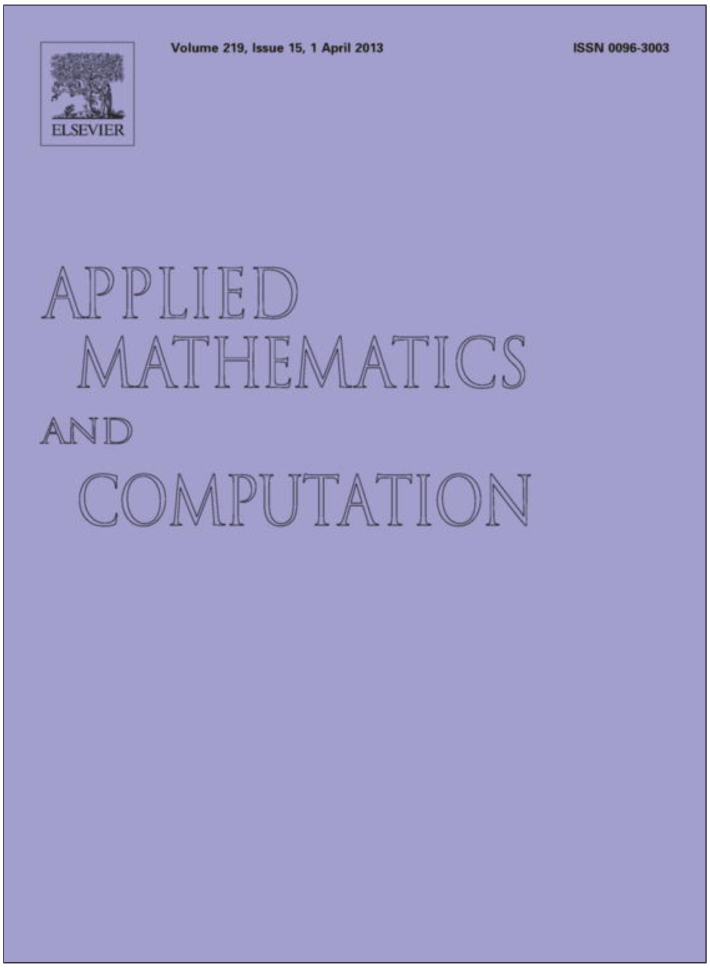
Volume 219, Issue 15, 1 April 2013
ISSN 0096-3003
ELSEVIER
APPLIED MATHEMATICS AND COMPUTATION

Applied Mathematics and Computation 219 (2013) 8033–8041

Contents lists available at SciVerse ScienceDirect

Applied Mathematics and Computation

journal homepage: www.elsevier.com/locate/amc

An adaptive nonmonotone trust-region method with curvilinear search for minimax problem ☆

Fu-Sheng Wang *, Chuan-Long Wang

Department of Mathematics, Taiyuan Normal University, Taiyuan 030012, PR China

ARTICLE INFO

Keywords:
Minimax problem
Nonlinear programming
Curvilinear search
Nonmonotone trust-region
Second-order correction

ABSTRACT

In this paper we propose an adaptive nonmonotone algorithm for minimax problem. Unlike traditional nonmonotone method, the nonmonotone technique applied to our method is based on the nonmonotone technique proposed by Zhang and Hager [H.C. Zhang, W.W. Hager, A nonmonotone line search technique and its application to unconstrained optimization, SIAM J. Optim. 14(4)(2004) 1043–1056] instead of that presented by Grippo et al. [L. Grippo, F. Lampariello, S. Lucidi, A nonmonotone line search technique for Newton's method, SIAM J. Numer. Anal. 23(4)(1986) 707–716]. Meanwhile, by using adaptive technique, it can adaptively perform the nonmonotone trust-region step or nonmonotone curvilinear search step when the solution of subproblems is unacceptable. Global and superlinear convergences of the method are obtained under suitable conditions. Preliminary numerical results are reported to show the effectiveness of the proposed algorithm.

1. Introduction

Consider the minimax problem

$$\min_{x\in\mathbf{R}^n} F(x) = \max_{i\in P} f_i(x), \tag{1.1}$$

where $P = \{1, 2, \ldots, p\}, f_i : \mathbf{R}^n \to \mathbf{R}, i \in P$ are continuously differentiable.

This problem is a nonsmooth optimization problem because the objective function $F(x)$ contains a "max" operator which is not differentiable. Therefore, many unconstrained optimization algorithms with the use of derivatives can not be applied to solve the problem (1.1) directly.

The minimax problem (1.1) is equivalent to the following nonlinear programming problem with $n+1$ variables.

$$\begin{cases} \min\limits_{(x,z)\in\mathbf{R}^{n+1}} z, \\ s.t. \quad f_i(x) - z \leqslant 0, \quad i \in P, \end{cases} \tag{1.2}$$

where $x \in \mathbf{R}^n$ and $z \in \mathbf{R}$. From problem (1.2), the first order optimal conditions of problem (1.1) is obtained as follows:

☆ This work is supported by the National Natural Science Foundation of China (11171250,11071184); The Natural Science Foundation of Shanxi Province of China (2011011002-2).

* Corresponding author.
E-mail address: fswang2005@163.com (F.-S. Wang).

0096-3003/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.amc.2013.01.071

$$\begin{cases} \sum_{i=1}^{m} \lambda_i^* \nabla f_i(x^*) = 0, \\ \sum_{i=1}^{m} \lambda_i^* = 1, \\ \lambda_i^* \geqslant 0, \quad i \in P, \\ \lambda_i^*(f_i(x^*) - F(x^*)) = 0, \quad i \in P. \end{cases} \tag{1.3}$$

where $\lambda^* = (\lambda_1^*, \ldots, \lambda_p^*)^T$ is the corresponding Lagrange multiplier, and x^* is an optimal solution of problem (1.1). The point x^* satisfied (1.3) is defined as a K-T point of the original problem (1.1).

However, Polak et al. [2] pointed out that there are some drawbacks to directly solve the form (1.2) using smooth SQP type algorithms. To tackle these drawbacks, there are several smooth approaches that have been developed to exploit the Kuhn–Tucker optimality conditions for this problem by solving either unconstrained subproblems or constrained subproblems [3]. For example, Zhu [4] developed a line search algorithm for minimax problems, and Xue [5,6] developed a class of Newton-like algorithms, Zhou [7] proposed a nonmonotone SQP line search method with second-order correction. Recently, Jian et al. [8] proposed a SQP algorithm. Typically, under mild assumptions, these algorithms have a locally superlinear convergence.

Trust-region methods are powerful methods that can lead to the strong global convergence (see [9,15,22–30]). However, unlike the smooth unconstrained optimization, the trust-region methods for nonsmooth optimization, in general, have the drawback of the Maratos effect, which badly slow down the rate of convergence. An approach based on trust-region strategy was proposed in [10,11] for a class of composite optimization problems, which can be applied to solve minimax problems. To overcome the Maratos effect, this approach employed the second-order correction step, and thus the rate of superlinear convergence was obtained by Yuan [12]. Another trust-region based algorithms for minimax problems were developed in [13,14], which were combined trust-region methods with line or curve search methods. This technique is similar to the technique of combining trust-region methods with line search methods [15], which can avoid repeatedly solving the trust-region subproblems in each iteration, and thus greatly decrease in computation work. Recently, Ye et al. [16] proposed a trust region Newton-CG method for minimax problem that used a new smoothing technique and solved a smooth unconstrained subproblem in each iteration. Numerical results show that the algorithm is efficient.

Grippo et al. [1] first introduced a nonmonotone line-search technique for Newton's method, in which the stepsize α_k satisfies the following condition:

$$f(x_k + \alpha_k d_k) \leqslant \max_{0 \leqslant j \leqslant m(k)} f(x_{k-j}) - \beta \alpha_k g_k^T d_k. \tag{1.4}$$

which $\beta \in (0,1), 0 \leqslant m(k) \leqslant \min\{m(k-1)+1, M\}$, and M is a fixed nonnegative integer. Since then many authors generalized the nonmonotone technique to different methods and proposed various nonmonotone line search algorithms or nonmonotone trust region algorithms, such as Mo et al. [9], Zhang and Zhang [23], Zhang [24], Deng et al. [26], Toint[27], Sun [28]. Dai [17] and Toint[27] pointed out that the nonmonotone method can enhance the possibility of finding a global optimum. Furthermore, it can improve the rate of convergence in cases where a monotone scheme is forced to creep along the bottom of a narrow curved valley. In addition, as is shown in Panier and Tits[21], the Moratos effect in the context of smooth constrained optimization can also be avoided by means of nonmonotone technique, and Zhou and Tits[7], Yu and Gao[20] generalized the nonmonotone technique to minimax problems based on line search method accordingly. Recently, Wang and Wang [13] generalized the nonmonotone technique to minimax problem based on trust region method. Theoretical analysis and lots of numerical results indicated that these algorithms with a nonmonotone scheme were more efficient than those with a monotone scheme.

Although these nonmonotone techniques based on (1.4) work well in many cases, there are some drawbacks (see [18]). First, a good function value generated in any iteration is essentially discarded due to the max function in (1.4). Second, in some cases, the numerical performance is very dependent on the choice of M (see [19] and [27]). Furthermore, Dai [17] has presented an example to show that for a strongly convex function, although an iterative method is generating R-linearly convergent sequence, the iterates may not satisfy the condition (1.4) for k sufficiently large, for any fixed bound M on the memory.

In order to overcome these disadvantages, Zhang and Hager [18] proposed an improved nonmonotone line search technique, they replaced the "max" function value in (1.4) with a weighted average of the successive function values, which find a stepsize α_k satisfying the following inequality

$$f(x_k + \alpha_k d_k) \leqslant C_k - \beta \alpha_k g_k^T d_k, \tag{1.5}$$

where

$$C_k = \begin{cases} f(x_0), & k = 0, \\ (\eta_{k-1} Q_{k-1} C_{k-1} + f_k)/Q_k, & k \geqslant 1, \end{cases} \tag{1.6}$$

$$Q_k = \begin{cases} 1, & k = 0, \\ \eta_{k-1}Q_{k-1} + 1, & k \geqslant 1, \end{cases} \tag{1.7}$$

and $\eta_{k-1} \in [\eta_{min}, \eta_{max}]$, $0 \leqslant \eta_{min} \leqslant \eta_{max} \leqslant 1$. Mo et al. [9] first extended this nonmonotone technique to traditional trust region method for unconstrained optimization problem. The numerical results showed that this nonmonotone technique is superior to that of (1.4) (see [9,18]).

In this paper, motivated by Mo et al. [9], as a continued study on basis of Wang and Wang [13], we generalize this nonmonotone technique proposed by Zhang and Hager [18] to the minimax problem, and propose a new algorithm that combine the nonmonotone trust region method with the nonmonotone curvilinear search method. Under mild conditions, the global and superlinear convergence are obtained. Preliminary numerical experiments show that the proposed algorithm is competitive.

This paper is organized as follows: The algorithm is presented in Section 2. In Section 3, we analyzed the convergence. In Section 4, we report some numerical results obtained. Finally, we end the paper with the conclusions.

We shall use the following notation and terminology. Unless otherwise stated, the vector norm used in this paper is Euclidean vector norm on R^n, and the matrix norm is the induced operator norm on $R^{n\times n}$. In addition, we denote $I_A(x) = \{i : f_i(x) = F(x)\}, I_N(x) = \{i : f_i(x) < F(x)\}$, $F_k = F(x_k)$.

2. Algorithm

In this section, we describe a new nonmonotone trust-region algorithm for problem (1.1). We want to compute a new iterate x_{k+1} from the current iterate x_k by using SQP trial step d_k, which is the solution of the following trust region subproblem

$$\begin{cases} \min\limits_{d\in\mathbf{R}^n} F'(x_k; d) + \frac{1}{2}\langle d, B_k d\rangle, \\ s.t. \quad \|d\|_\infty \leqslant \Delta_k, \end{cases} \tag{2.1}$$

where $B_k \in \mathbf{R}^{n\times n}$ is a symmetric approximation matrix, and

$$F'(x_k; d) = \max_{i\in P}\{\langle \nabla f_i(x_k), d\rangle + f_i(x_k)\} - F(x_k) \tag{2.2}$$

is a directional derivative along with direction d at point x_k. The solution of (2.1) can be obtained also by solving the following equivalent trust-region subproblem

$$\begin{cases} \min\limits_{(d,z)\in\mathbf{R}^{n+1}} \frac{1}{2}\langle d, B_k d\rangle + z = q_k(d, z), \\ s.t. \ \langle \nabla f_i(x_k), d\rangle - z \leqslant F(x_k) - f_i(x_k), \quad i \in P, \\ \|d\|_\infty \leqslant \Delta_k. \end{cases} \tag{2.3}$$

In trust-region subproblems (2.1) and (2.3), here we use the ℓ_∞ norm on the trust-region bound, so that the subproblems become standard quadratic problems. In addition, unlike the traditional quadratic subproblem in smooth SQP methods, subproblem (2.3) is always a feasible problem. In fact, it always has a feasible solution $(0, 0)$. Therefore, we do not need to consider the inconsistent of quadratic subproblem that usually occurs in SQP method for solving smooth optimization problems. In order to circumvent the Maratos effect, Fletcher [10,11] requires to solve a second-order correction step $\tilde{d}_k$, and Zhou [7] additionally performs a curvilinear search as follows

$$f(x_k + t_k d_k + t_k^2 \tilde{d}_k) \leqslant \max_{0\leqslant j\leqslant 2} f(x_{k-j}) - \alpha t_k \langle d_k, B_k d_k\rangle. \tag{2.4}$$

In this paper, we embed the nonmonotone technique proposed by Zhang and Hager [18] into an adaptive algorithm that combine the nonmonotone trust-region method with the nonmonotone curvilinear search method for minimax problem, that is, if the stepsize $d_k + \tilde{d}_k$ can not be acceptable in the nonmonotone trust-region scheme, we perform a nonmonotone curvilinear search, such that a stepsize t_k satisfy the following inequality:

$$F(x_k + t_k d_k + t_k^2 \tilde{d}_k) \leqslant C_k - \beta t_k \langle d_k, B_k d_k\rangle. \tag{2.5}$$

where $\beta \in (0, \frac{1}{2})$, (d_k, z_k) is a solution of (2.3), and $\tilde{d}_k, \tilde{z}_k$ is the second-order correction step which is the solution of following quadratic problem

$$\begin{cases} \min\limits_{(\tilde{d},\tilde{z})\in\mathbf{R}^{n+1}} \frac{1}{2}\langle d_k + \tilde{d}, B_k(d_k + \tilde{d})\rangle + \tilde{z}, \\ s.t. \ \langle \nabla f_i(x_k), \tilde{d}\rangle - \tilde{z} \leqslant F(x_k + d_k) - f_i(x_k + d_k), \quad i \in P, \\ \|d_k + \tilde{d}\|_\infty \leqslant \Delta_k. \end{cases} \tag{2.6}$$

In addition, we use $F(x)$ to be the merit function, and for problem (1.1), the Lagrangian is defined by $L(x, \lambda) = \sum_{i\in P}\lambda_i f_i(x)$. Now we give the algorithm as follows

Algorithm 2.1.

Step 0 Choose initial point $x_0 \in \mathbf{R}^n$, and parameters $0 \leqslant \eta_{min} \leqslant \eta_{max} < 1$, $0 < \Delta_0 < \Delta_{max}, 0 < \tau_1 < 1 < \tau_2$, $\beta \in (0, 0.25)$, $0 < \mu \leqslant 2\beta < \eta < 1, \theta \in (0,1), \varepsilon > 0$. Set $C_0 = F(x_0), Q_0 = 1, B_0 = I, k = 0$.
Step 1 Compute a solution (d_k, z_k) by solving the quadratic program (2.3). If $\|d_k\| \leqslant \varepsilon$, stop; Otherwise,
Step 2 Compute a solution $(\tilde{d}_k, \tilde{z}_k)$ by solving the quadratic program (2.6). If $\|\tilde{d}_k\| > \|d_k\|$, set $\tilde{d}_k = 0$;
Step 3 Compute the following ratio

$$r_k = \frac{C_k - F(x_k + d_k + \tilde{d}_k)}{q_k(0,0) - q_k(d_k, z_k)}. \tag{2.7}$$

Step 4 If $r_k > \mu$, set $s_k = d_k + \tilde{d}_k$, $x_{k+1} = x_k + s_k$, go to *Step 6*; Otherwise;
Step 5 Compute t_k, which is the first value in the sequence of $\{1, \theta, \theta^2, \ldots\}$ such that

$$F(x_k + t_k d_k + t_k^2 \tilde{d}_k) \leqslant C_k - \beta t_k \langle d_k, B_k d_k \rangle. \tag{2.8}$$

Set $s_k = t_k d_k + t_k^2 \tilde{d}_k, x_{k+1} = x_k + s_k$.
Step 6 If $r_k \leqslant \mu, \Delta_{k+1} \in [\|s_k\|, \tau_1 \Delta_k]$;
If $r_k \geqslant \eta$ and $\|s_k\| = \Delta_k$, $\Delta_{k+1} = \min(\tau_2 \Delta_k, \Delta_{max})$;
Otherwise, $\Delta_{k+1} = \Delta_k$.
Step 7 Choose $\eta_k \in [\eta_{min}, \eta_{max}]$, and set $Q_{k+1} = \eta_k Q_k + 1$, $C_{k+1} = (\eta_k Q_k C_k + F_{k+1})/Q_{k+1}$.
Step 8 Update B_k to B_{k+1}; $k := k+1$, go to *Step 1*.

In Algorithm 2.1, since the scheme includes the curvilinear search (2.8), we use Damped BFGS formula for updating B_k to B_{k+1} [30]. We recall the following results on minimax problem [13].

Lemma 2.1. *Suppose that* (d_k, z_k) *is the solution of SQP trust-region subproblem* (2.3). *Then*

(1) *If* $d_k = 0$, *there holds* $z_k = 0$.
(2) *If* $d_k \neq 0$, *there holds* $z_k < 0$.

Theorem 2.1. *Suppose that* $\{x_k\}$ *and* $\{d_k\}$ *are generated by* Algorithm 2.1, then

(1) $d_k = 0$ *if and only if* x_k *is the K-T point of the problem* (1.1),
(2) $\{x_k\}$ *is well-defined.*

3. Convergence Analysis

In order to analyze the convergence, we suppose that the following standard assumptions hold throughout the analysis.

Assumption 3.1. For any point $x_0 \in \mathbf{R}^n$, the level set
$L(x_0) = \{x \in \mathbf{R}^n : F(x) \leqslant F(x_0)\}$ is compact.

Assumption 3.2. For each $x \in L(x_0)$, the vectors

$$\begin{pmatrix} \nabla f_i(x) \\ -1 \end{pmatrix}, \quad i \in I_A(x)$$

are linearly independent.

Assumption 3.3. There exist $\sigma_1, \sigma_2 > 0$ such that for every $x \in \mathbf{R}^n$ and every integer k, $\sigma_1 \|x\|^2 \leqslant \langle x, B_k x \rangle \leqslant \sigma_2 \|x\|^2$ holds.

Assumption 3.1 is introduced in order to ensure the existence of a solution to problem (1.1); Assumption 3.2 is a common assumption in the literature on the minimax problems, and Assumption 3.3 is generally required to get global convergence in the context of SQP-type methods which using line-search.

Lemma 3.1. *Suppose that the sequence* $\{x_k\}$ *and* $\{d_k\}$ *are generated by*Algorithm 2.1, *the entire sequence* $\{x_k\}$ *converges to* x^*. *Then,*

(1) *If* x^* *is the K-T point of problem* (1.1), *then the corresponding sequence of multipliers* $\{\lambda_k\}$ *converges to* λ^*, *and* λ^* *is the corresponding multiplier of* x^*.
(2) x^* *is the K-T point of problem* (1.1), *if and only if* $\{d_k\}$ *converges to zero.*

Proof. The proof is similar to Lemma 3.1 in [13]. □

Lemma 3.2. *Suppose that the sequence $\{x_k\}$ is generated by Algorithm 2.1, and (d_k, z_k) is the solution of subproblem (2.3), u_k is the corresponding multiplier, then $z_k \leqslant -\langle d_k, (B_k + u_k I)d_k\rangle$.*

Proof. The proof is similar to Lemma 3.2 in [13]. □

Now we define two index sets as follows

$$D_1 = \{k : r_k > \mu\}, \quad D_2 = \{k : r_k \leqslant \mu\}.$$

The following lemmas are important to the analysis of the convergence of Algorithm 2.1.

Lemma 3.3. *Suppose that the sequence $\{x_k\}$ is generated by Algorithm 2.1, then the following inequality holds for all k.*

$$F_{k+1} < C_{k+1} < C_k. \tag{3.1}$$

Proof. In order to get the conclusion, we consider three cases, respectively.

Case (1). When $k \in D_1$.

Since $r_k > \mu$, from (2.7) we have

$$F_{k+1} < C_k - \mu(q_k(0,0) - q_k(d_k, z_k)) \leqslant C_k. \tag{3.2}$$

This, together with (1.5) and (1.6), we have

$$C_{k+1} = \frac{\eta_k Q_k C_k + F_{k+1}}{Q_{k+1}} < \frac{\eta_k Q_k C_k + C_k}{Q_{k+1}} = C_k \tag{3.3}$$

and

$$F_{k+1} - C_{k+1} = \eta_k Q_k (C_{k+1} - C_k). \tag{3.4}$$

It follows from (3.3) and (3.4) that the inequality (3.1) holds.

Case (2). When $k \in D_2$.

From (2.8) in Step5, since C_k is the same as that in case (1), and note that $x_{k+1} = x_k + t_k d_k + t_k^2 \tilde{d}_k$, we have again that (3.2), (3.3) and (3.4) hold, which yield the inequality (3.1).

Therefore, (3.1) holds for all k, which completes the proof. □

Lemma 3.3 implies that the sequence $\{C_k\}$ is monotonic non-increasing.

Lemma 3.4. *Suppose that Assumptions 3.1–3.3 hold, and the sequence $\{x_k\}$ is generated by Algorithm 2.1*, then

(1) *The sequence $\{C_k\}$ is convergent.*

(2) $$\lim_{k\to\infty} \hat{t}_k \|d_k\| = 0, \tag{3.5}$$

where

$$\hat{t}_k = \begin{cases} \frac{\mu}{2\beta}, & k \in D_1, \\ t_k, & k \in D_2. \end{cases} \tag{3.6}$$

(3) $$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{3.7}$$

Proof. (1) By Lemma 3.3, we know that $F_{k+1} < C_{k+1}$ for all k and $\{C_k\}$ is a monotonic non-increasing sequence. On the other hand, from Assumption 3.1 and the continuity of $F(x)$, we have that $\{F_k\}$ is bounded below. Thus $\{C_k\}$ is convergent. Let

$$\lim_{k\to\infty} C_k = C^*. \tag{3.8}$$

When $k \in D_1$, by Algorithm 2.1, it follows from Lemma 3.2 that

$$F_{k+1} = F(x_k + d_k + \tilde{d}_k) \leqslant C_k + \mu\left(z_k + \frac{1}{2}\langle d_k, B_k d_k\rangle\right) \leqslant C_k + \mu\left(-u_k\|d_k\|^2 - \frac{1}{2}\langle d_k, B_k d_k\rangle\right) \leqslant C_k - \frac{1}{2}\mu\sigma_1\|d_k\|^2, \tag{3.9}$$

Meanwhile, in view of $\|\tilde{d}_k\| \leqslant \|d_k\|$, we obtain

$$\|x_{k+1} - x_k\| = \|d_k + \tilde{d}_k\| \leqslant 2\|d_k\|. \tag{3.10}$$

If $k \in D_2$, by Algorithm 2.1, in this case, we perform curvilinear search.

$$F_{k+1} = F(x_k + t_k d_k + t_k^2 \tilde{d}_k) \leqslant C_k - \beta t_k \langle d_k, B_k d_k \rangle \leqslant C_k - \beta\sigma_1 t_k \|d_k\|^2. \tag{3.11}$$

Meanwhile, in view of $t_k < 1$, we have

$$\|x_{k+1} - x_k\| = \|t_k d_k + t_k^2 \tilde{d}_k\| \leqslant 2t_k \|d_k\|. \tag{3.12}$$

Denote

$$\hat{t}_k = \begin{cases} \frac{\mu}{2\beta}, & k \in D_1, \\ t_k, & k \in D_2. \end{cases} \tag{3.13}$$

Combining (3.9) and (3.11), it gives

$$F_{k+1} \leqslant C_k - \beta\sigma_1 \hat{t}_k \|d_k\|^2, \tag{3.14}$$

Next we claim that $|F_{k+1} - C_k| \to 0$ as $k \to \infty$, which gives

$$\lim_{k\to\infty} \hat{t}_k \|d_k\| = 0. \tag{3.15}$$

In fact, from (3.4) we have

$$|F_{k+1} - C_{k+1}| \leqslant \eta_{max} Q_k |C_{k+1} - C_k|. \tag{3.16}$$

On the other hand, from (1.7) and note that $\eta_{max} \in [0, 1)$ (see in Algorithm 2.1), we have

$$Q_k = \eta_{k-1} Q_{k-1} + 1 = \sum_{j=1}^{k} \prod_{i=1}^{j} \eta_{k-i} + 1 \leqslant \sum_{j=1}^{k} \eta_{max}^j + 1 \leqslant \sum_{j=0}^{\infty} \eta_{max}^j \leqslant \frac{1}{1 - \eta_{max}}. \tag{3.17}$$

It follows from (3.8), (3.16) and (3.17) that

$$\lim_{k\to\infty} |F_{k+1} - C_{k+1}| = 0. \tag{3.18}$$

In addition, we have

$$|F_{k+1} - C_k| \leqslant |F_{k+1} - C_{k+1}| + |C_{k+1} - C_k|. \tag{3.19}$$

Combining (3.8), (3.18) and (3.19), we immediately obtain the result.

In addition, by the definition of $\hat{t}_k$, if $k \in D_1$, there is a constant ρ_1 such that $\rho_1 \hat{t}_k > 2$. Let $\rho = \max\{\rho_1, 2\}$, it follows from (3.10) and (3.12) that

$$\|x_{k+1} - x_k\| \leqslant \rho \hat{t}_k \|d_k\|, \tag{3.20}$$

which gives from (3.5) that (3.7) holds, and the proof is completed. □

Theorem 3.1. *Suppose that $\{x_k\}$ is generated by Algorithm 2.1, Assumptions 3.1–3.3 hold and the multiplies u_k is bounded, if the algorithm does not stop in finite step, then any accumulation of $\{x_k\}$ is a K-T point of problem (1.2).*

Proof. By Assumption 3.1, if Algorithm 2.1 does not stop in finite step, since the sequence $\{x_k\}$ is contained in the compact set $L(x_0)$, there exists a convergent subsequence, we might as well denote it by $\{x_k\}$, and suppose that $\lim_{k\to\infty} x_k = x^*, x^* \in L(x_0)$.

Next we prove that the corresponding subsequence $\{d_k\}$ converges to zero. When $k \in K \cap D_1$, by Lemma 3.4, obviously, the assertion is true. So, it suffices to prove that it is also true when $k \in K \cap D_2$. Suppose on contrary that the assertion is false, then there exists an infinite subset $K_1 \subset K \cap D_2$, such that $\inf_{k\in K_1} \|d_k\| > 0$, i.e., there exists $\delta > 0$, such that $\|d_k\| > \delta$ for all $k \in K_1$.

We first show that there exits a positive scaler $t_{min} > 0$ independent of k such that the curvilinear search (2.8) is always satisfied for some $t_k > t_{min}$ and for all $k \in K_1$. By the Taylor expansion at x_k, there holds

$$f_i(x_k + td_k + t^2\tilde{d}_k) = f_i(x_k) + \langle \nabla f_i(x_k), td_k + t^2\tilde{d}_k \rangle + o(t\|d_k\|).$$

By Lemma 3.2 and note that d_k and $\tilde{d}_k$ are all bounded, for $t \in (0, 1)$ and every $i \in P$, we have

$$\begin{aligned} f_i(x_k + td_k + t^2\tilde{d}_k) &= f_i(x_k) + t\langle \nabla f_i(x_k), d_k \rangle + o(t) \leqslant f_i(x_k) + t(F_k + z_k - f_i(x_k)) + o(t) \\ &= (1-t)f_i(x_k) + tF_k + tz_k + o(t) \leqslant F_k + tz_k + o(t) \leqslant F_k - t\langle (B_k + u_k I)d_k, d_k \rangle + o(t) \\ &\leqslant F_k - t\langle B_k d_k, d_k \rangle + o(t). \end{aligned} \tag{3.21}$$

It follows by Assumption 3.3 and the contradiction hypothesis that

$$f_i(x_k + td_k + t^2\tilde{d}_k) \leqslant F_k - \beta t\langle B_k d_k, d_k\rangle - t(1-\beta)\langle B_k d_k, d_k\rangle + o(t) \leqslant F_k - \beta t\langle B_k d_k, d_k\rangle - t(1-\beta)\sigma_1\delta^2 + o(t). \tag{3.22}$$

This, together with $t < 1$, suggests that there exists $\tilde{t}_i > 0$, for all $t \in (0, \tilde{t}_i]$ and k, it holds

$$f_i(x_k + td_k + t^2\tilde{d}_k) \leqslant F_k - \beta t\langle B_k d_k, d_k\rangle. \tag{3.23}$$

Let $t_{min} = \min_{i\in P} \tilde{t}_i, t_k = t_{min}/2$. Then the following inequality

$$F(x_k + t_k d_k + t_k^2\tilde{d}_k) \leqslant F_k - \beta t_k\langle B_k d_k, d_k\rangle \tag{3.24}$$

holds for all $k \in K_1$. Thus $\{t_k d_k\}$ is uniformly bounded below on K_1 by $\delta t_{min}/2$, which contradicts to Lemma 3.4. Therefore, $d_k \to 0$ as $k \to \infty$. By Lemma 3.1, x^* is just a K-T point of problem (1.1). □

Remark 3.1. From the proof of Theorem 3.1, we can see that, there exists a stepsize t_k in step 5 of Algorithm 2.1 such that (2.8) is satisfied, which means that Algorithm 2.1 is well defined.

In fact, at the current iterate x_k, if Algorithm 2.1 is proceeding in step 5, under the conditions of Theorem 3.1, for $t \in (0,1)$ and every $i \in P$, the inequality (3.21) also holds. Furthermore, if $\|d_k\| \neq 0$ in the iteration k, similarly to (3.22), we have

$$f_i(x_k + td_k + t^2\tilde{d}_k) \leqslant F_k - \beta t\langle B_k d_k, d_k\rangle - t(1-\beta)\langle B_k d_k, d_k\rangle + o(t) \leqslant F_k - \beta t\langle B_k d_k, d_k\rangle - t(1-\beta)\sigma_1\|d_k\|^2 + o(t). \tag{3.25}$$

Since (3.25) holds for every $i \in P$, it follows that

$$F(x_k + td_k + t^2\tilde{d}_k) \leqslant F_k - \beta t\langle B_k d_k, d_k\rangle - t(1-\beta)\sigma_1\|d_k\|^2 + o(t) \tag{3.26}$$

and this implies that, there exists a constant $t_k, t_k \in (0,1)$, so small such that

$$F(x_k + t_k d_k + t_k^2\tilde{d}_k) \leqslant F_k - \beta t_k\langle B_k d_k, d_k\rangle. \tag{3.27}$$

It follows from the inequality in Lemma 3.3 that, for above t_k obtained, the inequality (2.8) holds.

In order to analyze the locally superlinear convergence, let $\{x_k\}$ be generated by Algorithm 2.1, $x_k \to x^*, x^*$ be a K-T point of problem (1.1), and λ^* be the corresponding multiplier of x^*. Denote $I_A(x^*) = \{i : \lambda_i^* > 0\}$, and make the following hypothesis further.

Assumption 3.4. The second order sufficiency conditions with strict complementary slackness are satisfied at x^* as follows

$$\begin{cases} \lambda_i^* > 0, & i \in I_A(x^*), \\ \langle d, \nabla_{xx}^2 L(x^*, \lambda^*)d\rangle > 0, & \forall 0 \neq d \in \mathbf{S}, \end{cases} \tag{3.28}$$

where

$$\mathbf{S} = \{d \in \mathbf{R}^n : \langle d, \nabla f_i(x^*)\rangle = \langle d, \nabla f_j(x^*)\rangle \quad \forall i,j \in I_A(x^*)\}. \tag{3.29}$$

Assumption 3.5. The approximation matrices sequence $\{B_k\}$ satisfy the Dennis-More condition

$$\lim_{k\to\infty} \frac{\left\|\left(B_k - \sum_{i=1}^m \lambda_i^* \nabla^2 f_i(x^*)\right)d_k\right\|}{\|d_k\|} = 0. \tag{3.30}$$

Similarly to the analysis in [13], it is not difficult to obtain the following results, and so we omitted here.

Theorem 3.2. *Suppose that Assumption 3.4 and 3.5 are satisfied, $\{x_k\}$ is generated by Algorithm 2.1, and $\lim_{k\to\infty} x_k = x^*$, where x^* is a K-T point of of problem (1.1). Then, $\{x_k\}$ converges to x^* superlinearly.*

4. Numerical experiments

In this section we do some experiments to show the computational efficiency of the proposed algorithm, and compare the performance of Algorithm 2.1 with the algorithm proposed in [13] and the algorithm given in [7]. All experiments are run on a PC computer with CPU Pentium 4, 2.00 GHz and coded in MATLAB programs. The stopping rule in the experiments is $\|d\| \leqslant \epsilon$. The other parameters are as follows: $\Delta_0 = 1, \Delta_{max} = 10, \tau_1 = 0.5, \tau_2 = 2, \beta = 0.2, \theta = 0.5, \mu = 0.25, \eta = 0.75, B_0 = I, \varepsilon = 1.0e-5$.

In the experiments, the update of the parameters η_k is that: $\eta_0 = 0.2, \eta_1 = 0.1, \eta_k = (\eta_{k-1} + \eta_{k-2})/2, k \geqslant 2$. In addition, $C_0 = F(x_0), Q_0 = 1$. The tested examples were all from [13], which is a set of standard test problems for minimax problem.

The computation results are shown in Tables 1–3, where Table 1 lists the results obtained by Algorithm 2.1, Table 2 and Table 3 list the results obtained by the algorithms proposed in [13,7], respectively. In all Tables, the columns have the following meanings: problem denotes the name of the test problem, n denotes the dimension of the problem, m denotes the

Table 1
Numerical results.

Fun.	n/m	NI	NF	NG	$F(x)$	$\|d\|$	$I_A(x)$	time (s)
1	2/3	5	9	5	1.9522	4.6322e−007	1,2	0.5470
2	2/3	4	7	4	2.0000	2.5249e−012	1,2,3	0.5000
3	4/4	11	25	11	−44.0000	1.9957e−006	1,2,4	3.6409
4	2/3	9	19	9	0.6164	2.8766e−006	1,3	0.9689
5	3/6	10	21	10	3.5997	5.6719e−006	2,5	1.8130
6	3/30	8	15	8	0.0508	6.3159e−011	9,18,23,30	5.4210
7	3/21	14	27	14	2.3470e−06	3.9756e−008	2,3,8,11, 12,16,17,21	10.0000
8	7/5	17	42	17	678.6796	5.9932e−006	1,2	7.4380
9	10/9	16	33	16	24.3062	1.0981e−006	1,2,3,5, 6,7,9	11.4220
10	20/18	20	45	20	131.2477	6.8821e−006	1,2,3,5,6, 7,9,11,12, 15,16,17,18	53.2030

Table 2
Numerical results.

Fun.	n/m	NI	NF	NG	$F(x)$	$\|d\|$	$I_A(x)$	time (s)
1	2/3	6	6	6	1.9522	8.2328e−007	1,2	1.7230
2	2/3	5	5	5	2.0000	9.4108e−010	1,2,3	1.2720
3	4/4	11	17	11	-44.0000	3.3857e−007	1,2,4	10.9360
4	2/3	11	11	11	0.6164	1.6139e−008	1,3	2.2230
5	3/6	11	11	11	3.5997	1.3166e−006	2,5	4.5470
6	3/30	7	7	7	0.0508	4.2234e−006	9,23	15.2420
7	3/21	15	15	15	2.3470e−06	9.8131e−008	2,3,8,11, 12,16,17,21	27.8300
8	7/5	20	32	20	678.6796	5.7390e−006	1,2	26.6880
9	10/9	14	14	14	24.3062	7.9824e−006	1,2,3,5, 6,7,9	23.7750
10	20/18	21	29	21	131.2477	4.9794e−006	1,2,3,5,6, 7,9,11,12, 15,16,17,18	92.2590

Table 3
Numerical results.

Fun.	n/m	NI	NF	NG	$F(x)$	$\|d\|$	$I_A(x)$	time (s)
1	2/3	6	6	6	1.9522	8.2328e−007	1,2	1.6220
2	2/3	5	5	5	2.0000	9.4108e−010	1,2,3	1.2110
3	4/4	9	26	9	−44.0000	1.3009e−006	1,2,4	12.5680
4	2/3	9	12	9	0.6164	6.7713e−008	1,3	2.1530
5	3/6	11	11	11	3.5997	1.3166e−006	2,5	4.6870
6	3/30	7	7	7	0.0508	4.2234e−006	9,23	13.6800
7	3/21	15	15	15	2.3470e−06	9.8131e−008	2,3,8,11, 12,16,17,21	27.3490
8	7/5	16	56	16	678.6796	6.1430e−006	1,2	36.2620
9	10/9	15	33	15	24.3062	3.9247e−007	1,2,3,5, 6,7,9	41.3890
10	20/18	22	73	22	131.2477	1.6683e−006	1,2,3,5,6, 7,9,11,12, 15,16,17,18	166.2890

number of functions in "max" operator. NI, NG and NF denote the numbers of iterations, gradient evaluations and function evaluations, respectively. $F(x), \|d\| \leqslant \varepsilon$, $I_A(x)$ and "time(s)" stand for the optimal value of objective function, the norm of the solution of SQP subproblem, the set of indices of active functions and the cost of CPU running time in seconds, respectively.

From the tables we can see that for most problems, the numbers NI and NG of the new algorithm are in general smaller than those of the other three algorithms. Especially in terms of the cost of CPU running time, the new algorithm performed much better than others. This means that the new adaptive nonmonotone trust-region method with curvilinear search is more effective.

5. Conclusions

In this paper, we have introduced an adaptive nonmonotone algorithm for minimizing minimax problem. The new algorithm takes not only the advantages of nonmonotone technique proposed in [18] but also the advantages of adaptive technique combining the nonmonotone trust-region method with nonmonotone curvilinear search method, which enable the Maratos effect to be avoided. The new algorithm has strongly global convergence and superlinear convergence, and the numerical experiments show the efficiency.

Acknowledgement

The authors are indebted to the editors and anonymous referees for their a number of helpful comments and suggestions that improved the quality of this manuscript.

References

[1] L. Grippo, F. Lampariello, S. Lucidi, A nonmonotone line search for Newton's method, SIAM Journal on Numerical Analysis 23 (4) (1986) 707–716.
[2] E. Polak, D.H. Mayne, J.E. Higgins, Superlinearly convergent algorithm for min-max problems, Journal of Optimization Theory and Applications 69 (1991) 407–439.
[3] E. Ploak, Basics of minimax algorithms, in: F.H. Clarke, V.F. Demyanov, F. Giannessi (Eds.), Nonsmooth Optimization and Related Topics, Plenum, New York, 1989, pp. 343–369.
[4] Z.B. Zhu, Xiang Cai, Jinbao Jian, An improved SQP algorithm for solving minimax problems, Applied Mathematics Letters 22 (2009) 464–469.
[5] Y. Xue, The sequential quadratic programming method for solving minimax problem, Journal of Systems Science and Mathematical Sciences 22 (2002) 355–364.
[6] Y. Xue, A Newton like algorithm for solving minimax problem, Journal on Numerical Methods and Computer Applications 2 (2004) 108–115.
[7] J.L. Zhou, A.L. Tits, Nonmonotone Line Search for Minimax Problems, Journal of Optimization Theory and Applications 76 (1993) 455–476.
[8] J.B. Jian, Q. Ran, Q.J. Hu, A new superlinearly convergent SQP algorithm for nonlinear minimax problem, Acta Mathematicae Applicatae Sinica 23 (2007) 395–410 (English series).
[9] J.T. Mo, C.Y. Liu, S.C. Yan, A nonmonotone trust region method based on nonincreasing technique of weighted average of the successive function values, Journal of Computational and Applied Mathematics 209 (2007) 97–108.
[10] R. Fletcher, Second order correction for nondifferentiable optimization problems, in: G.A. Watson (Ed.), Numerical Analysis, Springer, Berlin, 1982, pp. 97–108.
[11] R. Fletcher, A model algorithm for composite nondifferentiable optimization problems, Mathematical Programming Study 17 (1982) 67–76.
[12] Y. Yuan, On the superlinear convergence of a trust-region algorithm for nonsmooth optimization, Mathematical Programming 31 (1985) 269–285.
[13] F.S. Wang, Y.P. Wang, Nonmonotone algorithm for minimax optimization problems, Applied Mathematics and Computation 217 (2011) 6296–6308.
[14] F.S. Wang, K.C. Zhang, A hybrid algorithm for minimax optimization, Annals of Operations Research 164 (2008) 167–191.
[15] E.M. Gertz, A quasi-Newton trust-region method, Mathematical Programming 100 (2004) 447–470.
[16] F. Ye, H.W. Liu, S.S. Zhou, S.Y. Liu, A smoothing trust-region Newton-CG method for minimax problem for minimax problem, Applied Mathematics and Computation 199 (2008) 581–589.
[17] Y.H. Dai, On the nonmonotone line search, Journal of Optimization Theory and Applications 112 (2002) 315–330.
[18] H.C. Zhang, W.W. Hager, A nonmonotone line search technique and its application to unconstrained optimization, SIAM Journal on Optimization 14 (4) (2004) 1043–1056.
[19] M. Rayday, The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem, SIAM Journal on Optimization 7 (1) (1997) 26C33.
[20] Y.H. Yu, L. Gao, Nonmonotone line search for constrained minimax problems, Journal of Optimization Theory and Application 115 (2002) 419–446.
[21] E.R. Panier, A.L. Tits, Avoiding the Maratos effect by means of a nonmonotone line search, Part 1: general constrained problems, SIAM Journal on Numerical Analysis 28 (1991) 1183–1195.
[22] M.J.D. Powell, On the global convergence of trust region algorithm for unconstrained minimization, Mathematical Programming 29 (1984) 297–303.
[23] J.L. Zhang, X.S. Zhang, A nonmonotone adaptive trust region method and its convergence, Computers and Mathematics with Applications 45 (2003) 1469–1477.
[24] J.L. Zhang, X.S. Zhang, A modified SQP method with nonmonotone line search technique, Journal of Global Optimization 21 (2001) 201–218.
[25] F.S. Wang, K.C. Zhang, C.L. Wang, L. Wang, A variant of trust-region methods for unconstrained optimization, Applied Mathematics and Computation 203 (2008) 297–307.
[26] N.Y. Deng, Y. Xiao, F.J. Zhou, Nonmonotic trust region algorithm, Journal of Optimization Theory and Application 76 (2) (1993) 259–285.
[27] P.L. Toint, An assessment of non-monotone line search techniques for unconstrained optimization, SIAM Journal on Scientific and Statisical Computing 17 (3) (1996) 725–739.
[28] W.Y. Sun, Nonmonotone trust region method for solving optimization problems, Applied Mathematics and Computation 156 (1) (2004) 159–174.
[29] Y. Yuan, W.Y. Sun, The Theory and Methods of Optimization, Science Press, Beijing, 1997.
[30] J. Nocedal, S.J. Wright, Numerical Optimization, Science Press, Beijing, 2006.