobian matrix of the equality constraints
j	Standard index
k	Iteration counter
L	Lagrangian function
m	Number of equality constraints
$N_{SOC}^{\max}$	Maximum number of second order corrections
n	Number of variables
n_H	No. of past values stored
n_p	Number of penalty parameter levels
P	“Wildcat” linesearch penalty parameter
p_λ	Search direction for Lagrange multipliers
p_x	Search direction for primal variables x
r	Penalty function parameter
x	Primal variables
x^L	Lower bound on variables x
x^U	Upper bound on variables x

Abbreviations

NLP	Nonlinear programming problem
SOC	Second order correction

Greek Symbols

α	Line search step size
ε	Tolerance
ε_C	Complimentarity tolerance
ε_F	Feasibility tolerance
ε_L	Lagrangian gradient norm tolerance
ε_V	Violation tolerance
γ_μ	Barrier parameter reducing factor
λ	Lagrange multipliers
$\lambda_{i,0}$	Initial Lagrange multiplier
ω	Line start factor
Φ_B	Barrier function
μ	Barrier parameter
μ_{start}	Initial barrier parameter
$\varphi(.)$	Merit function
θ	Step size reduction parameter used in backtracking scheme

1. Introduction

Minimax is an interesting and developing area in optimisation, and receives a lot of attention from the users as well as the problem solvers. It is widely applied in various areas for example in *n*-person games, economics, policy optimisation, finance, science, engineering and design [1, 2, 3, 4]. The main idea behind the minimax problem is basically trying to solve the worst case scenario where each of the function $f_i(x)$ represents a possible scenario [2].

That will help the decision maker to decide the optimal solution base on worst-case scenario. For example, to design an optical element which reflect/transmit energy extremising a functional under the worse case scenario [4], in economic decision making, the multiple objective functions derived from rival decision models are pooled together to choose only one of these models to represent the statistical observations and analysis of the real system [5], in finance, minimax hedging strategy is aimed to minimise the maximum potential hedging error between two consecutive time [1], and in engineering, to validate a model of a fed-batch reactor from collected experimental data [1].

Minimax problem can be represented in mathematical form as

$$\min_{x} F(x) = \max_{1\le i\le m} f_i(x) \tag{1}$$

subject to

$$h(x) = 0 \tag{2}$$

$$x^L \le x \le x^U \tag{3}$$

where $x \in \Re^{n_x}$ and m is a finite number.

The main problem of the minimax problems as stated in equations (1) to (3) lies in its objective contour. All the functions $f_i(x)$ are smooth and continuous, but the $\max f_i(x)$ function will always be non-smooth and discontinuous in its first partial derivative at certain points, the points or segments where there are two or more $f_i(x)$ overlapping, producing kinks in the contour. Therefore, it is impossible to use normal gradient based nonlinear programming methods to minimise $F(x)$. Apart from that, the non-convex and nonlinear characteristics of the objective and constraints make it difficult to obtain global solution. As for linear equality constraints $Ax = b$, it can be simply solved by determining an initial feasible point and the feasibility can be maintained by performing all arithmetic in the null space of A [2].

One subclass of minimax problems to be included here is the discrete minimax or discrete Chebyshev problem [6], where there is absolute function in the problem formulation. The problem is represented as

$$\min_{x} F(x) = \max_{1\le i\le m} |f_i(x)| \tag{4}$$

where $x \in \Re^{n_x}$, m is a finite number and $f_i(x)$ is continuously differentiable. Again, the objective function is not continuously differentiable due to the absolute function as well as the existence of kinks in the contour as discussed before. However, this problem can be reformulated as a normal minimax problem [6] by adding the whole set of inverted $f_i(x)$ to the function to remove the absolute function. The reformulated discrete Chebyshev will become

$$\min_{x} F(x) = \max_{1\leq i\leq m} f_i(x) \tag{5}$$

where now

$$m = \{1, 2, ..., m, m+1, ..., 2m\} \tag{6}$$

and

$$f_{i+m} = (x) = -f_i(x), \quad i = 1, ..., m \tag{7}$$

2. Minimax to NLP Reformulation

Minimax problems of type (1) and (4) can be reformulated to a smooth nonlinear programming (NLP) problem by replacing the functions by constraints using a support variable ε. This extra variable serves as an upper bound of each of functions $f_i(x)$, which means that ε is the maximum of the functions at a certain state, and we are looking for the state that minimises the maximum. The reformulation of (1) is represented as

$$\min_{x,\varepsilon} \varepsilon$$

(8)

subject to

$$f_i(x) \leq \varepsilon, \quad i = 1, 2, ..., m \tag{9}$$

with the dimension of variable space increases to $n_x + 1$. Therefore, instead of minimizing the max function, we are now minimizing the upper bound of the functions [2]. From the new equivalent formulations, the minimax problems are now solvable by any NLP algorithm.

However, as the nature of an NLP problem is highly non-convex and nonlinear, it is difficult to find a global solution for the above problem. Nonetheless, we are still choosing to solve it in NLP reformulation as we have a new method which we believe will perform well on them.

3. Primal-Dual Interior Point Methods for Nonlinear Optimisation

In this paper, we are interested in solving the constrained reformulated minimax nonlinear programming (NLP) problem using primal-dual interior point methods. Generally constrained NLP problems are formulated as

$$\min_{x} f(x) \tag{10}$$

subject to

$$h(x) = 0 \tag{11}$$

$$x^L \le x \le x^U \tag{12}$$

where $x \in \Re^{n_x}$. Any inequality constraints can be reformulated into equalities with appropriately bounded slack variables. Effectively with the use of slack variables the NLP problem becomes a sequence of equality constrained problems with primal-dual method by solving

$$\min \Phi_B = f(x) - \mu \sum_{i=1}^{n_x} \left[\ln(x_i^U - x_i) + \ln(x_i - x_i^L) \right] \tag{13}$$

subject to

$$h(x) = 0 \tag{14}$$

where $\mu > 0$ is the barrier parameter and μ must be chosen such as $\mu \to 0$ so that $x^*(\mu) \to x^*$, x^* is the optimal and feasible solution of NLP problem of equations (10) to (12).

Finally, the constraints are included in a Lagrangian function by using primal-dual equation

$$L(x,\lambda) = \Phi_B(x) + \lambda^T h(x) \tag{15}$$

From the Lagrangian function $L(x,\lambda)$, we obtain a Newton iteration subproblem as

$$\left. \begin{pmatrix} H_L & J^T \\ J & 0 \end{pmatrix} \right|_{(x^{(k)}\lambda^{(k)})} \begin{pmatrix} p_x \\ p_\lambda \end{pmatrix} = - \left. \begin{pmatrix} \nabla_x L(x,\lambda) \\ h(x) \end{pmatrix} \right|_{(x^{(k)}\lambda^{(k)})} \tag{16}$$

where

$$H_L = \frac{\partial f}{\partial x^2} + \sum_{i=1}^{m} \lambda_i \frac{\partial^2 h_i}{\partial x^2} \tag{17}$$

and

$$J = \frac{\partial h}{\partial x} \tag{18}$$

are the Hessian and the Jacobian matrices.

The search directions p_x and p_λ obtained from the solution of the Newton iteration subproblem (16) are then used to generate new points in updating the iteration. The updated points are

$$x^{(k+1)} = x^{(k)} + \alpha p_x \tag{19}$$

$$\lambda^{(k+1)} = \lambda^{(k)} + \alpha p_\lambda \tag{20}$$

and $\alpha \in (0,1]$ is the step cutting size in a linesearch procedure. If the new solution found with $x^{(k+1)}$ and $\lambda^{(k+1)}$ is better than the previous solution with $x^{(k)}$ and $\lambda^{(k)}$ then the full stepsize $\alpha = 1$ is accepted. Otherwise, backtracking procedure will take place where the stepsize is cut to $\theta\alpha$ where $0 < \theta < 1$.

In this work, we are interested in using a novel line search method which was originally developed by Vassiliadis *et al.* [7] as a measure of accepting or rejecting the updated points as in equations (19) and (20) as well as to determine α. The procedure is to be outlined next.

4. A Novel Non-Monotone Line Search Method for Constrained Optimisation

In our case, we are focusing in using merit functions with a penalty parameter to decide on acceptance or rejection of a trial step size. For these, it is very important to choose a suitable representation to balance the objective function with the feasibility of the constraints in a constrained NLP problem.

We have chosen to use the exact l_1-penalty function, where it is represented as:

$$\phi(x,r) = f(x) + \sum_{i=1}^{m} r|h_i(x)| \tag{21}$$

where r is a positive penalty parameter and m is the number of equality constraints present. According to the theory of penalty functions [8], it does not require the penalty parameter r to grow to infinity as at r bigger than a threshold $\bar{r}$, that is as if $r > \bar{r}$, $\bar{r} = \|\lambda^*\|_\infty$ (λ^* is the optimal Lagrange multiplier values associated with each equality constraint), this function will have the same minimum as the problem given in equations (10) to (12) [9, 10]. However, the finite $\bar{r}$ is not know *a priori* and needs to be estimated and adjusted as well as it is unreliable except if x is in the neighbourhood of x^* [11]. Too big a value of r will overpenalise constraint satisfaction causing a severe impact to the line search procedure. Another common problem is the Maratos effect [12] where the convergence of the algorithm is slowed down even when the iterations are close to the optimal solution.

To solve the problem of unknown r and to counter Maratos effect, we are proposing a new line search method which we call the “Wildcat” line search. In our new “Wildcat” line search procedure, we implement non-monotonic line search using a penalty function with varying penalty parameter. It is based on an idea by Panier and Tits [13] who proposed a non-monotone line search technique coupled with SOC for constrained NLP problems to counter the Maratos effect. Here, the merit function is allowed to increase in certain iterations. The crucial parameter in a non-monotone technique is to keep the penalty parameter constant, as otherwise the value will be dissimilar subject to changes of its value. Furthermore, the penalty merit function scheme must be able to update the parameter to reflect the behaviour of Lagrange multiplier; with delay mechanism if the Lagrange multiplier to be decreased, and immediately if it is to be increased.

Our proposed Wildcat line search technique is a coarse-grained non-monotone line search scheme [7]. The procedure is:

Descretise the possible range of penalty parameters between the highest and lowest meaningful number (10^{-16} to 10^{16}) in coarse grain as

$$P_i = 10^{-16+i0.1}, \quad i = 0,1,2,...,n_p \tag{22}$$

where n_p is the number of penalty parameter levels, $n_p = 320$.

Store n_H past values in each penalty parameter level as

$$H_{ij}, \quad j = 1,2,...,n_h, \quad i = 1,2,...,n_p \tag{23}$$

with a relaxed initialisation by setting

$$H_{ij} = \Phi_B(x) + \sum_{k=1}^{n_h} |h_k(x)| + \omega \max\left[1, \Phi_B(x) + P_i \sum_{k=1}^{n_h} |h_k(x)|\right],$$

$$j = 1,2,...,n_h, \quad i = 1.,2.,...,n_p \tag{24}$$

ω used throughout this work is set to $\omega = 0.001$.

Select an operating level i where the operating merit function will be activated for comparison at each iteration. The selection implementation used is

$$\hat{i} = \left\{ \arg \min_{i \in \{1,2,...,n_p\}} P_{\hat{i}} : \max\left\{ \left\|\lambda^{(k)}\right\|_\infty, \left\|\lambda^{(k)} + p_\lambda\right\|_\infty \right\} < P_i \right\} \tag{25}$$

with the infinity norm l_∞ used to measure the Lagrange multiplier values as it is the dual norm of the l_1 norm used in the exact penalty merit function. The "operating" merit function becomes

$$\phi(x; P_{\hat{i}}) = \Phi_B(x) + P_{\hat{i}} \sum_{k=1}^{n_h} |h_k(x)| \tag{26}$$

Accept the step if the primal direction is a descent direction for the merit function

$$\left.\frac{D\phi}{Dx}\right|_{x;P_{\hat{i}}} p_x < 0 \tag{27}$$

The non-differentiability of the absolute values is dealt as used by Mahdavi-Amiri and Bartels [14] with

$$\left.\frac{D\phi}{Dx}\right|_{x,P_{\hat{i}}} = P_{\hat{i}} \sum_{\substack{k=1 \\ k:|h_k(x)|>\varepsilon_V}} \operatorname{sgn}(h_K(x)) \left.\frac{\partial h_k}{\partial x}\right|_x \tag{28}$$

$\varepsilon_V = \varepsilon_F$ where ε_F is the feasibility tolerance for satisfaction of the constraints at the solution. The Armijo's non-monotone acceptance criterion [15] is used to decide on acceptance where

$$\phi\left(x^{(k)} + \alpha p_x ; P_{\hat{i}}\right) \leq \max(H_{\hat{i}}) + \alpha\sigma \left.\frac{D\phi}{Dx}\right|_{x;P_{\hat{i}}} p_x \tag{29}$$

with the stepsize α set to $\alpha = \alpha/2$ and a parameter $\sigma < 1$ where we are using $\sigma = 10^{-6}$. Levenberg-Marquardt scheme will be applied if the criterion (27) is not fulfilled. The Lagrangian Hessian block of the Lagrange-Newton matrix is incremented by adding 10^{-6} initially and a factor of 10 consecutively until the descent condition is met.

Implement second-order correction (SOC) steps as in the work of Wachter and Biegler [16] every time a step size not accepted before backtracking procedure to counter the Maratos effect. The number of maximum SOC steps allowed is $N_{\max}^{SOC}$.

Update the history lists for all parameter values after a step is accepted.

The Wildcat line search is implemented in a sparse NLPOPT1 solver which is based on primal-dual interior point method as described earlier. All the programming and tests were done using Mathematica$^{\text{TM}}$ version 5.0. Mathematica is an intuitive, equation based optimisation model description modeling interface and is capable to handle parsing and translation. The evaluation of first and second derivatives given are exact and in sparse format using symbolic incidence-driven differentiation. A feature within the platform of Mathematica$^{\text{TM}}$ 5.0 allows symbolic vector expressions to be compiled on-the-fly in memory to increase the processing speed for large realistic problems with speeds approaching those of FORTRAN or C++. Further details of the implementation can be found in our paper [7].

5. Minimax Optimisation Case Studies and Computational Results

In this section we present a number of selected case studies that have been solved with NLPOPT1 using the new line search algorithm. The case studies are all extracted from Luksan and Vlcek's Test Problems for Nonsmooth Unconstrained and Linearly Constrained Optimisation [17]. All the problems are solved using the Pentium IV 224 MB RAM and 1.19 GHz CPU clock. We have grouped the case studies into Type I and Type II problems, where Type I consists of normal minimax problems of cases 1 to 18 and Type II the Chebyshev minimax problems of cases 19 to 25 and are presented in the Appendix. There is several numbers of problems where we modified the structure of the problems to ease the parsing and translation in the solver. This will be noted in the Appendix as well where appropriate. The main parameter settings are as presented in Table 1 and any changes to the values outlined will be highlighted.

Table 1. Parameters Setting for NLPOPT1.

Parameters	**Symbols**	**Values for Type I**	**Values for Type II**
No. of past values stored	n_h	5	5
Feasibility tolerance	ε_F	10^{-8}	10^{-8}
Lagrangian gradient norm tolerance	ε_L	10^{-8}	10^{-8}
Complimentarity tolerance	ε_C	10^{-10}	10^{-16}
Line start factor	ω	10^{-3}	10^{-3}
Maximum number of SOC	$N_{\max}^{SOC}$	2	2
Initial barrier parameter	μ_{start}	0.01	0.01
Barrier parameter reducing factor	γ_{μ}	10	10
Initial Lagrange multiplier	$\lambda_{i,0}$	1	1

The results for the optimisation procedure on the selected problems are reported in tabular form. Tables 2, 3 and 4 summarise the problem sizes. Numbers of variables, lower and upper bounds, non-zeros in Jacobian, non-zeros in Hessian and non-zeros in Lagrange-Newton overall matrix in these tables include the automatically generated slack variables for inequality constraints. Tables 5, 6, 7 and 8 give a concise overview of the solution statistics. For these tables, the number of NLP cycles is based on the equality-constrained (inner) problem solution, functions evaluations are calculated as number of NLP cycles + line searches + SOC cycles, Jacobian evaluations are calculated as number of Newton's + SOC cycles, Hessian evaluations are calculated as number of Newton's, and Hessian factorisation evaluations are calculated as number of Newton's + SOC cycles + Hessian correction cycles.

However there are a few case studies of interest that we would like to highlight further.

Table 2. Summary of the Problems Sizes (I).

	Case study number								
Number of	1	2	3	4	5	6	7	8	9
Variables	6	6	5	10	9	9	13	20	39
Constraints	3	3	2	6	4	4	5	9	18
Equality constraints	0	0	0	0	0	0	0	0	0
Inequality constraints	3	3	2	6	4	4	5	9	18
Lower bounds	0	0	0	0	0	0	0	0	0
Upper bounds	3	3	2	6	4	4	5	9	18
Linear constraints	0	0	0	2	0	0	0	0	0
Nonlinear constraints	3	3	2	4	4	4	5	9	18
Non-zeros in Jacobian	12	12	8	29	24	24	45	108	396
Non-zeros in Hessian	8	6	7	12	9	15	17	22	41
Non-zeros in Lagrange-Newton overall matrix	32	30	23	70	57	63	107	238	833

Table 3. Summary of the Problems Sizes (II).

	Case study number							
Number of	10	11	12	13	14	15	16	17
Variables	22	7	7	7	178	57	20	39
Constraints	10	4	4	4	172	49	9	18
Equality constraints	0	0	0	0	2	41	0	0
Inequality constraints	10	4	4	4	170	8	9	18
Lower bounds	0	1	1	2	7	8	0	0
Upper bounds	10	3	3	3	163	8	9	18
Linear constraints	0	1	1	1	9	9	3	4
Nonlinear constraints	10	3	3	3	163	40	6	14
Non-zeros in Jacobian	130	13	13	13	1490	424	87	328
Non-zeros in Hessian	22	9	9	9	178	753	22	41
Non-zeros in Lagrange-Newton overall matrix	282	35	35	35	3158	1601	196	697

Table 4. Summary of the problems sizes (III).

	Case study number							
Number of	18	19	20	21	22	23	24	25
Variables	16	97	67	38	89	68	69	96
Constraints	7	76	63	33	84	63	63	90
Equality constraints	0	0	21	11	42	21	21	30
Inequality constraints	7	76	42	22	42	42	42	60
Lower bounds	8	10	1	1	0	0	1	0
Upper bounds	15	76	42	22	42	42	42	60
Linear constraints	4	0	43	22	63	42	43	60
Nonlinear constraints	3	76	20	11	21	21	20	30
Non-zeros in Jacobian	37	1672	207	121	273	231	248	360
Non-zeros in Hessian	38	97	73	84	89	72	189	216
Non-zeros in Lagrange-Newton overall matrix	112	3441	487	326	635	534	685	936

Table 5. Summary of the solution statistics (I).

	Case study number						
	1	**2**	**3**	**4**	**5**	**6**	**7**
Translation time (s)	0	0.02	0.02	0.01	0	0.06	0.04
Solution time (s)	0.76	1.21	2.46	0.91	0.93	1.25	1.11
Optimal objective	1.952225	0	0	3.599719	-44	-44	680.63006
Reported best objective	1.952225	0	0	3.599719	-44	-44	680.63006
$\|\lambda^*\|_\infty$	0.5696	0.5025	0.5000	0.8767	0.7000	0.7000	0.8492
Final penalty parameter used	0.6310	0.6310	0.5012	1.0000	0.7943	0.7943	1.0000
NLP cycles	9	9	9	9	9	9	9
Newton's cycles	29	26	74	23	24	30	30
Line search cycles	23	22	88	16	18	25	25
SOC cycles	0	1	32	0	1	4	5
Hessian corrections	0	5	0	0	0	0	0
Function evaluations	32	32	129	25	28	38	39
Jacobian evaluations	29	27	106	23	25	34	35
Hessian evaluations	29	26	74	23	24	30	30
Hessian factorisation evaluations	29	32	106	23	25	34	35

Table 6. Summary of the solution statistics (II).

	Case study number					
	8	**9**	**10**	**11**	**12**	**13**
Translation time (s)	0.08	0.51	0.31	0.01	0.01	0.01
Solution time (s)	1.34	2.09	2.48	1.31	1.45	1.90
Optimal objective	24.30621	133.72990	3.70348	-0.389659	-0.330357	-0.448911
Reported best objective	24.30621	133.72825	261.08258	-0.389659	-0.330357	-0.448911
$\lVert\lambda^*\rVert_\infty$	0.5813	0.4769	0.7243	0.5854	1.0000	1.6126
Final penalty parameter used	0.6310	0.5012	0.7943	0.6310	1.0000	1.9953
NLP cycles	9	9	9	9	9	9
Newton's cycles	30	33	39	29	27	34
Line search cycles	26	35	32	21	20	28
SOC cycles	0	12	0	0	0	0
Hessian corrections	0	0	0	0	0	0
Function evaluations	35	56	41	30	29	37
Jacobian evaluations	30	45	39	29	27	34
Hessian evaluations	30	33	39	29	27	34
Hessian factorisation evaluations	30	45	39	29	27	34

Table 7. Summary of the solution statistics (III).

	Case study number					
	14	**15**	**16**	**17**	**18**	**19**
Translation time (s)	12.67	37.02	0.071	0.45	0	0.88
Solution time (s)	23.07	4.477	1.40	2.52	1.74	8.15
Optimal objective	0.040133	0	24.306209	133.72828	7049.248	0.506948
Reported best objective	0.101831	0	24.306209	133.72828	7049.248	0.506948
$\lVert\lambda^*\rVert_\infty$	0.3518	0.3384	1.7165	1.7038	5210.67	0.0736
Final penalty parameter used	0.3981	1.0000	1.9953	1.9953	6309.57	0.0794
NLP cycles	9	9	9	9	9	15
Newton's cycles	40	33	31	40	32	53
Line search cycles	33	36	26	35	37	44
SOC cycles	2	17	1	6	24	11
Hessian corrections	2	2	0	0	0	0
Function evaluations	44	62	36	50	70	70
Jacobian evaluations	42	50	32	46	56	64
Hessian evaluations	40	33	31	40	32	53
Hessian factorisation eval.	44	52	32	46	56	64

Table 8. Summary of the solution statistics (IV).

	Case study number					
	20	**21**	**22**	**23**	**24**	**25**
Translation time (s)	0.26	0.08	0.19	0.20	0.17	0.22
Solution time (s)	5.63	1.89	7.71	5.54	3.91	7.07
Optimal objective	0.004202	0.008084	0.002636	0.002016	0.000122	0.022340
Reported best objective	0.004202	0.008084	0.002636	0.002016	0.000122	0.022340
$\lVert\lambda^*\rVert_\infty$	0.3248	1.2362	0.6499	0.2906	0.2905	0.2645
Final penalty parameter used	0.3981	1.2589	0.7943	0.3162	0.3162	0.3162
NLP cycles	15	16	15	15	15	15
Newton's cycles	45	43	57	52	35	48
Line search cycles	36	30	48	39	24	38
SOC cycles	11	0	13	2	6	7
Hessian corrections	0	0	0	5	0	0
Function evaluations	62	46	78	56	45	60
Jacobian evaluations	56	43	70	54	41	55
Hessian evaluations	45	43	57	52	35	48
Hessian factorisation evaluations	56	43	70	59	41	55

Highlight I: Problem 10 (Polak 3)

This case study is of particular interest as we managed to find a better solution than previously reported by Luksan and Vlcek [17]. The problem formulation is as presented in the Appendix.

From the experiment, it is found that NLPOPT1 with Type I solver parameter settings as in Table 1 was able to solve the problem effectively. We found a new optimal solution of 3.70348 as compared to 261.08258 by Luksan and Vlcek [17]. The starting values and the optimum values obtained are as detailed in Table 9.

To show the computational details from the exercise, some convergence plots and run details are included. The convergence plots are as in Figures 1, 2 and 3. The details of the run are presented in Tables 3 and 6.

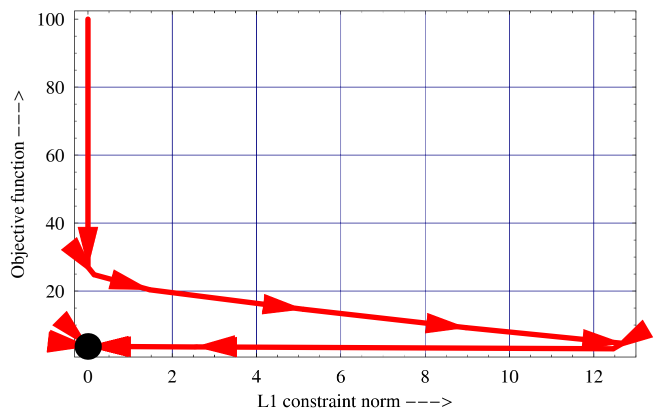

Fig. 1. Objective Function versus Constraint Norm Plot for Problem 10.

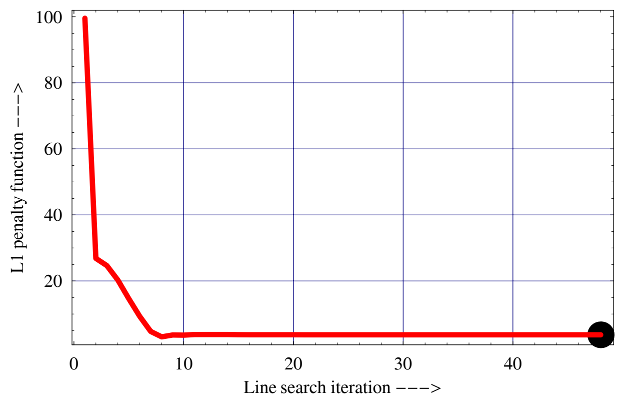

Fig. 2. Penalty Function Evolution Plot for Problem 10.

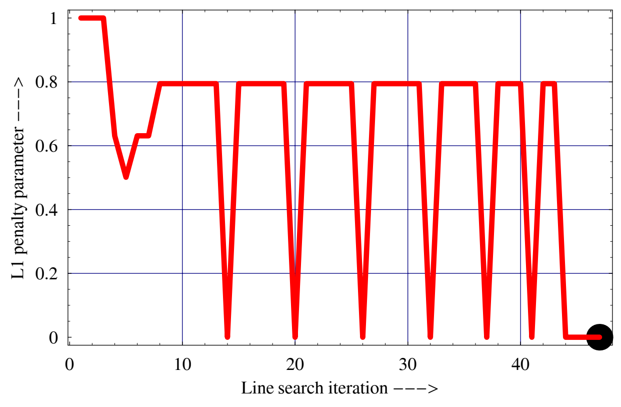

Fig. 3. Penalty Parameter Evolution for Problem 10.

Table 9. Starting and optimum values for Problem 10.

	x_1	x_2	x_3	x_4	x_5	x_6
Starting value	1	1	1	1	1	1
Optimum value	0.01245	0.29069	-0.33461	-0.12645	0.23281	-0.27654

	x_7	x_8	x_9	x_{10}	x_{11}	ε
Starting value	1	1	1	1	1	100
Optimum value	-0.16658	0.22909	-0.18291	-0.17044	0.24018	3.70348

In the tables to follow, the feasibility norm value used is $\|\bullet\|_\infty$. The line search iteration axes for the convergence plots are including all line searches, across all NLP and Newton's subproblems, along with NLP initial values. It is also important to note that for the penalty parameter evolution plots (e.g. Figure 3), 0 for the penalty parameter does not mean the penalty parameter is actually 0 at the specific iteration. It is a way to indicate the initialisation of a new NLP cycle.

Highlight II: Problem 14 (MAD6)

This is another problem that we would like to highlight as we managed to obtain a better optimal solution than reported by previous researchers. The problem formulation is as presented in the Appendix.

The problem was solved using the preset solver settings Type I given in Table 1, initial variables as in Table 10, and initial Lagrange multipliers for each constraint set as $\lambda_{i,start} = 1.0$.

Comparatively, the optimum objective value found with our NLPOPT1 is slightyly better than as obtained by Luksan [18] and Luksan and Vlcek [17] where we found $F(x) = 0.040133$ as opposed to the one reported as $F(x) = 0.101831$.The new optimum values are as summarised in Table 10. The convergence plots are presented in Figures 4, 5 and 6 respectively. Finally, the summary of the sizes and the solution details can be found in Tables 3 and 7 respectively. In Figure 6, 0 is not a value but the initiatialisation of a new NLP.

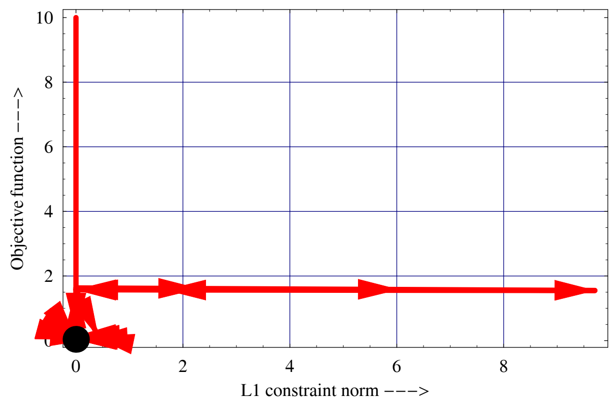

Fig. 4. Objective Function versus Constraint Norm Plot for Problem 14.

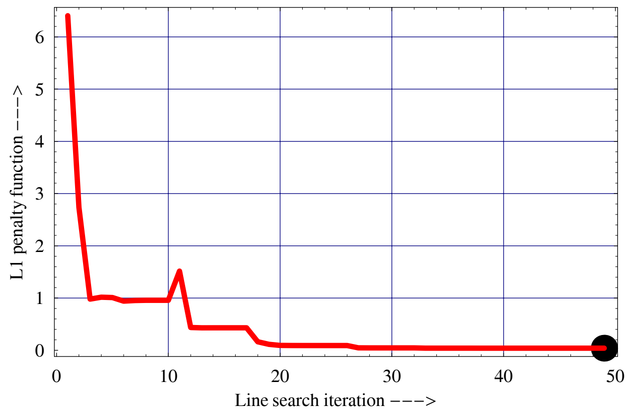

Fig. 5. Penalty Function Evolution for Problem 14.

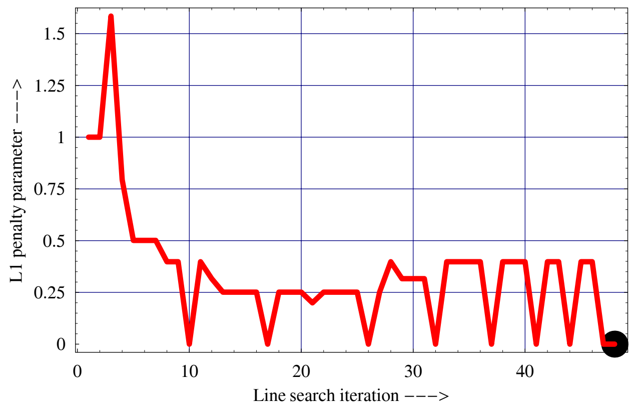

Fig. 6. Penalty Parameter Evolution for Problem 14.

Table 10. Starting and Optimum Values for Problem 14.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	ε
Starting value	0.5	1.0	1.5	2.0	2.5	3.0	3.5	10.0
Optimum value	0.595	0.995	1.467	1.867	2.360	2.867	3.500	0.0401331

Highlight III: Problem 22 (OET5)

This third case study to be highlighted is chosen due to its slightly difficult converging behaviour. It took extra number of iterations before converging to the optimal solution. The problem formulation is as presented in the Appendix.

This is again solved using the Type I NLPOPT1 settings and initial dual variables as in Table 1. The starting ε used was $\varepsilon_{start} = 9$. Even with slightly more number of iterations in the initial NLP cycle, we managed to get an optimum solution which is as good as reported by Luksan and Vlcek [17] at $F(x) = 2.6359735 \times 10^{-3}$. The details of the computations are given in Figures 7, 8 and 9, and in Tables 4 and 8. Again, readers are reminded that the zeros observed in Figure 6 indicate the initialisation of a new NLP cycle, and not a value of the actual penalty parameter used.

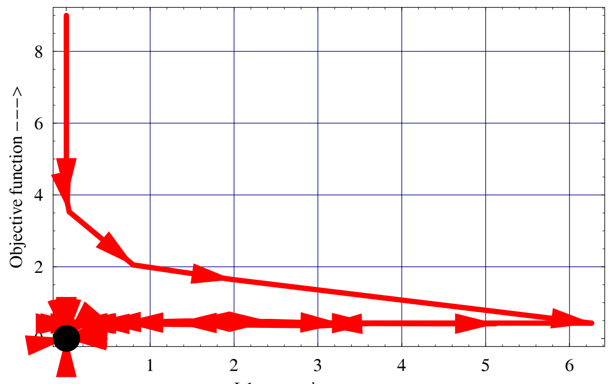

Fig. 7. Objective function against constraint norm plot for Problem 22.

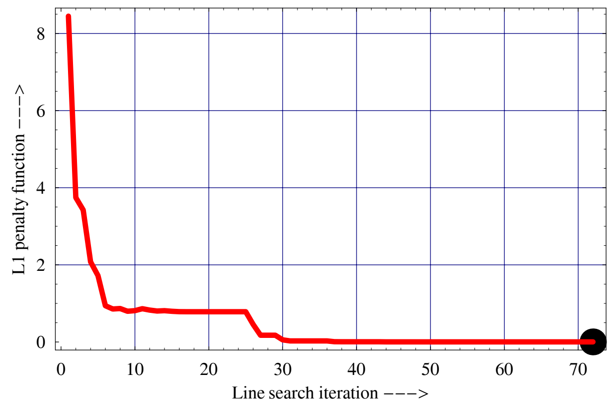

Fig. 8. Penalty function evolution for Problem 22.

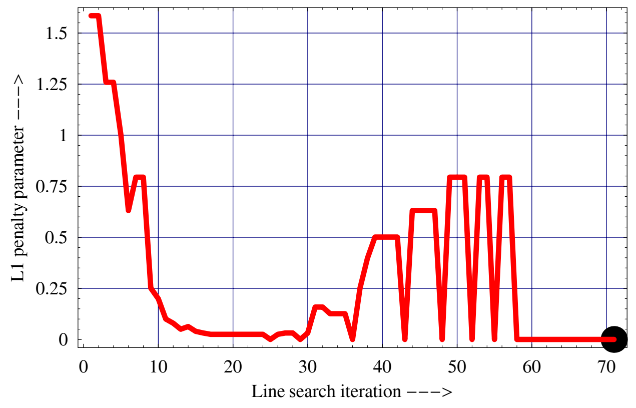

Fig. 9. Penalty Parameter Evolution for Problem 22.

Table 11. Starting and Optimum Values for Problem 22.

	x_1	x_2	x_3	x_4	ε
Starting value	1	1	1	1	9
Optimum value	-0.0880	0.4954	-1.1186	1.5032	0.0026359735

6. Conclusions and Future Work

The focal point of this work has been the dedicated study of minimax optimisation models by reformulation into standard nonlinear optimisation models and the application of a novel interior point method to solve them. In particular, the novelty in our approach lies in the use of a new non-monotone line search scheme which not only has the necessary algorithmic ingredients to overcome the Maratos effect, but also due to its non-monotonicity it can potentially lead to better solutions for difficult optimisation problems.

Specifically, the reformulation of minimax problems into an NLP model can suffer from non-convexity issues, in the case of nonlinear objectives and constraints. The new algorithm has been shown to be very promising in terms of addressing this important class of optimisation problems, and the computational results reported indicate that it is possible with its use to obtain improved solutions to problems found in the open literature.

Future work will focus on improvement of our implementation of the algorithm, as well as detailed theoretical studies to derive properties under which convergence might be guaranteed.

Acknowledgments

We would like to thank the Department of Public Services, Malaysia and University of Putra, Malaysia for the scholarship awarded to the first author.

References

1. Asprey, B., Rustem, S. & Zakovic, S. (2001). Minimax algorithms with applications in finance and engineering. In *Modeling and Control of Economic Systems* (R. Neck, editor). Pergamon Press, Oxford.
2. Erdmann, G. D. (2003). *A New Minimax Algorithm and Its Applications to Optics Problems*. PhD thesis, School of Mathematics, University of Minnesota, Minnesota, USA.
3. Wrobel, M. (2003). *Minimax Optimization without Second Order Information*. PhD thesis, Technical University of Denmark, Denmark.
4. Erdmann, G., and Santosa, F. (2004). Minimax design of optically transparent and reflective coatings. *J. Opt. Soc. Am. A*, 21(9), 1730-1739.
5. Rustem, B. (1998). *Algorithms for Nonlinear Programming and Multiple-Objective Decisions*. John Wiley and Sons, 1998.
6. Conn, A., & Li, Y. (1988). The computational structure and characterization of nonlinear chebyshev problems. Technical Report 88-956, Dept. of Computer Science, Cornell University, New York, USA.
7. Vassiliadis, V. S., Ahamad, I. S., & Conejeros, R. (2006). A novel non-monotone line search method for constrained nonlinear programming: Algorithmic concepts and preliminary computational studies. *Ind. Eng. Chem. Res.,* 45(25), 8270-8281.
8. Han, S. & Mangasarian, O. (1979). Exact penalty functions in nonlinear programming. *Mathematical Programming*, 17, 251-269.
9. Gill, P. E., Murray, W. & Wright, M. H. (1988). *Practical Optimization*. Academic Press, San Diego.
10. Herskovits, J. (1996). *A View on Nonlinear Optimization*. Kluwer.
11. Byrd, R. Nocedal, H. J. and Waltz, R. A. Steering exact penalty methods. Technical report, Northwestern University.
12. Maratos, N. (1978). Exact *Penalty Function Algorithms for Finite Dimensional and Optimization Problems*. PhD thesis, University of London, Imperial College, London, U.K.
13. Panier, E. & Tits, A. (1991). Avoiding the maratos effect by means of a nonmonotone line search i. general constrained problems. *SIAM J. Num. Anal.*, 28, 1183-1195.

14. Mahdavi-Amiri, N. & Bartels, R. (1989). Constrained nonlinear least squares: An exact penalty approach with projected structured quasi-Newton updates. *ACM Transactions on Mathematical Software*, 15, 220-242.
15. Armijo, L. (1966). Minimization of functions having lipschitz-continuous first partial derivatives. *Pacific Journal of Mathematics*, 16, 1-3.
16. Wachter, A. & Biegler, L. (2004). On the implementation of an interior-point _lter line-search algorithm for large-scale nonlinear programming. Technical Report RC 23149, IBM T.J. Watson Research Center, Yorktown, USA.
17. Luksan, L. & Vlcek, J. (2000). Test problems for nonsmooth unconstrained and linearly constrained optimization. Technical Report 798, Academy of Sciences of the Czech Republic, Prague, Czech Republic.
18. Luksan, L. (1985). An implementation of recursive quadratic programming variable metric methods for linearly constrained nonlinear minimax approximation. *Kybernetika*, 21(1), 22-40.