Mathematical Programming 20 (1981) 49–62.
North-Holland Publishing Company

COMBINED LP AND QUASI-NEWTON METHODS FOR MINIMAX OPTIMIZATION

Jørgen HALD and Kaj MADSEN
Technical University of Denmark, Lyngby, Denmark

Received 27 June 1979
Revised manuscript received 28 April 1980

We present an algorithm for minimax optimization that combines LP methods and quasi-Newton methods. The quasi-Newton algorithm is used only if an irregular solution is detected, in which case second-order derivative information is needed in order to obtain a fast final rate of convergence. We prove that the algorithm can converge only to a stationary point and that normally the final rate of convergence will be either quadratic or superlinear. The performance is illustrated through some numerical examples.

Key words: Algorithms, Optimization, Minimax, Quasi-Newton, Superlinear Convergence.

1. Introduction

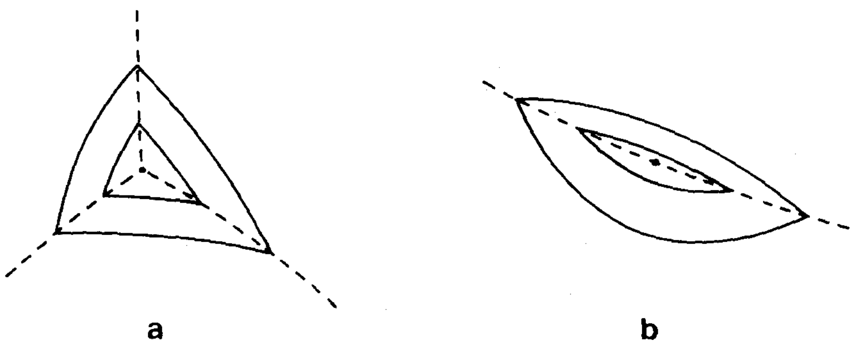
In this paper we consider the problem of minimizing the minimax objective function

$$F(x) \equiv \max_{1 \le j \le m} f_j(x) \tag{1}$$

where the functions f_j are supposed to be smooth, and $x = (x_1, x_2, \dots, x_n)^T$.

An excellent theoretical treatment of minimax optimization can be found in the book of Dem'yanov and Malozemov [10]. Algorithms for minimizing (1) by using only first derivative information have been published by several authors during the past 10 years. Lately it has become clear that in some situations second derivative information is necessary in order to obtain fast final convergence. Examples of algorithms based upon this are those of Hettich [14], Han [13], Charalambous and Moharram [6], Hald and Madsen [12], Watson [21] and Conn [8].

The objective function (1) is in general a non-differentiable function having directional derivatives in all directions. Normally, the minimum is situated at an edge, that is a point where two or more functions are equal, cf. Fig. 1 which shows level curves for minimax objective functions in 2 variables. In Fig. 1a there is no smooth valley through the solution and the minimum is numerically very well determined: no second derivative information (positive definiteness) is needed, the minimum is characterized by only first derivatives of the 3 functions f_j which determine the minimum. Therefore it is possible to construct algorithms based on first derivative information, with fast final convergence in cases like 1a.



It was proved in [17] that the stage 1 (see below) algorithm of this paper, which is of the type mentioned, has quadratic final rate of convergence to the solution x^* when any subset of the set $\{f_i'(x^*) \mid f_i(x^*) = F(x^*)\}$ has maximal rank. This condition is the so-called *Haar-condition*, which ensures that no smooth valley passes through the solution.

In 1b of Fig. 1 there is a smooth valley through the solution, namely along the dotted line. In this case some second-order information may be needed: For directions through the valley the minimum is not characterized by first derivatives only (however, this is still the case for all other directions). This suggests that in situations like 1b (where the number of functions determining the minimum is not larger than the number of unknowns) some second-order information, or approximate second-order information, is needed in order to obtain a fast final rate of convergence. But the fact that the level curves of F have sharp corners is still useful: In Fig. 1b, for example, first derivatives will, with a quadratic rate of convergence, give the information that the solution is at the dotted line, so the dimension of the problem is reduced from 2 to 1 in this case. In general such a valley is always characterized by the fact that some functions are equal. Suppose that the number of such functions is s and the functions are f_j, $j \in A(x^*)$, i.e. $F(x^*) = f_j(x^*) > f_i(x^*)$ for $j \in A(x^*)$ and $i \notin A(x^*)$. Then the following must hold in the valley and at the solution,

$$f_{j_0}(x) - f_j(x) = 0, \quad j \in A(x^*), \tag{2}$$

where $j_0 \in A(x^*)$, so by linearizing these, we can obtain a quadratic convergence to the valley. If the Haar-condition is satisfied at the solution then $s \geq n+1$ and the Jacobian of the system (2) has rank n, so there is no valley (Fig. 1a). In this case a Newton iteration applied to (2) gives quadratic convergence to the solution and it requires only first derivatives of the functions f_j. If, however, $s \leq n$ or the Jacobian of (2) is rank deficient at x^*, then some information is needed in addition to (2). We use some equations that correspond to the Kuhn–Tucker equations in nonlinear programming, namely the following (see for instance [14]):

$$\sum_{j\in A(x)} \lambda_j f_j'(x) = \mathbf{0},$$
$$\sum_{j\in A(x)} \lambda_j = 1 \tag{3}$$

for $x = x^*$. Furthermore $\lambda_j \geq 0$ for $j \in A(x^*)$. Solving (2) and (3) for the minimax problem is equivalent to solving the Kuhn–Tucker equations in nonlinear programming when the number of active constraints is less than the number of unknowns [14].

The minimax algorithm that we propose in this paper consists in two parts. Normally, the algorithm of [15] is used; this is called *stage 1*. But if a smooth valley through the solution is detected, then a switch is made to an algorithm which solves (2) and (3) through a quasi-Newton iteration. This is called *stage 2*. If it turns out that the stage 2 iteration is unsuccessful (for instance if the active set $f_1, \dots, f_s$ has been wrongly chosen), then a switch is made back to stage 1. The algorithm may switch several times between stage 1 and stage 2. However, our experiences indicate that normally only very few switches will take place, so the iteration will finish either as a quadratically convergent stage 1 iteration or as a superlinearly convergent stage 2 iteration according as the Haar-condition is satisfied or not at the solution.

2. Details of the algorithm

In the following we give a rather complete description of the algorithm. We omit only a few computational details which are not of importance for the understanding of the algorithm. These may be found in [12].

The algorithm consists of four parts: The stage 1 iteration, the stage 2 iteration and two sets of criteria for switching.

2.1. The stage 1 iteration

This is the iteration of [15]. At the kth stage of the algorithm we have an approximation x_k of the solution and we wish to use the gradient information at x_k to find a better approximation x_{k+1}. Therefore we find the increment as a solution h_k of the linear minimax problem

$$\begin{aligned}&\text{Minimize (as a function of } h) \quad \bar{F}(x_k, h) \equiv \max_{1\leq i\leq m}\{f_i(x_k) + f_i'(x_k)^{\mathrm{T}} h\},\\ &\text{subject to the constraints} \qquad \|h\|_\infty \leq \Lambda_k\end{aligned} \tag{4}$$

where $\Lambda_k = c_{k-1}\|h_{k-1}\|_\infty$, with $c_{k-1} = 0.5$, 1, or 2 according as the iteration number $k-1$ is unsuccessful, not unsuccessful, or successful. If the objective function F decreases, then $x_{k+1} = x_k + h_k$, otherwise $x_{k+1} = x_k$. Note that no line search is involved.

2.2. The stage 2 iteration

We suppose that a set of indices A has been determined in 2.3; A is called the current *active set* and is an approximation of the set of indices that are active at the solution $\boldsymbol{x}^*$,

$$A^* \equiv A(\boldsymbol{x}^*) \equiv \{j \mid f_j(\boldsymbol{x}^*) = F(\boldsymbol{x}^*)\}. \tag{5}$$

Now an approximate Newton iteration is applied to the system

$$\begin{aligned} &\sum_{j\in A} \lambda_j \boldsymbol{f}'_j(\boldsymbol{x}) = \boldsymbol{0}, \\ &\sum_{j\in A} \lambda_j - 1 = 0, \\ &f_{j_0}(\boldsymbol{x}) - f_j(\boldsymbol{x}) = 0, \quad j \in A, \quad j \neq j_0 \end{aligned} \tag{6}$$

where $j_0 \in A$ is fixed. (The unknowns are $(\boldsymbol{x}, \boldsymbol{\lambda})$). The Newton iteration is approximate because we don't use $f''_j(x)$, so the derivatives of the first set of equations in (6) with respect to x are approximated by finite differences on the first iteration and then updated for the subsequent iterations. Every time a switch from stage 1 to stage 2 is made a new finite difference approximation is calculated.

2.3. Conditions for switching to stage 2

It is not disastrous to start a quasi-Newton iteration with an incorrect active set A, since in that case a switch back to stage 1 will be made after some iterations (see [12, Theorem 1]). But in order to avoid unnecessary iterations we use a rather restrictive set of criteria which must all be satisfied before the quasi-Newton iteration is started: the switch is only made if (8)–(10) below are satisfied.

For each stage 1 iteration the active set $A = A_k$ is defined as

$$A_k \equiv \{j \mid F(\boldsymbol{x}_k) - f_j(\boldsymbol{x}_k) \leq \epsilon_1\} \tag{7}$$

where ϵ_1 is a small positive number specified by the user. Suppose that the latest 3 different iterates, $\boldsymbol{x}_k$, $\boldsymbol{x}_{k-j_1}$, $\boldsymbol{x}_{k-j_2}$, have been calculated in stage 1. Then a switch to stage 2 is made if there exist $\lambda_j \geq 0$, $j \in A_k$, with $\sum \lambda_j = 1$, such that

$$\|\boldsymbol{h}_k\|_\infty = \Lambda_k, \tag{8}$$

$$A_{k-j_2} = A_{k-j_1} = A_k, \tag{9}$$

$$\left\| \sum_{j\in A_k} \lambda_j \boldsymbol{f}'_j(\boldsymbol{x}_k) \right\|_2 \leq \epsilon_2 \tag{10}$$

where $\epsilon_2 > 0$. (In practice (10) is only tested when (8) and (9) are satisfied, and $\{\lambda_j\}$ is found by a linear least squares calculation.)

The condition (8) will hold near a non-Haar solution (see Lemma 1 below), whereas it will not hold near a solution that satisfies the Haar-condition because of the quadratic convergence in this case (cf. [17]). The condition (9) tests the stability of the active set, whereas (10) holds when x_k is close to a solution x^* with $A^* = A_k$ (see (3)).

2.4. *Causes for switching back to stage 1*

The rules of this section are tested for in each quasi-Newton step. These rules guarantee that the convergence properties of the method used in stage 1 are not wasted by the stage 2 iteration. A switch to stage 1 is made if one of the conditions (11), (12), or (13) below fails to hold.

We denote by $r(x, \lambda)$ the vector of left-hand sides of the nonlinear system (6), i.e. the residual vector of (6) at the point (x, λ). In order to continue the stage 2 iteration we require that the residuals decrease:

$$\|r(x_{k+1}, \lambda_{k+1})\| \le \eta \|r(x_k, \lambda_k)\| \tag{11}$$

where $0 < \eta < 1$. We have used $\eta = 0.999$.

It is also tested that no function with an index from outside the active set becomes dominating, i.e.

$$F(x_{k+1}) = \max_{j \in A} f_j(x_{k+1}). \tag{12}$$

Finally we test that all multipliers corresponding to the active set are non-negative,

$$\lambda_i^{(k+1)} \ge 0, \quad i \in A. \tag{13}$$

Notice, that it is *not* required that the minimax objective function F decreases in each stage 2 iteration. Often F will increase in the first stage 2 iteration.

3. Convergence properties

We use the same smoothness assumption as in [12], namely the following,

$$\begin{aligned} &f_j(x + h) = f_j(x) + f_j'(x)^{\mathrm{T}} h + o(\|h\|), \\ &f_j' \text{ is continuous}, \quad j = 1, \dots, m. \end{aligned} \tag{14}$$

A *stationary point* of F is a point for which the generalized gradient (cf. [7])

$$\partial^* F(x) \equiv \operatorname{conv}\{f_j'(x) \mid f_j(x) = F(x)\}, \tag{15}$$

conv being the convex hull, contains $\mathbf{0}$. It follows that a stationary point is a point where the equalities (3) are satisfied. It is shown, for instance in [17], that any local minimum of F is a stationary point, and on the other hand that if the Haar-condition is satisfied at a stationary point x^*, then x^* is a strict local minimum.

In [12] we examined the global convergence properties of the algorithm presented here and proved the following result:

Theorem 1. *If the sequence $\{x_k\}$ generated by the algorithm is convergent, then the limit point is a stationary point.*

Here we concentrate on the local convergence properties. It is assumed that the algorithm generates a convergent sequence of points, and we prove that if a few assumptions are satisfied then the iteration finishes either as a quadratically convergent stage 1 iteration or as a superlinearly convergent stage 2 iteration.

One of the assumptions we need for the theorems of this paper is that the functions which correspond to the active set A^* are strongly active:

3.1. Strongly active functions

We say that the function f_i is *strongly active* at the stationary point $\boldsymbol{x}$ if

$$\begin{aligned} &f_i(\boldsymbol{x}) = F(\boldsymbol{x}), \\ &\mathbf{0} \notin \operatorname{conv}\{f_j'(\boldsymbol{x}) \mid f_j(\boldsymbol{x}) = F(\boldsymbol{x}), j \neq i\}. \end{aligned} \tag{16}$$

We know that any local minimum y of F is a stationary point. It follows that any function f_i that is strongly active at y is necessary for y being a minimum: y is not a stationary point (and thus not a local minimum) of the minimax objective function (1) when the strongly active function f_i is left out.

Following the terminology of linear programming we say that a stationary point is *degenerate* if there are active functions that are not strongly active. Otherwise the stationary point is *non-degenerate.*

In the local convergence theorems we assume that the solution $\boldsymbol{x}^*$ is a non-degenerate stationary point.

We will need the following results, Propositions 1 and 2, where y is a stationary point, s is the number of elements in the active set $A = A(y)$, λ_j are the multipliers of (3), and $\boldsymbol{v}_j$ are the vectors

$$\boldsymbol{v}_j = \begin{pmatrix} 1 \\ f_j'(y) \end{pmatrix}, \quad j \in A. \tag{17}$$

Proposition 1. *If the stationary point y is non-degenerate, then*

(i) $\lambda_j > 0$, $j \in A$,

(ii) $\{\boldsymbol{v}_j \mid j \in A\}$ *is linearly independent,*

(iii) $\{\lambda_j\}$ *is the only solution of* (3) *when* $\boldsymbol{x} = y$,

(iv) $s \leq n + 1$.

Proof. When $\boldsymbol{x} = y$, (3) is identical to the equation

$$\sum_{j \in A} \lambda_j \boldsymbol{v}_j = (1, 0, 0, \dots, 0)^{\mathrm{T}}, \quad \lambda_j \geq 0. \tag{18}$$

If $\lambda_j = 0, j \in A$, then f_j would not be strongly active, and therefore (i) is true. If $\Sigma\ \mu_j v_j = 0$, then every multiplier $\mu_j, j \in A$, must be 0, since otherwise it would be possible to satisfy (18) with a set of non-negative multipliers, one of which could be 0, and this contradicts (i). Thus (ii) is true. (iii) and (iv) are consequences of (18) and (ii). This completes the proof.

Proposition 2. *If $s = n + 1$, then the stationary point y is non-degenerate if and only if the Haar-condition holds at y.*

Proof. Under the Haar-condition $\mathbf{0}$ cannot be a non-trivial linear combination of less than $(n + 1)$ derivatives $f_j'(y), j \in A$. Therefore, if $s = n + 1$, then every function $f_j, j \in A$, must be strongly active.

On the other hand, suppose that y is non-degenerate, and that the Haar condition is not satisfied at y. This means that there exists a non-trivial linear combination of less than $(n + 1)$ of the derivatives $f_j'(y), j \in A$, which is $\mathbf{0}$. We formulate this in the following way:

$$\sum_{j \in A} \mu_j f_j'(y) = \mathbf{0}, \quad \mu_i = 0. \tag{19}$$

Here $\Sigma\ \mu_j \neq 0$, since otherwise $\{\lambda_j + \mu_j\}$ would be a solution of (3) in conflict to (iii) of Proposition 1. It is not a restriction to assume $\Sigma\ \mu_j = 1$. Thus $\{\mu_j\}$ is a solution of (3) which is a contradiction because of (i) and (iii) in Proposition 1, since $\mu_i = 0$. This proves Proposition 2.

3.2. Local convergence results

First we prove that the local bound in (4) will become active when a non-Haar solution is approximated by a stage 1 iteration.

Lemma 1. *If the number of active functions at the stationary point y is less than $(n + 1)$, then there exist $\delta > 0$ and a neighbourhood V of y such that for any $\boldsymbol{x} \in V$ either the system of linear functions*

$$\{f_k'(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{h} + f_k(\boldsymbol{x}) \mid k = 1, \ldots, m\} \tag{20}$$

has no minimax solution or it has a minimax solution $\boldsymbol{h}$ with $\|\boldsymbol{h}\| \geq \delta$.

Proof. By definition $F(y) = f_j(y) > f_i(y)$ for $j \in A(y)$ and $i \notin A(y)$. Since f_k and f_k' are continuous there exist $\delta > 0$ and a neighbourhood V of y with the following property: if $j \in A(y)$ and $i \notin A(y)$, then

$$f_j'(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{h} + f_j(\boldsymbol{x}) > f_i'(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{h} + f_i(\boldsymbol{x})$$

for any $\boldsymbol{x} \in V$ and $\|\boldsymbol{h}\| < \delta$. This means that if any linear function with index $i \notin A(y)$ is active at a minimax solution $\boldsymbol{h}$ of (20), then $\|\boldsymbol{h}\| \geq \delta$.

Therefore, if (20) has a minimax solution with $\|\boldsymbol{h}\| \leq \delta$, then this solution is a minimax solution of the system

$$\{f_j'(x)^{\mathrm{T}}h + f_j(x) \mid j \in A(y)\}. \tag{21}$$

If (21) has a minimax solution $\hat{h}$, then there exist non-negative multipliers λ_j such that $\Sigma\ \lambda_j f_j'(x) = \mathbf{0}$ and $\Sigma\ \lambda_j = 1$, the sums being taken over $j \in A(y)$. Since $A(y)$ has at most n elements this means that $\{f_j'(x) \mid j \in A(y)\}$ is linearly dependent. But in this case there exists a vector $v \neq \mathbf{0}$ which is orthogonal to $\{f_j'(x) \mid j \in A(y)\}$. Therefore any vector $h = \hat{h} + tv, t \in R$ is a minimax solution of (21). By choosing t suitably we obtain $\|h\| \geq \delta$.

Thus (20) always has a minimax solution with $\|h\| \geq \delta$ unless no minimax solution exists and the lemma is proved.

Lemma 2. *Under the assumptions of Lemma* 1 *the following holds: If a stage* 1 *iteration is performed at the point $x_k \in V$ and with $\Lambda_k \leq \delta$, then $\|h_k\| = \Lambda_k$ provided $\|h_k\|$ is chosen as large as possible in case of several solutions in* (4).

Proof. If (20) has no solution for $x = x_k$, then the solution of the restricted problem only exists because of the local bounds. Therefore $\|h_k\| = \Lambda_k$ in this case.

If (20) has only solutions with $\|h\| > \Lambda_k$ the local bounds must be active at the solution h_k of (4), i.e. $\|h_k\| = \Lambda_k$. Finally, if (20) has a solution h_1 with $\|h_1\| \leq \Lambda_k$, then there is also a solution h_2 of (20) with $\|h_2\| > \Lambda_k$ because of Lemma 1. Since the set of minimax solutions of (20) is a convex set this implies that there is a solution h_3 of (20) with $\|h_3\| = \Lambda_k$. Thus we have again $\|h_k\| = \Lambda_k$. This proves Lemma 2.

In the following two convergence theorems it is assumed that $x_k \rightarrow x^*$ which means, according to Theorem 1, that x^* is a stationary point. s^* is the number of active functions at x^*.

Theorem 2. *Suppose that $x_k \rightarrow x^*$ and that x^* is non-degenerate. Then $A_k = A^* \equiv A(x^*)$ for all large values of k provided ϵ_1 in* (7) *is chosen small enough. Furthermore, if there is only a finite number of stage* 2 *iterations, then $s^* = (n+1)$ and the rate of convergence is quadratic.*

Proof. Because of the continuity of the functions f_j, $j = 1, \ldots, m$, the set defined in (7) equals A^*, for $k \geq K_1$ say, if $\epsilon_1 > 0$ is sufficiently small. Therefore, if iteration number $k \geq K_1$ is a stage 1 iteration, then $A_j = A^*$ for $j \geq k$. However, since (7) is only tested in stage 1 we have to deal with the case where the iteration finishes in stage 2, and the last switch takes place for $k < K_1$. Therefore, suppose that every iteration with $k \geq K_1$ is a stage 2 iteration. Then the active set is constant for $k \geq K_1$, $A_k \equiv A$, say. We must prove that $A = A^*$. Since (11) is always true for $k \geq K_1$ the equation (6) must hold for $x = x^*$. Because of (12) some of the indices in A belong to A^*, and then the last set of equations in (6) implies that $A \subseteq A^*$. But since all functions $f_j, j \in A^*$, are strongly active we cannot have $A \subset A^*$ because of (i) in Proposition 1. Therefore $A_k = A = A^*$ for $k \geq K_1$ and the first part of Theorem 2 is proved.

Next suppose that there is only a finite number of stage 2 iterations: every iteration with index $k \geq K_2 \geq K_1$ is a stage 1 iteration. Since $A_k = A^*$ for $k \geq K_1$ (9) is true for $k \geq K_3 \geq K_2$. Since equations (3) are satisfied at $x = x^*$ (10) is true for any $k \geq K_4 \geq K_3$. Since no shift to stage 2 takes place for $k \geq K_4$ this means that (8) never holds for $k \geq K_4$. But since the convergence of $\{x_k\}$ implies that $\Lambda_k \to 0$ it follows from Lemma 2 that (8) will be satisfied unless $s^* = n + 1$. Thus $s^* = n + 1$. The theorem is now a consequence of Proposition 2 since it was proved in [17] that the iteration used in stage 1 has quadratic convergence under the Haar-condition.

In our numerical experiments with the algorithm we have noticed that when the Haar-condition holds at the limit point x^* of the iteration then no shift to stage 2 takes place, i.e. the iteration is identical to that of [15]. It is a consequence of Theorem 2 that if the Haar condition does *not* hold at a non-degenerate limit point of the iteration [15], then the two iterations are not identical: there will be infinitely many stage 2 iterations.

Until now we have not made any assumptions about the quasi-Newton iteration used in stage 2. In the following theorem we assume that the stage 2 iteration is locally and linearly convergent in the sense of Dennis and Moré [11]: the iteration $z_{k+1} = z_k - B_k^{-1} r(z_k)$ is *locally and linearly convergent* at z^* if there exists $\epsilon > 0$ and $\delta > 0$ such that $\|z_0 - z^*\| < \epsilon$ and $\|B_0 - r'(z^*)\| < \delta$ imply that the iteration is well-defined and converges linearly to z^*.

Theorem 3. *Suppose that x_k converges to the non-degenerate stationary point x^* and that the quasi-Newton iteration used is locally and linearly convergent at (x^*, λ^*). If there is an infinite number of stage 2 iterations, then every iteration from a certain point is a quasi-Newton iteration with the active set $A = A^*$ provided that ϵ_1 in (7) is chosen small enough, that η in (11) is close enough to 1, and that the step length used in the finite differences of 2.2 is sufficiently small.*

Proof. It follows from Theorem 2 that we can assume $A_k = A^*$ for every k. Because of the continuity of f_j we can also assume that $f_j(x_k) \leq F(x_k)$ for every $j \notin A^*$ and $k \geq 0$.

Now suppose that there are infinitely many switches from stage 1 to stage 2. When a stage 2 iteration is started $\{\lambda_j\}$ is found by a least squares calculation (see (10)), and from (iii) of Proposition 1 and the continuity it follows that λ_j is close to $\lambda_j^*, j \in A^*$, when $\|x_k - x^*\|$ is small. If the finite difference approximations of 2.2 are sufficiently accurate then the conditions for local and linear convergence are satisfied on every start of a quasi-Newton iteration with

x_k close to x^*, i.e. $z_k = (x_k, \lambda_k)$ close to $z^* = (x^*, \lambda^*)$. Then the quasi-Newton iteration will converge (at least) linearly to z^* unless it is stopped. But if the shift is made close enough to x^*, then (13) will always hold since $\lambda_j^* > 0$ (Proposition 1), and because of the linear convergence. Also (11) will always be satisfied provided η is not less than the linear convergence constant q. Therefore no switch to stage 1 will take place, and thus the assumption of infinitely many switches is contradicted. This proves the theorem.

Theorems 2 and 3 show that when x_k converges to a non-degenerate point then the rate of convergence is at least as good as the rate of convergence of the method used in stage 2 provided we use a locally and linearly convergent quasi-Newton iteration. Broyden's method, [2], has the required properties, and if $\underline{r}'(x^*, \lambda^*)$ is regular and r satisfies a Lipschitz condition at (x^*, λ^*), then the rate of convergence is superlinear as shown in [3]. Since equations (6) can be set up such that the Jacobian is symmetric it is also possible to use Powell's symmetric Broyden update [18] which has superlinear convergence under the conditions mentioned above (see [3]). Under a few additional assumptions we will also have superlinear convergence when using the DFP or the BFGS update [3].

4. Numerical examples

The algorithm which has been implemented and tested, differs slightly from the one, which is described in the previous chapters.

The algorithm, which has been tested, minimizes the objective function

$$\hat{F}(x) \equiv \max_{1 \le i \le m} |f_i(x)|.$$

Therefore in Examples 3 and 4 some rather large constants have been introduced in the residuals f_i to prevent negative functions from determining $\hat{F}$.

The matrix B approximating the Jacobian matrix r' in stage 2 is updated using Powell's symmetric Broyden update. However, some parts of r' consist of only first derivative information, which is available, so in order to utilize this information, these parts of r' are substituted in B. This modification of the update turned out to improve convergence properties, as might be expected.

The requirement, that $\|r\|_\infty$ must be reduced by a factor at least η in each stage-2 iteration is dispensed in the first stage-2 step after a switch: $\|r\|_\infty$ is allowed to be increased by a factor up to 3. This does not influence the convergence properties.

The performance of the algorithm is illustrated through five mathematical test examples and one set of examples originating from network design. We have chosen the parameters in the optimizations as described below.

ϵ_1 in (7) and ϵ_2 in (10) are chosen by the algorithm in the following way:

$$\epsilon_1(x_k) = 0.01\hat{F}(x_k),$$

$$\epsilon_2(x_k) = \begin{cases} 0.5 \min\limits_{i \in A_k} \|f_i'(x_k)\|_\infty, & s_k > 1, \\ 0.01\hat{F}(x_k)/\Lambda_{\max}, & s_k = 1 \end{cases}$$

where $\Lambda_{\max}$ is a user specified upper bound on the stage-2 step lengths, and s_k is the number of elements in the current active set A_k.

The step length t used during numerical differentiations has been given the value $t = 10^{-5}$ everywhere. Also $\Lambda_{\max} = 1$ in all cases. In Examples 1–5 we have used $\Lambda_0 = 0.5$ and in 6 and 7 $\Lambda_0 = 0.1$.

In Table 1 the performance of the new algorithm, the stage 1 algorithm alone and some other algorithms are compared.

The computer used is an IBM 370/165, the calculations being performed in double precision, i.e. 14 hexadecimal digits.

Example 1. This is the set of functions

$$f_1(x) = 10 * (x_2 - x_1^2),$$
$$f_2(x) = 1 - x_1.$$

We minimize the function $\hat{F}(x) = \max|f_i(x)|$ which has the same banana-shaped valley as the Rosenbrock function [20]. Starting point $(-1.2, 1)^T$, solution $(1, 1)^T$, where $\hat{F}(x) = 0$.

Table 1
Number of function evaluations. Note that in [6] second derivatives are required

				This algorithm		Other algorithms, $\epsilon \gtrsim \frac{1}{2} \cdot 10^{-5}$						
Example	n	m	Stage 2	$\epsilon = \frac{1}{2} \cdot 10^{-14}$	$\epsilon = \frac{1}{2} \cdot 10^{-5}$	[1]	[4]	[5]	[6]	[8]	[15]	[16]
I	2	2	0	21	21						21	21
II	5	21	0	12	10						10	10
III.1	7	5	1	28	23			150		106	490	83
III.2	7	5	2	33	29				61	77	531	84
IV.1	4	4	1	19	16			37		64	22	44
IV.2	4	4	1	21	18				52		37	48
V	4	20	1	28	25						50	60
VI.1	6	11	2	51	46	155	67			80	707	48
VI.2	6	11	1	26	21	95	162			67	253	29
VII.1	8	11	3	76	73						2300	262
VII.2	8	11	4	81	74						2466	217
VII.3	8	11	2	58	53						61	31
VII.4	8	11	1	33	26						1097	165
VII.5	8	11	2	83	74						1680	320
VII.6	8	11	2	67	57						998	252

Example 2. This is the set of functions

$$f_j(x) = \frac{x_1 + x_2 y_j}{1 + x_3 y_j + x_4 y_j^2 + x_5 y_j^3} - e^{y_j}, \quad j = 1, \dots, 21$$

where $y_j = -1(0.1)1$, and $\hat{F}$ is minimized. The example was used in [15]. Starting point $(0.5, 0, 0, 0, 0)^T$, solution $(0.999878, 0.253588, -0.746608, 0.245202, -0.037490)^T$, where $\hat{F}(x) = 0.000122$.

Example 3. We consider the nonlinear programming problem:

$$\begin{aligned} \text{minimize} \quad & f(x) = (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 \\ & \qquad + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7 + 1000, \\ \text{subject to} \quad & g_2(x) = -2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 + 127 \geq 0, \\ & g_3(x) = -7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 + 282 \geq 0, \\ & g_4(x) = -23x_1 - x_2^2 - 6x_6^2 + 8x_7 + 196 \geq 0, \\ & g_5(x) = -4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0 \end{aligned}$$

which is used in [5]. Following [5] the solution is found by minimizing the minimax objective F where $f_1 = f$ and $f_i = f - 10g_i$, $i = 2, 3, 4, 5$. In 3.I the starting point is $(3, 3, 0, 5, 1, 3, 0)^T$ as in [5] and in 3.II it is $(1, 2, 0, 4, 0, 1, 1)^T$ as in [6]. The solution is $(2.33050, 1.95137, -0.47754, 4.36573, -0.62449, 1.03813, 1.59423)^T$, where $F(x) = 680.63$.

Example 4. This is the Rosen–Suzuki problem, [19]:

$$\begin{aligned} \text{minimize} \quad & f(x) = x_1^2 + x_2^2 + 2x_3^2 + x_4^2 - 5x_1 - 5x_2 - 21x_3 + 7x_4 + 100, \\ \text{subject to} \quad & g_2(x) = -x_1^2 - x_2^2 - x_3^2 - x_4^2 - x_1 + x_2 - x_3 + x_4 + 8 \geq 0, \\ & g_3(x) = -x_1^2 - 2x_2^2 - x_3^2 - 2x_4^2 + x_1 + x_4 + 10 \geq 0, \\ & g_4(x) = -x_1^2 - x_2^2 - x_3^2 - 2x_1 + x_2 + x_4 + 5 \geq 0. \end{aligned}$$

The solution is found by our minimax algorithm after having used the transformation described in Example 3. In example 4.I we use the starting point $(0,0,0,0)^T$, and in 4.II the starting point is $(2,2,5,0)^T$. The solution is $(0,1,2,-1)^T$, where $f(x) - 100 = -44$.

Example 5. This is a minimax version of the Davidon 2 problem, [9]. The minimax problem is to

$$\text{minimize} \quad \hat{F}(x) = \max_{1 \leq i \leq m} |f_i(x)|,$$

where

$$f_i(x) = (x_1 + x_2 t_i - \exp(t_i))^2 + (x_3 + x_4 \sin(t_i) - \cos(t_i))^2,$$

$$t_i = 0.2i, \qquad m = 20, \qquad n = 4.$$

The starting point is $(25, 5, -5, -1)^T$ and the solution is $(-12.24368, 14.02180, -0.45151, -0.01052)^T$, where $\hat{F}(x) = 115.70644$.

Example 6. As in [16] we use minimax optimization for minimizing the maximum reflection coefficient of a 10:1 three-section transformer with 100% bandwidth. We use the same two starting points as in [16].

Example 7. The same problem as in Example 6 except that the transformer has four s ctions. We use the same six starting points as in [16].

The results are shown in Table 1. In the 5th and 6th column it is shown how many function (including gradient) evaluations was required to obtain 14 and 5 decimals accuracy respectively. The 4th column gives the number of times a switch to stage 2 is made. The rightmost 7 columns give the number of function evaluation used by other algorithms to solve the same problems. In none of these cases the accuracy is better than 5 decimals. The algorithm of [15] is identical to the new algorithm, if in this no switch to stage 2 is allowed.

The first two examples are regular, i.e. the Haar-condition is satisfied at the solution, and no switch to stage 2 is made. Notice that the algorithm passes through the valley of Example 1 without detecting a non-Haar solution. In all the examples a fast final convergence was observed which means that the true active set has been identified in the Examples 3–7 all of which represent non-Haar solutions.

5. Conclusion

It has been proved (Theorem 2 and Theorem 3) that if there is convergence to a non-degenerate stationary point and if a couple of constants are properly chosen, then either we get quadratic final rate of convergence in stage 1 (Haar-solution) or the iteration will be a pure quasi-Newton iteration from some point (non-Haar solution), provided the quasi-Newton method used in stage 2 is locally and linearly convergent. If Powell's symmetric Broyden (PSB) formula was used, then the quasi-Newton algorithm would be locally and superlinearly convergent. It is easy to show, that our modified PSB quasi-Newton method is locally and linearly convergent and our experiments have indicated that it is slightly faster than the straightfoward use of PSB. Further it has been proved (Theorem 1) that if there is convergence, then the limit point is a stationary point. Our numerical experiments seem to confirm these theoretical results.

References

[1] J.W. Bandler and C. Charalambous, "New algorithms for network optimization", *IEEE Transactions on Microwave Theory Technique* MTT-21 (1973) 815–818.

[2] C.G. Broyden, "A class of methods for solving nonlinear simultaneous equations", *Mathematics of Computation* 19 (1965) 577–593.

[3] C.G. Broyden, J.E. Dennis and J.J. Moré, "On the local and superlinear convergence of quasi-Newton methods", *Mathematics of Computation* 12 (1973) 223–246.

[4] C. Charalambous and A.R. Conn, "Optimization of microwave networks", *IEEE Transactions on Microwave Technique* MTT-23 (1975) 834–838.

[5] C. Charalambous and A.R. Conn, "An efficient method to solve the minimax problem directly", *SIAM Journal on Numerical Analysis* 15 (1978) 162–187.

[6] C. Charalambous and O. Moharram, "A new approach to minimax optimization", Department of Systems Design, University of Waterloo, Waterloo, Ont. (1978) 1–4.

[7] F.H. Clarke, "Generalized gradients and applications", *Transactions of the American Mathematical Society* 205 (1975) 247–262.

[8] A.R. Conn, "An efficient second order method to solve the (constrained) minimax problem", Department of Combinatorics and Optimization, University of Waterloo, Waterloo, Ont., Report CORR 79-5 (1979) 1–48.

[9] W.C. Davidon, "Optimally conditional optimization algorithms without line searches", *Mathematical Programming* 9 (1975) 1–30.

[10] V.F. Dem'yanov and V.N. Malozemov, *Introduction to minimax* (Wiley, New York, 1974). Translated from: *Vvedenie v minimaks* (Izdatel'stvo "Nauka", Moscow, 1972).

[11] J.E. Dennis and J.J. Moré, "Quasi-Newton methods, motivation and theory", *SIAM Review* 19 (1977) 46–89.

[12] J. Hald and K. Madsen, "A 2-stage algorithm for minimax optimization", in: *International symposium on systems optimization and analysis*, Lecture notes in control and information sciences 14 (Springer, Heidelberg, 1978) 225–239.

[13] S.P. Han, "Superlinear convergence of a minimax method", Report TR 78-336, Department of Computer Science, Cornell University, Ithaca, NY (1978) 1–21.

[14] R. Hettich, "A Newton-method for nonlinear Chebyshev approximation", in: R. Schaback and K. Scherer, eds., Approximation theory, Lecture Notes in Mathematics, 556 (Springer, Berlin, 1976) 222–236.

[15] K. Madsen, "An algorithm for minimax solution of overdetermines systems of non-linear equations", *Journal of the Institute of Mathematics and its Applications* **16** (1975) 321–328.

[16] K. Madsen and H. Schjaer-Jacobsen, "Singularities in minimax optimization of networks", *IEEE Transactions on Circuits and Systems* (1976) 456–460.

[17] K. Madsen and H. Schjaer-Jacobsen, "Linearly constrained minimax optimization", *Mathematical Programming* 14 (1978) 208–223.

[18] M.J.D. Powell, "A new algorithm for unconstrained optimization", in: J.B. Rosen, O.L. Mangasarian and K. Ritter, Eds., *Nonlinear programming* (Academic Press, New York, 1970).

[19] J.B. Rosen and S. Suzuki, "Construction of non-linear programming test problems", *Communication of the Association for Computing Machinery* 8 (1965) 113.

[20] H.H. Rosenbrock, "An automatic method for finding the greatest or least value of a function", *The Computer Journal* 3 (1960) 175–184.

[21] G.A. Watson, "The minimax solution of an overdetermined system of non-linear equations", *Journal of the Institute of Mathematics and its Applications* 23 (1979) 167–180.