Ladislav Lukšan

A compact variable metric algorithm for linearly constrained nonlinear minimax approximation

Kybernetika, Vol. 21 (1985), No. 6, 405--427

Persistent URL: http://dml.cz/dmlcz/125885

KYBERNETIKA — VOLUME 21 (1985), NUMBER 6

A COMPACT VARIABLE METRIC ALGORITHM FOR LINEARLY CONSTRAINED NONLINEAR MINIMAX APPROXIMATION

LADISLAV LUKŠAN

The paper contains a description of an efficient algorithm for linearly constrained nonlinear minimax approximation. This algorithm uses an elimination of constraints together with the product-form variable metric updates. The main advantage of the new algorithm consists in a simple determination of the direction vector. No quadratic programming subproblem has to be solved. The efficiency of the algorithm is demonstrated on test problems.

1. INTRODUCTION

This paper contains a description of an efficient algorithm for linearly constrained nonlinear minimax approximation, where a point $x^* \in R_n$ is sought such that

$$F(x^*) = \min_{x \in L_n} F(x) \tag{1.1}$$

where

$$F(x) = \max_{i \in K} f_i(x)$$

and

$$L_n = \{x \in R_n : a_j^T x \geqq b_j, j \in L\}$$

where $f_i(x)$, $i \in K$, are real valued functions in the n-dimensional vector space R_n with continuous second-order derivatives, and where K, L are some index sets.

The problem (1.1) can be transformed into the equivalent problem of nonlinear programming, where we seek a pair $(x^*, z^*) \in R_{n+1}$ such that

$$z^* = \min_{(x,z) \in N_{n+1}} z \tag{1.2}$$

where

$$N_{n+1} = \{(x, z) \in R_{n+1} : f_i(x) \leqq z, i \in K, a_j^T x \geqq b_j, j \in L\} .$$

It is clear that $z^* = F(x^*)$. Let $I(x^*) = \{i \in K : f_i(x^*) = F(x^*)\}$ and $J(x^*) = \{j \in L : a_j^T x^* = b_j\}$ be the sets of indices of active functions and active constraints respectively and suppose that the vectors $[g_i^T(x^*), 1]^T$, $i \in I(x^*)$, $[a_j^T, 0]^T$, $j \in J(x^*)$,

are linearly independent. Then necessary conditions for the solution of the problem (1.2) have the following form

$$\sum_{i \in I(x^*)} u_i^* \, g_i(x^*) = \sum_{j \in J(x^*)} v_j^* a_j \,,$$

$$\sum_{i \in I(x^*)} u_i^* = 1 \,, \tag{1.3}$$

$$f_i(x^*) = z \,, \quad i \in I(x^*) \,,$$

$$a_j^{\mathrm{T}} x^* = b_j \,, \quad j \in J(x^*) \,,$$

$$u_i^* \geqq 0 \,, \quad v_j^* \geqq 0 \,, \tag{1.4}$$

where $g_i(x^*)$, $i \in I(x^*)$, are gradients of the functions $f_i(x^*)$, $i \in I(x^*)$, and where u_i^*, $i \in I(x^*)$, v_j^*, $j \in J(x^*)$, are nonnegative Lagrange multipliers.

The algorithm for solving (1.1), which is under examination in this paper, is essentially the quasi-Newton method applied to (1.3). This algorithm is a generalization of the variable metric method described in [10] and it uses an elimination of constraints together with the product-form variable metric updates as it was described in [8]. In each iteration beginning at the feasible point $x \in L_n$ we determine a set $J(x)$ of indices of active constraints and a set $I_\varepsilon(x)$ which approximates, in some sense, the set $I(x^*)$. Using $I_\varepsilon(x)$ instead of $I(x^*)$ and $J(x)$ instead of $J(x^*)$ in (1.3) and linearizing (1.3), we find a direction vector $s \in R_n$ which is a feasible descent direction for the problem (1.1) at the feasible point $x \in L_n$. This linearization makes use of an approximation of the Hessian matrix of the Lagrangian function associated with the problem (1.2). The direction vector $s \in R_n$ is used for finding a new point $x^+ \in L_n$ such that $x^+ = x + \alpha s$ where the steplength α is chosen to guarantee a sufficient decrease of the function (1.1). The main advantage of the proposed algorithm consists in a simple determination of the direction vector $s \in R_n$. It is a considerable simplification as compared with the method [11], which has to solve a quadratic programming subproblem in each step. The computation of the direction vector is the same in all steps. We need to combine neither different methods nor different direction vectors. Numerical experiments show that the new algorithm is comparable with the efficient method of recursive quadratic programming [11]. It also surpasses, in most cases, the method of recursive linear programming [12] and it is more efficient than the variable metric method described in [8], which uses horizontal steps only.

2. DETERMINATION OF THE DIRECTION VECTOR

Let $x \in L_n$ and let $I(x) = \{i \in K : f_i(x) = F(x)\}$ and $J(x) = \{j \in L : a_j^{\mathrm{T}} x = b_j\}$ be the sets of indices of active functions and active constraints respectively at the feasible point $x \in L_n$. We assume, in this paper, that each feasible point $x \in L_n$ is regular, i.e. that the vectors $[g_i^{\mathrm{T}}(x), 1]^{\mathrm{T}}$, $i \in I(x)$, $[a_j^{\mathrm{T}}, 0]^{\mathrm{T}}$, $j \in J(x)$, are linearly independent.

This section is devoted to the determination of a feasible descent direction for the problem (1.1). Let $\varepsilon > 0$ and let

$$I_\varepsilon(x) = \{i \in K : f_i(x) \geqq F(x) - \varepsilon\} . \tag{2.1}$$

In order to simplify the notation, we omit the parameter x. We denote by u the vector containing all u_i, $i \in I_\varepsilon$, by f the vector containing all f_i, $i \in I_\varepsilon$, by e the vector containing all e_i, $i \in I_\varepsilon$, where $e_i = 1$, $i \in I_\varepsilon$, and by B the matrix containing all g_i, $i \in I_\varepsilon$, as its columns. Furthermore we denote by v the vector containing all v_j, $j \in J$, and by A the matrix containing all a_j, $j \in J$, as its columns.

Using I_ε instead of $I(x^*)$ and J instead of $J(x^*)$ in (1.3) and linearizing (1.3), we get

$$\begin{aligned} Gs + Bu &= Av , \\ B^T s &= ze - f , \\ A^T s &= 0 , \\ e^T u &= 1 , \end{aligned} \tag{2.2}$$

where

$$G = \sum_{i \in I_\varepsilon} u_i G_i$$

is the Hessian matrix of the Lagrangian function associated with the problem (1.2) (G_i, $i \in I_\varepsilon$, are Hessian matrices of the functions f_i, $i \in I_\varepsilon$, at the feasible point $x \in L_n$).

Definition 2.1. Let S be a matrix such that $[A, S]$ is a nonsingular square matrix of order n and $A^T S = 0$. Then we say that A, S is an orthogonal pair of matrices generated by the set J.

Lemma 2.1. Let the Hessian matrix G be positive definite. Then there exists a matrix S such that A, S is an orthogonal pair of matrices generated by the set J, and the direction vector $s \in R_n$ given by (2.2) can be expressed in the form

$$s = S\tilde{s} \tag{2.3}$$

where

$$\begin{aligned} \tilde{s} &= -\tilde{B}u , \\ \tilde{B}^T \tilde{B} u &= f - ze , \\ e^T u &= 1 , \end{aligned} \tag{2.4}$$

and where $\tilde{B} = S^T B$.

Proof. Using (2.2) we get

$$s = -HBu + HAv$$

and

$$\begin{aligned} -B^T HBu + B^T HAv &= ze - f , \\ A^T HBu - A^T HAv &= 0 , \end{aligned}$$

where $H = G^{-1}$. The matrix H exists and is positive definite since G is positive

definite. Eliminating the vector v from the last equations, we obtain

$$s = -(H - HA(A^THA)^{-1}A^TH)Bu \tag{2.5}$$

and

$$B^T(H - HA(A^THA)^{-1}A^TH)Bu = f - ze\,.$$

The matrix $H - HA(A^THA)^{-1}A^TH$ is positive semidefinite since H is positive definite. Therefore, there exists a matrix S with linearly independent columns such that

$$SS^T = H - HA(A^THA)^{-1}A^TH\,. \tag{2.6}$$

Obviously $A^TS = 0$ since $A^TSS^TA = A^THA - A^THA(A^THA)^{-1}A^THA = 0$. Moreover $[A, S]$ is a nonsingular square matrix of order n since the matrices A, S have linearly independent columns, $A^TS = 0$ and $S^Tw = 0$ implies $w = Ay$ with $y = (A^THA)^{-1}A^THw$. Setting the matrix (2.6) into (2.5), we get

$$s = -SS^TBu$$

and

$$B^TSS^TBu = f - ze$$

which prove (2.3) and (2.4). □

The Hessian matrix G is usually unknown. Therefore, it will be supposed in the rest of this paper that H is only a positive definite approximation of the matrix G^{-1}. We shall be working only with the matrix S which will be updated by means of the product-form variable metric corrections.

Definition 2.2. We say that the nonsingularity condition is satisfied for a given $\varepsilon > 0$ (at the feasible point $x \in L_n$) if the matrix $\tilde{B}^T\tilde{B} + \lambda ee^T$ is nonsingular for an arbitrary parameter $\lambda > 0$.

Lemma 2.2. Let the feasible point $x \in L_n$ be regular. Then there exists a number $\varepsilon > 0$ such that the nonsingularity condition is satisfied for it.

Proof. Since

$$\tilde{B}^T\tilde{B} + \lambda ee^T = [\tilde{B}^T, e]\begin{bmatrix} I, & 0 \\ 0, & \lambda \end{bmatrix}\begin{bmatrix} B \\ e^T \end{bmatrix},$$

the nonsingularity condition is violated only if the matrix $[\tilde{B}^T, e]^T$ has linearly dependent columns, i.e. only if there exists a nonzero vector w such that

$$\tilde{B}w = 0\,,$$

$$e^Tw = 0\,.$$

But $\tilde{B}w = 0$ together with $\tilde{B} = S^TB$ implies that $Bw = Ay$ for some vector y. Therefore the nonsingularity condition is violated only if the vectors $[g_i^T, 1]^T$, $i \in I_\varepsilon$, $[a_j^T, 0]^T$, $j \in J$, are linearly dependent. Since the set K is finite, there exists a nonzero number $\varepsilon > 0$ such that $I_\varepsilon = I$. Therefore, the nonsingularity condition cannot be

violated for such a number $\varepsilon > 0$ since the vectors $[g_i^T, 1]^T$, $i \in I$, $[a_j^T, 0]^T$, $j \in J$, are linearly independent at a regular feasible point. □

The equations (2.4) have the same form as those used in [10]. Therefore we can continue in the same way as in [10]. Let us introduce the notation.

$$C = (\tilde{B}^T \tilde{B} \times \lambda e e^T)^{-1}, \tag{2.7}$$

$$p = Ce,$$

in case the nonsingularity condition is satisfied.

Lemma 2.3. Let the nonsingularity condition be satisfied for a given $\varepsilon > 0$. Then the solution of the system (2.4) can be expressed in the following form

$$\tilde{s} = -\tilde{B}u,$$

$$u = Cf - (z - \lambda)\, p, \tag{2.8}$$

$$z = \lambda + (p^T f - 1)/e^T p.$$

Proof. See [10]. □

Lemma 2.4. Let the nonsingularity condition be satisfied for a given $\varepsilon > 0$. Then $\lambda e^T p \leqq 1$. Moreover, $\lambda e^T p = 1$ if and only if the vectors g_i, $i \in I_\varepsilon$, a_j, $j \in J$, are linearly dependent. In the last case, $\tilde{B}p = 0$.

Proof. See [10]. Note that the vectors g_i, $i \in I_\varepsilon$, a_j, $j \in J$, are linearly dependent if and only if the matrix $\tilde{B}$ has linearly dependent columns (see proof of Lemma 2.2). □

Lemma 2.5. Let the nonsingularity condition be satisfied for a given $\varepsilon > 0$. Let $\tilde{s}$ be the vector obtained from (2.4). Then $\tilde{s} = 0$ if and only if the feasible point $x \in L_a$ is the solution of the system (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J.

Proof. The feasible point $x \in L_a$ is the solution of the system (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J if and only if $Bu = Av$, $f - ze = 0$ and $e^T u = 1$. (The conditions $a_j^T x = b_j$, $j \in J$, are satisfied automatically by definition of the set J.) But $Bu = Av$ if and only if $\tilde{B}u = 0$, which is satisfied together with $f - ze = 0$ and $e^T u = 1$ if and only if $\tilde{s} = 0$, where $\tilde{s}$ is the vector obtained from (2.4). □

Lemma 2.6. Let the nonsingularity condition be satisfied for a given $\varepsilon > 0$. Let u be the Lagrange multiplier vector determined by (2.4) and let u_k be a component of the vector u with the index $k \in I_\varepsilon$. Let $I_\varepsilon^- = I_\varepsilon \setminus \{k\}$ and let the triple $(\tilde{s}^-, u^-, z^-)$ be the solution of the system which results from the system (2.4) after replacing I_ε by I_ε^-. Then

$$g_k^T s^- = u_k(\beta_k \gamma_k + \delta_k) - (f_k - z^-) \tag{2.9}$$

where $s^- = S\tilde{s}^-$ and $\beta_k \gamma_k + \delta_k > 0$.

Proof. See [10]. Note that $g_k^T s^- = \tilde{g}_k^T \tilde{s}^-$ where $\tilde{g}_k = S^T g_k$. □

The above lemmas can be used to prove the following theorems which are generalizations of the theorems stated in [10].

Theorem 2.1. Let the nonsingularity condition be satisfied for a given $\varepsilon > 0$. Let the triple $(\tilde{s}, u, z)$ be defined by (2.8) such that $\tilde{s} \neq 0$ and $u \geqq 0$. Let the functions f_i, $i \in I_\varepsilon$, have bounded Hessian matrices in a neighbourhood of a feasible point $x \in L_n$. Then $s = S\tilde{s}$ is a feasible descent direction for the problem (1.1) at the feasible point $x \in L_n$.

Proof. Let us define

$$g = Bu$$

$$\text{and } \tilde{g} = \tilde{B}u \,. \tag{2.10}$$

Then (2.8) and (2.10) imply

$$\tilde{s}^{\mathrm{T}}\tilde{g} = -\tilde{s}^{\mathrm{T}}\tilde{s} < 0$$

since $\tilde{s} \neq 0$ by the assumption. Using (2.4), we get

$$B^{\mathrm{T}}s = \tilde{B}^{\mathrm{T}}\tilde{s} = -\tilde{B}^{\mathrm{T}}\tilde{B}u = -(f - ze) \,. \tag{2.11}$$

Now we are going to prove

$$\lim_{\alpha \to 0_+} \frac{F(x + \alpha s) - F(x)}{\alpha} \leqq z - F \,. \tag{2.12}$$

Taking sufficiently small steplength $0 < \alpha < 1$ we can suppose that $F(x + \alpha s) > f_i(x + \alpha s)$ for all indices $i \notin I_\varepsilon$. Let $k \in I_\varepsilon$ be the index such that $F(x + \alpha s) = f_k(x + \alpha s)$. The function f_k has a bounded Hessian matrix in a neighbourhood of the point $x \in R_n$. Therefore, there exists a constant K such that

$$F(x + \alpha s) \leqq f_k + \alpha g_k^{\mathrm{T}} s + \frac{\alpha^2}{2} K \|s\|^2 \,.$$

Using (2.11), we get

$$F(x + \alpha s) \leqq f_k + \alpha g_k^{\mathrm{T}} s + \frac{\alpha^2}{2} K \|s\|^2 = f_k + \alpha(z - f_k) + \frac{\alpha^2}{2} K \|s\|^2 \leqq$$

$$\leqq F + \alpha(z - F) + \frac{\alpha^2}{2} K \|s\|^2$$

since $f_k \leqq F$ and $\alpha < 1$. Therefore we obtain (2.12). Finally we are going to prove that $z - F \leqq \tilde{s}^{\mathrm{T}}\tilde{g}$. Since $u \geqq 0$ and $f_i \leqq F$ for all indices $i \in I_\varepsilon$, the inequality $(Fe - f)^{\mathrm{T}} u \geqq 0$ is valid. Using (2.4) and (2.11) we get

$$z - F = (ze - Fe)^{\mathrm{T}} u = (ze - f)^{\mathrm{T}} u - (Fe - f)^{\mathrm{T}} u \leqq$$

$$\leqq (ze - f)^{\mathrm{T}} u = \tilde{s}^{\mathrm{T}}\tilde{B}u = \tilde{s}^{\mathrm{T}}\tilde{g} \,.$$

Using all proven inequalities, we obtain

$$\lim_{\alpha \to 0_+} \frac{F(x + \alpha s) - F(x)}{\alpha} \leqq z - F \leqq \tilde{s}^{\mathrm{T}}\tilde{g} < 0 \,. \qquad \square$$

Theorem 2.2. Let a feasible point $x \in L_n$ be regular. Then there exists a number $\varepsilon > 0$ such that the nonsingularity condition is satisfied end either the feasible point $x \in L_n$ is a solution of the problem (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J or the direction vector $s \in R_n$, obtained from (2.3), is a feasible descent direction for the problem (1.1) at the feasible point $x \in L_n$.

Proof. Lemma 2.2 guarantees that there exists a number $\varepsilon > 0$ for which the nonsingularity condition is satisfied. Three cases can occur for such a number $\varepsilon > 0$.

(a) The conditions

$$\lambda e^T p = 1 , \tag{2.13}$$

$$f = Fe \tag{2.14}$$

are satisfied simultaneously.

(b) The case (a) does not occur but

$$\varepsilon < (1 - \lambda e^T p)/\|p\|_1 . \tag{2.15}$$

(c) Neither the case (a) nor the case (b) occurs.

In the case (a) we can use Lemma 2.4 and Lemma 2.5. The equality (2.13) implies $\tilde{B}p = 0$ (see Lemma 2.4). Using (2.8) and (2.14) we obtain

$$\tilde{B}u = (F - z + \lambda)\,\tilde{B}p = 0 ,$$

so that $\tilde{s} = -\tilde{B}u = 0$ and the feasible point $x \in L_n$ is a solution of the problem (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J (see Lemma 2.5).

In the case (b) we can define numbers $\varepsilon_i = F - f_i$ for $i \in I_\varepsilon$. It is clear that $0 \leqq \leqq \varepsilon_i \leqq \varepsilon$ for $i \in I_\varepsilon$. Using (2.8) and (2.11) we obtain

$$g_i^T s = \tilde{g}_i^T \tilde{s} = -(f_i - z) = -f_i + \lambda + (p^T f - 1)/e^T p$$

for an arbitrary index $i \in I_\varepsilon$ so that

$$g_i^T s e^T p = \lambda e^T p - 1 + \varepsilon_i \sum_{j \in I_\varepsilon} p_j - \sum_{j \in I_\varepsilon} \varepsilon_j p_j \leqq \lambda e^T p - 1 + \varepsilon \|p\|_1 < 0$$

by (2.15). Since $e^T p > 0$, we get $g_i^T s < 0$ for all indices $i \in I_\varepsilon$. Consequently $s \in R_n$ is a descent direction for the function (1.1). It is also a feasible direction since $A^T S = 0$ by Definition 2.1 and, therefore, $A^T s = A^T S \tilde{s} = 0$ by (2.3).

In the case (c) we can reduce the number ε (divide it by 10 for example) and we can repeat all the prosess. Since the set K is finite, there exists a nonzero number $\varepsilon > 0$ such that $I_\varepsilon = I$. The condition (2.14) is satisfied in this case. If in addition (2.13) holds, we obtain the case (a). If (2.13) does not hold, we get

$$(1 - \lambda e^T p)/\|p\|_1 > 0$$

since the vector p is finite. This number remains unchanged on further decrease of ε. Therefore the condition (2.15) is satisfied for a sufficiently small value $\varepsilon > 0$ and we obtain the case (b). We have proved that for a sufficiently small value $\varepsilon > 0$ the case (c) does not occur, which proves the theorem. □

The theorems we have proved can be used in case the feasible point $x \in R_n$ is not a solution of the problem (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J. Theorem 2.1 does not depend on the value $\varepsilon > 0$ provided the nonsingularity condition is satisfied, but the Lagrange multipliers have to be nonnegative. On the other side Theorem 2.2 does not depend on the sign of Lagrange multipliers but the value $\varepsilon > 0$ is limited by the condition (2.15).

Now we are considering the case when the feasible point $x \in L_n$ is a solution of the problem (1.3) but the conditions (1.4) are violated.

Lemma 2.7. Let the nonsingularity condition be satisfied for a given $\varepsilon > 0$. Let $\lambda e^{\mathrm{T}} p = 1$. Let $k \in I_\varepsilon$ and let the vectors e^-, p^- be defined in the same way as the vectors e, p, respectively, with I_ε replaced by $I_\varepsilon^- = I_\varepsilon \setminus \{k\}$. Then $\lambda(e^-)^{\mathrm{T}} p^- = 1$ if and only if $p_k = 0$ where p_k is a component of the vector p with the index $k \in I_\varepsilon$.

Proof. Since $\lambda e^{\mathrm{T}} p = 1$, we get $\tilde{B} p = 0$ by Lemma 2.4. Let $\tilde{B}^-$ be the matrix defined in the same way as the matrix $\tilde{B}$ with I_ε replaced by $I_\varepsilon^- = I_\varepsilon \setminus \{k\}$. If $p_k = 0$ then $\tilde{B}^-$ has linearly dependent columns so that $\lambda(e^-)^{\mathrm{T}} p^- = 1$ by Lemma 2.4. If, on the other side, $\lambda(e^-)^{\mathrm{T}} p^- = 1$, we get $\tilde{B}^- p^- = 0$ by Lemma 2.4. Denote by $\tilde{p}$ the vector that is of the same dimension as the vector p, such that $\tilde{p}_i = p_i^-$ for $i \in I_\varepsilon^-$ and $\tilde{p}_k = 0$. Then $\tilde{B}\tilde{p} = 0$. If $p \neq \tilde{p}$ then, by setting $w = \lambda(p - \tilde{p})$, we obtain $\tilde{B} w = 0$ and $e^{\mathrm{T}} w = \lambda e^{\mathrm{T}} p - \lambda(e^-)^{\mathrm{T}} p^- = 1 - 1 = 0$ which is in contradiction with the nonsingularity condition (see proof of Lemma 2.2). Therefore, $p = \tilde{p}$ so that $p_k = \tilde{p}_k = 0$. □

Theorem 2.3. Let the feasible point $x \in L_n$ be a solution of the problem (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J. Let the assumptions of Lemma 2.6 be satisfied with $u_k < 0$. Then $s^- = S\tilde{s}^-$ is a feasible descent direction for the problem (1.1) at the feasible point $x \in L_n$.

Proof. Since the feasible point $x \in L_n$ is a solution of the problem (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J, the conditions (2.13) and (2.14) hold so that $I_\varepsilon = I$. Moreover, from (2.8) we get $u = (F - z + \lambda) p$ which together with $u_k < 0$ implies $p_k \neq 0$. Therefore, $\lambda(e^-)^{\mathrm{T}} p^- < 1$ by Lemma 2.7. Since $I_\varepsilon = I$, the set I_ε remains unchanged for a small enough number $\varepsilon > 0$. Therefore, we can consider the condition

$$\varepsilon < (1 - \lambda(e^-)^{\mathrm{T}} p^-)/\|p^-\|_1 \tag{2.16}$$

as satisfied and, as in the proof of Theorem 2.2, we get $g_i^{\mathrm{T}} s^- < 0$ for all indices $i \in I_\varepsilon^-$. Moreover, since $u_k < 0$, we get

$$g_k^{\mathrm{T}} s^- = u_k(\beta_k \gamma_k + \delta_k) - (f_k - z^-) < -(f_k - z^-)$$

by Lemma 2.2 so that $g_k^{\mathrm{T}} s^- < 0$ (see again the proof of Theorem 2.2). Consequently $g_i^{\mathrm{T}} s^- < 0$ for all indices $i \in I_\varepsilon$ which implies that the direction vector $s^- \in R_n$ is a descent direction for the function (1.1). It is also a feasible direction since $A^{\mathrm{T}} S = 0$ by Definition 2.1 and, therefore, $A^{\mathrm{T}} s^- = A^{\mathrm{T}} S \tilde{s}^- = 0$. □

Lemma 2.8. Let the nonsingularity condition be satisfied for a given $\varepsilon > 0$. Let A, S be an orthogonal pair of matrices generated by the set J. Let $v_l < 0$ for some $l \in J$, where

$$v = (A^{\mathrm{T}}A)^{-1} A^{\mathrm{T}} g \,, \tag{2.17}$$

with g defined by (2.10). Let A^-, S^- be an orthogonal pair of matrices defined by the set $J^- = J \setminus \{l\}$ such that $S^- = [S, s_0]$ and $S^{\mathrm{T}} s_0 = 0$. Then $s^{(-)} = -S^-(S^-)^{\mathrm{T}} g$ is a feasible direction for the problem (1.1) at the feasible point $x \in L_n$. Moreover $a_l^{\mathrm{T}} s^{(-)} > 0$.

Proof. See [7]. □

Theorem 2.4. Let the feasible point $x \in L_n$ be a solution of the problem (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J. Let the assumptions of Lemma 2.8 be satisfied with $v_l < 0$. Let the triple $(\tilde{s}^-, u^-, z^-)$ be a solution of the system which results from the system (2.4) after replacing $\tilde{B} = S^{\mathrm{T}}B$ by $\tilde{B}^- = (S^-)^{\mathrm{T}} B$. Then $s^- = S^-\tilde{s}^-$ is a feasible descent direction for the problem (1.1) at the feasible point $x \in L_n$. Moreover $a_l^{\mathrm{T}} s^- > 0$.

Proof. Let $g = Bu$ and $g^- = Bu^-$. First we prove that

$$s_0^{\mathrm{T}} g^- = \frac{s_0^{\mathrm{T}} g}{1 + s_0^{\mathrm{T}} B \tilde{C} B^{\mathrm{T}} s_0} \tag{2.18}$$

where $\tilde{C}$ is a positive semidefinite matrix. The pair (u^-, z^-) is a solution of the system

$$\left(\begin{bmatrix} C^{-1}, & e \\ e^{\mathrm{T}}, & 0 \end{bmatrix} + \begin{bmatrix} B^{\mathrm{T}} s_0 \\ 0 \end{bmatrix} [s_0^{\mathrm{T}} B, \, 0] \right) \begin{bmatrix} u^- \\ z^- \end{bmatrix} = \begin{bmatrix} f + \lambda e \\ 1 \end{bmatrix},$$

which has the following form

$$(D + ww^{\mathrm{T}})\, y^- = d \,.$$

Using the Sherman-Morrison formula, we get

$$y^- = (D + ww^{\mathrm{T}})^{-1} d = \left(D^{-1} - \frac{D^{-1} w w^{\mathrm{T}} D^{-1}}{1 + w^{\mathrm{T}} D^{-1} w} \right) d = y - \frac{D^{-1} w w^{\mathrm{T}} y}{1 + w^{\mathrm{T}} D^{-1} w}$$

where $y = D^{-1}d$. With respect to the original system, we can write $y^- = [(u^-)^{\mathrm{T}}, z^-]^{\mathrm{T}}$, $y = [u^{\mathrm{T}}, z]^{\mathrm{T}}$, $w = [s_0^{\mathrm{T}} B, 0]^{\mathrm{T}}$ and

$$D^{-1} = \begin{bmatrix} C^{-1}, & e \\ e^{\mathrm{T}}, & 0 \end{bmatrix}^{-1} = \begin{bmatrix} C - \dfrac{Cee^{\mathrm{T}}C}{e^{\mathrm{T}}Ce}, & \dfrac{Ce}{e^{\mathrm{T}}Ce} \\ \dfrac{e^{\mathrm{T}}C}{e^{\mathrm{T}}Ce}, & -\dfrac{1}{e^{\mathrm{T}}Ce} \end{bmatrix}$$

so that

$$u^- = u - \frac{\tilde{C} B^{\mathrm{T}} s_0 s_0^{\mathrm{T}} B u}{1 + s_0^{\mathrm{T}} B \tilde{C} B^{\mathrm{T}} s_0}$$

where $\tilde{C} = C - Cee^{\mathrm{T}}C/e^{\mathrm{T}}Ce$ is a positive semidefinite matrix. Using the last formula, we get

$$s_0^{\mathrm{T}}g^- = s_0^{\mathrm{T}}Bu^- = s_0^{\mathrm{T}}Bu - \frac{s_0^{\mathrm{T}}B\tilde{C}B^{\mathrm{T}}s_0 s_0^{\mathrm{T}}Bu}{1 + s_0^{\mathrm{T}}B\tilde{C}B^{\mathrm{T}}s_0} = \frac{s_0^{\mathrm{T}}Bu}{1 + s_0^{\mathrm{T}}B\tilde{C}B^{\mathrm{T}}s_0} = \frac{s_0^{\mathrm{T}}g}{1 + s_0^{\mathrm{T}}B\tilde{C}B^{\mathrm{T}}s_0},$$

which is exactly (2.18). Note that positive semidefiniteness of the matrix $\tilde{C}$ implies $1 + s_0^{\mathrm{T}}B\tilde{C}B^{\mathrm{T}}s_0 \geqq 1$ so that $s_0^{\mathrm{T}}g^-$ has the same sign as $s_0^{\mathrm{T}}g$.

Now we are going to prove that $s^- = S^-\tilde{s}^-$ is a feasible direction and $a_l^{\mathrm{T}}s^- > 0$. Suffice it to prove $a_l^{\mathrm{T}}s^- > 0$ since $(A^-)^{\mathrm{T}}s^- = (A^-)^{\mathrm{T}}S^-\tilde{s}^- = 0$ follows from Definition 2.1. We can write

$$s^- = S^-\tilde{s}^- = -S^-(S^-)^{\mathrm{T}}Bu^- = -S^-(S^-)^{\mathrm{T}}g^-$$

by (2.3) and (2.4) with S replaced by S^-. Therefore,

$$a_l^{\mathrm{T}}s^- = -a_l^{\mathrm{T}}S^-(S^-)^{\mathrm{T}}g^- = -a_l^{\mathrm{T}}SS^{\mathrm{T}}g^- - a_l^{\mathrm{T}}s_0 s_0^{\mathrm{T}}g^- = -a_l^{\mathrm{T}}s_0 s_0^{\mathrm{T}}g^-$$

since $a_l^{\mathrm{T}}S = 0$ by Definition 2.1. Using Lemma 2.8, we get

$$a_l^{\mathrm{T}}s^{(-)} = -a_l^{\mathrm{T}}S^-(S^-)^{\mathrm{T}}g = -a_l^{\mathrm{T}}s_0 s_0^{\mathrm{T}}g > 0\,.$$

Since $s_0^{\mathrm{T}}g^-$ has the same sign as $s_0^{\mathrm{T}}g$ and since $a_l^{\mathrm{T}}s_0 \neq 0$ (which follows from $s_0 \neq 0$, $S^{\mathrm{T}}s_0 = 0$ and $(A^-)^{\mathrm{T}}s_0 = 0$), the last inequality implies $a_l^{\mathrm{T}}s^- > 0$, which was to be proved.

Finally we prove that $s^- = S^-\tilde{s}^-$ is a descent direction for the function (1.1). Since the feasible point $x \in L_n$ is a solution of the problem (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J, the conditions (2.13) and (2.14) are satisfied so that $I_\varepsilon = I$. Define

$$C^- = ((\tilde{B}^-)^{\mathrm{T}}\tilde{B}^- + \lambda ee^{\mathrm{T}})^{-1} = (\tilde{B}^{\mathrm{T}}\tilde{B} + \lambda ee^{\mathrm{T}} + B^{\mathrm{T}}s_0 s_0^{\mathrm{T}}B)^{-1}\,.$$

Using Sherman-Morrison formula, we get

$$C^- = C - \frac{CB^{\mathrm{T}}s_0 s_0^{\mathrm{T}}BC}{1 + s_0^{\mathrm{T}}BCB^{\mathrm{T}}s_0}$$

(see (2.7)) so that

$$e^{\mathrm{T}}C^-e = e^{\mathrm{T}}Ce - \frac{pB^{\mathrm{T}}s_0 s_0^{\mathrm{T}}Bp}{1 + s_0^{\mathrm{T}}BCB^{\mathrm{T}}s_0}\,.$$

Note that $1 + s_0^{\mathrm{T}}BCB^{\mathrm{T}}s_0 \geqq 1$ since the matrix C is positive definite. We prove that $s_0^{\mathrm{T}}Bp \neq 0$, which implies $e^{\mathrm{T}}C^-e < e^{\mathrm{T}}Ce$ and consequently $\lambda e^{\mathrm{T}}C^-e < 1$. Using (2.13) and Lemma 2.4, we get $S^{\mathrm{T}}Bp = 0$ so that $Bp = Av$ by Definition 2.1. Since $s_0^{\mathrm{T}}A^- = 0$ by Definition 2.1, $s_0^{\mathrm{T}}a_l \neq 0$ (which follows from $s_0 \neq 0$, $S^{\mathrm{T}}s_0 = 0$ and $(A^-)^{\mathrm{T}}s_0 = 0$), and $v_l < 0$, we obtain

$$s_0^{\mathrm{T}}Bp = s_0^{\mathrm{T}}Av = s_0^{\mathrm{T}}a_l v_l \neq 0\,,$$

which was to be proved. Therefore, $\lambda e^{\mathrm{T}}C^-e < 1$. Since $I_\varepsilon = I$, the set I_ε remains

unchanged for a small enough number $\varepsilon > 0$. Therefore, we can consider the condition

$$\varepsilon < (1 - \lambda e^{\mathrm{T}} C^{-} e) / \|C^{-} e\|_1$$

satisfied and, as in the proof of Theorem 2.2, we get $g_i^{\mathrm{T}} s^{-} < 0$, for all indices $i \in I_\varepsilon$, which implies that the direction vector $s^{-} = S^{-} \tilde{s}^{-}$ is a descent direction for the function (1.1). □

We have used the same notation $(\tilde{s}^{-}, u^{-}, z^{-})$ in Theorem 2.4 as in Theorem 2.3 even if they are the solutions of quite different systems. But the above theorems cannot be used simultaneously so, by our opinion, no misunderstanding can happen.

The theorems we have proved can be used for the construction of a feasible descent direction for the problem (1.1) at the feasible point $x \in L_n$. If the feasible point $x \in L_n$ is not a solution of the problem (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J, we can use either Theorem 2.1 or Theorem 2.2. Using Theorem 2.2 in case the condition $u \geqq 0$ does not hold, we have to check the validity of the condition $z - F < 0$ which guarantee the direction vector $s \in R_n$ is a descent one for function (1.1) at the feasible point $x \in L_n$. When the condition $z - F < 0$ is violated, we have to decrease the value ε (dividing it by 10 for example) and repeat the determination of the direction vector $s \in R_n$. If the feasible point $x \in L_n$ is a solution of the problem (1.3) with $I(x^*)$ replaced by I_ε and $J(x^*)$ replaced by J, we can use either Theorem 2.3 in case $u_k < 0$ for an index $k \in I_\varepsilon$ or Theorem 2.4 in case $v_l < 0$ for an index $l \in J$. Therefore, a feasible descent direction $s \in R_n$ can be determined in each feasible point $x \in L_n$ which is not a solution of the problem (1.1) and the new feasible point $x^{+} \in L_n$, where $x^{+} = x + \alpha s$, can be found such that $F(x^{+}) < F(x)$.

Numerical experiments show that it is advantageous to construct the set $I_\varepsilon^{-} = I_\varepsilon \setminus \{k\}$ and compute the direction vector $s^{-} \in R_n$ by means of (2.3) and (2.4) with I_ε replaced by I_ε^{-} whenever $u_k < 0$ for some $k \in I_\varepsilon$. The fact that the computed direction vector is a feasible descent direction for the problem (1.1) at the feasible point $x \in L_n$ can be verified by the combination of Lemma 2.6 and Theorem 2.2. Note that the validity of the condition $z - F < 0$ has to be checked in this case and the value $\varepsilon > 0$ has to be decreased if it does not hold.

3. DESCRIPTION OF THE ALGORITHM

In this section we are describing an algorithm which uses the direction vector determined by (2.3) and (2.8). First, we must introduce some details concerning certain steps of the algorithm.

To improve the stability of the algorithm, we use the triangular decomposition

$$\tilde{R}^{\mathrm{T}} \tilde{R} = \tilde{B}^{\mathrm{T}} \tilde{B} + \lambda e e^{\mathrm{T}} \tag{3.1}$$

instead of the inversion $C = (\tilde{B}^{\mathrm{T}} \tilde{B} + \lambda e e^{\mathrm{T}})^{-1}$. The upper triangular matrix $\tilde{R}$ is

computed recursively. Let $(\tilde{R}^+)^T \tilde{R}^+ = (\tilde{B}^+)^T \tilde{B}^+ + \lambda e^+(e^+)^T$, where $e^+ = [e^T, 1]^T$ and $\tilde{B}^+ = [\tilde{B}, \tilde{b}_i]$ with $\tilde{b}_i = S^T g_i$. Then

$$\tilde{R}^+ = \begin{bmatrix} \tilde{R}, & \tilde{r}_1 \\ 0, & \tilde{r}_2 \end{bmatrix} \tag{3.2}$$

where

$$\tilde{R}^T \tilde{r}_1 = \tilde{B}^T \tilde{b}_i + \lambda e\,,$$

$$\tilde{r}_2^2 = \tilde{b}_i^T \tilde{b}_i + \lambda - \tilde{r}_1^T \tilde{r}_1$$

(see for instance [15]).

After deleting the index k from the set I_e, we need to find the decomposition $(\tilde{R}^-)^T \tilde{R}^- = (\tilde{B}^-)^T \tilde{B}^- + \lambda e^-(e^-)^T$, where e^- results from the vector e after deleting the component e_k and $\tilde{B}^-$ results from the matrix $\tilde{B}$ after deleting the column $\tilde{b}_k$. Let $\tilde{P}$ be a permutation matrix which transfers the column $\tilde{b}_k$ of the matrix $\tilde{B}$ to the last position so that $\tilde{R}\tilde{P}$ is an upper Hessenberg matrix. Let $\tilde{Q}$ be an orthogonal matrix such that $\tilde{Q}\tilde{R}\tilde{P} = \tilde{T}$, where $\tilde{T}$ is an upper triangular matrix. Then $\tilde{T}^T\tilde{T} = \tilde{P}^T\tilde{R}^T\tilde{Q}^T\tilde{Q}\tilde{R}\tilde{P} = \tilde{P}^T\tilde{R}^T\tilde{R}\tilde{P}$, so that

$$\tilde{T} = \begin{bmatrix} \tilde{T}^-, & \tilde{t}_1 \\ 0, & \tilde{t}_2 \end{bmatrix} \tag{3.3}$$

and

$$(\tilde{T}^-)^T \tilde{T}^- = (\tilde{B}^-)^T \tilde{B}^-\,.$$

Therefore,

$$\tilde{R}^- = \tilde{T}^-$$

is a matrix satisfying the desired condition.

The number $\lambda > 0$ has to be taken to guarantee the positive definiteness of the matrix (3.1). The most advantageous choice of the number $\lambda > 0$ is such that makes the matrix (3.1) optimally conditioned. However, it is computationally time-consuming. Therefore, we use the simple choice $\lambda = 1$ in the algorithm.

The algorithm makes use of three additional matrices A, S and R which represent a current set of active constraints. Here A, S is an orthogonal pair of matrices generated by the set J (see Definition 2.1) and R is an upper triangular matrix such that $R^TR = A^TA$. The matrix R serves for the computation of the Lagrange multipliers from the equation

$$R^TRv = A^Tg\,, \tag{3.4}$$

which is equivalent to (2.17).

The matrices A, S and R must be adjusted after each change of the current set of active constraints. In this case the second-order information obtained by the variable metric updates in the previous steps has to be conserved. A more detailed investigation of this problem is given for instance in [7]. Suppose that a new constraint with the normal a_j, say, has to be added to the set of active constraints. Suppose that the normal a_j is not a linear combination of the normals a_i, $i \in J$ and let's set

$j^+ = J \cup \{j\}$. Then $A^+ = [A, a_j]$ and

$$R^+ = \begin{bmatrix} R, & r_1 \\ 0, & r_2 \end{bmatrix} \tag{3.5}$$

where

$$R^{\mathrm{T}} r_1 = A^{\mathrm{T}} a_j ,$$

$$r_2^2 = a_j^{\mathrm{T}} a_j - r_1^{\mathrm{T}} r_1$$

(see [3]). The matrix S^+ can be determined as in [13]. Then

$$S^+ = \tilde{S} - \left(\frac{1 - t s_k^{\mathrm{T}} a_j}{s^{\mathrm{T}} a_j} s - t s_k \right) a_j^{\mathrm{T}} \tilde{S} \tag{3.6}$$

where $\tilde{S}$ is a matrix resulting from the matrix S after deleting an arbitrary column s_k and $s = SS^{\mathrm{T}} a_j$. Furthermore t is a root of the quadratic equation

$$\omega^2 t^2 + 2 t s_k^{\mathrm{T}} a_j - 1 = 0$$

where $\omega = a_j^{\mathrm{T}} \tilde{S} \tilde{S}^{\mathrm{T}} a_j = s^{\mathrm{T}} a_j - (s_k^{\mathrm{T}} a_j)^2$. Note that A^+, S^+ is an orthogonal pair of matrices generated by the set $J^+ = J \cup \{j\}$ and

$$S^+ (S^+)^{\mathrm{T}} = SS^{\mathrm{T}} - \frac{SS^{\mathrm{T}} a_j a_j^{\mathrm{T}} SS^{\mathrm{T}}}{a_j^{\mathrm{T}} SS^{\mathrm{T}} a_j}$$

holds which guarantee that the second order information, obtained in the previous steps, has not been lost.

After deleting the index l from the set J, we need to find the matrices A^-, S^- and R^-, where A^-, S^- is an orthogonal pair of matrices generated by the set $J^- = J \setminus \{l\}$ and R^- is an upper triangular matrix such that $(R^-)^{\mathrm{T}} R^- = (A^-)^{\mathrm{T}} A^-$. A procedure described in [4] can be used in this case. Let P be a permutation matrix which transfers the column a_l of the matrix A to the last position so that RP is an upper Hessenberg matrix. Let Q be an orthogonal matrix such that $QRP = T$ where T is an upper triangular matrix. Denote

$$T = \begin{bmatrix} T^-, & t_1 \\ 0, & t_2 \end{bmatrix}, \quad h = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Let A^- be a matrix resulting from the matrix A after deleting the column a_l and

$$R^- = T^- , \tag{3.7}$$

$$S^- = [S, s_0]$$

where

$$s_0 = APT^{-1} h$$

then $(R^-)^{\mathrm{T}} R^- = (A^-)^{\mathrm{T}} A^-$ and A^-, S^- is an orthogonal pair of matrices generated by the set $J^- = J \setminus \{l\}$. Moreover $S^{\mathrm{T}} s_0 = 0$, which is the necessary assumption

of Theorem 2.2, and

$$SS^{\mathrm{T}} = S^{-}(S^{-})^{\mathrm{T}} - \frac{S^{-}(S^{-})^{\mathrm{T}} a_l a_l^{\mathrm{T}} S^{-}(S^{-})^{\mathrm{T}}}{a_l^{\mathrm{T}} S^{-}(S^{-})^{\mathrm{T}} a_l}$$

holds which guarantee that the second order information, obtained in the previous steps, has not been lost.

Having the direction vector $s \in R_n$, we need to determine a steplength α and compute the point $x^+ = x + \alpha s$, which is a new approximation of the solution of the problem (1.1). To guarantee the global convergence of the algorithm, we choose the steplength α so that

$$F(x^+) - F(x) \leqq \varepsilon_2 \alpha s^{\mathrm{T}} g$$

where g is the vector defined by (2.10) and $0 < 2\varepsilon_2 < 1$. The theoretical motivation for this choice is given for instance in [5]. The steplength α is usually determined by means of a bisection or by means of a safeguarded quadratic interpolation which uses function values only. The inequality $\alpha \leqq \alpha_2$ has to be satisfied, where

$$\alpha_2 = \min_{j \in \tilde{J}} \left(\frac{b_j - a_j^{\mathrm{T}} x}{a_j^{\mathrm{T}} s} \right) \tag{3.8}$$

and

$$\tilde{J} = \{ j \in L \setminus J : a_j^{\mathrm{T}} s < 0 \} ,$$

to maintain the point $x^+ \in L_n$ to be feasible. The efficiency of a line search strongly depends upon the initial estimate of steplength α. This initial estimate is most frequently determined by means of the linearization of the functions $f_i(x)$, $i \in K$, in the case of nonlinear minimax approximation. Let

$$\alpha_1 = \min_{i \in \tilde{I}_e} \left(\frac{F(x) - f_i(x)}{s^{\mathrm{T}} g_i - s^{\mathrm{T}} g} \right) \tag{3.9}$$

where

$$\tilde{I}_e = \{ i \in K \setminus I_e : s^{\mathrm{T}} g_i > s^{\mathrm{T}} g \} .$$

Then

$$\alpha = \min (1, \alpha_1, \alpha_2) \tag{3.10}$$

is a suitable initial estimate of the steplength.

The matrix S should be chosen in such way that SS^{T} approximates the matrix $G^{-1} - G^{-1}A(A^{\mathrm{T}}G^{-1}A)^{-1} A^{\mathrm{T}}G^{-1}$ as closely as possible, where G is the Hessian matrix of the Lagrangian function associated with the problem (1.2) (see (2.6)). Therefore, it is advantageous to use the product-form of variable metric updates which belong to Broyden's class. Let $\tilde{g} = S^{\mathrm{T}} g$ and $\tilde{g}^+ = S^{\mathrm{T}} g^+$, where

$$g = \sum_{i \in I_e} u_i g_i(x)$$

and

$$g^+ = \sum_{i \in I_\varepsilon} u_i \, g_i(x^+) ,$$

be reduced gradients of the Lagrangian function associated with the problem (1.2) at the feasible points $x \in L_n$ and $x^+ \in L_n$ respectively. Denote $\tilde{d} = \alpha\tilde{s}$, $\tilde{y} = \tilde{g}^+ - \tilde{g}$ and $\varrho = \tilde{d}^T\tilde{d}$, $\sigma = \tilde{y}^T\tilde{d}$, $\tau = \tilde{y}^T\tilde{y}$, where α is a steplength obtained by line search (so that $x^+ = x + \alpha s$). Then the matrix S can be updated by one of the formulae

$$S^+ = S + \frac{1}{\tau} S \left(\sqrt{\left(\frac{\tau}{\sigma}\right)} \tilde{d} - \tilde{y} \right) \tilde{y}^T , \tag{3.11a}$$

$$S^+ = S + \frac{1}{\sigma} S\tilde{d} \left(\sqrt{\left(\frac{\sigma}{\varrho}\right)} \tilde{d} - \tilde{y} \right)^T , \tag{3.11b}$$

$$S^+ = S + \frac{\sqrt{\left(\dfrac{\varrho - \sigma}{\sigma - \tau}\right)} - 1}{\varrho - 2\sigma + \tau} S(\tilde{d} - \tilde{y})(\tilde{d} - \tilde{y})^T , \tag{3.11c}$$

which are the DFP method, the BFGS method and the rank-one method in the product form respectively (see [1] and [14]). Note that (3.11) can be used only if $\sigma > 0$. This condition cannot be satisfied automatically since the steplength α is chosen to reduce the minimax objective function $F(x)$ while σ is computed from the difference between the reduced gradients of the Lagrangian function. Therefore, the computation of the matrix S^+ has to be slightly modified. Two procedures exhibited good efficiency during the implementation of the algorithm:

(a) The matrix S is updated by (3.11) only if

$$\sigma \geqq \varepsilon_3 \tau \tag{3.12}$$

where $0 < \varepsilon_3 < 1$. In the opposite case we set $S^+ = S$.

(b) The procedure described in [13] is used. In this case we replace, in the formulae (3.11), $\tilde{d}$ and σ by $\mu d + (1 - \mu)\tilde{y}$ and $\mu\sigma + (1 - \mu)\tau$, where

$$\mu = \min\left(1, \frac{(1 - \varepsilon_4)\tau}{\tau - \sigma}\right). \tag{3.13}$$

Now we are in a position to describe the complete algorithm. Since it is more advantageous to work with relative tolerances, we define the set I_ε by the formula

$$I_\varepsilon = \{i \in K : f_i \geqq F - \varepsilon \max(\varepsilon, |F|)\} \tag{3.14}$$

instead of (2.1). To ensure the numerical stability of the algorithm we use the set

$$J = \{j \in L : |a_j^T x - b_j| \leqq \varepsilon_1\} \tag{3.15}$$

instead of that defined in Section 2.

Algorithm 3.1.

Step 1: Determine an initial feasible point $x \in L_n$ (it can be determined by the procedure described in [2]). Compute the values $f_i := f_i(x)$, $i \in K$, and the gradients $g_i := g_i(x)$, $i \in K$. Compute the value of the objective function $F := \max_{i \in K} f_i(x)$. Set $M := 0$.

Step 2: (Restart.) Using (3.15), determine the set $J \subset L$ of indices of linear constraints, which are active at the feasible point $x \in L_n$ and which have linearly independent normals. Determine an orthogonal pair of matrices A, S generated by the set J and compute the upper triangular matrix R such that $R^T R = A^T A$. These matrices can be computed recursively by the formulae (3.5) and (3.6) where j is the index passing over all J and where A, R are empty matrices and $S = I$ (the unit matrix of order n) at the beginning of this processes. Go to Step 5.

Step 3: Set $\tilde{d} := -\alpha \tilde{g}_1$ and $\tilde{y} := \tilde{g} - \tilde{g}_1$. Compute the values $\varrho := \tilde{d}^T \tilde{d}$, $\sigma := \tilde{y}^T \tilde{d}$ and $\tau := \tilde{y}^T \tilde{y}$. If $MOD = 0$ and $\sigma \geqq \varepsilon_3 \tau$, go to Step 4. If $MOD = 0$ and $\sigma < \varepsilon_3 \tau$, go to Step 5. If $MOD = 1$ then set $\mu := \min(1, (1 - \varepsilon_4)\tau : (\tau - \sigma))$ and compute $\tilde{d} := \mu \tilde{d} + (1 - \mu)\tilde{y}$ and $\sigma := \mu\sigma + (1 - \mu)\tau$.

Step 4: (VM update.) Compute the matrix S^+ by (3.11a), (3.11b) or (3.11c) according to whether $MET = 1$, $MET = 2$ or $MET = 3$. Set $S := S^+$.

Step 5: Set $\varepsilon := \varepsilon_0$. If $NEW = 0$ then go to Step 7 else go to Step 6.

Step 6: (Addition of an active constraint.) Set $j := NEW$. Let A^+, S^+ and R^+ be matrices determined from the matrices A, S and R by (3.5) and (3.6). Set $A := A^+$, $S := S^+$ and $R := R^+$.

Step 7: Set $REM := 0$.

Step 8: Set $\lambda := 1$. Using (3.14) determine the set $I_\varepsilon \subset K$ of indices of functions, which are nearly active at the feasible point $x \in L_n$ and which have linearly independent gradients. Compute the upper triangular matrix $\tilde{R}$ such that (3.1) holds. This matrix can be computed recursively by the formula (3.2) where i is the index passing over all I_ε and where $\tilde{B}$, $\tilde{R}$ are empty matrices at the beginning of this process. If the decomposition (3.1) does not exist (i.e. if $\tilde{r}_2 \leqq \varepsilon_5$ in (3.2)) then go to Step 13.

Step 9: Determine the vector u and the number z according to (2.8) where we use the triangular decomposition (3.1) instead of the inversion (2.7). If either $REM = 1$ or $\lambda e^T p \leqq 1 - \varepsilon_6$ go to Step 11.

Step 10: Determine the index $k \in I_\varepsilon$ in such way that $u_k = \min_{i \in I_\varepsilon} (u_i)$. If $u_k + \varepsilon_5 > 0$, go to Step 11. If $u_k + \varepsilon_5 \leqq 0$ then set $I_\varepsilon := I_\varepsilon \setminus \{k\}$, $\tilde{R} := \tilde{R}^-$ where $\tilde{R}^-$ is the upper triangular matrix determined by (3.3), set $REM := 1$ and go to Step 9.

Step 11: Compute $g := Bu$ and $\tilde{g} := S^T g$ where B is the matrix which contains the gradients g_i, $i \in I_\varepsilon$, as its columns. If $REM = 0$, go to Step 14.

Step 12: If $z - F + \varepsilon_5 \leqq 0$, go to Step 17.

Step 13: If $\varepsilon \leqq \varepsilon_5$ then terminate the computation (the algorithm fails). If $\varepsilon > \varepsilon_5$, then set $\varepsilon := \varepsilon/10$ and go to Step 7.

Step 14: If $\|\tilde{g}\| > \varepsilon_5$, go to Step 17.

Step 15: Compute the vector v by means of (3.4). Determine the index $l \in J$ in such way that $v_l = \min_{j \in J}(v_j)$ and set $OLD := j$. If $v_l + \varepsilon_5 > 0$ then terminate the computation (the solution of the problem (1.1) has been found with required precision).

Step 16: (Deletion of an active constraint.) Set $j := OLD$, Let A^-, S^- and R^- be matrices determined from the matrices A, S and R by (3.7). Set $A := A^-$, $S := S^-$ and $R := R^-$. Set $REM := 1$ and go to Step 8.

Step 17: (Determination of the direction vector.) Set $s := -S\tilde{g}$. Determine the trial steplength α_1 by (3.9) and the maximum steplength α_2 by (3.8). Set $NEW := j$, where j is the index of the constraint, which becomes active for the maximum steplength has been chosen.

Step 18: (Line search.) Set $x_1 := x$, $\tilde{g}_1 := \tilde{g}$ and $F_1 := F$. Determine the steplength α to satisfy the conditions $0 < \alpha \leqq \alpha_2$ and $F_1 - F \geqq \varepsilon_2 \alpha \|g\|$ (the initial estimate of α is given by (3.10)). Set $x := x_1 + \alpha s$. Compute the values $f_i := f_i(x)$, $i \in K$ and the gradients $g_i := g_i(x)$, $i \in K$. Compute the value of the objective function $F := \max_{i \in K} f_i(x)$.

Step 19: Compute $g := Bu$ and $\tilde{g} := S^T g$, where B is the matrix which contains the gradients g_i, $i \in I_\varepsilon$, as its columns.

Step 20: If $(\alpha_2 - \alpha)\|s\| > \varepsilon_2$, set $NEW := 0$. Set $M := M + 1$. If the restart is required (after a certain number of iterations) then go to Step 2 else go to Step 3.

Algorithm 3.1 can be controlled by the integers MET and MOD. The parameter MET serves for the selection of the variable metric update from (3.11) (standard value is $MET = 2$). The parameter MOD determines a modification of the variable metric method. If $MOD = 0$ then the procedure (a) is used (see (3.12)) while if $MOD = 1$ then the procedure (b) is used (see (3.13)). Algorithm 3.1 uses additional integers M, REM, NEW and OLD. Here M is an iteration count, REM is an integer indicating whether the index k was deleted from the set I_ε in case the corresponding Lagrange multiplier u_k was negative, NEW is the index of the constraint added to the set J and OLD is the index of the constraint deleted from the set J. Besides, Algorithm 3.1 uses several tolerances. The values $10^{-2} \leqq \varepsilon_0 \leqq 10^{-1}$, $\varepsilon_1 = 10^{-6}$, $\varepsilon_2 = 10^{-2}$, $\varepsilon_3 = 10^{-2}$, $\varepsilon_4 = 10^{-1}$, $\varepsilon_5 = 10^{-10}$ and $\varepsilon_6 = 10^{-2}$ were used in the implementation of this algorithm on an IBM 370/135 computer in double precision arithmetic.

Algorithm 3.1 can fail in Step 13. This situation arises only if either the regularity condition is satisfied for no value $\varepsilon \geqq \varepsilon_5$ (see Definition 2.2) or the condition (2.15) is violated for all values $\varepsilon \geqq \varepsilon_5$.

4. NUMERICAL EXPERIMENTS

The efficiency of Algorithm 3.1 has been tested by means of test problems which are given in [11]. Results of the tests are shown in several tables. Each row of each table corresponds to one example numbered as in [11]. Each column of each table corresponds to one run of the algorithm. Several numbers are given for each run and each example:

NI – number of iterations (value of integer M after termination),

NF – number of different points at which the values $f_i(x)$, $i \in K$, were computed,

NG – number of different points at which the gradients $g_i(x)$, $i \in K$, were computed,

P – relative precision of the computed optimal value of the objective function $F(x)$ (only in Table 3a and Table 3b).

All tests have been performed with two different line search procedures. The letter A indicates the line search procedure in which the usual quadratic interpolation has been used while the letter B refers to that where each function $f_i(x)$, $i \in K$, has been approximated by means of a special parabola (see [11] for details).

Table 1a and Table 1b demonstrate a considerable influence of the value of parameter ε_0, which serves for the definition of nearly active functions. These tables correspond to the choice $MET = 2$ and $MOD = 0$.

The numerical experiments summarized in Table 1a and Table 1b show that the value $\varepsilon_0 = 10^{-1}$ is advantageous in all cases with the exception of the problem U 5 where the value $\varepsilon_0 = 10^{-2}$ gives better results. The tests of Algorithm 3.1 have been done also for values $\varepsilon_0 < 10^{-2}$. In all these cases the efficiency of Algorithm 3.1 was less than in those given in Table 1a and Table 1b. Similar results have been reached for the choice $MET = 2$ and $MOD = 1$ but the previous one was slightly more efficient.

Table 2a and Table 2b contain results of the tests for different values of the controlling parameter MET. The value $\varepsilon_0 = 10^{-1}$ was used in all cases together with the choice $MOD = 0$.

Table 2a and Table 2b show that the choice of the variable metric method has no expressive influence on the efficiency of Algorithm 3.1.

To compare several methods for linearly constrained nonlinear minimax approximation, Table 3a and Table 3b have been set. These tables contain results of the tests for the method of recursive linear programming [12] (LP), the variable metric method described in [8] (VM), the method of recursive quadratic programming [11] with dual quadratic programming algorithm [9] (QP) and the new method described in this paper (Algorithm 3.1). The last two methods have been tested in two different runs which have used two different line search procedures (the same as in the previous tables). The values $\varepsilon_0 = 10^{-1}$, $MET = 2$ and $MOD = 0$ were used in the Algorithm 3.1 with exception of the case U 5 where the value $\varepsilon_0 = 10^{-2}$ was used instead

Table 1a.

	Line search A			Line search B		
	$\varepsilon_0 = 10^{-1}$	$\varepsilon_0 = 5 \cdot 10^{-2}$	$\varepsilon_0 = 10^{-2}$	$\varepsilon_0 = 10^{-1}$	$\varepsilon_0 = 5 \cdot 10^{-2}$	$\varepsilon_0 = 10^{-2}$
U 1	NI = 10 NF = 12 NG = 12	NI = 11 NF = 13 NG = 13	NI = 13 NF = 15 NG = 15	NI = 10 NF = 12 NG = 12	NI = 11 NF = 13 NG = 13	NI = 13 NF = 15 NG = 15
U 2	NI = 11 NF = 15 NG = 13	NI = 14 NF = 21 NG = 16	NI = 15 NF = 20 NG = 17	NI = 10 NF = 15 NG = 12	NI = 11 NF = 16 NG = 13	NI = 11 NF = 17 NG = 13
U 3	NI = 61 NF = 71 NG = 63	NI = 51 NF = 58 NG = 53	NI = 49 NF = 56 NG = 51	NI = 51 NF = 56 NG = 53	NI = 50 NF = 61 NG = 52	NI = 57 NF = 68 NG = 59
U 4	NI = 18 NF = 21 NG = 20	NI = 22 NF = 30 NG = 24	NI = 36 NF = 45 NG = 38	NI = 20 NF = 26 NG = 22	NI = 21 NF = 28 NG = 23	NI = 23 NF = 32 NG = 25
U 5	NI = 197 NF = 387 NG = 199	NI = 103 NF = 199 NG = 105	NI = 25 NF = 42 NG = 27	NI = 17 NF = 30 NG = 19	NI = 19 NF = 30 NG = 21	NI = 18 NF = 28 NG = 20
U 6	NI = 19 NF = 26 NG = 21	NI = 55 NF = 87 NG = 57	NI = 44 NF = 62 NG = 46	NI = 17 NF = 22 NG = 19	NI = 41 NF = 58 NG = 43	NI = 36 NF = 43 NG = 38
U 7	NI = 26 NF = 29 NG = 28	NI = 32 NF = 37 NG = 34	NI = 52 NF = 77 NG = 61	NI = 26 NF = 29 NG = 28	NI = 32 NF = 35 NG = 34	NI = 57 NF = 79 NG = 59

Table 1b.

	Line search A			Line search B		
	$\varepsilon_0 = 10^{-1}$	$\varepsilon_0 = 5 \cdot 10^{-2}$	$\varepsilon_0 = 10^{-2}$	$\varepsilon_0 = 10^{-1}$	$\varepsilon_0 = 5 \cdot 10^{-2}$	$\varepsilon_0 = 10^{-2}$
L 1	NI = 9 NF = 11 NG = 11	NI = 11 NF = 13 NG = 13	NI = 11 NF = 13 NG = 13	NI = 9 NF = 11 NG = 11	NI = 11 NF = 13 NG = 13	NI = 11 NF = 13 NG = 13
L 2	NI = 4 NF = 6 NG = 6	NI = 4 NF = 6 NG = 6	NI = 4 NF = 6 NG = 6	NI = 4 NF = 6 NG = 6	NI = 4 NF = 6 NG = 6	NI = 4 NF = 6 NG = 6
L 3	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15
L 4	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77
L 5	NI = 28 NF = 30 NG = 30	NI = 26 NF = 28 NG = 28	NI = 75 NF = 77 NG = 77	NI = 28 NF = 30 NG = 30	NI = 26 NF = 28 NG = 28	NI = 75 NF = 77 NG = 77

Table 2a.

	Line search A			Line search B		
	MET = 1	*MET* = 2	*MET* = 3	*MET* = 1	*MET* = 2	*MET* = 3
U 1	NI = 10 NF = 12 NG = 12	NI = 10 NF = 12 NG = 12	NI = 10 NF = 12 NG = 12	NI = 10 NF = 12 NG = 12	NI = 10 NF = 12 NG = 12	NI = 10 NF = 12 NG = 12
U 2	NI = 12 NF = 17 NG = 14	NI = 11 NF = 15 NG = 13	NI = 11 NF = 14 NG = 13	NI = 10 NF = 15 NG = 12	NI = 10 NF = 15 NG = 12	NI = 10 NF = 15 NG = 12
U 3	NI = 45 NF = 53 NG = 47	NI = 61 NF = 71 NG = 63	NI = 46 NF = 53 NG = 48	NI = 38 NF = 41 NG = 40	NI = 51 NF = 56 NG = 53	NI = 43 NF = 49 NG = 45
U 4	NI = 37 NF = 40 NG = 39	NI = 18 NF = 21 NG = 20	NI = 17 NF = 21 NG = 19	NI = 98 NF = 149 NG = 100	NI = 20 NF 26 NG = 22	NI = 22 NF = 30 NG = 24
U 5	NI = 196 NF = 386 NG = 198	NI = 197 NF = 387 NG = 199	NI = 197 NF = 386 NG = 198	NI = 20 NF = 34 NG = 22	NI = 17 NF = 30 NG = 19	NI = 23 NF = 39 NG = 25
U 6	NI = 33 NF = 54 NG = 35	NI = 19 NF = 26 NG = 21	NI = 35 NF = 59 NG = 37	NI = 16 NF = 21 NG = 18	NI = 17 NF = 22 NG = 19	NI = 18 NF = 26 NG = 20
U 7	NI = 30 NF = 41 NG = 32	NI = 26 NF = 29 NG = 28	NI = 30 NF = 45 NG = 32	NI = 27 NF = 36 NG = 29	NI = 26 NF = 29 NG = 28	NI = 25 NF = 35 NG = 27

Table 2b.

	Line search A			Line search B		
	MET = 1	*MET* = 2	*MET* = 3	*MET* = 1	*MET* = 2	*MET* = 3
L 1	NI = 9 NF = 11 NG = 11	NI = 9 NF = 11 NG = 11	NI = 9 NF = 11 NG = 11	NI = 9 NF = 11 NG = 11	NI = 9 NF = 11 NG = 11	NI = 9 NF = 11 NG = 11
L 2	NI = 4 NF = 6 NG = 6	NI = 4 NF = 6 NG = 6	NI = 3 NF = 5 NG = 5	NI = 4 NF = 6 NG = 6	NI = 4 NF = 6 NG = 6	NI = 3 NF = 5 NG = 5
L 2	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15	NI = 13 NF = 15 NG = 15
L 4	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77	NI = 75 NF = 77 NG = 77
L 5	NI = 39 NF = 41 NG = 41	NI = 28 NF = 30 NG = 30	NI = 36 NF = 40 NG = 38	NI = 39 NF = 41 NG = 41	NI = 28 NF = 30 NG = 30	NI = 37 NF = 40 NG = 39

Table 3a.

	LP	VM	QP — A	QP — B	New — A	New — B
U 1	NI = 13 NF = 25 NG = 25 P ≐ 10^{-7}	NI = 18 NF = 20 NG = 20 P ≐ 10^{-7}	NI = 8 NF = 10 NG = 10 P ≐ 10^{-10}	NI = 8 NF = 10 NG = 10 P ≐ 10^{-10}	NI = 10 NF = 12 NG = 12 P ≐ 10^{-10}	NI = 10 NF = 12 NG = 12 P ≐ 10^{-10}
U 2	NI = 18 NF = 28 NG = 28 P ≐ 10^{-6}	NI = 26 NF = 36 NG = 28 P ≐ 10^{-4}	NI = 12 NF = 19 NG = 14 P ≐ 10^{-10}	NI = 9 NF = 13 NG = 11 P ≐ 10^{-10}	NI = 11 NF = 15 NG = 13 P ≐ 10^{-10}	NI = 10 NF = 15 NG = 12 P ≐ 10^{-10}
U 3	NI = 10 NF = 14 NG = 14 P ≐ 10^{-6}	NI = 134 NF = 142 NG = 136 P ≐ 10^{-5}	NI = 298 NF = 589 NG = 300 P ≐ 10^{-6}	F	NI = 61 NF = 71 NG = 63 P ≐ 10^{-9}	NI = 51 NF = 56 NG = 53 P ≐ 10^{-9}
U 4	NI = 230 NF = 252 NG = 252 P ≐ 10^{-4}	NI = 63 NF = 73 NG = 65 P ≐ 10^{-5}	NI = 14 NF = 19 NG = 16 P ≐ 10^{-10}	NI = 14 NF = 17 NG = 16 P ≐ 10^{-10}	NI = 18 NF = 21 NG = 20 P ≐ 10^{-10}	NI = 20 NF = 26 NG = 22 P ≐ 10^{-9}
U 5	NI = 43 NF = 51 NG = 51 P ≐ 10^{-5}	NI = 48 NF = 78 NG = 50 P ≐ 10^{-4}	NI = 249 NF = 493 NG = 251 P ≐ 10^{-9}	NI = 17 NF = 30 NG = 19 P ≐ 10^{-10}	NI = 25 NF = 42 NG = 27 P ≐ 10^{-10}	NI = 18 NF = 28 NG = 20 P ≐ 10^{-10}
U 6	NI = 66 NF = 78 NG = 78 P ≐ 10^{-5}	NI = 102 NF = 120 NG = 104 P ≐ 10^{-4}	NI = 15 NF = 20 NG = 17 P ≐ 10^{-10}	NI = 16 NF = 20 NG = 18 P ≐ 10^{-10}	NI = 19 NF = 26 NG = 21 P ≐ 10^{-10}	NI = 17 NF = 22 NG = 19 P ≐ 10^{-10}
U 7	NI = 122 NF = 128 NG = 128 P ≐ 10^{-9}	F	NI = 21 NF = 33 NG = 23 P ≐ 10^{-10}	NI = 19 NF = 30 NG = 21 P ≐ 10^{-10}	NI = 26 NF = 29 NG = 28 P ≐ 10^{-10}	NI = 26 NF = 29 NG = 28 P ≐ 10^{-10}

of the value $\varepsilon_0 = 10^{-1}$. The letter F in Table 3a and Table 3b denotes the case when the algorithm fails.

Results of the tests show a considerable reliability and efficiency of the new algorithm (the efficiency is measured by the number of calls of the subroutine for computing values $f_i(x)$, $i \in K$, and the gradients $g_i(x)$. $i \in K$. Algorithm 3.1 is comparable with the QP method but it requires no solution of a quadratic programming subproblem. It is also more efficient than the LP method and the VM method.

The LP algorithm, VM algorithm, QP algorithm and the new algorithm were implemented as Fortran subroutines POMX 61, POMX 62, POMX 66 and POMX 65, respectively, in the Software Package for Optimization and Nonlinear Approximation SPONA (see [6]). All results given in this section have been obtained using these subroutines.

(Received May 12, 1984.)

Table 3b.

	LP	VM	QP − A	QP − B	New − A	New − B
L 1	NI = 7 NF = 9 NG = 9 P ≐ 10^{-8}	NI = 11 NF = 13 NG = 13 P ≐ 10^{-8}	NI = 6 NF = 8 NG = 8 P ≐ 10^{-10}	NI = 6 NF = 8 NG = 8 P ≐ 10^{-10}	NI = 9 NF = 11 NG = 11 P ≐ 10^{-10}	NI = 9 NF = 11 NG = 11 P ≐ 10^{-10}
L 2	NI = 56 NF = 76 NG = 76 P ≐ 10^{-10}	NI = 4 NF = 6 NG = 6 P ≐ 10^{-10}	NI = 4 NF = 6 NG = 6 P ≐ 10^{-10}	NI = 4 NF = 6 NG = 6 P ≐ 10^{-10}	NI = 4 NF = 6 NG = 6 P ≐ 10^{-10}	NI = 4 NF = 6 NG = 6 P ≐ 10^{-10}
L 3	NI = 9 NF = 11 NG = 11 P ≐ 10^{-6}	NI = 12 NF = 14 NG = 14 P ≐ 10^{-4}	NI = 7 NF = 9 NG = 9 P ≐ 10^{-9}	NI = 7 NF = 9 NG = 9 P ≐ 10^{-9}	NI = 13 NF = 15 NG = 15 P ≐ 10^{-10}	NI = 13 NF = 15 NG = 15 P ≐ 10^{-10}
L 4	NI = 7 NF = 9 NG = 9 P ≐ 10^{-6}	NI = 74 NF = 76 NG = 76 P ≐ 10^{-6}	NI = 75 NF = 77 NG = 77 P ≐ 10^{-10}	NI = 75 NF = 77 NG = 77 P ≐ 10^{-10}	NI = 75 NF = 77 NG = 77 P ≐ 10^{-10}	NI = 75 NF = 77 NG = 77 P ≐ 10^{-10}
L 5	NI = 7 NF = 11 NG = 11 P ≐ 10^{-7}	NI = 35 NF = 37 NG = 37 P ≐ 10^{-6}	NI = 10 FN = 14 NG = 12 P ≐ 10^{-9}	NI = 10 NF = 14 NG = 12 P ≐ 10^{-10}	NI = 28 NF = 30 NG = 30 P ≐ 10^{-10}	NI = 28 NF = 30 NG = 20 P ≐ 10^{-10}

REFERENCES

[1] K. W. Brodlie, A. R. Gourlay, and J. Greenstadt: Rank one and two corrections to positive definite matrices expressed in product form. J. Inst. Math. Appl. *11* (1973), 1, 73—82.

[2] R. Fletcher: The calculation of feasible points for linearly constrained optimization problems. A.E.R.E. Harwell Rept. No. R-6354 (1970).

[3] P. E. Gill and W. Murray: A numerically stable form of the simplex algorithm. Linear Algebra Appl. *7* (1973), 2, 99—138.

[4] P. E. Gill and W. Murray: Newton-type methods for unconstrained and linearly constrained optimization. Math. Programming *7* (1974), 3, 311—350.

[5] S. P. Han: Variable metric methods for minimizing a class of nondifferentiable functions. Math. Programming *20* (1981), 1, 1—13.

[6] L. Lukšan: Software package for optimization and nonlinear approximation. Proc. of the 2nd IFAC/IFIP Symposium on Software for Computer Control, Prague 1979.

[7] L. Lukšan: Quasi-Newton methods without projections for linearly constrained minimization. Kybernetika *18* (1982), 4, 307—319.

[8] L. Lukšan: Variable metric methods for linearly constrained nonlinear minimax approximation. Computing *30* (1983), 3, 315—334.

[9] L. Lukšan: Dual method for solving a special problem of quadratic programming as a subproblem at linearly constrained nonlinear minimax approximation. Kybernetika *20* (1984), 6, 445—457.

[10] L. Lukšan: A compact variable metric algorithm for nonlinear minimax approximation. Computing (to appear).

[11] L. Lukšan: An implementation of recursive quadratic programming variable metric methods for linearly constrained nonlinear minimax approximation. Kybernetika *21* (1985), 1, 22—40.
[12] K. Madsen and H. Schjaer-Jacobsen: Linearly constrained minimax optimization. Math. Programming *14* (1978), 2, 208—223.
[13] M. J. D. Powell: A fast algorithm for nonlinearly constrained optimization calculations. In: Numerical Analysis, Dundee 1977 (G. A. Watson, ed.), Lecture Notes in Mathematics 630, Springer-Verlag, Berlin 1978.
[14] K. Ritter: A variable metric method for linearly constrained minimization problems. In: Nonlinear Programming 3 (O. L. Mangasarian, R. R. Meyer and S. M. Robinson, eds.), Academic Press, London 1978.
[15] P. Wolfe: Finding the nearest point in a polytope. Math. Programming *11* (1976), 2, 128—149.

Ing. Ladislav Lukšan, CSc., Středisko výpočetní techniky ČSAV (General Computing Centre — Czechoslovak Academy of Sciences), Pod vodárenskou věží 4, 182 07 Praha 8. Czechoslovakia.