PMIN - A Recursive Quadratic Programming Variable Metric Algorithm for Minimax Optimization

Lukšan, Ladislav
1997

INSTITUTE OF COMPUTER SCIENCE

ACADEMY OF SCIENCES OF THE CZECH REPUBLIC

Prague

PMIN - A Recursive Quadratic Programming Variable Metric Algorithm for Minimax Optimization

L. Lukšan, J. Vlček

Technical Report No. V-717

September 1997

Akademie věd České republiky

ÚSTAV INFORMATIKY A VÝPOČETNÍ TECHNIKY

Institute of Computer Science, Academy of Sciences of the Czech Republic

Pod vodárenskou věží 2, 182 07 Prague 8, Czech Republic

E-mail: ICS@uivt.cas.cz

Fax: (+422) 8585789 Phone: (+422) 846669, (+422) 66051111

PMIN - A Recursive Quadratic Programming Variable Metric Algorithm for Minimax Optimization [1]

L. Lukšan and J. Vlček
Institute of Computer Science, Academy of Sciences of the Czech Republic,
Pod vodárenskou věží 2, 182 07 Prague 8, Czech Republic

Abstract. We present FORTRAN subroutines for nonlinear minimax optimization with simple bounds and general linear constraints based on a recursive quadratic programming variable metric algorithm.

Categories and Subject Descriptors:

General Terms: Algorithms

Additional Key Words and Phrases: Minimax optimization, discrete Chebyshev approximation, recursive quadratic programming methods, variable metric methods, general linear constraints

---

1. Introduction

The double-precision FORTRAN 77 basic subroutine PMIN is designed to find a close approximation to a local minimum of a special objective function

$$F(x) = \max_{1 \le i \le n_a} f_i(x)$$

(minimax) with simple bounds on variables and general linear constraints. Here $x \in R^n$ is a vector of n variables and $f_i : R^n \to R$, $1 \le i \le m$, are twice continuously differentiable functions. Simple bounds are assumed in the form (I^x, I^c correspond to the arays `IX, IC` in Section 3)

$$\begin{aligned}
x_i - \text{unbounded} \quad &, \quad I_i^x = 0, \\
x_i^l \le x_i \quad &, \quad I_i^x = 1, \\
x_i \le x_i^u \quad &, \quad I_i^x = 2, \\
x_i^l \le x_i \le x_i^u \quad &, \quad I_i^x = 3, \\
x_i = x_i^l = x_i^u \quad &, \quad I_i^x = 5,
\end{aligned}$$

where $1 \le i \le n$. General linear constraints are assumed in the form

$$a_i^T x - \text{unbounded} \quad , \quad I_i^c = 0,$$

[1] This work was supported under grant No. 201/96/0918 given by the Czech Republic Grant Agency

$$
\begin{aligned}
c_i^l \le a_i^T x \quad &, \quad I_i^c = 1, \\
a_i^T x \le c_i^u \quad &, \quad I_i^c = 2, \\
c_i^l \le a_i^T x \le c_i^u \quad &, \quad I_i^c = 3, \\
a_i^T x = c_i^l = c_i^u \quad &, \quad I_i^c = 5,
\end{aligned}
$$

where $1 \le i \le n_c$ and n_c is a number of general linear constraints. To simplify user's work, three additional easy to use subroutines are added. They call the basic general subroutine PMIN:

PMINU - unconstrained minimax optimization

PMINS - minimax optimization with simple bounds

PMINL - minimax optimization with simple bounds and general linear constraints

All subroutines contain a description of formal parameters and extensive comments. Furthermore, two test programs TMINU and TMINL are included, which contain 7 and 6 test problems (see e.g. [6]). These test programs serve as examples for using the subroutines, verify their correctness and demonstrate their efficiency.

2. The method

To simplify the description of the method, we will consider a simpler problem written in the following compact form

$$
x^\star = \arg \min_{x \in L^n} (\max_{i \in M_1} f_i(x)), \tag{1}
$$

where

$$
L^n = \{x \in R^n : a_i^T x \le b_i, i \in M_2\}
$$

with $M_1 \cap M_2 = \emptyset$. It is clear that the application of the method described below to the general problem stated in the previous section is straightforward, but this requires to consider each type of constraint separately as it is realized in the subroutine PMIN.

2.1 Recursive quadratic programming variable metric method for nonlinear minimax optimization

If we introduce a new variable z, then the problem (1) can be reformulated as a nonlinear programming problem

$$
(x^\star, z^\star) = \arg \min_{(x,z) \in N^{n+1}} z, \tag{2}
$$

where

$$
N^{n+1} = \{(x,z) \in R^{n+1} : f_i(x) \le e_i z, i \in M_1 \cup M_2\}
$$

with $e_i = 1$ for $i \in M_1$ and $e_i = 0$, $f_i(x) = a_i^T x - b_i$ for $i \in M_2$. This nonlinear programming problem can be solved by a recursive quadratic programming method that uses a quadratic approximation of the Lagrangian function and a linear approximation of the constraints in each iteration. Let $x^k \in R^n$ be a current approximation to the minimum $x^\star$. Then the resulting quadratic programming subproblem has the form

$$(d^k, z^k) = \arg \min_{(d,z) \in L_k^{n+1}} (\frac{1}{2} d^T G^k d + z), \tag{3}$$

where G^k is an approximation of the Hessian matrix of the Lagrangian function and

$$L_k^{n+1} = \{(d, z) \in R^{n+1} : f_i^k + (a_i^k)^T d \le e_i z, i \in M_1 \cup M_2\}$$

with $e_i = 1$, $a_i^k = \nabla f_i(x^k)$ for $i \in M_1$ and $e_i = 0$, $a_i^k = a_i$ for $i \in M_2$. The solution of the quadratic programming subproblem (3) has to satisfy the Karush-Kuhn-Tucker conditions

$$\begin{aligned} d^k &= -H^k g^k \\ e^T u^k &= 1, \\ u^k &\ge 0, \\ v^k &\ge 0, \\ (v^k)^T u^k &= 0, \end{aligned}$$

where u^k is the vector of Lagrange multipliers, $H^k = (G^k)^{-1}$, $A^k = [a_1^k, \dots, a_m^k]$, $e = [e_1, \dots, e_m]^T$, $f^k = [f_1^k, \dots, f_m^k]^T$, $g^k = A^k u^k$ is the gradient of the Lagrangian function and $v^k = z^k e - f^k - (A^k)^T d^k$ is the vector of constraint violations. Note that if $g^k = 0$, then we obtain the Karush-Kuhn-Tucker conditions for the nonlinear programming problem (2) exactly so that the minimax problem (1) is solved. Therefore, the condition $\|g^k\|_\infty \le TOLG$ is used in the subroutine PMIN as the basic stopping criterion (when it is fulfilled then ITERM=4).

The direction vector $d^k \in R^n$ obtained as the solution to the quadratic programming subproblem (3) is used for the definition of the new approximation x^{k+1} to the minimum $x^\star$ by the formula

$$x^{k+1} = x^k + \alpha^k d^k,$$

where $0 < \alpha^k \le 1$ is a steplength, which is chosen in such a way that

$$F(x^k + \alpha^k d^k) - F(x^k) \le \underline{\varepsilon} \alpha^k (d^k)^T g^k,$$

where $0 < \underline{\varepsilon} < 1$ is a tolerance for function decrease in the line search (parameter TOLS in the subroutine PMIN). The steplength α^k is chosen iteratively either by the bisection (MES=1) or by two point quadratic interpolation (MES=2) or by three point quadratic interpolation (MES=3) or by three point cubic interpolation (MES=4) (MES is a parameter of the subroutine PMIN).

Having the new approximation x^{k+1} to the minimum $x^\star$, we can compute the new matrix $A^{k+1} = [a_1^{k+1}, \dots, a_m^{k+1}]$, where $a_i^{k+1} = \nabla f_i(x^{k+1})$ for $i \in M_1$ and $a_i^{k+1} = a_i$ for $i \in M_2$. If we denote $s^k = x^{k+1} - x^k$ and $y^k = A^{k+1}u^k - A^k u^k = A^{k+1}u^k - g^k$, then the BFGS ([1], [2], [4], [8]) method consists in the following update

$$G^{k+1} = \frac{1}{\gamma^k}(G^k + \gamma^k \frac{y^k (y^k)^T}{(s^k)^T y^k} - \frac{G^k s^k (G^k s^k)^T}{(s^k)^T G^k s^k}) = \frac{1}{\gamma^k}(G^k + \gamma^k \frac{y^k (y^k)^T}{(s^k)^T y^k} + \alpha^k \frac{g^k (g^k)^T}{(s^k)^T g^k}),$$

where $\gamma^k > 0$ is a self scaling parameter. This parameter is usually equal to one with the exception of the first iteration (or iteration after the restart) where either $\gamma^k = 1$ if MET=1 or $\gamma^k = (s^k)^T G^k s^k / (s^k)^T y^k = -\alpha^k (s^k)^T g^k / (s^k)^T y^k$ if MET=2 (MET is a parameter of the subroutine PMIN). The BFGS method requires the condition $(s^k)^T y^k > 0$ to be satisfied, which guarantees a positive definiteness of the matrix G^{k+1}. Unfortunately, this condition does not hold in the minimax optimization case automatically. Therefore, we set $G^{k+1} = G^k$ whenever $(s^k)^T y^k \le 0$.

2.2 Dual range space method for a special quadratic programming subproblem

Consider a quadratic programming problem in which we seek a pair $(d^\star, z^\star) \in R^{n+1}$ in such a way that

$$\phi(d^\star, z^\star) = \min_{(d,z) \in L^{n+1}} \phi(s, z), \tag{4}$$

where

$$\phi(s, z) = \frac{1}{2} d^T G d + z$$

and

$$L^{n+1} = \{(d, z) \in R^{n+1} : f_i + a_i^T d \le e_i z, i \in M_1 \cup M_2\}$$

(see (3). The fact that the matrix G is positive definite implies that the the problem (4) is convex and we can apply the duality theory to obtain a dual quadratic programming problem which consists in seeking a vector $u^\star \in R^m$ (vector of Lagrange multipliers of (4)) so that

$$\psi(u^\star) = \min_{u \in L^m} \psi(u), \tag{5}$$

where

$$\psi(u) = \frac{1}{2} u^T A^T H A u - f^T u$$

and

$$L^m = \{u \in R^m : e^T u = 1, u \ge 0\}.$$

Here $H = G^{-1}$, $A = [a_1, \dots, a_m]$, $f = [f_1, \dots, f_m]^T$, $e = [e_1, \dots, e_m]^T$, where $f_i = f_i(x)$, $e_i = 1$ for $i \in M_1$ and $f_i = a_i^T x - b_i$, $e_i = 0$ for $i \in M_2$. The solution of (4) can be obtained from the solution of (5) by the formulas

$$d^\star = -HAu^\star \tag{6}$$

and

$$z^\star = f^T u^\star - (u^\star)^T A^T H A u^\star. \tag{7}$$

The solution $u^\star$ of (5) is the optimum vector of Lagrange multipliers for (4). Since the problem (5) is convex, $u^\star$ is its solution if and only if the Karush-Kuhn-Tucker conditions are valid, i.e. if and only if

$$e^T u^\star = 1, \quad u^\star \geq 0 \tag{8}$$

and a number $z^\star$ exists in such a way that

$$v^\star = A^T H A u^\star - f + z^\star e \geq 0, \quad (v^\star)^T u^\star = 0. \tag{9}$$

Vector $v^\star$ is the vector of Lagrange multipliers of the problem (5). Conditions (6) and (9) imply that $z^\star$ in (9) is identical with $z^\star$ in (7). This in turn implies that $v^\star$ is, at the same time, the vector of constraint values of the problem (4).

Consider any subset $I \subset M = M_1 \cup M_2$ and denote the vectors of elements u_i, f_i, e_i, $i \in I$ by u, f, e, respectively. Similarly, let A be the matrix of columns a_i, $i \in I$. To connect two separate cases which can occur in an investigation of a dual range space method together, we introduce an artificial parameter $\eta > 0$ and denote

$$\tilde{A} = \begin{bmatrix} A \\ -e^T \end{bmatrix}, \quad \tilde{H} = \begin{bmatrix} H & 0 \\ 0 & \eta \end{bmatrix}.$$

We will suppose that the subset $I \subset M = M_1 \cup M_2$ was chosen in such a way that the columns of $\tilde{A}$ are linearly independent.

If $I = I^*$ were the set of active constraints of the problem (4), then we could compute the dual variables $z^\star$ and $u^\star$ from (8)-(9). Unfortunately, this set is not known a priory. Therefore, we start with the set $I = \{k\}$, where $k \in M_1$ is arbitrary. Then $z = f_k - a_k^T H a_k$ and $u = [1]$. Suppose that $I \subset M = M_1 \cup M_2$ is a current subset and z, u are corresponding dual variables. Then we can proceed in the following way. First we compute the direction vector $d = -HAu$ and the value of the most violated primal constraint

$$v_k = z e_k - f_k - a_k^T d = \min_{i \in M \setminus I} \{z e_i - f_i - a_i^T d\}.$$

If $v_k \geq 0$ then the set of active constraints has been detected and the solutions of (4) and (5) have been found. Otherwise, we set $u_k = 0$ and compute the primal and dual steplengths

$$\begin{aligned} \alpha_P &= -\frac{v_k}{\beta_k \gamma_k + \delta_k} \\ \alpha_D &= \frac{u_j}{q_{kj} + \gamma_k p_j} = \min_{i \in \overline{I}} \frac{u_i}{q_{ki} + \gamma_k p_i}, \end{aligned}$$

where $p = (\tilde{A}^T \tilde{H} \tilde{A})^{-1} e$, $q_k = (\tilde{A}^T \tilde{H} \tilde{A})^{-1} \tilde{A}^T \tilde{H} \tilde{a}_k$, $\beta_k = e_k - e^T q_k$, $\gamma_k = \beta_k / p^T e$, $\delta_k = \tilde{a}_k^T (\tilde{H} - \tilde{H} \tilde{A} (\tilde{A}^T \tilde{H} \tilde{A})^{-1} \tilde{A}^T \tilde{H}) \tilde{a}_k$ (with $\tilde{a}_k = [a_k, -e_k]^T$) and $\overline{I} = \{i \in I : q_{ki} + \gamma_k p_i > 0\}$.

If $\beta_k \gamma_k + \delta_k = 0$, then we set $\alpha_P = \infty$. If $\overline{I} = \emptyset$, then we set $\alpha_D = \infty$. If simultaneously $\alpha_P = \infty$ and $\alpha_D = \infty$, then the problem has no feasible solution. Otherwise we set $\alpha = \min\{\alpha_P, \alpha_D\}$ and compute $z := z + \alpha\gamma_k$, $u := u - \alpha(q_k + \gamma_k p)$, $u_k := u_k + \alpha$, $v_k := (1 - \alpha/\alpha_P)v_k$.

If $\alpha_P \le \alpha_D$, then the primal step is realized, i.e. we set $I := I \cup \{k\}$, $u := [u^T, u_k]^T$, $e := [e^T, e_k]^T$, $A := [A, a_k]$, $\tilde{A} := [\tilde{A}, \tilde{a}_k]$, recompute $d = -HAu$ and determine a new value of the most violated primal constraint and a new index k.

If $\alpha_P > \alpha_D$, then the dual step is realized, i.e. we set $I := I \backslash \{j\}$, $u := u^{(j)}$, $e := e^{(j)}$, $A := A^{(j)}$, $\tilde{A} := \tilde{A}^{(j)}$, where the upper index in parentheses denotes an element or column which are deleted. Now, two cases can occur. If $I \cap M_1 \neq \emptyset$, then we recompute the primal and dual steplengths and repeat the process with the same index k. If $I \cap M_1 = \emptyset$, then we compute $z := z - v_k$, set $I := I \cup \{k\}$, $u := [u^T, u_k]^T$, $e := [e^T, e_k]^T$, $A := [A, a_k]$, $\tilde{A} := [\tilde{A}, \tilde{a}_k]$, recompute $d = -HAu$ and determine the new value of the most violated primal constraint and the new index k.

In [5], it was proved that the above dual range space finds the solutions of quadratic programming problems (4) and (5) after a finite number of steps.

3. Description of the subroutines

In this section we describe all subroutines which can be called from the user's program. In the description of formal parameters we introduce a type of the argument that specifies whether the argument must have a value defined on entry to the subroutine (I), whether it is a value which will be returned (O), or both (U), or whether it is an auxiliary value (A). Note that the arguments of the type I can be changed on output under some circumstances, especially if improper input values were given. Besides formal parameters, we can use a COMMON /STAT/ block containing statistical information. This block, used in each subroutine has the following form:

```
COMMON /STAT/ NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
```

The arguments have the following meaning.

Argument	Type	Significance
NDECF	O	Positive INTEGER variable that indicates the number of matrix decompositions.
NRES	O	Positive INTEGER variable that indicates the number of restarts.
NRED	O	Positive INTEGER variable that indicates the number of reductions.
NREM	O	Positive INTEGER variable that indicates the number of constraint deletions during the QP solutions.
NADD	O	Positive INTEGER variable that indicates the number of constraint additions during the QP solutions.

NIT	O	Positive INTEGER variable that indicates the number of iterations.
NFV	O	Positive INTEGER variable that indicates the number of function evaluations.
NFG	O	Positive INTEGER variable that specifies the number of gradient evaluations.
NFH	O	Positive INTEGER variable that specifies the number of Hessian evaluations.

3.1 Subroutines PMINU, PMINS, PMINL

The calling sequences are

```
      CALL PMINU(NF,NA,X,AF,IA,RA,IPAR,RPAR,F,GMAX,IEXT,ITERM)

      CALL PMINS(NF,NA,NB,X,IX,XL,XU,AF,IA,RA,IPAR,RPAR,F,GMAX,
     & IEXT,ITERM)

      CALL PMINL(NF,NA,NB,NC,X,IX,XL,XU,CF,IC,CL,CU,CG,AF,IA,RA,
     & IPAR,RPAR,F,GMAX,IEXT,ITERM)
```

The arguments have the following meaning.

Argument	Type	Significance
NF	I	Positive INTEGER variable that specifies the number of variables of the objective function.
NA	I	INTEGER variable that specifies the number of functions in the minimax criterion.
NB	I	INTEGER variable that specifies whether the simple bounds are suppressed (NB $= 0$) or accepted (NB $=$ NF).
NC	I	INTEGER variable that specifies the number of linear constraints; if NC $= 0$ the linear constraints are suppressed.
X(NF)	U	On input, DOUBLE PRECISION vector with the initial estimate to the solution. On output, the approximation to the minimum.
IX(NF)	I	On input (significant only if NB > 0) INTEGER vector containing the simple bounds types: IX(I) $= 0$: the variable X(I) is unbounded, IX(I) $= 1$: the lower bound X(I) $\geq$ XL(I), IX(I) $= 2$: the upper bound X(I) $\leq$ XU(I), IX(I) $= 3$: the two side bound XL(I) $\leq$ X(I) $\leq$ XU(I), IX(I) $= 5$: the variable X(I) is fixed (given by its initial estimate).

XL(NF)	I	DOUBLE PRECISION vector with lower bounds for variables (significant only if NB > 0).
XU(NF)	I	DOUBLE PRECISION vector with upper bounds for variables (significant only if NB > 0).
CF(NC)	A	DOUBLE PRECISION vector which contains values of constraint functions (only if NC > 0).
IC(NC)	I	On input (significant only if NC > 0) INTEGER vector which contains constraint types: IC(K) $= 0$: the constraint CF(K) is not used, IC(K) $= 1$: the lower constraint CF(K) $\geq$ CL(K), IC(K) $= 2$: the upper constraint CF(K) $\leq$ CU(K), IC(K) $= 3$: the two side constraint CL(K) $\leq$ CF(K) $\leq$ CU(K), IC(K) $= 5$: the equality constraint CF(K) $=$ CL(K).
CL(NC)	I	DOUBLE PRECISION vector with lower bounds for constraint functions (significant only if NC > 0).
CU(NC)	I	DOUBLE PRECISION vector with upper bounds for constraint functions (significant only if NC > 0).
CG(NF*NC)	I	DOUBLE PRECISION matrix whose columns are normals of the linear constraints (significant only if NC > 0).
AF(NA)	O	DOUBLE PRECISION vector which contains values of functions in the minimax criterion.
IA(NIA)	A	INTEGER working array of the dimension of at least NIA=NF+NA+1.
RA(NRA)	A	DOUBLE PRECISION working array of the dimension of at least NRA=(NF+NA+8)*NF+2*NA+4.
IPAR(5)	A	INTEGER parameters. IPAR(1)=MET, IPAR(2)=MES, IPAR(3)= MIT, IPAR(4)=MFV, IPAR(5)=IPRNT. These parameters (MET, MES, MIT, MFV, IPRNT) are described in Section 3.2.
RPAR(7)	A	DOUBLE PRECISION parameters. RPAR(1)=TOLX, RPAR(2)= TOLF, RPAR(3)=TOLB, RPAR(4)=TOLG, RPAR(5)=TOLD, RPAR(6)= TOLS, RPAR(7)=XMAX. These parameters (TOLX, TOLF, TOLB, TOLG, TOLD, TOLS, XMAX) are described in Section 3.2).
F	O	DOUBLE PRECISION value of the objective function at the solution X.
GMAX	O	DOUBLE PRECISION maximum absolute value of a partial derivative of the Lagrangian function.
IEXT	I	INTEGER variable that specifies the minimax criterion:

		IEXT ≤ 0:	maximum of positive values,
		IEXT $= 0$:	maximum of absolute values,
		IEXT ≥ 0:	maximum of negative values,
ITERM	O	INTEGER variable that indicates the cause of termination:	
		ITERM $= 1$:	if $\vert x - x_{old}\vert$ was less than or equal to TOLX in MTESX subsequent iterations,
		ITERM $= 2$:	if $\vert F - F_{old}\vert$ was less than or equal to TOLF in MTESF subsequent iterations,
		ITERM $= 3$:	if F is less than or equal to TOLB,
		ITERM $= 4$:	if GMAX is less than or equal to TOLG,
		ITERM $= 11$:	if NFV exceeded MFV,
		ITERM $= 12$:	if NIT exceeded MIT,
		ITERM < 0:	if the method failed.

The subroutines PMINU, PMINS, PMINL require the user supplied subroutines FUN and DER that define the values and the gradients of the functions in the minimax criterion and have the form

```
SUBROUTINE FUN(NF,KA,X,FA)

SUBROUTINE DER(NF,KA,X,GA)
```

The arguments of user supplied subroutines have the following meaning.

Argument	Type	Significance
NF	I	Positive INTEGER variable that specifies the number of variables of the objective function.
KA	I	Positive INTEGER variable that specifies the index of a function in the minimax criterion.
X(NF)	I	DOUBLE PRECISION an estimate to the solution.
FA	O	DOUBLE PRECISION value of a function with the index KA at the point X.
GA(NF)	O	DOUBLE PRECISION gradient of a function with the index KA at the point X.

3.2 Subroutine PMIN

This general subroutine is called from all the subroutines described in Section 3.1. The calling sequence is

```
 CALL PMIN(NF,NA,NB,NC,X,IX,XL,XU,CF,IC,CL,CU,CG,AF,IA,AFO,AFD,
& GA,AG,IAA,AR,AZ,G,H,S,XO,GO,TOLX,TOLF,TOLB,TOLG,TOLD,TOLS,
& XMAX,GMAX,F,IEXT,MET,MES,MIT,MFV,IPRNT,ITERM).
```

The arguments NF, NA, NB, NC, X, IX, XL, XU, CF, IC, CL, CU, CG, AF, GMAX, F, IEXT, ITERM, have the same meaning as in Section 3.1. Other arguments have the following meaning.

Argument	Type	Significance
IA(NA)	A	INTEGER vector containing types of functions in the minimax criterion.
AFO(NA)	A	DOUBLE PRECISION vector of saved values of functions in the minimax criterion.
AFD(NA)	A	DOUBLE PRECISION vector of increments of functions in the minimax criterion.
GA(NF)	A	DOUBLE PRECISION gradient of the selected function in the minimax criterion.
AG(NF*NA)	A	DOUBLE PRECISION matrix whose columns are gradients of functions in the minimax criterion.
IAA(NA)	A	INTEGER vector containing indices of active functions.
AR((NF+1) *(NF+2)/2)	A	DOUBLE PRECISION matrix containing triangular decomposition of the orthogonal projection kernel.
AZ(NF+1)	A	DOUBLE PRECISION vector of Lagrange multipliers.
G(NF)	A	DOUBLE PRECISION gradient of the Lagrangian function.
H((NF+1) *NF/2)	A	DOUBLE PRECISION Hessian matrix of the Lagrangian function.
S(NF+1)	A	DOUBLE PRECISION direction vector.
XO(NF)	A	DOUBLE PRECISION vector which contains increments of variables.
GO(NF+1)	A	DOUBLE PRECISION vector which contains increments of partial derivatives.
TOLX	I	DOUBLE PRECISION tolerance for the change of the coordinate vector X; the choice TOLX $= 0$ causes that the default value 10^{-16} will be taken.
TOLF	I	DOUBLE PRECISION tolerance for the change of function values; the choice TOLF $= 0$ causes that the default value 10^{-8} will be taken.
TOLB	I	DOUBLE PRECISION minimum acceptable function value; the choice TOLB $= 0$ causes that the default value -10^{60} will be taken.

TOLG	I	DOUBLE PRECISION tolerance for the Lagrangian function gradient; the choice TOLG $= 0$ causes that the default value 10^{-6} will be taken.
TOLD	I	DOUBLE PRECISION tolerance for a descent direction; the choice TOLD $= 0$ causes that the default value 10^{-4} will be taken.
TOLS	I	DOUBLE PRECISION tolerance parameter for a function decrease in a line search; the choice TOLS $= 0$ causes that the default value 10^{-2} will be taken.
XMAX	I	DOUBLE PRECISION maximum stepsize; the choice XMAX $= 0$ causes that the default value 10^3 will be taken.
MET	I	INTEGER variable that specifies self scaling for variable metric updates: MET $= 1$: self scaling is suppressed, MET $= 2$: self scaling is used only in the first iteration (initial self scaling), The choice MET $= 0$ causes that the default value MET $= 2$ will be taken.
MES	I	INTEGER variable that specifies the interpolation method selection in a line search: MES $= 1$: bisection, MES $= 2$: two point quadratic interpolation. MES $= 3$: three point quadratic interpolation. MES $= 4$: three point cubic interpolation. The choice MES $= 0$ causes that the default value MES $= 1$ will be taken.
MIT	I	INTEGER variable that specifies the maximum number of iterations; the choice MIT $= 0$ causes that the default value 200 will be taken.
MFV	I	INTEGER variable that specifies the maximum number of function evaluations; the choice MFV $= 0$ causes that the default value 500 will be taken.
IPRNT	I	INTEGER variable that specifies PRINT: IPRNT $= 0$: print is suppressed, IPRNT $= 1$: basic print of final results, IPRNT $= 1$: extended print of final results, IPRNT $= 1$: basic print of intermediate and final results, IPRNT $= 1$: extended print of intermediate and final results,

The subroutine PMIN has a modular structure. The following list contains its most important subroutines.

PA1MX2	Minimax criterion evaluation.
PDDXQ1	Determination of the descent direction using quadratic programming subroutine.
PLQDF1	Dual range space method for solving the quadratic programming problem with linear constraints (see [5]).
PS0LA2	Line search using only function values.
PUDBG1	The BFGS variable metric update applied to the Choleski decomposition of the approximate Hessian matrix.

The subroutine PMIN requires the user supplied subroutines FUN and DER. User supplied subroutines FUN and DER are described in Section 3.1.

3.3 Subroutine PLQDF1

Since the dual range space method for special quadratic programming subproblems arising in nonlinear minimax optimization can be used separately in many applications (e.g. in bundle-type methods for nonsmooth optimization), we describe the subroutine PLQDF1 in more details. The calling sequence is

```
      CALL PLQDF1(NF,NA,NC,X,IX,XL,XU,AF,AFD,IA,IAA,AG,AR,AZ,
     &  CF,IC,CL,CU,CG,G,H,S,MFP,KBF,KBC,IDECF,ETA0,ETA2,ETA9,
     & EPS7,EPS9,XNORM,UMAX,GMAX,N,ITERQ)
```

The arguments NF, NA, NC, X, IX, XL, XU, AF, CF, IC, CL, CU, CG have the same meaning as in Section 3.1 (only with the difference that the arguments X and AF are of the type (I), i.e. they must have a value defined on entry to PLQDF1 and they are not changed). The arguments AFD, IA, IAA, AG, AR, AZ have the same meaning as in Section 3.2 (only with the difference that the arguments AFD, IAA, AR, AZ are of the type (O), i.e. their values can be used subsequently). Other arguments have the following meaning.

Argument	Type	Significance
G(NF+1)	O	DOUBLE PRECISION gradient of the Lagrangian function.
H((NF+1)*NF/2)	U	DOUBLE PRECISION Choleski decomposition of the approximate Hessian matrix.
S(NF+1)	O	DOUBLE PRECISION direction vector.
MFP	I	INTEGER variable that specifies the type of the computed feasible point. MFP = 1: computation is terminated whenever an arbitrary feasible point is found, MFP = 2: computation is terminated whenever an optimum feasible point is found,

		`MFP` = 3: computation starts from the previously reached point and is terminated whenever an optimum feasible point is found.
`KBF`	I	INTEGER variable that specifies simple bounds on variables. `KBF` = 0: simple bounds are suppressed, `KBF` = 2: one sided simple bounds, `KBF` = 2: two sided simple bounds.
`KBC`	I	INTEGER variable that specifies general linear constraints. `KBC` = 0: linear constraints are suppressed, `KBC` = 1: one sided linear constraints, `KBC` = 2: two sided linear constraints.
`IDECF`	U	INTEGER variable that specifies the type of matrix decomposition. `IDECF` = 0: no decomposition, `IDECF` = 1: Choleski decomposition, `IDECF` = 9: inversion, `IDECF` = 10: diagonal matrix.
`ETA0`	I	DOUBLE PRECISION machine precision (the recommended value is 10^{-15}).
`ETA2`	I	DOUBLE PRECISION tolerance for positive definiteness in the Choleski decomposition.
`ETA9`	I	DOUBLE PRECISION maximum floating point number.
`EPS7`	I	DOUBLE PRECISION tolerance for linear independence of constraints (the recommended value is 10^{-10}).
`EPS9`	I	DOUBLE PRECISION tolerance for the definition of active constraints (the recommended value is 10^{-8}).
`XNORM`	O	DOUBLE PRECISION value of the linearized minimax function.
`UMAX`	O	DOUBLE PRECISION maximum absolute value of the negative Lagrange multiplier.
`GMAX`	O	DOUBLE PRECISION infinity norm of the gradient of the Lagrangian function.
`N`	O	INTEGER dimension of a manifold defined by active constraints.
`ITERQ`	O	INTEGER variable that indicates the type of the computed feasible point. `ITERQ` = 1: an arbitrary feasible point was found, `ITERQ` = 2: the optimum feasible point was found,

ITERQ $= -1$: an arbitrary feasible point does not exist,
ITERQ $= -2$: the optimum feasible point does not exist.

3.4 Form of printed results

The form of printed results is specified by the parameter IPRNT as is described above. Here we demonstrate individual forms of printed results by the simple use of the program TMINL described in the next section (with NEXT=6). If we set IPRNT=1, then the printed results will have the form

```
NIT=  16  NFV=  18  NFG=  17  F=  .50694800D+00  G= .2872D-06  ITERM=  4
```

If we set IPRNT=-1, then the printed results will have the form

```
EXIT FROM PMIN :
NIT=  16  NFV=  18  NFG=  17  F=  .50694800D+00  G= .2872D-06  ITERM=  4
X=  .5000000D+00  .5000000D+00  .5000000D+00  .5000000D+00  .5000000D+00
    .5000000D+00  .5000000D+00  .5000000D+00  .5000000D+00  .5000000D+00
   -.4166693D+00 -.4166693D+00 -.4166693D+00 -.4166693D+00 -.4166693D+00
   -.4166693D+00 -.4166693D+00 -.4166693D+00 -.4166693D+00 -.5069240D+00
```

If we set IPRNT=2, then the printed results will have the form

```
ENTRY TO PMIN :
NIT=   0  NFV=   1  NFG=   1  F=  .21899000D+05  G= .1000D+61
NIT=   1  NFV=   2  NFG=   2  F=  .13670000D+05  G= .2200D+02
NIT=   2  NFV=   3  NFG=   3  F=  .35097538D+04  G= .1050D+02
NIT=   3  NFV=   4  NFG=   4  F=  .90439182D+03  G= .2478D+01
NIT=   4  NFV=   5  NFG=   5  F=  .22346618D+03  G= .5013D+00
NIT=   5  NFV=   6  NFG=   6  F=  .39130423D+02  G= .1630D+00
NIT=   6  NFV=   7  NFG=   7  F=  .11062302D+02  G= .1489D+01
NIT=   7  NFV=   8  NFG=   8  F=  .13659872D+01  G= .6371D-01
NIT=   8  NFV=   9  NFG=   9  F=  .79850964D+00  G= .1994D+00
NIT=   9  NFV=  10  NFG=  10  F=  .74156516D+00  G= .1780D+00
NIT=  10  NFV=  11  NFG=  11  F=  .72019631D+00  G= .8374D-01
NIT=  11  NFV=  12  NFG=  12  F=  .69644369D+00  G= .1351D+00
NIT=  12  NFV=  13  NFG=  13  F=  .65212038D+00  G= .1845D+00
NIT=  13  NFV=  15  NFG=  14  F=  .52670051D+00  G= .4901D-01
NIT=  14  NFV=  16  NFG=  15  F=  .51183315D+00  G= .2369D-01
NIT=  15  NFV=  17  NFG=  16  F=  .50695236D+00  G= .3043D-03
NIT=  16  NFV=  18  NFG=  17  F=  .50694800D+00  G= .2872D-06
EXIT FROM PMIN :
NIT=  16  NFV=  18  NFG=  17  F=  .50694800D+00  G= .2872D-06  ITERM=  4
```

If we set IPRNT=-2, then the printed results will have the form

```
ENTRY TO PMIN :
NIT=   0  NFV=   1  NFG=   1  F=  .21899000D+05  G= .1000D+61
NIT=   1  NFV=   2  NFG=   2  F=  .13670000D+05  G= .2200D+02
NIT=   2  NFV=   3  NFG=   3  F=  .35097538D+04  G= .1050D+02
NIT=   3  NFV=   4  NFG=   4  F=  .90439182D+03  G= .2478D+01
NIT=   4  NFV=   5  NFG=   5  F=  .22346618D+03  G= .5013D+00
NIT=   5  NFV=   6  NFG=   6  F=  .39130423D+02  G= .1630D+00
NIT=   6  NFV=   7  NFG=   7  F=  .11062302D+02  G= .1489D+01
NIT=   7  NFV=   8  NFG=   8  F=  .13659872D+01  G= .6371D-01
NIT=   8  NFV=   9  NFG=   9  F=  .79850964D+00  G= .1994D+00
NIT=   9  NFV=  10  NFG=  10  F=  .74156516D+00  G= .1780D+00
NIT=  10  NFV=  11  NFG=  11  F=  .72019631D+00  G= .8374D-01
NIT=  11  NFV=  12  NFG=  12  F=  .69644369D+00  G= .1351D+00
NIT=  12  NFV=  13  NFG=  13  F=  .65212038D+00  G= .1845D+00
NIT=  13  NFV=  15  NFG=  14  F=  .52670051D+00  G= .4901D-01
NIT=  14  NFV=  16  NFG=  15  F=  .51183315D+00  G= .2369D-01
NIT=  15  NFV=  17  NFG=  16  F=  .50695236D+00  G= .3043D-03
NIT=  16  NFV=  18  NFG=  17  F=  .50694800D+00  G= .2872D-06
EXIT FROM PMIN :
NIT=  16  NFV=  18  NFG=  17  F=  .50694800D+00  G= .2872D-06  ITERM=  4
X=  .5000000D+00  .5000000D+00  .5000000D+00  .5000000D+00  .5000000D+00
    .5000000D+00  .5000000D+00  .5000000D+00  .5000000D+00  .5000000D+00
   -.4166693D+00 -.4166693D+00 -.4166693D+00 -.4166693D+00 -.4166693D+00
   -.4166693D+00 -.4166693D+00 -.4166693D+00 -.4166693D+00 -.5069240D+00
```

4. Verification of the subroutines

In this section we introduce the main programs TMINU and TMINL, which serve as demonstration, verification and testing of the subroutines PMINU and PMINL.

4.1 Program TMINU

The following main program demonstrates the usage of the subroutine PMINU.

```
C
C     TEST PROGRAM FOR THE SUBROUTINE PMINU
C
      INTEGER NF,NA,IA(200),IEXT,IPAR(5),ITERM
      REAL*8 X(40),AF(200),RA(4000),RPAR(7),F,GMAX
      REAL*8 FMIN
      INTEGER NAL,NEXT,IERR,I
      COMMON /PROB/ NEXT
      INTEGER NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
      COMMON /STAT/ NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
```

```
C
C     LOOP FOR 7 TEST PROBLEMS
C
      DO 3 NEXT=1,7
C
C     CHOICE OF INTEGER AND REAL PARAMETERS
C
      DO 1 I=1,5
      IPAR(I)=0
    1 CONTINUE
      DO 2 I=1,7
      RPAR(I)=0.0D 0
    2 CONTINUE
      IPAR(5)=1
C
C     PROBLEM DIMENSION
C
      NF=20
      NA=30
C
C     INITIATION OF X AND CHOICE OF RPAR(7)
C
      CALL TIUD06(NF,NA,NAL,X,FMIN,RPAR(7),NEXT,IEXT,IERR)
      IF (IERR.NE.0) GO TO 3
C
C     SOLUTION
C
      CALL PMINU(NF,NA,X,AF,IA,RA,IPAR,RPAR,F,GMAX,IEXT,ITERM)
    3 CONTINUE
      STOP
      END
C
C     USER SUPPLIED SUBROUTINE (CALCULATION OF FA)
C
      SUBROUTINE FUN(NF,KA,X,FA)
      INTEGER NF,KA
      REAL*8 X(*),FA
      INTEGER NEXT
      COMMON /PROB/ NEXT
C
C     FUNCTION EVALUATION
C
      CALL TAFU06(NF,KA,X,FA,NEXT)
```

```
      RETURN
      END
C
C     USER SUPPLIED SUBROUTINE (CALCULATION OF GA)
C
      SUBROUTINE DER(NF,KA,X,GA)
      INTEGER NF,KA
      REAL*8 X(*),GA(*)
      INTEGER NEXT
      COMMON /PROB/ NEXT
C
C     GRADIENT EVALUATION
C
      CALL TAGU06(NF,KA,X,GA,NEXT)
      RETURN
      END
```

This main program uses subroutines TIUD06 (initiation), TAFU06 (function evaluation) and TAGU06 (subgradient evaluation) containing 7 standard test problems with at most 20 variables which were taken from the UFO system [7]. The results obtained by this main program have the following form.

```
NIT=  8  NFV=  8  NFG=  8  F=  .19522245D+01  G= .2841D-08  ITERM= 4
NIT= 12  NFV= 17  NFG= 13  F= -.44000000D+02  G= .1200D-07  ITERM= 4
NIT= 10  NFV= 12  NFG= 11  F=  .12237125D-03  G= .1883D-08  ITERM= 4
NIT= 14  NFV= 16  NFG= 15  F=  .19729063D+00  G= .3416D-07  ITERM= 4
NIT= 14  NFV= 21  NFG= 15  F=  .68063006D+03  G= .1753D-06  ITERM= 4
NIT= 18  NFV= 27  NFG= 19  F=  .24306209D+02  G= .7582D-07  ITERM= 4
NIT= 22  NFV= 30  NFG= 23  F=  .13372828D+03  G= .2567D-06  ITERM= 4
```

The rows corresponding to individual test problems contain the number of iterations NIT, the number of function evaluations NFV, the number of gradient evaluations NFG, the final value of the objective function F, the value of the criterion for the termination G and the cause of termination ITERM.

4.2 Program TMINL

The following main program demonstrates the usage of the subroutine PMINL.

```
C
C     TEST PROGRAM FOR THE SUBROUTINE PMINL
C
      INTEGER NF,NA,NB,NC,IX(40),IC(10),IA(200),IEXT,IPAR(5),ITERM
      REAL*8 X(40),XL(40),XU(40),CF(10),CL(10),CU(10),CG(200),AF(200),
     & RA(4000),RPAR(7),F,GMAX
```

```
      REAL*8 FMIN
      INTEGER NAL,NCL,NEXT,IERR,I
      COMMON /PROB/ NEXT
      INTEGER NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
      COMMON /STAT/ NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
C
C     LOOP FOR 6 TEST PROBLEMS
C
      DO 3 NEXT=1,6
C
C     CHOICE OF INTEGER AND REAL PARAMETERS
C
      DO 1 I=1,5
      IPAR(I)=0
    1 CONTINUE
      DO 2 I=1,7
      RPAR(I)=0.0D 0
    2 CONTINUE
      IPAR(5)=1
C
C     PROBLEM DIMENSION
C
      NF=20
      NA=165
      NB=20
      NC=10
C
C     INITIATION OF X AND CHOICE OF RPAR(7)
C
      CALL TIUD22(NF,NA,NAL,NC,NCL,X,IX,XL,XU,IC,CL,CU,CG,FMIN,RPAR(7),
     & NEXT,IEXT,IERR)
      IF (IERR.NE.0) GO TO 3
C
C     SOLUTION
C
      CALL PMINL(NF,NA,NB,NC,X,IX,XL,XU,CF,IC,CL,CU,CG,AF,IA,RA,IPAR,
     & RPAR,F,GMAX,IEXT,ITERM)
    3 CONTINUE
      STOP
      END
C
C     USER SUPPLIED SUBROUTINE (CALCULATION OF FA)
C
```

```
      SUBROUTINE FUN(NF,KA,X,FA)
      INTEGER NF,KA
      REAL*8 X(*),FA
      INTEGER NEXT
      COMMON /PROB/ NEXT
C
C     FUNCTION EVALUATION
C
      CALL TAFU22(NF,KA,X,FA,NEXT)
      RETURN
      END
C
C     USER SUPPLIED SUBROUTINE (CALCULATION OF GA)
C
      SUBROUTINE DER(NF,KA,X,GA)
      INTEGER NF,KA
      REAL*8 X(*),GA(*)
      INTEGER NEXT
      COMMON /PROB/ NEXT
C
C     GRADIENT EVALUATION
C
      CALL TAGU22(NF,KA,X,GA,NEXT)
      RETURN
      END
```

This main program uses subroutines TIUD22 (initiation), TAFU22 (function evaluation), TAGU22 (subgradient evaluation) containing 6 standard test problems with at most 20 variables which were taken from the UFO system [7]. The results obtained by this main program have the following form.

```
NIT=   6  NFV=   7  NFG=   7  F= -.38965952D+00  G= .6129D-08  ITERM=  4
NIT=   5  NFV=   5  NFG=   5  F= -.33035714D+00  G= .0000D+00  ITERM=  4
NIT=   8  NFV=   8  NFG=   8  F= -.44891078D+00  G= .2632D-09  ITERM=  4
NIT=  75  NFV=  75  NFG=  75  F= -.42928061D+00  G= .4375D-10  ITERM=  4
NIT=  12  NFV=  13  NFG=  12  F=  .10183089D+00  G= .1106D-09  ITERM=  4
NIT=  16  NFV=  18  NFG=  17  F=  .50694800D+00  G= .2872D-06  ITERM=  4
```

The rows corresponding to individual test problems contain the number of iterations NIT, the number of function evaluations NFV, the number of gradient evaluations NFG, the final value of the objective function F, the value of the criterion for the termination G and the cause of termination ITERM.

References

[1] Broyden C.G.: The Convergence of a Class of Double Rank Minimization Algorithms. Part 1 - General Considerations. Part 2 - the New Algorithm. J. Inst. Math. Appl. 6 (1970) 76-90, 222-231.

[2] Fletcher R.: A New Approach to Variable Metric Algorithms. Computer J. 13 (1970) 317-322.

[3] Goldfarb D.: A Family of Variable Metric Algorithms Derived by Variational Means. Math Comput. 24 (1970) 23-26.

[4] Han S.P.: Variable Metric Methods for Minimizing a Class of Nondifferentiable Functions. Math. Programming 20 (1981) 1, 1-13 Derived by Variational Means. Math Comput. 24 (1970) 23-26.

[5] Lukšan L. 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.

[6] Lukšan L. An Implementation of Recursive Quadratic Programming Variable Metric Methods for Linearly Constrained Nonlinear Minimax Approximation, *Kybernetika 21* (1985), 1, 22-40.

[7] Lukšan L., Šiška M., Tůma M., Vlček J., Ramešová N. Interactive System for Universal Functional Optimization (UFO), Version 1996. Research Report No. V-701, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague, Czech Republic, 1996.

[8] Shanno D.F.: Conditioning of Quasi-Newton Methods for Function Minimization. Math. Comput. 24 (1970) 647-656.