INTERNAL REPORTS IN

SIMULATION, OPTIMIZATION AND CONTROL

No. SOC-292

MMLC - A FORTRAN PACKAGE FOR LINEARLY CONSTRAINED MINIMAX OPTIMIZATION

J.W. Bandler and W.M. Zuberek

May 1982

FACULTY OF ENGINEERING
McMASTER UNIVERSITY
HAMILTON, ONTARIO, CANADA

MMLC - A FORTRAN PACKAGE FOR LINEARLY CONSTRAINED MINIMAX OPTIMIZATION

J.W. Bandler and W.M. Zuberek

Abstract

MMLC is a package of subroutines for solving linearly constrained minimax optimization problems. It is an extension and modification of the MMLA1Q package due to Hald. First derivatives of all functions with respect to all variables are assumed to be known. The solution is found by an iteration that uses either linear programming applied in connection with first-order derivatives or a quasi-Newton method applied in connection with first-order and approximate second-order derivatives. The method has been described by Hald and Madsen. The package and documentation are developed for the CDC 170/730 system with the NOS 1.4 operating system and the Fortran 4.8508 compiler.

This work was supported by the Natural Sciences and Engineering Research Council of Canada under Grant G0647.

J.W. Bandler and W.M. Zuberek are with the Group on Simulation, Optimization and Control, and the Department of Electrical and Computer Engineering, McMaster University, Hamilton, Canada L8S 4L7.

W.M. Zuberek is on leave from the Institute of Computer Science, Technical University of Warsaw, Warsaw, Poland.

I. INTRODUCTION

The package for linearly constrained minimax optimization of a set of nonlinear functions [1] has recently been extended and modified to provide a uniform printed output of input parameters as well as intermediate and final results of optimization. Consequently, the calling sequences have been modified appropriately, however, the original call of the subroutine MMLA1Q has been preserved to ensure compatibility with the previous version of the package.

The whole package is written in Fortran IV for the CDC 170/730 system. At McMaster University it is available in the form of a library of binary relocatable subroutines which is linked with the user's program by the appropriate call of the main subroutine in the package. The name of the library is LIBRMML. The library is available as a group indirect file under the charge RJWBAND. The general sequence of NOS commands to use the package can be as follows:

```
/GET(LIBRMML/GR)       - fetch the library,
/LIBRARY(LIBRMML)      - indicate the library to the loader,
/FTN(...,GO)           - compile, load and execute the program.
```

The user's program should be composed (at least) of:

- the main segment which prepares parameters and calls the main subroutine of the package,
- the segment which calculates the values of residual functions and their first partial derivatives at points determined by the package; the name of this subroutine can be arbitrary because it is

transferred to the package as one of the parameters.

II. GENERAL DESCRIPTION

Given a set of nonlinear differentiable residual functions $f_i(\underset{\sim}{x})$, $i=1,2,\ldots,m$, of n variables $\underset{\sim}{x} = [x_1\ x_2\ \ldots\ x_n]^T$, it is the purpose of the package to find a local minimum of the minimax objective function

$$F(\underset{\sim}{x}) = \max_{1 \leq i \leq m} f_i(\underset{\sim}{x})$$

subject to linear constraints

$$\underset{\sim}{c}_i^T \underset{\sim}{x} + b_i = 0, \quad i = 1, \ldots, \ell_{eq},$$
$$\underset{\sim}{c}_i^T \underset{\sim}{x} + b_i \geq 0, \quad i = \ell_{eq}+1, \ldots, \ell,$$

where $\underset{\sim}{c}_i$ and b_i, $i = 1, \ldots, \ell$, are constants.

The objective function is in general a non-differentiable function and normally the minimum is situated at a point where two or more residual functions are equal and/or some of the constraints are active (a constraint is active if its value is equal to zero). If there is no smooth valley through the solution and the minimum is numerically well-defined then the minimum is characterized by only first derivatives of the residual functions and the constraints which determine it. For such cases it is possible to construct algorithms based on first derivative information only with fast final convergence. It has been proved [2],[3] that if the so-called Haar condition (which ensures that no smooth valley passes through the solution) is satisfied then quadratic final rate of convergence can be obtained. If there is, however, a smooth valley through the solution, the first-order derivatives may be insufficient and some second-order information may be needed to obtain a fast final convergence. For such cases the quasi-

Newton iteration has been proposed [3] in which the second-order derivatives are approximated by Powell's method.

The minimax algorithm is a two-stage one [3]. Initially, Stage 1 is used and at each point the nonlinear residual functions are approximated by linear functions using the first derivative information. However, if a smooth valley through the solution is detected, a switch to Stage 2 is made and the quasi-Newton iteration is used. If it turns out that the Stage 2 iteration is unsuccessful (for instance, if the set of active functions has been wrongly choosen) then a switch is made back to Stage 1. The algorithm may switch several times between Stage 1 and Stage 2 but normally only a few switches will take place and the iteration will terminate either in Stage 1 with quadratic rate of convergence or in Stage 2 with superlinear rate of convergence [3].

The algorithm is a feasible point algorithm which means that the residual functions are only evaluated at points satisfying the linear constraints. Initially a feasible point is determined by the package, and from that point feasibility is retained.

Stage 1

The Stage 1 algorithm is similar to that of [2]. At the kth iteration the change $\underset{\sim}{h}^k$ of the approximation $\underset{\sim}{x}^{k-1}$ is determined as the solution of the linear minimax problem

$$\underset{\underset{\sim}{h}^k}{\text{Minimize}}\ \tilde{F}(\underset{\sim}{x}^{k-1}, \underset{\sim}{h}^k) = \max_{1 \leq i \leq m} (f_i(\underset{\sim}{x}^{k-1}) + \underset{\sim}{f}_i'^T(\underset{\sim}{x}^{k-1})\ \underset{\sim}{h}^k)$$

subject to constraints

$$\underset{\sim}{c}_i^T(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) + b_i = 0, \quad i = 1, \ldots, \ell_{eq},$$

$$c_i^T(x^{k-1} + h^k) + b_i \geq 0, \quad i = \ell_{eq}+1, \ldots, \ell,$$

$$\|h^k\| \leq \delta_x^{k-1} ,$$

where δ_x^k is equal to $0.25\|h^{k-1}\|$, $\|h^{k-1}\|$, or $2\|h^{k-1}\|$ according to an unsuccessful, not unsuccessful or successful (k-1)th iteration. The jth iteration is unsuccessful if

$$F(x^{j-1}) - F(x^{j-1} + h^j) \leq 0.25\ (F(x^{j-1}) - \tilde{F}(x^{j-1}, h^j)),$$

it is successful if

$$F(x^{j-1}) - F(x^{j-1} + h^j) \geq 0.75\ (F(x^{j-1}) - \tilde{F}(x^{j-1}, h^j))$$

and is not unsuccessful otherwise. In each iteration of Stage 1, the step size is thus updated according to the goodness of the linear approximation. If the change of the objective function F slightly differs from the change predicted by linear approximation, the step size is increased; if it differs significantly, the step size is decreased. The initial step size δ_x^0 is defined by the user (argument DX).

In order to accept $x^{k-1} + h^k$ as the next point it is usually required that the value of the objective function F decreases, namely,

$$F(x^{k-1} + h^k) < F(x^{k-1}).$$

It is shown in [4], however, that this criterion is not always sufficient to guarantee convergence and, therefore, the stronger condition is used. If

$$F(x^{k-1}) - F(x^{k-1} + h^k) \geq 0.01\ (F(x^{k-1}) - \tilde{F}(x^{k-1}, h^k))$$

then $x^k = x^{k-1} + h^k$, otherwise $x^k = x^{k-1}$.

The algorithm terminates in Stage 1 when any one of the following conditions is satisfied:

(1) the number of residual function evaluations exceeds the limit defined by the user (argument MAXF),

(2) the consecutive change $\underset{\sim}{h}^k$ of the approximation $\underset{\sim}{x}^k$ of the solution is sufficiently small

$$\|\underset{\sim}{h}^k\| \leq \varepsilon \, \|\underset{\sim}{x}^k\|,$$

where ε is defined by the user (argument EPS),

(3) the consecutive change $\underset{\sim}{h}^k$ reaches the machine accuracy

$$\|\underset{\sim}{h}^k\| \leq \varepsilon_0 \, \|\underset{\sim}{x}^k\|,$$

where ε_0 is the smallest positive number such that

$$1 + \varepsilon_0 > 1,$$

(4) the consecutive change $\underset{\sim}{h}^k$ is insignificantly small, namely,

$$\|\underset{\sim}{h}^k\| \leq 10^{-50}$$

(when the solution $\underset{\sim}{x}^*$ is equal to $\underset{\sim}{0}$, the conditions (2) and (3) may be insufficient to terminate the iteration),

(5) the consecutive solution of the linear minimax problem does not decrease the value of the objective function

$$\tilde{F}\,(\underset{\sim}{x}^{k-1},\, \underset{\sim}{h}^k) \geq F\,(\underset{\sim}{x}^{k-1}).$$

Moreover, the user can terminate the iterative procedure and cause the return from the package by setting one of parameters during evaluation of residual functions (see argument FDF).

<u>Switch to Stage 2</u>

For each kth Stage 1 iteration the set $A^k = A_f^k + A_c^k$ of active residual functions A_f^k and active constraints A_c^k is determined. Initially this set contains all the equality constraints provided that the equality and inequality constraints are satisfied for the starting point (otherwise the starting point is adjusted appropriately by the package). Subsequently, the sets A^k, $k = 1,2,...,$ are updated in consecutive iterations, corresponding to consecutive approximations $\underset{\sim}{x}^k$

of the solution. A switch to Stage 2 is made after the kth Stage 1 iteration if the following conditions are satisfied simultaneously:

(1) the sets of active residual functions and constraints for the last t Stage 1 iterations are identical

$$A^{k-t+1} = A^{k-t+2} = \dots = A^k$$

(parameter t is defined by the user - argument KEQS - and normally t = 3 is an appropriate value),

(2) there have been at least n Stage 1 iterations (n is the number of optimization variables)

$$k \geq n,$$

(3) the approximation of the Hessian matrix is positive definite for the set A^k of active residual functions and constraints,

(4) the value of the objective function $F(\underset{\sim}{x}^k)$ decreases in consecutive switches to Stage 2 (for the first switch this condition is omitted)

$$F(\underset{\sim}{x}^k) \leq F(\underset{\sim}{x}^{k-s}) - \delta|F(\underset{\sim}{x}^{k-s})|$$

where $\underset{\sim}{x}^{k-s}$ is the point at which the previous switch to Stage 2 has been made, and δ is a small positive number ($\delta = 10^{-14}$ is used in the package).

Stage 2

At the kth Stage 2 iteration an approximate Newton method is applied to the following system of equations

$$\sum_{j \in A_f^k} \lambda_j^k f'_{ji} (\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) + \sum_{j \in A_c^k} \lambda_j^k (\underset{\sim}{c}_j^T(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) + b_j) = 0,$$

$$i = 1,\dots,n; \quad f'_{ji} = \partial f_j/\partial x_i,$$

$$\sum_{j \in A^k} \lambda_j^k = 1,$$

$$\underset{\sim}{c}_j^T(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) + b_j = 0, \qquad j \in A_c^k,$$

$$f_j(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) - f_{j_0}(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) = 0, \qquad j \in A_f^k,\ j_0 \in A_f^k,\ j \neq j_0,$$

where the unknowns are $[\underset{\sim}{h}^k, \underset{\sim}{\lambda}^k]$, and $A^k = A_f^k + A_c^k$ is the set of active residual functions A_f^k and active constraints A_c^k. The iteration is approximate because instead of $f_j''(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k)$ the approximated second-order derivatives are used.

If the solution of the given system of equations is non-singular, the residual $r(\underset{\sim}{x}, \underset{\sim}{\lambda}, A)$ is evaluated at the point $\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k$

$$\begin{aligned} r(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k, \underset{\sim}{\lambda}^k, A^k) = \| & \{\lambda_j^k f_{ji}'(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) \mid j \in A_f^k,\ i = 1,2,\ldots,n\}, \\ & \{\lambda_j^k(\underset{\sim}{c}_j^T(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) + b_j) \mid j \in A_c^k\}, \\ & \{\underset{\sim}{c}_j^T(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) + b_j \mid j \in A_c^k\}, \\ & \{f_j(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) - f_{j_0}(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k) \mid j \in A_f^k - \{j_0\}\}\| \end{aligned}$$

and if the residual decreases

$$r(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k, \underset{\sim}{\lambda}^k, A^k) \leq 0.999\, r(\underset{\sim}{x}^{k-1}, \underset{\sim}{\lambda}^{k-1}, A^{k-1})$$

then $(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k)$ is accepted as the next point, $\underset{\sim}{x}^k = \underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k$, otherwise $\underset{\sim}{x}^k = \underset{\sim}{x}^{k-1}$.

Moreover, in each Stage 2 iteration the approximation of the Hessian matrix is updated similarly as in Stage 1, and persistence of the set A^k of active residual functions and active constraints is checked.

The algorithm terminates in Stage 2 if any one of the following conditions is satisfied:

(1) the number of residual function evaluations exceeds the limit defined by the user (argument MAXF),

(2) the consecutive change $\underset{\sim}{h}^k$ of the approximation $\underset{\sim}{x}^k$ of the solution is sufficiently small

$$\|\underset{\sim}{h}^k\| \leq \varepsilon \|\underset{\sim}{x}^k\|,$$

where ε is defined by the user (argument EPS),

(3) the consecutive change $\underset{\sim}{h}^k$ reaches the machine accuracy

$$\|\underset{\sim}{h}^k\| \leq \varepsilon_0 \|\underset{\sim}{x}^k\|,$$

where ε_0 is the smallest positive number such that

$$1 + \varepsilon_0 > 1,$$

(4) the consecutive change $\underset{\sim}{h}^k$ is insignificantly small, namely,

$$\|\underset{\sim}{h}^k\| \leq 10^{-50}$$

(when the solution $\underset{\sim}{x}^*$ is equal to $\underset{\sim}{0}$, the conditions (2) and (3) may be insufficient to terminate the iteration).

Moreover, the user can terminate the iterative procedure by setting one of the parameters during the evaluation of residual functions (see the argument FDF).

Switch to Stage 1

At each kth Stage 2 iteration the following conditions are checked:

(1) whether the set of active residual functions and active constraints

is preserved

$$A^k = A^{k-1},$$

(2) whether residuals $r(\underset{\sim}{x}, \underset{\sim}{\lambda}, A)$ are decreasing

$$r(\underset{\sim}{x}^{k-1} + \underset{\sim}{h}^k, \underset{\sim}{\lambda}^k, A^k) \leq 0.999\, r(\underset{\sim}{x}^{k-1}, \underset{\sim}{\lambda}^{k-1}, A^{k-1}),$$

(3) whether the system of equations solved by the approximate Newton method has a non-singular solution.

The Stage 2 iteration is continued when all the conditions are satisfied, otherwise the algorithm returns to Stage 1.

III. STRUCTURE OF THE PACKAGE

There are 2 different entries to the package and 2 corresponding "main" (or interfacing) subroutines:

1. subroutine MMLC1A - standard entry which provides uniform printing of input parameters as well as intermediate and final results,
2. subroutine MMLA1Q - original entry, as defined by Hald [1]; this entry is preserved to ensure the compatibility with the previous version of the package.

Block diagrams of the package, corresponding to entries 1 and 2 are shown in Fig. 1 and 2, respectively. It can be observed that the PRINTOUT package of subroutines is used only when entry 1 (subroutine MMLC1A) is called, and that the subroutine MMX00Q (Fig. 1), which is responsible for printing the values of functions and their first derivatives, is replaced by dummy subroutine MMX00Z (Fig. 2) when entry 2 is used.

The common part of the package is composed of subroutines MMLC8A, MMLC9A, FEASI, MMLPA, S2LA1Q, BFGS, LINSYS and a set of subroutines

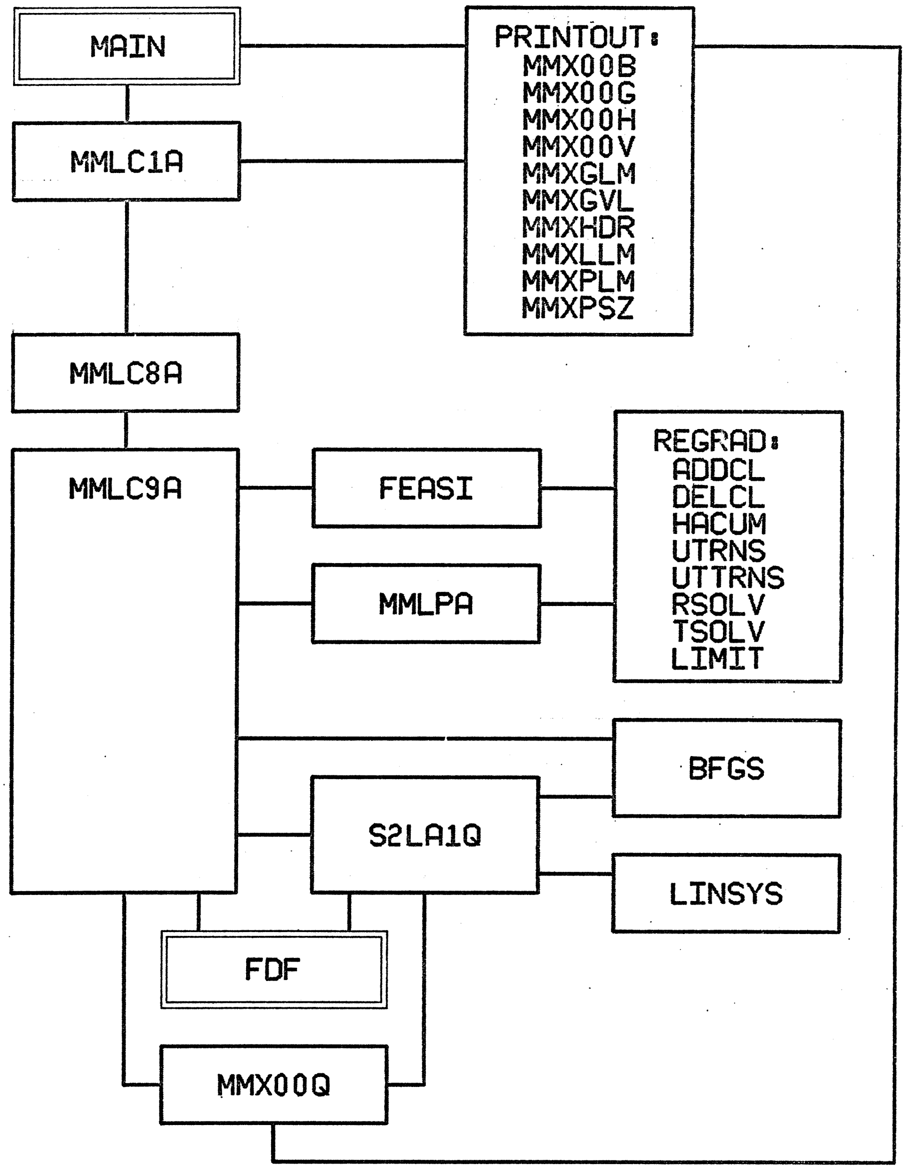

Fig. 1 Structure of the MMLC package corresponding to the standard entry (subroutine MMLC1A).

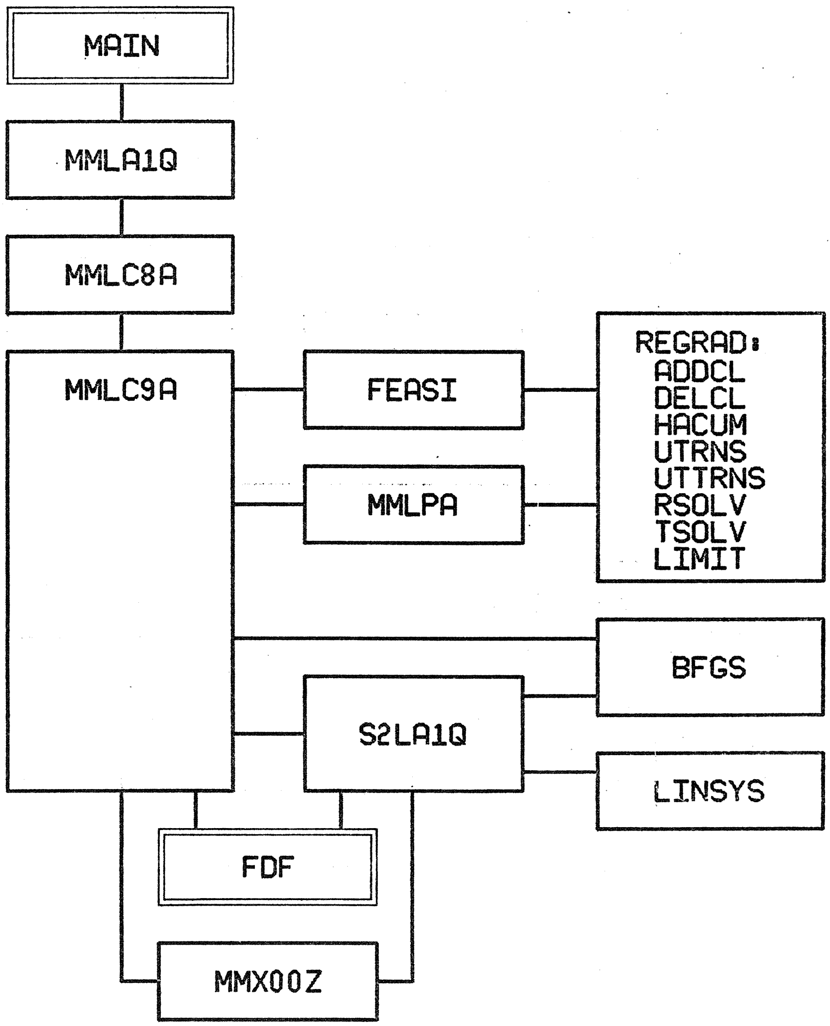

Fig. 2 Structure of the MMLC package corresponding to the original entry (subroutine MMLA1Q).

REGRAD. Checking of input parameters and subdivision of the working space (defined by the user) is performed in MMLC8A. The Stage 1 algorithm is implemented in MMLC9A, and the Stage 2 algorithm in S2LA1Q. FEASI determines a feasible starting point, and the linear subproblems of Stage 1 are solved by MMLPA. Both, MMLPA and FEASI, use the set of subroutines REGRAD for projected gradient calculations. The subroutine BFGS is an implementation of the BFGS formula for updating an approximate Hessian matrix containing second-order information. LINSYS uses Gaussian elimination for solving systems of linear equations.

The main segment MAIN and the subroutine FDF for evaluation of residual functions and their first-order derivatives must be supplied by the user.

When the standard entry (Fig. 1) is used, the subroutine MMLC1A and the set of subroutines PRINTOUT provide printed output containing principal input parameters of the minimax problem to be solved, and the solution obtained by the package. Moreover, the subroutine MMX00Q outputs the values of residual functions and their derivatives according to the argument IPR in the call of MMLC1A.

IV. LIST OF ARGUMENTS

Standard entry (subroutine MMLC1A)

The subroutine call is

```
CALL MMLC1A (FDF,N,M,L,LEQ,B,C,LC,X,DX,EPS,MAXF,KEQS,W,IW,ICH,IPR,IFALL)
```

The arguments are as follows.

FDF is the name of a subroutine supplied by the user. It must have the form

```
SUBROUTINE FDF(N,M,X,DF,F)
DIMENSION X(N),DF(M,N),F(M)
```

and it must calculate the values of the residual functions $f_i(\underset{\sim}{x})$ and their derivatives $\partial f_i(\underset{\sim}{x})/\partial x_j$ at the point $\underset{\sim}{x}$ corresponding to X(1),X(2),...,X(N), and store the values in the following way:

F(I) = $f_I(\underset{\sim}{x})$, I=1,...,M,

DF(I,J) = $\partial f_I(\underset{\sim}{x})/\partial x_J$, I=1,...,M, J=1,...,N.

Note: The name FDF can be arbitrary (user's choice) and must appear in the EXTERNAL statement in the segment calling MMLC1A.

The user can terminate the iterative procedure and force the return from the package by setting to zero (in the subroutine FDF) the variable MARK in the common area MML000

```
COMMON /MML000/ MARK
```

(on entry to the package MARK is set to 1).

N is an INTEGER argument which must be set to n, the number of optimization parameters. Its value must be positive and it is not changed by the package.

M is an INTEGER argument which must be set to m, the number of residual functions defining the minimax objective function. Its value must be positive and it is not changed by the package.

L is an INTEGER argument which must be set to ℓ, the total number of equality and inequality constraints. Its value must be positive or zero, and it is not changed by the package.

LEQ is an INTEGER argument which must be set to ℓ_{eq}, the number of equality constraints. Its value must be positive or zero and not greater than N, and not greater than L. Its value is not changed by the package.

B is a REAL array of length LC $\geq$ L. The elements of B must be set to the constant terms in the linear constraints, i.e. B(I) = b_I, I = 1,..., L. The contents of B is not changed by the package.

C is a REAL matrix of dimensions (LC,N). The first L rows of C must be set to the coefficients of $\underset{\sim}{x}$ in the linear constraints, i.e.

$$(C(I,1),\ C(I,2),\dots,C(I,N)) = \underset{\sim}{c}_I^T,\quad I = 1,\dots,L.$$

LC is an INTEGER argument which must be set to the length of the array B and to the number of rows of the matrix C. Its value must be not less than L, and it is not changed by the package.

X is a REAL array of the length at least N which, on entry, must be set to the initial approximation of the solution, X(I)=x_I^0, I=1,...,N. <u>On exit</u> X contains the best solution found by the package.

DX is a REAL variable which controls the step length of the iterative algorithm. <u>On entry</u> it must be set to such an initial value that in the region $\{x \mid \|x-x^0\| < DX\}$ the residual functions $f_i(x)$ can be approximated reasonably well by linear functions. If the residual functions are nearly linear, DX should be set to an approximate value of the distance between the initial approximation x^0 and the solution, but if more curvature is present this value may be too large. Normally $DX=0.1*\|x^0\|$ is an appropriate value, but an improper choice of DX is usually not critical, since the value of DX is adjusted by the package during the iteration. The value of DX must be positive. On exit DX contains the last value of the step size δ_x^k.

EPS is a REAL variable which on entry must be set to the required accuracy of the solution. The iteration terminates when $\|h^k\| \leq EPS*\|x^k\|$, where h^k is the correction to the kth approximation x^k of the solution. If EPS is chosen too small, the iteration terminates when no better estimation of the solution can be obtained because of rounding errors. On exit EPS contains the length of the last step taken in the iteration.

MAXF is an INTEGER variable which must be set to an upper bound on the number of calls of FDF (i.e., the maximum number of residual functions evaluations). On exit MAXF contains the number of calls of FDF that have been performed by the package.

KEQS is an INTEGER variable which must be set to the number of successive iterations with identical sets of active residual functions and active constraints that is required before a switch to Stage 2 is made. Normally, KEQS=3 is an appropriate value. If KEQS $\geq$ MAXF, the Stage 2 is never used. On exit KEQS contains the number of switches to Stage 2 that have taken place.

W is a REAL array which is used for working space. Its length is given by IW. <u>On exit</u> the first M elements of W contain the residual function values at the solution, i.e., $W(I)=f_I(\underset{\sim}{x})$, I=1,...,M.

IW is an INTEGER argument which must be set to the length of W. Its value must be at least

IWR = 2*M*N+5*N*N+4*M+8*N+4*LC+3.

The values of IWR-4*LC for a set of initial values of arguments M and N are given in Table 1.

ICH is an INTEGER argument which must be set to the unit number (or channel number) that is to be used for the printed output generated by the package. Usually it is the unit number of the file OUTPUT. If ICH is less than or equal to zero, no printed output will be generated by the package. The value of ICH is not changed by the package.

IPR is an INTEGER argument which controls the printed output generated by the package. It must be set by the user and is not changed by the package. The absolute value of IPR, as a decimal number, is "logically" composed of 4 fields

$$|IPR| = pqrs$$

where q, r and s are the least significant one-digit fields, and p is the remaining part of the number. If q is not equal to zero (i.e. q=1, ..., 9) then the first q evaluations of residual functions (i.e., the first q calls of FDF) are reported in the printed output. Further, if p is not equal to zero then every pth evaluation of residual functions is reported in the printed

TABLE I

MINIMUM WORKSPACE FOR THE MMLC PACKAGE FOR UNCONSTRAINED PROBLEMS

M: \ N:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	22	47	82	127	182	247	322	407	502	607	722	847	982	1127	1282	1447	1622	1807	2002	2207
2	28	55	92	139	196	263	340	427	524	631	748	875	1012	1159	1316	1483	1660	1847	2044	2251
3	34	63	102	151	210	279	358	447	546	655	774	903	1042	1191	1350	1519	1698	1887	2086	2295
4	40	71	112	163	224	295	376	467	568	679	800	931	1072	1223	1384	1555	1736	1927	2128	2339
5	46	79	122	175	238	311	394	487	590	703	826	959	1102	1255	1418	1591	1774	1967	2170	2383
6	52	87	132	187	252	327	412	507	612	727	852	987	1132	1287	1452	1627	1812	2007	2212	2427
7	58	95	142	199	266	343	430	527	634	751	878	1015	1162	1319	1486	1663	1850	2047	2254	2471
8	64	103	152	211	280	359	448	547	656	775	904	1043	1192	1351	1520	1699	1888	2087	2296	2515
9	70	111	162	223	294	375	466	567	678	799	930	1071	1222	1383	1554	1735	1926	2127	2338	2559
10	76	119	172	235	308	391	484	587	700	823	956	1099	1252	1415	1588	1771	1964	2167	2380	2603
11	82	127	182	247	322	407	502	607	722	847	982	1127	1282	1447	1622	1807	2002	2207	2422	2647
12	88	135	192	259	336	423	520	627	744	871	1008	1155	1312	1479	1656	1843	2040	2247	2464	2691
13	94	143	202	271	350	439	538	647	766	895	1034	1183	1342	1511	1690	1879	2078	2287	2506	2735
14	100	151	212	283	364	455	556	667	788	919	1060	1211	1372	1543	1724	1915	2116	2327	2548	2779
15	106	159	222	295	378	471	574	687	810	943	1086	1239	1402	1575	1758	1951	2154	2367	2590	2823
16	112	167	232	307	392	487	592	707	832	967	1112	1267	1432	1607	1792	1987	2192	2407	2632	2867
17	118	175	242	319	406	503	610	727	854	991	1138	1295	1462	1639	1826	2023	2230	2447	2674	2911
18	124	183	252	331	420	519	628	747	876	1015	1164	1323	1492	1671	1860	2059	2268	2487	2716	2955
19	130	191	262	343	434	535	646	767	898	1039	1190	1351	1522	1703	1894	2095	2306	2527	2758	2999
20	136	199	272	355	448	551	664	787	920	1063	1216	1379	1552	1735	1928	2131	2344	2567	2800	3043

TABLE I

MINIMUM WORKSPACE FOR THE MMLC PACKAGE FOR UNCONSTRAINED PROBLEMS

M: \ N:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
21	142	207	282	367	462	567	682	807	942	1087	1242	1407	1582	1767	1962	2167	2382	2607	2842	3087
22	148	215	292	379	476	583	700	827	964	1111	1268	1435	1612	1799	1996	2203	2420	2647	2884	3131
23	154	223	302	391	490	599	718	847	986	1135	1294	1463	1642	1831	2030	2239	2458	2687	2926	3175
24	160	231	312	403	504	615	736	867	1008	1159	1320	1491	1672	1863	2064	2275	2496	2727	2968	3219
25	166	239	322	415	518	631	754	887	1030	1183	1346	1519	1702	1895	2098	2311	2534	2767	3010	3263
26	172	247	332	427	532	647	772	907	1052	1207	1372	1547	1732	1927	2132	2347	2572	2807	3052	3307
27	178	255	342	439	546	663	790	927	1074	1231	1398	1575	1762	1959	2166	2383	2610	2847	3094	3351
28	184	263	352	451	560	679	808	947	1096	1255	1424	1603	1792	1991	2200	2419	2648	2887	3136	3395
29	190	271	362	463	574	695	826	967	1118	1279	1450	1631	1822	2023	2234	2455	2686	2927	3178	3439
30	196	279	372	475	588	711	844	987	1140	1303	1476	1659	1852	2055	2268	2491	2724	2967	3220	3483
31	202	287	382	487	602	727	862	1007	1162	1327	1502	1687	1882	2087	2302	2527	2762	3007	3262	3527
32	208	295	392	499	616	743	880	1027	1184	1351	1528	1715	1912	2119	2336	2563	2800	3047	3304	3571
33	214	303	402	511	630	759	898	1047	1206	1375	1554	1743	1942	2151	2370	2599	2838	3087	3346	3615
34	220	311	412	523	644	775	916	1067	1228	1399	1580	1771	1972	2183	2404	2635	2876	3127	3388	3659
35	226	319	422	535	658	791	934	1087	1250	1423	1606	1799	2002	2215	2438	2671	2914	3167	3430	3703
36	232	327	432	547	672	807	952	1107	1272	1447	1632	1827	2032	2247	2472	2707	2952	3207	3472	3747
37	238	335	442	559	686	823	970	1127	1294	1471	1658	1855	2062	2279	2506	2743	2990	3247	3514	3791
38	244	343	452	571	700	839	988	1147	1316	1495	1684	1883	2092	2311	2540	2779	3028	3287	3556	3835
39	250	351	462	583	714	855	1006	1167	1338	1519	1710	1911	2122	2343	2574	2815	3066	3327	3598	3879
40	256	359	472	595	728	871	1024	1187	1360	1543	1736	1939	2152	2375	2608	2851	3104	3367	3640	3923

output. Consequently, if p=1, the value of q is insignificant because all function evaluations will be reported by the package. The fields p and q control the printing of residual function values only. Printing of partial derivatives is controlled by the fields r and s. If s is not equal to zero (and is not greater than q) then the values of partial derivatives calculated in the first s calls of FDF are reported in the printed output. If r is not equal to zero (and p is greater than zero) then every (p*r)th evaluation of partial derivatives is reported as well. Moreover, if q is equal to zero and p is not equal to 1 (i.e., when the first call of FDF is not reported by the package), then the "starting point" values of optimization variables $\underset{\sim}{x}^0$ and corresponding residual function values $\underset{\sim}{f}(\underset{\sim}{x}^0)$ are printed; if, at the same time, s is greater than zero, the values of partial derivatives are included in the "starting point" information. It should be noted that the values of partial derivatives can only be printed for those evaluations for which printing of residual function values is indicated.

Note: The function evaluations reported by the package are indexed by two numbers in the form i/j where

i is the consecutive number of function evaluation,

j is the stage of the iterative algorithm:

0 - initial function evaluation,

1 - Stage 1 iteration,

2 - Stage 2 iteration.

If the value of IPR is negative, the partial derivatives calculated by FDF are verified numerically by comparing values supplied by FDF with the differences of residual function values in the small environment of the starting point. All partial derivatives which differ from the numerically approximated ones by more than 1% (with respect to the numerical approximation) are reported in the printed output.

IFALL is an INTEGER variable which, on exit, contains information about the solution:

IFALL = -2 feasible region is empty,

IFALL = -1 incorrect input data,

IFALL = 0 regular solution; required accuracy obtained,

IFALL = 1 singular solution; required accuracy obtained,

IFALL = 2 machine accuracy reached,

IFALL = 3 maximum number of function evaluations reached,

IFALL = 4 iteration terminated by the user.

Original entry (subroutine MMLA1Q)

The subroutine call is

```
CALL MMLA1Q (FDF,N,M,L,LEQ,B,C,LC,X,DX,EPS,MAXF,KEQS,W,IW,IFALL)
```

The arguments are generally the same as for the foregoing standard entry. The detailed description is given in [1].

V. AUXILIARY SUBROUTINES

The package contains several auxiliary subroutines which can be used to change or to set the values of additional parameters controlling the form of the printed output generated by the package. All these subroutines (if used) should be called before the standard entry to the package.

Subroutine MMXHDR

Subroutine MMXHDR defines the title line which is printed within the page header. The title must be a string of up to 80 characters which is stored in consecutive elements of a REAL array, 10 characters in one element.

The subroutine call is

```
CALL MMXHDR(L,T)
```

where L is the number of array elements required for the title, and T is the name of an array or the first element storing the title. If L is equal to zero, no title line is printed by the package.

Subroutine MMXPSZ

Subroutine MMXPSZ defines the "page size", that is the maximum number of lines printed on a page. The preset value is 65.

The subroutine call is

```
CALL MMXPSZ(L)
```

where L is the defined page size. If the value of L is equal to zero, the printed output is generated without page control.

Subroutine MMXPLM

Subroutine MMXPLM defines the limit of printed pages. The preset value of this limit is 10, and it cannot be changed to more than 50.

The subroutine call is

```
CALL MMXPLM (L)
```

where L is the defined limit of pages.

When the limit of pages is reached the further output generated by the package is suppressed except of the results of optimization.

Subroutine MMXLLM

Subroutine MMXLLM defines the limit of printed lines. The preset value of this limit is 750.

The subroutine call is

```
CALL MMXLLM(L)
```

where L is the defined limit of lines.

When the limit of printed lines is reached the further output generated by the package is suppressed except of the results of optimization.

Subroutine MMXGLM

Subroutine MMXGLM defines the bounds on the number of variables and the number of residual functions when the matrix of partial derivatives is printed by the package (for some problems this matrix can be quite large and it can be reasonable to print the initial part of it only). The preset bound on the number of variables is 10, and on the number of functions is 25.

The subroutine call is

CALL MMXGLM(K,L)

where K is the defined bound on the number of variables, and L is the defined bound on the number of residual functions.

Subroutine MMXGVL

Subroutine MMXGVL defines, for the matrix of partial derivatives, the number of columns printed in one line. The preset value is 10, and it corresponds to 120 character lines. If the standard form of generated output is to be preserved this number should be defined as 6.

The subroutine call is

CALL MMXGVL(K)

where K is the defined number of columns per line.

VI. GENERAL INFORMATION

Use of COMMON: COMMON/MMX000/ (for standard entry only), COMMON/MML000/ (see argument FDF).

Workspace: Provided by the user; see arguments W and IW.

Input/output: Output (for standard entry only) as defined by the user; see argument ICH.

Subroutines: MMLC8A, MMLC9A, S2LA1Q, FEASI, MMLPA, LINSYS, BFGS, ADDCL, DELCL, UTTRNS, UTRNS, RSOLV, TSOLV, HACUM, LIMIT and:

a) for standard entry: MMLC1A, MMX00Q, MMX00V, MMX00G, MMX00H, MMX00B, MMXPSZ, MMXPLM, MMXLLM, MMXHDR, MMXGLM, MMXGVL;

b) for original entry: MMLA1Q, MMX00Z.

Restrictions: N>0, M>0, L≥0, LEQ≥0, LEQ≤L, LEQ≤N, LC≥L, DX>0, EPS≥0, MAXF>0, KEQS>0, IW≥IWR.

Date: April 1982.

VII. EXAMPLES

Example 1 [1, Example 1]

Minimize

$$F(\underset{\sim}{x}) = \max_{1 \le i \le 3} f_i(\underset{\sim}{x})$$

subject to

$$-3x_1 - x_2 - 2.5 \ge 0,$$

where

$$f_1(\underset{\sim}{x}) = x_1^2 + x_2^2 + x_1 x_2 - 1,$$

$$f_2(\underset{\sim}{x}) = \sin(x_1),$$

$$f_3(\underset{\sim}{x}) = -\cos(x_2).$$

The starting point is

$$\underset{\sim}{x}^0 = \begin{bmatrix} -2 \\ -1 \end{bmatrix}.$$

To show the influence of the parameters DX and KEQS the optimization has been performed several times for different values of DX and KEQS. The resulting numbers of residual function evaluations required to achieve the accuracy EPS = 10^{-6}, as well as the numbers of shifts to Stage 2 are summarized in the following table (the numbers of shifts are given in parentheses):

DX	KEQS 2	3	4
0.1	10(2)	10(2)	12(1)
0.2	9(2)	9(1)	10(1)
0.4	12(2)	12(1)	14(1)

It can be observed that the increasing values of KEQS correspond, generally, to smaller numbers of shifts to Stage 2 (some too early shifts are eliminated), and to slightly increased numbers of residual function evaluations. Moreover, too small and too large values of DX require more residual function evaluations because of adjustments which are performed by the package.

```
      PROGRAM TRMML1(OUTPUT,TAPE1=OUTPUT)
C
C    J.HALD - EXAMPLE 1.
C
      DIMENSION X(2),W(67),B(1),C(1,2),H(4)
      EXTERNAL FDF
      DATA H/10HPROGRAM TR,10HMML1 : J.H,10HALD - EXAM,10HPLE 1      /
      CALL MMXHDR(4,H)
      N=2
      M=3
      L=1
      LEQ=0
      LC=1
      B(1)=-2.5E0
      C(1,1)=-3.0
      C(1,2)=-1.0
      X(1)=-2.0
      X(2)=-1.0
      DX=0.2
      EPS=1.E-6
      MAXF=50
      KEQS=3
      IW=67
      ICH=1
      IPC=-10
      CALL MMLC1A(FDF,N,M,L,LEQ,B,C,LC,X,DX,EPS,MAXF,KEQS,W,IW,
     1            ICH,IPC,IFALL)
      STOP
      END
C
C
      SUBROUTINE FDF(N,M,X,DF,F)
      DIMENSION X(N),F(M),DF(M,N)
      X1=X(1)
      X2=X(2)
      F(1)=X1*X1+X2*X2+X1*X2-1.0
      F(2)=SIN(X1)
      F(3)=-COS(X2)
      DF(1,1)=X1+X1+X2
      DF(1,2)=X2+X2+X1
      DF(2,1)=COS(X1)
      DF(2,2)=0.0
      DF(3,1)=0.0
      DF(3,2)=SIN(X2)
      RETURN
      END
```

DATE : 82/04/22. TIME : 15.17.59. PAGE : 1
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE) (V:82.04)

PROGRAM TRMML1 : J.HALD - EXAMPLE 1

INPUT DATA

NUMBER OF VARIABLES (N) 2

NUMBER OF FUNCTIONS (M) 3

TOTAL NUMBER OF LINEAR CONSTRAINTS (L) 1

NUMBER OF EQUALITY CONSTRAINTS (LEQ) 0

STEP LENGTH (DX) 2.000E-01

ACCURACY (EPS) 1.000E-06

MAX NUMBER OF FUNCTION EVALUATIONS (MAXF) 50

NUMBER OF SUCCESSIVE ITERATIONS (KEQS) 3

WORKING SPACE (IW) 67

PRINTOUT CONTROL (IPR) . . . . -10

STARTING POINT :

	VARIABLES		FUNCTION VALUES
1	-2.000000000000E+00	1	6.000000000000E+00
2	-1.000000000000E+00	2	-9.092974268257E-01
		3	-5.403023058681E-01

VERIFICATION OF PARTIAL DERIVATIVES PERFORMED.

SOLUTION

	VARIABLES		FUNCTION VALUES
1	-8.928571428571E-01	1	-3.303571428571E-01
2	1.785714285714E-01	2	-7.788668934368E-01
		3	-9.840984453126E-01

TYPE OF SOLUTION (IFALL) 1

NUMBER OF FUNCTION EVALUATIONS 9

NUMBER OF SHIFTS TO STAGE-2 1

EXECUTION TIME (IN SECONDS) 029

<u>Example 2</u> [5, Example 3]

This is the problem proposed by Brent [6] as an example in which the continuous analog of the Newton-Raphson method is not globally convergent. The problem is to solve a system of 2 nonlinear equations

$$4(x_1+x_2) = 0 ,$$
$$(x_1-x_2)((x_1-2)^2 + x_2^2) + 3x_1 + 5x_2 = 0 .$$

More details and some solutions are given in [5]. It can be observed, however, that the solution can be obtained by minimizing the objective function

$$F(\underset{\sim}{x}) = \max\ (f(\underset{\sim}{x}),\ -\ f(\underset{\sim}{x}))$$

subject to linear equality constraint

$$4x_1 + 4x_2 = 0,$$

where

$$f(\underset{\sim}{x}) = (x_1-x_2)((x_1-2)^2 + x_2^2) + 3x_1 + 5x_2 .$$

The solutions are shown for 4 different starting points $\underset{\sim}{x}^0$

$$\begin{bmatrix} 2 \\ 2 \end{bmatrix}, \begin{bmatrix} -2 \\ -2 \end{bmatrix}, \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

as in [5]. For this example all the solutions have been found in Stage 1 only.

```
      PROGRAM TRMML2(OUTPUT,TAPE6=OUTPUT)
C
C   BRENT EXAMPLE
C
      DIMENSION X(2),XX(4,2),B(1),C(1,2),T(3),W(59)
      EXTERNAL FDF
      DATA XX/2.0,-2.0,2.0,2.0,
     1        2.0,-2.0,0.0,1.0/
      DATA B/0.0/,C/4.0,4.0/
      DATA T/10HTRMML2 : B,10HRENT EXAMP,10HLE        /
      CALL MMXHDR(3,T)
      N=2
      M=2
      LEQ=1
      L=1
      IL=1
      IPR=-10
      DO 20 I=1,4
      X(1)=XX(I,1)
      X(2)=XX(I,2)
      DX=0.2
      EPS=1.E-6
      MAXF=50
      KEQS=2
      IW=59
      ICH=6
      CALL MMLC1A(FDF,N,M,L,LEQ,B,C,IL,X,DX,EPS,MAXF,KEQS,W,IW,ICH,
     1 IPR,IFLAG)
      IPR=0
   20 CONTINUE
      STOP
      END
C
C
      SUBROUTINE FDF(N,M,X,DF,F)
      DIMENSION X(N),DF(M,N),F(M)
      X1=X(1)
      X2=X(2)
      R1=X1-X2
      R2=(X1-2.0)**2+X2*X2
      F(1)=R1*R2+3.0*X1+5.0*X2
      F(2)=-F(1)
      DF(1,1)=R2+(R1+R1)*(X1-2.0)+3.0
      DF(1,2)=-R2+R1*(X2+X2)+5.0
      DF(2,1)=-DF(1,1)
      DF(2,2)=-DF(1,2)
      RETURN
      END
```

DATE : 82/04/22. TIME : 15.26.07. PAGE : 1
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE) (V:82.04)

TRMML2 : BRENT EXAMPLE

INPUT DATA

NUMBER OF VARIABLES (N)	2
NUMBER OF FUNCTIONS (M)	2
TOTAL NUMBER OF LINEAR CONSTRAINTS (L)	1
NUMBER OF EQUALITY CONSTRAINTS (LEQ)	1
STEP LENGTH (DX)	2.000E-01
ACCURACY (EPS)	1.000E-06
MAX NUMBER OF FUNCTION EVALUATIONS (MAXF)	50
NUMBER OF SUCCESSIVE ITERATIONS (KEQS)	2
WORKING SPACE (IW)	59
PRINTOUT CONTROL (IPR)	-10

STARTING POINT :

	VARIABLES		FUNCTION VALUES
1	2.000000000000E+00	1	1.600000000000E+01
2	2.000000000000E+00	2	-1.600000000000E+01

VERIFICATION OF PARTIAL DERIVATIVES PERFORMED.

SOLUTION

	VARIABLES		FUNCTION VALUES
1	-1.894780628693E-14	1	3.635071051258E-27
2	1.326346440086E-13	2	-3.635071051258E-27

TYPE OF SOLUTION (IFALL)	0
NUMBER OF FUNCTION EVALUATIONS	3
NUMBER OF SHIFTS TO STAGE-2	0
EXECUTION TIME (IN SECONDS)	.011

```
DATE : 82/04/22.                    TIME : 15.26.07.                     PAGE :   1
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)                 (V:82.04)

TRMML2 : BRENT EXAMPLE


INPUT DATA
----------

  NUMBER OF VARIABLES (N) . . . . . . . . . . . . . . . . . . . . . . . . .     2

  NUMBER OF FUNCTIONS (M) . . . . . . . . . . . . . . . . . . . . . . . . .     2

  TOTAL NUMBER OF LINEAR CONSTRAINTS (L)  . . . . . . . . . . . . . . . . .     1

  NUMBER OF EQUALITY CONSTRAINTS (LEQ)  . . . . . . . . . . . . . . . . . .     1

  STEP LENGTH (DX)  . . . . . . . . . . . . . . . . . . . . . . . . .  2.000E-01

  ACCURACY (EPS)  . . . . . . . . . . . . . . . . . . . . . . . . . .  1.000E-06

  MAX NUMBER OF FUNCTION EVALUATIONS (MAXF) . . . . . . . . . . . . . . . .    50

  NUMBER OF SUCCESSIVE ITERATIONS (KEQS)  . . . . . . . . . . . . . . . . .     2

  WORKING SPACE (IW) . . . . . . . . . . . . . . . . . . . . . . . . . . .    59

  PRINTOUT CONTROL (IPR) . . . . . . . . . . . . . . . . . . . . . . . . .     0

  STARTING POINT :

                          VARIABLES                     FUNCTION VALUES

                  1  -2.000000000000E+00           1  -1.600000000000E+01
                  2  -2.000000000000E+00           2   1.600000000000E+01

SOLUTION
--------

                          VARIABLES                     FUNCTION VALUES

                  1   1.894780628694E-14           1  -2.019483917366E-27
                  2  -1.326346440086E-13           2   2.019483917366E-27

  TYPE OF SOLUTION (IFALL)  . . . . . . . . . . . . . . . . . . . . . . . .     0

  NUMBER OF FUNCTION EVALUATIONS  . . . . . . . . . . . . . . . . . . . . .     3

  NUMBER OF SHIFTS TO STAGE-2 . . . . . . . . . . . . . . . . . . . . . . .     0

  EXECUTION TIME (IN SECONDS) . . . . . . . . . . . . . . . . . . . . . .   .010
```

```
DATE : 82/04/22.                    TIME : 15.28.23.                  PAGE :  1
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)              (V:82.04)

TRMML2 : BRENT EXAMPLE


INPUT DATA
----------

  NUMBER OF VARIABLES (N) . . . . . . . . . . . . . . . . . . . . . . . . .      2

  NUMBER OF FUNCTIONS (M) . . . . . . . . . . . . . . . . . . . . . . . . .      2

  TOTAL NUMBER OF LINEAR CONSTRAINTS (L)  . . . . . . . . . . . . . . . . .      1

  NUMBER OF EQUALITY CONSTRAINTS (LEQ)  . . . . . . . . . . . . . . . . . .      1

  STEP LENGTH (DX)  . . . . . . . . . . . . . . . . . . . . . . . . .  2.000E-01

  ACCURACY (EPS)  . . . . . . . . . . . . . . . . . . . . . . . . . .  1.000E-06

  MAX NUMBER OF FUNCTION EVALUATIONS (MAXF) . . . . . . . . . . . . . . . .     50

  NUMBER OF SUCCESSIVE ITERATIONS (KEQS)  . . . . . . . . . . . . . . . . .      2

  WORKING SPACE (IW) . . . . . . . . . . . . . . . . . . . . . . . . . . .     59

  PRINTOUT CONTROL (IPR) . . . . . . . . . . . . . . . . . . . . . . . . .      0

  STARTING POINT :

                            VARIABLES                     FUNCTION VALUES

                     1   2.000000000000E+00           1   6.000000000000E+00
                     2   0.                           2  -6.000000000000E+00

SOLUTION
--------

                            VARIABLES                     FUNCTION VALUES

                     1  -1.514612938024E-28           1  -9.087677628146E-28
                     2   1.514612938024E-28           2   9.087677628146E-28

  TYPE OF SOLUTION (IFALL)  . . . . . . . . . . . . . . . . . . . . . . . .      2

  NUMBER OF FUNCTION EVALUATIONS  . . . . . . . . . . . . . . . . . . . . .     17

  NUMBER OF SHIFTS TO STAGE-2 . . . . . . . . . . . . . . . . . . . . . . .      2

  EXECUTION TIME (IN SECONDS) . . . . . . . . . . . . . . . . . . . . . . .   .046
```

```
DATE : 82/04/22.                      TIME : 15.28.23.                    PAGE :  1
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)                 (V:82.04)

TRMML2 : BRENT EXAMPLE


INPUT DATA
----------

  NUMBER OF VARIABLES (N) . . . . . . . . . . . . . . . . . . . . . . . . .        2

  NUMBER OF FUNCTIONS (M) . . . . . . . . . . . . . . . . . . . . . . . . .        2

  TOTAL NUMBER OF LINEAR CONSTRAINTS (L)  . . . . . . . . . . . . . . . . .        1

  NUMBER OF EQUALITY CONSTRAINTS (LEQ)  . . . . . . . . . . . . . . . . . .        1

  STEP LENGTH (DX)  . . . . . . . . . . . . . . . . . . . . . . . . . .    2.000E-01

  ACCURACY (EPS)  . . . . . . . . . . . . . . . . . . . . . . . . . . .    1.000E-06

  MAX NUMBER OF FUNCTION EVALUATIONS (MAXF) . . . . . . . . . . . . . . . .       50

  NUMBER OF SUCCESSIVE ITERATIONS (KEQS)  . . . . . . . . . . . . . . . . .        2

  WORKING SPACE (IW) . . . . . . . . . . . . . . . . . . . . . . . . . . .        59

  PRINTOUT CONTROL (IPR) . . . . . . . . . . . . . . . . . . . . . . . . .         0

  STARTING POINT :

                              VARIABLES                      FUNCTION VALUES

                    1    2.000000000000E+00            1    1.200000000000E+01
                    2    1.000000000000E+00            2   -1.200000000000E+01


SOLUTION
--------

                              VARIABLES                      FUNCTION VALUES

                    1   -2.389010899710E-16            1   -1.433406539826E-15
                    2    2.389010899710E-16            2    1.433406539826E-15

  TYPE OF SOLUTION (IFALL)  . . . . . . . . . . . . . . . . . . . . . . . .        2

  NUMBER OF FUNCTION EVALUATIONS  . . . . . . . . . . . . . . . . . . . . .        8

  NUMBER OF SHIFTS TO STAGE-2 . . . . . . . . . . . . . . . . . . . . . . .        1

  EXECUTION TIME (IN SECONDS) . . . . . . . . . . . . . . . . . . . . . . .     .023
```

Example 3

Minimize the Beale constrained function

$$f_1(\underset{\sim}{x}) = 9 - 8x_1 - 6x_2 - 4x_3 + 2x_1^2 + 2x_2^2 + x_3^2 + 2x_1x_2 + 2x_1x_3$$

subject to constraints

$$x_i \geq 0 \ , \ i = 1,2,3 \ ,$$

$$3 - x_1 - x_2 - 2x_3 \geq 0 \ .$$

The function has a minimum $f_1(\underset{\sim}{x}^*) = 1/9$ at the point $\underset{\sim}{x}^* = [4/3 \;\; 7/9 \;\; 4/9]^T$.

The numbers of residual function evaluations required to achieve the accuracy EPS = 10^{-6}, as well as the numbers of shifts to Stage 2, for the starting point

$$\underset{\sim}{x}^0 = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}$$

and several values of parameters DX and KEQS are summarized in the following table:

	KEQS		
DX	2	3	4
0.125	10(1)	10(1)	13(1)
0.25	9(1)	10(1)	9(1)
0.5	11(1)	11(1)	12(1)
1.0	11(1)	11(1)	11(1)

It should be noted that the obtained results are much better then the results reported in [7, Example 5], where the constraints have been converted to additional residual functions.

```
      PROGRAM TRMML3(OUTPUT,TAPE2=OUTPUT)
C
C  BEALE CONSTRAINED FUNCTION
C
      DIMENSION X(3),W(98),C(4),DC(4,3),T(4)
      EXTERNAL FDF
      DATA C/0.0,0.0,0.0,3.0/
      DATA DC/1.0,0.0,0.0,-1.0,
     1        0.0,1.0,0.0,-1.0,
     2        0.0,0.0,1.0,-2.0/
      DATA T/10HTRMML3 : B,10HEALE CONST,10HRAINED FUN,5HCTION/
      CALL MMXHDR(4,T)
      N=3
      M=1
      L=4
      LEQ=0
      IC=4
      X(1)=0.5
      X(2)=0.5
      X(3)=0.5
      DX=0.25
      EPS=1.E-6
      MAXF=50
      KEQS=2
      IW=98
      IPR=-10
      LCH=2
      CALL MMLC1A(FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQS,W,IW,
     1            LCH,IPR,IFALL)
      STOP
      END
C
C
      SUBROUTINE FDF(N,M,X,DF,F)
      DIMENSION X(N),F(M),DF(M,N)
      X1=X(1)
      X2=X(2)
      X3=X(3)
      F(1)=9.0-8.0*X1-6.0*X2-4.0*X3+2.0*(X1*(X1+X2+X3)+X2*X2)+X3*X3
      DF(1,1)=4.0*X1+2.0*(X2+X3)-8.0
      DF(1,2)=4.0*X2+2.0*X1-6.0
      DF(1,3)=2.0*(X1+X3)-4.0
      RETURN
      END
```

DATE : 82/04/22. TIME : 15.33.23. PAGE : 1
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE) (V:82.04)

TRMML3 : BEALE CONSTRAINED FUNCTION

INPUT DATA

NUMBER OF VARIABLES (N) 3

NUMBER OF FUNCTIONS (M) 1

TOTAL NUMBER OF LINEAR CONSTRAINTS (L) 4

NUMBER OF EQUALITY CONSTRAINTS (LEQ) 0

STEP LENGTH (DX) 2.500E-01

ACCURACY (EPS) 1.000E-06

MAX NUMBER OF FUNCTION EVALUATIONS (MAXF) 50

NUMBER OF SUCCESSIVE ITERATIONS (KEQS) 2

WORKING SPACE (IW) 98

PRINTOUT CONTROL (IPR) . . . . -10

STARTING POINT :

	VARIABLES		FUNCTION VALUES
1	5.000000000000E-01	1	2.250000000000E+00
2	5.000000000000E-01		
3	5.000000000000E-01		

VERIFICATION OF PARTIAL DERIVATIVES PERFORMED.

SOLUTION

	VARIABLES		FUNCTION VALUES
1	1.333333333333E+00	1	1.111111111109E-01
2	7.777777777774E-01		
3	4.444444444448E-01		

TYPE OF SOLUTION (IFALL) 1

NUMBER OF FUNCTION EVALUATIONS 9

NUMBER OF SHIFTS TO STAGE-2 1

EXECUTION TIME (IN SECONDS) 030

Example 4

This is again the Beale constrained function (Example 3)

$$f_1(\underset{\sim}{x}) = 9 - 8x_1 - 6x_2 - 4x_3 + 2x_1^2 + 2x_2^2\, x_3^2 + 2x_1x_2 + 2x_1x_3$$

but in this case the constraint

$$3 - x_1 - x_2 - 2x_3 \geq 0$$

which is the only constraint active at the solution, is transformed into additional residual function by the common technique [8]

$$f_2(\underset{\sim}{x}) = f_1(\underset{\sim}{x}) - \alpha\,(3 - x_1 - x_2 - 2x_3),$$

and $\alpha = 1$ is assumed (as in [7]). The objective function is thus

$$F(\underset{\sim}{x}) = \max(f_1(\underset{\sim}{x}),\ f_2(\underset{\sim}{x}))$$

and it is minimized subject to constraints

$$x_i \geq 0,\ i = 1, 2, 3.$$

The results obtained for the same starting point and the same parameters DX and KEQS as in Example 3, are summarized in the following table:

	KEQS		
DX	2	3	4
0.125	10(1)	13(1)	15(1)
0.25	10(1)	11(1)	12(1)
0.5	11(1)	12(1)	11(1)
1.0	10(1)	11(1)	12(1)

The results obtained in Example 3 seem to be slightly better than those of Example 4 (the total number of function evaluations is 128 for Example 3, and 138 for Example 4), however, the differences are not significant.

```
      PROGRAM TRMML4(OUTPUT,TAPE2=OUTPUT)
C
C   BEALE CONSTRAINED FUNCTION
C
      DIMENSION X(3),W(104),C(3),DC(3,3),T(4)
      EXTERNAL FDF
      DATA C/0.0,0.0,0.0/
      DATA DC/1.0,0.0,0.0,
     1        0.0,1.0,0.0,
     2        0.0,0.0,1.0/
      DATA T/10HTRMML4 : B,10HEALE CONST,10HRAINED FUN,5HCTION/
      CALL MMXHDR(4,T)
      N=3
      M=2
      L=3
      LEQ=0
      IC=3
      X(1)=0.5
      X(2)=0.5
      X(3)=0.5
      DX=0.25
      EPS=1.E-6
      MAXF=50
      KEQS=2
      IW=104
      LCH=2
      IPR=-10
      CALL MMLC1A(FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQS,W,IW,
     1            LCH,IPR,IFALL)
      STOP
      END
C
C
      SUBROUTINE FDF(N,M,X,DF,F)
      DIMENSION X(N),F(M),DF(M,N)
      X1=X(1)
      X2=X(2)
      X3=X(3)
      F(1)=9.0-8.0*X1-6.0*X2-4.0*X3+2.0*(X1*(X1+X2+X3)+X2*X2)+X3*X3
      DF(1,1)=4.0*X1+2.0*(X2+X3)-8.0
      DF(1,2)=4.0*X2+2.0*X1-6.0
      DF(1,3)=2.0*(X1+X3)-4.0
      F(2)=F(1)+X1+X2+X3+X3-3.0
      DF(2,1)=DF(1,1)+1.0
      DF(2,2)=DF(1,2)+1.0
      DF(2,3)=DF(1,3)+2.0
      RETURN
      END
```

DATE : 82/04/22 TIME : 16.40.03 PAGE : 1
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE) (V:82.04)

TRMML4 : BEALE CONSTRAINED FUNCTION

INPUT DATA

NUMBER OF VARIABLES (N) 3

NUMBER OF FUNCTIONS (M) 2

TOTAL NUMBER OF LINEAR CONSTRAINTS (L) 3

NUMBER OF EQUALITY CONSTRAINTS (LEQ) 0

STEP LENGTH (DX) 2.500E-01

ACCURACY (EPS) 1.000E-06

MAX NUMBER OF FUNCTION EVALUATIONS (MAXF) 50

NUMBER OF SUCCESSIVE ITERATIONS (KEQS) 2

WORKING SPACE (IW) 104

PRINTOUT CONTROL (IPR) . . . . -10

STARTING POINT :

	VARIABLES		FUNCTION VALUES
1	5.000000000000E-01	1	2.250000000000E+00
2	5.000000000000E-01	2	1.250000000000E+00
3	5.000000000000E-01		

VERIFICATION OF PARTIAL DERIVATIVES PERFORMED.

SOLUTION

	VARIABLES		FUNCTION VALUES
1	1.333333333174E+00	1	1.111111111109E-01
2	7.777777778903E-01	2	1.111111111109E-01
3	4.444444444676E-01		

TYPE OF SOLUTION (IFALL) 1

NUMBER OF FUNCTION EVALUATIONS 10

NUMBER OF SHIFTS TO STAGE-2 1

EXECUTION TIME (IN SECONDS) 036

Example 5

The problem is to determine an optimally centered point $\underset{\sim}{x}^* = [x_1^*\ x_2^*]^T$ that maximizes the relative tolerance r in the region R_c defined by the inequalities

$$2 + 2x_1 - x_2 \geq 0,$$
$$143 - 11\,x_1 - 13\,x_2 \geq 0,$$
$$-60 + 4x_1 + 15x_2 \geq 0,$$

i.e., to find a point $\underset{\sim}{x}^*$ and a tolerance r such that the tolerance region R_ε

$$R_\varepsilon = \{\underset{\sim}{x} \mid (1-r)x_i^* \leq x_i \leq (1+r)x_i^*,\ i = 1,2\}$$

is in the constraint region R_c and is as large as possible.

It can be shown [9] that if the constraint region R_c is one-dimensionally convex (and it is in this case) then it is sufficient that all vertices of R_ε belong to R_c to guarantee that the whole tolerance region R_ε is in the constraint region R_c.

For minimax formulation of the problem it is convenient to assume that the tolerance r is an additional optimization variable; then, however, the vertices of the tolerance region R_ε will be described by nonlinear expressions

$$[(1\pm r)x_1^*\ \ (1\pm r)x_2^*]^T$$

and therefore it is reasonable to introduce independent tolerances for variables x_1 and x_2 (say x_3 and x_4, respectively), and to require that

$$\frac{x_3^*}{x_1^*} = \frac{x_4^*}{x_2^*}$$

(provided that $x_1^* > 0$ and $x_2^* > 0$). The minimax objective function can then take the form

$$f(\underset{\sim}{x}) = \max(f_1(\underset{\sim}{x}),\ f_2(\underset{\sim}{x}))$$

subject to constraints

$$2 + 2(x_1 \pm x_3) - (x_2 \pm x_4) \geq 0,$$

$$143 - 11(x_1 \pm x_3) - 13(x_2 \pm x_4) \geq 0,$$

$$-60 + 4(x_1 \pm x_3) + 15(x_2 \pm x_4) \geq 0,$$

$$x_3 \geq 0,$$

$$x_4 \geq 0,$$

where

$$f_1(\underset{\sim}{x}) = -x_3/x_1,$$

$$f_2(\underset{\sim}{x}) = -x_4/x_2,$$

since x_3 and x_4 are to be maximized.

It should be observed that due to $x_3 \geq 0$ and $x_4 \geq 0$, the first 3 constraints (and in fact, 12 constraints) can be simplified to the form

$$2 + 2(x_1 - x_3) - (x_2 + x_4) \geq 0,$$

$$143 - 11(x_1 + x_3) - 13(x_2 + x_4) \geq 0,$$

$$-60 + 4(x_1 - x_3) + 15(x_2 - x_4) \geq 0,$$

or, finally,

$$2 + 2x_1 - x_2 - 2x_3 - x_4 \geq 0,$$

$$143 - 11x_1 - 13x_2 - 11x_3 - 13x_4 \geq 0,$$

$$-60 + 4x_1 + 15x_2 - 4x_3 - 15x_4 \geq 0.$$

The solution is shown for the starting point $\underset{\sim}{x}^0 = \underset{\sim}{1}$, which is infeasible, and is adjusted by the package. The resulting relative tolerance r is equal to 0.3414 or 34.1%.

```
      PROGRAM TRMML5 (OUTPUT,TAPE6=OUTPUT)
C
C  TOLERANCING EXAMPLE
C
      DIMENSION X(4),B(5),C(5,4),W(159),H(3)
      EXTERNAL FT
      DATA B/2.0,143.0,-60.0,0.0,0.0/
      DATA C/2.0,-11.0,4.0,0.0,0.0,
     1       -1.0,-13.0,15.0,0.0,0.0,
     2       -2.0,-11.0,-4.0,1.0,0.0,
     3       -1.0,-13.0,-15.0,0.0,1.0/
      DATA H/10HTRMML5 : T,10HOLERANCING,10H EXAMPLE  /
      CALL MMXHDR(3,H)
      N=4
      M=2
      DX=1.0
      EPS=1.E-6
      IC=5
      L=5
      LEQ=0
      X(1)=1.0
      X(2)=1.0
      X(3)=1.0
      X(4)=1.0
      MAXF=25
      KEQS=3
      IW=159
      ICH=6
      IPR=-1000
      CALL MMLC1A(FT,N,M,L,LEQ,B,C,IC,X,DX,EPS,MAXF,KEQS,W,IW,ICH,IPR,
     1 IFLAG)
      STOP
      END
C
C
      SUBROUTINE FT(N,M,X,D,F)
      DIMENSION X(N),D(M,N),F(M)
      X1=X(1)
      X2=X(2)
      X3=X(3)
      X4=X(4)
      F(1)=-X3/X1
      F(2)=-X4/X2
      D(1,1)=X3/(X1*X1)
      D(1,2)=0.0
      D(1,3)=-1.0/X1
      D(1,4)=0.0
      D(2,1)=0.0
      D(2,2)=X4/(X2*X2)
      D(2,3)=0.0
      D(2,4)=-1.0/X2
      RETURN
      END
```

```
DATE : 82/05/19.                    TIME : 14.56.41.                    PAGE :   1
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)                (V:82.04)

TRMML5 : TOLERANCING EXAMPLE


INPUT DATA
----------

 NUMBER OF VARIABLES (N) . . . . . . . . . . . . . . . . . . . . . . . . . .     4

 NUMBER OF FUNCTIONS (M) . . . . . . . . . . . . . . . . . . . . . . . . . .     2

 TOTAL NUMBER OF LINEAR CONSTRAINTS (L) . . . . . . . . . . . . . . . . . . .    5

 NUMBER OF EQUALITY CONSTRAINTS (LEQ) . . . . . . . . . . . . . . . . . . . .    0

 STEP LENGTH (DX) . . . . . . . . . . . . . . . . . . . . . . . . . . . 1.000E+00

 ACCURACY (EPS) . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1.000E-06

 MAX NUMBER OF FUNCTION EVALUATIONS (MAXF) . . . . . . . . . . . . . . . . .    25

 NUMBER OF SUCCESSIVE ITERATIONS (KEQS) . . . . . . . . . . . . . . . . . . .    3

 WORKING SPACE (IW) . . . . . . . . . . . . . . . . . . . . . . . . . . . .    159

 PRINTOUT CONTROL (IPR) . . . . . . . . . . . . . . . . . . . . . . . . . . -1000

VERIFICATION OF PARTIAL DERIVATIVES PERFORMED.

FUNCTION EVALUATION :    1 / 0

                              VARIABLES                         FUNCTION VALUES

                        1   1.700389105058E+00             1  -1.762013729977E-01
                        2   3.626459143969E+00             2   0.
                        3   2.996108949416E-01
                        4   0.

FUNCTION EVALUATION :    2 / 1

                              VARIABLES                         FUNCTION VALUES

                        1   1.871126283894E+00             1  -1.938686527610E-01
                        2   4.307257078478E+00             2  -1.647196841922E-01
                        3   3.627527318041E-01
                        4   7.094900257014E-01

FUNCTION EVALUATION :    3 / 1

                              VARIABLES                         FUNCTION VALUES

                        1   3.331240161110E+00             1  -2.887243906439E-01
                        2   5.053505892104E+00             2  -3.335019083561E-01
                        3   9.618102856049E-01
                        4   1.685353858905E+00
```

```
DATE : 82/05/19.                    TIME : 14.56.41.                    PAGE :  2
LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE)               (V:82.04)

TRMML5 : TOLERANCING EXAMPLE


FUNCTION EVALUATION :    4 / 1

                              VARIABLES                         FUNCTION VALUES

                        1    3.664248088532E+00            1   -3.379898381485E-01
                        2    5.102333540799E+00            2   -3.428245890865E-01
                        3    1.238478618379E+00
                        4    1.749205399507E+00

FUNCTION EVALUATION :    5 / 1

                              VARIABLES                         FUNCTION VALUES

                        1    3.670134774875E+00            1   -3.414041140767E-01
                        2    5.094850908381E+00            2   -3.414075210309E-01
                        3    1.252999111358E+00
                        4    1.739420418652E+00

FUNCTION EVALUATION :    6 / 1

                              VARIABLES                         FUNCTION VALUES

                        1    3.670138928952E+00            1   -3.414065195725E-01
                        2    5.094845628088E+00            2   -3.414065195742E-01
                        3    1.253009358081E+00
                        4    1.739413513653E+00

FUNCTION EVALUATION :    7 / 2

                              VARIABLES                         FUNCTION VALUES

                        1    3.670138928954E+00            1   -3.414065195737E-01
                        2    5.094845628085E+00            2   -3.414065195737E-01
                        3    1.253009358086E+00
                        4    1.739413513650E+00


SOLUTION
--------

                              VARIABLES                         FUNCTION VALUES

                        1    3.670138928954E+00            1   -3.414065195737E-01
                        2    5.094845628085E+00            2   -3.414065195737E-01
                        3    1.253009358086E+00
                        4    1.739413513650E+00

  TYPE OF SOLUTION (IFALL) . . . . . . . . . . . . . . . . . . . . . . . .      0

  NUMBER OF FUNCTION EVALUATIONS . . . . . . . . . . . . . . . . . . . . .      7

  NUMBER OF SHIFTS TO STAGE-2 . . . . . . . . . . . . . . . . . . . . . . .      1

  EXECUTION TIME (IN SECONDS) . . . . . . . . . . . . . . . . . . . . . .     .185
```

VIII. REFERENCES

[1] J. Hald (Adapted and Edited by J.W. Bandler and W.M. Zuberek), "MMLA1Q - a Fortran package for linearly constrained minimax optimization", Faculty of Engineering, McMaster University, Hamilton, Canada, Report SOC-281, 1981.

[2] K. Madsen and H. Schjaer-Jacobsen, "Linearly constrained minimax optimization", Mathematical Programming, vol. 14, 1978, pp. 208-223.

[3] J. Hald and K. Madsen, "Combined LP and quasi-Newton methods for minimax optimization", Mathematical Programming, vol. 20, 1981, pp. 49-62.

[4] R. Fletcher, "An algorithm for solving linearly constrained optimization problems", Mathematical Programming, vol. 2, 1972, pp. 133-165.

[5] S. Incerti, V. Parisi and F. Zirilli, A new method for solving nonlinear simultaneous equations", SIAM J. Numerical Analysis, vol. 16, 1979, pp. 779-789.

[6] R.P. Brent, "On the Davidenko-Branin method for solving simultaneous nonlinear equations", IBM J. Research and Development, vol. 16, 1972, pp. 434-436.

[7] J.W. Bandler and W.M. Zuberek, "MMUM - a Fortran package for unconstrained minimax optimization", Faculty of Engineering, McMaster University, Hamilton, Canada, Report SOC-291, 1982.

[8] J.W. Bandler and C. Charalambous, "Nonlinear programming using minimax techniques", J. Optimization Theory and Applications, vol. 13, 1974, pp. 607-619.

[9] J.W. Bandler, "Optimization of design tolerances using nonlinear programming", J. Optimization Theory and Applications, vol. 14, 1974, pp. 99-114.

APPENDIX

LISTING OF THE MMLC PACKAGE

```
      SUBROUTINE MMLC1A (FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQS,W,IW,L
     1CH,IPR,IFALL)
      EXTERNAL FDF,MMX00Q,MMX00B
C
C           LEVEL 1 INTERFACE (STANDARD ENTRY)
C
      DIMENSION C(1), DC(1,1), X(1), W(1)
      COMMON /MMX000/ NCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      NCH=LCH
      IF (LCH.LE.0) GO TO 40
      I=IABS(IPR)
      J=I/10
      LG2=MOD(I,10)
      I=J/10
      LG1=MOD(J,10)
      J=I/10
      LV2=MOD(I,10)
      LV1=J
      LG1=LG1*LV1
      NRP=0
      CALL MMXPSZ (-1)
      CALL MMXPLM (-1)
      CALL MMXLLM (-1)
      CALL MMXHDR (-1,H)
      CALL MMXGLM (-1,-1)
      CALL MMXGVL (-1)
      IF (MXL.NE.0) LML=MXL*LMP+100
      IF (MXL.EQ.0) MXL=LML+100
      CALL DATE (DAT)
      CALL TIME (TIM)
      CALL MMX00B
      WRITE (LCH,10) N,M,L,LEQ,DX,EPS,MAXF,KEQS,IW,IPR
   10 FORMAT (11H0INPUT DATA/11H ----------//
     1 27H     NUMBER OF VARIABLES (N) ,25(2H. ),I4//
     2 27H     NUMBER OF FUNCTIONS (M) ,25(2H. ),I4//
     3 43H     TOTAL NUMBER OF LINEAR CONSTRAINTS (L)   ,17(2H. ),I4//
     4 41H     NUMBER OF EQUALITY CONSTRAINTS (LEQ)   ,18(2H. ),I4//
     5 21H     STEP LENGTH (DX)   ,25(2H. ),1PE10.3//
     6 19H     ACCURACY (EPS)   ,26(2H. ),1PE10.3//
     7 45H     MAX NUMBER OF FUNCTION EVALUATIONS (MAXF)  ,16(2H. ),I4//
     8 43H     NUMBER OF SUCCESSIVE ITERATIONS (KEQS)   ,17(2H. ),I4//
     9 22H     WORKING SPACE (IW)  ,26(2H. ),1H.,I6//
     * 26H     PRINTOUT CONTROL (IPR) ,24(2H. ),1H.,I6/)
      NRL=NRL-24
      LML=LML-24
      IF (LV2.NE.0.OR.LV1.EQ.1) GO TO 30
      WRITE (LCH,20)
   20 FORMAT (19H   STARTING POINT :)
      NRL=NRL-1
      LML=LML-1
      CALL FDF (N,M,X,W(M+1),W(1))
      CALL MMX00V (MMX00B,X,N,W,M)
      IF (LG2.NE.0) CALL MMX00G (MMX00B,W(M+1),M,N)
   30 IF (IPR.GE.0) GO TO 40
      I=M*N+M+1
      J=I+M
      K=J+M
      CALL MMX00H (MMX00B,FDF,N,M,X,W(M+1),W(1),W(J),W(K),W(I))
   40 CALL SECOND (TBEG)
      CALL MMLC8A (MMX00Q,MMX00B,FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQ
     1S,W,IW,IFALL)
      CALL SECOND (TEND)
      IF (LCH.LE.0) RETURN
      IF (IFALL.EQ.-1) GO TO 90
```

```
      IF (IFALL.EQ.-2) GO TO 70
      IF (NRL.LT.9) CALL MMX00B
      WRITE (LCH,50)
   50 FORMAT (//9H SOLUTION/9H --------)
      NRL=NRL-4
      LML=LML-4
      CALL MMX00V (MMX00B,X,N,W,M)
      CPU=TEND-TBEG
      IF (NRL.LT.9) CALL MMX00B
      WRITE (LCH,60) IFALL,MAXF,KEQS,CPU
   60 FORMAT (//29H   TYPE OF SOLUTION (IFALL)   ,24(2H. ),I4//
     1 35H   NUMBER OF FUNCTION EVALUATIONS   ,21(2H. ),I4//
     2 31H   NUMBER OF SHIFTS TO STAGE-2 ,23(2H. ),I4//
     3 31H   EXECUTION TIME (IN SECONDS) ,21(2H. ),1H.,F7.3/)
      RETURN
   70 WRITE (LCH,80)
   80 FORMAT (//42H E M P T Y   F E A S I B L E   R E G I O N/)
      RETURN
   90 WRITE (LCH,100)
  100 FORMAT (//40H I N C O R R E C T   P A R A M E T E R S/)
      RETURN
      END
C
C
      SUBROUTINE MMLA1Q (FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQS,W,IW,I
     1FALL)
      EXTERNAL FDF,MMX00Z
C
C          LEVEL 2 INTERFACE (J.HALD ENTRY)
C
      DIMENSION C(1), DC(1,1), X(1), W(1)
      CALL MMLC8A (MMX00Z,MMX00Z,FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQ
     1S,W,IW,IFALL)
      RETURN
      END
C
C
      SUBROUTINE MMX00Z (FUN,N,M,X,DF,F,K,NS)
C
C     DUMMY SUBROUTINE WHICH FOR BASIC AND ORIGINAL ENTRIES SUBSTITUTES
C     SUBROUTINE MMX00Q/11Q.
C
      EXTERNAL FUN
      DIMENSION X(N), DF(M,N), F(M)
      RETURN
      END
C
C
      SUBROUTINE MMX00Q (FHH,N,M,X,DF,F,K,NS)
C
C     PRINT RESULTS OF FUNCTION EVALUATION.
C
      EXTERNAL FHH
      DIMENSION X(N), DF(M,N), F(M)
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      IF (LCH.LE.0) RETURN
      IF (LV1+LV2.EQ.0) RETURN
      IF (K.LE.LV2) GO TO 10
      IF (LV1.EQ.0) RETURN
      IF (MOD(K,LV1).NE.0) RETURN
   10 IF (NRP.LE.LMP.AND.LML.GE.0) GO TO 30
      LV1=0
      LV2=0
      WRITE (LCH,20)
```

```
   20 FORMAT (//26H ( LISTING LIMIT REACHED )//)
      NRL=NRL-5
      LML=LML-5
      RETURN
   30 IF (NRL.LT.7) CALL FHH
      WRITE (LCH,40) K,NS
   40 FORMAT (22H0FUNCTION EVALUATION :,I4,2H /,I2)
      NRL=NRL-2
      LML=LML-2
      CALL MMX00V (FHH,X,N,F,M)
      IF (LG1+LG2.EQ.0) RETURN
      IF (K.LE.LG2) GO TO 50
      IF (K.LE.LV2) RETURN
      IF (LG1.EQ.0) RETURN
      IF (MOD(K,LG1).NE.0) RETURN
   50 CALL MMX00G (FHH,DF,M,N)
      RETURN
      END
C
C
      SUBROUTINE MMX00V (FHH,X,N,F,M)
C
C     PRINT VALUES OF VARIABLES AND RESIDUAL FUNCTIONS.
C
      DIMENSION X(N), F(M)
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      IF (LCH.LE.0) RETURN
      K=MAX0(N,M)
      IF (NRL.LT.5) CALL FHH
      WRITE (LCH,10)
   10 FORMAT (/30X,9HVARIABLES,18X,15HFUNCTION VALUES/)
      NRL=NRL-3
      LML=LML-3
      DO 40 I=1,K
      IF (NRL.LE.0) CALL FHH
      IF (I.LE.N.AND.I.LE.M) WRITE (LCH,20) I,X(I),I,F(I)
      IF (I.LE.N.AND.I.GT.M) WRITE (LCH,20) I,X(I)
      IF (I.GT.N.AND.I.LE.M) WRITE (LCH,30) I,F(I)
   20 FORMAT (18X,I4,2X,1PE19.12,5X,I4,2X,1PE19.12)
   30 FORMAT (48X,I4,2X,1PE19.12)
      NRL=NRL-1
      LML=LML-1
   40 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE MMX00G (FHH,G,M,N)
C
C     PRINT PARTIAL DERIVATIVES OF RESIDUAL FUNCTIONS.
C
      DIMENSION G(M,N)
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      IF (LCH.LE.0) RETURN
      IF (NRL.LT.7) CALL FHH
      MM=MIN0(M,LMF)
      NN=MIN0(N,LMV)
      WRITE (LCH,10)
   10 FORMAT (30H0  GRADIENTS ( DF.I / DX.J ) :)
      NRL=NRL-2
      LML=LML-2
      DO 60 K=1,NN,LGH
      IF (NRL.LT.5) CALL FHH
```

```
      J1=K
      J2=MIN0(NN,K+LGH-1)
      WRITE (LCH,20) (J,J=J1,J2)
   20 FORMAT (1H0,9X,12HVARIABLES(J),10(I5,5X))
      WRITE (LCH,30)
   30 FORMAT (10X,12HFUNCTIONS(I))
      NRL=NRL-3
      LML=LML-3
      DO 50 I=1,MM
      IF (NRL.LE.0) CALL FHH
      WRITE (LCH,40) I,(G(I,J),J=J1,J2)
   40 FORMAT (10X,I6,4X,10(1PE10.2))
      NRL=NRL-1
      LML=LML-1
   50 CONTINUE
   60 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE MMX00H (FHH,FDF,N,M,X,DF,F,DG,DH,G)
C
C     NUMERICAL VERIFICATION OF USER-DEFINED PARTIAL DERIVATIVES
C     (VARIABLES ARE DISTURBED ONE BY ONE).
C
      DIMENSION X(N), DF(M,N), F(M), DG(M), DH(M,N), G(M)
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      IF (LCH.LE.0) RETURN
      K=0
      CALL FDF (N,M,X,DF,F)
      DO 60 I=1,N
      Z=X(I)
      DX=1.E-6*Z
      IF (ABS(DX).LT.1.E-10) DX=1.E-10
      DX2=DX+DX
      X(I)=Z+DX
      CALL FDF (N,M,X,DH,F)
      DO 10 J=1,M
      DG(J)=DH(J,I)
   10 CONTINUE
      X(I)=Z-DX
      CALL FDF (N,M,X,DH,G)
      X(I)=Z
      DO 50 J=1,M
      Y=DF(J,I)
      Z=F(J)-G(J)
      IF (ABS(Z).LE.0.5E-13*(F(J)+G(J))) Z=0.0
      Z=Z/DX2
      IF (ABS(Y).LE.1.E-20.AND.ABS(Z).LE.1.E-20) GO TO 50
      IF (ABS(Z).LT.1.E-20) Z=SIGN(1.E-20,Z)
      R=100.0*ABS((Z-Y)/Z)
      IF (R.LE.1.0) GO TO 50
      IF (SIGN(1.0,DG(J))+SIGN(1.0,DH(J,I)).EQ.0.0) GO TO 50
      IF (K.NE.0) GO TO 30
      IF (NRL.LT.5) CALL FHH
      WRITE (LCH,20)
   20 FORMAT (38H0VERIFICATION OF PARTIAL DERIVATIVES :/
     1 1H0,18X,52H DF.I / DX.J   :   USER DEFINED   NUMERICAL   DIFFERENCE)
      NRL=NRL-4
      LML=LML-4
   30 K=K+1
      IF (NRL.LE.0) CALL FHH
      WRITE (LCH,40) J,I,Y,Z,R
   40 FORMAT (19X,I5,3X,I4,6X,1PE10.3,2X,1PE10.3,4X,0PF6.1,2H %)
```

```
      NRL=NRL-1
      LML=LML-1
   50 CONTINUE
   60 CONTINUE
      IF (K.NE.0) GO TO 80
      IF (NRL.LT.2) CALL FHH
      WRITE (LCH,70)
   70 FORMAT (47H0VERIFICATION OF PARTIAL DERIVATIVES PERFORMED.)
      NRL=NRL-2
      LML=LML-2
   80 RETURN
      END
C
C
      SUBROUTINE MMX00B
C
C     CHANGE PAGE AND PRINT PAGE HEADER.
C
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      IF (LCH.LE.0) RETURN
      IF (NRP.LT.LMP) GO TO 20
      LV1=0
      LV2=0
      WRITE (LCH,10)
   10 FORMAT (//27H ( LIMIT OF PAGES REACHED ))
   20 NRP=NRP+1
      NRL=MXL-5
      LML=LML-5
      WRITE (LCH,30) DAT,TIM,NRP
   30 FORMAT (1H1/7H DATE :,A10,19X,6HTIME :,A10,20X,6HPAGE :,I3/
     1 57H LINEARLY CONSTRAINED MINIMAX OPTIMIZATION (MMLC PACKAGE),15X,
     2 9H(V:82.04))
      IF (LHT.LE.0) GO TO 50
      WRITE (LCH,40) (H(J),J=1,LHT)
   40 FORMAT (1H0,8A10)
      NRL=NRL-2
      LML=LML-2
   50 WRITE (LCH,60)
   60 FORMAT (1H0)
      RETURN
      END
C
C
      SUBROUTINE MMXPSZ (L)
C
C     DEFINE THE PAGE SIZE (I.E. THE NUMBER OF LINES PER PAGE).
C
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      DATA LL/65/
      IF (L.GT.0) LL=MAX0(25,L)
      IF (L.EQ.0) LL=0
      MXL=LL
      RETURN
      END
C
C
      SUBROUTINE MMXPLM (L)
C
C     DEFINE THE LIMIT OF PRINTED PAGES.
C
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      DATA LL/10/
```

```
      IF (L.GT.0) LL=MIN0(50,L)
      LMP=LL
      RETURN
      END
C
C
      SUBROUTINE MMXLLM (L)
C
C     DEFINE THE LIMIT OF PRINTED LINES.
C
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      DATA LL/750/
      IF (L.GT.0) LL=L
      LML=LL
      RETURN
      END
C
C
      SUBROUTINE MMXHDR (L,T)
C
C     DEFINE THE HEADER LINE.
C
      DIMENSION T(1)
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      DATA LL/0/
      IF (L.GE.0) LL=MIN0(8,L)
      LHT=LL
      IF (L.LE.0) RETURN
      DO 10 I=1,LL
      H(I)=T(I)
   10 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE MMXGLM (K,L)
C
C     DEFINE THE SIZE OF PRINTED JACOBIAN.
C
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      DATA KK/25/,LL/10/
      IF (K.GT.0) KK=K
      IF (L.GT.0) LL=L
      LMF=KK
      LMV=LL
      RETURN
      END
C
C
      SUBROUTINE MMXGVL (L)
C
C     DEFINE THE NUMBER OF JACOBIAN COLUMNS PRINTED IN ONE LINE.
C
      COMMON /MMX000/ LCH,LV1,LV2,LG1,LG2,LMF,LMV,NRP,NRL,MXL,LMP,LML,LG
     1H,DAT,TIM,LHT,H(8)
      DATA LL/10/
      IF (L.GT.0) LL=MAX0(MIN0(10,L),5)
      LGH=LL
      RETURN
      END
C
C
```

```
      SUBROUTINE MMLC8A (FQQ,FHH,FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQ
     1S,W,IW,IFALL)
C
C     MMLC8A MINIMIZES THE MAXIMUM VALUE OF A SET OF NONLINEAR FUNCTIONS
C     SUBJECT TO LINEAR EQUALITY AND INEQUALITY CONSTRAINTS. DERIVATIVES
C     OF NONLINEAR FUNCTIONS ARE REQUIRED.
C
C     FOR A PROGRAM DESCRIPTION SEE:
C     J. HALD: "MMLA1Q, A FORTRAN SUBROUTINE FOR LINEARLY CONSTRAINED
C     MINIMAX OPTIMIZATION", REPORT NO. NI-81-1, INSTITUTE FOR NUMERICAL
C     ANALYSIS, TECHNICAL UNIVERSITY OF DENMARK, DK-2800 LYNGBY, DENMARK
C
C     THE SUBROUTINES: MMLPA,FEASI,S2LA1Q,BFGS,ADDCL,DELCL,HACUM,
C     UTRNS,UTTRNS,RSOLV,TSOLV,LIMIT,LINSYS   MUST BE AVAILABLE.
C
      DIMENSION C(1), DC(1,1), X(1), W(1)
      EXTERNAL FQQ,FHH,FDF
      COMMON /MML000/ MARK
      DATA ZERO/0.0/
      MARK=1
C
C        CHECK INPUT QUANTITIES
C
      IWR=2*M*N+5*N*N+4*M+8*N+4*IC+3
      IFALL=-1
      IF (IW.LT.IWR.OR.N.LT.1.OR.M.LT.1.OR.L.LT.0.OR.LEQ.LT.0.OR.LEQ.GT.
     1L.OR.LEQ.GT.N.OR.IC.LT.L.OR.DX.LE.ZERO.OR.EPS.LT.ZERO.OR.MAXF.LE.0
     2) GO TO 10
C
C        SPLIT UP THE WORK AREA
C
      N1=N+1
      NN=N+N
      NF=1
      NF1=NF+M
      NDF=NF1+M
      NDF1=NDF+M*N
      NX1=NDF1+M*N
      NB=NX1+N
      NU=NB+N*N
      NR=NU+N*N
      NA=NU
      NCL=NA+NN*NN
      NWL=NCL+IC
      NWL1=NWL+IC
      NXX=NWL1+IC
      NW=NXX+N
      NW1=NW+N
      NW2=NW1+N
      NWM=NW2+N
      NAS=NWM+M
      NKS=NAS+N1
      NKS0=NKS+N1
      NKSTC=NKS0+N1
      NKSTF=NKSTC+IC
      IL=MAX0(1,IC)
      CALL MMLC9A (FQQ,FHH,FDF,N,M,L,LEQ,C,DC,IL,X,DX,EPS,MAXF,KEQS,N1,N
     1N,W(NF),W(NF1),W(NDF),W(NDF1),W(NX1),W(NB),W(NU),W(NR),W(NA),W(NCL
     2),W(NWL),W(NWL1),W(NXX),W(NW),W(NW1),W(NW2),W(NWM),W(NAS),W(NKS),W
     3(NKS0),W(NKSTC),W(NKSTF),IFALL)
      IF (IFALL.LT.0) GO TO 10
      RETURN
   10 MAXF=0
      KEQS=0
      RETURN
```

```
      END
C
C
      SUBROUTINE MMLC9A (FQQ,FHH,FDF,N,M,L,LEQ,C,DC,IC,X,DX,EPS,MAXF,KEQ
     1S,N1,NN,F,F1,DF,DF1,X1,B,U,R,A,CLOC,WL,WL1,XX,W,W1,W2,WM,ASET,KSET
     2,KSET0,KSTATC,KSTATF,IFALL)
      DIMENSION C(IC), DC(IC,N), X(N), F(M), F1(M), WM(M), DF(M,N), DF1(
     1M,N), XX(NN), X1(N), U(N,N), R(N,N), B(N,N), A(NN,NN), CLOC(IC), W
     2L(IC), WL1(IC), W(N), W1(N), W2(N), ASET(N1)
      INTEGER KSET(N1),KSET0(N1),KSTATC(IC),KSTATF(M)
      LOGICAL DIV4,ACCUM,SHIFT
      EXTERNAL FQQ,FHH,FDF
      COMMON /MML000/ MARK
      DATA XZERO,XONE,XP73,XM50/0.0,1.0,1.E73,1.E-50/
C
C         SEPS IS AN EXPRESSION FOR THE MACHINE ACCURACY
C
      SEPS=2.0*16.0**(-12)
C
C         SET SOME CONSTANTS
C
      DIV4=.FALSE.
      LI=L-LEQ
C
C         INITIALIZE
C
      KEQSET=0
      NCALL=0
      NSHIFT=0
      NSTEP=0
      FMMREF=XP73
      DX0=DX
      DO 20 I=1,N
      DO 10 J=1,N
      B(I,J)=XZERO
   10 CONTINUE
      B(I,I)=XONE
   20 CONTINUE
C
C         FIND A FEASIBLE POINT
C
      IF (L.EQ.0) GO TO 80
      DO 40 I=1,L
      T=C(I)
      DO 30 J=1,N
      T=T+DC(I,J)*X(J)
   30 CONTINUE
      CLOC(I)=T
   40 CONTINUE
      NACT=0
      CALL FEASI (CLOC,DC,IC,LEQ,LI,N,XX,NACT,KSET,ASET,U,R,W1,W2,WL,WL1
     1,W,KSTATC,IFALL,ACCUM,SEPS)
      IF (IFALL.NE.0) RETURN
      DO 50 I=1,N
      X(I)=X(I)+XX(I)
   50 CONTINUE
      DO 70 I=1,L
      T=C(I)
      DO 60 J=1,N
      T=T+DC(I,J)*X(J)
   60 CONTINUE
      CLOC(I)=T
   70 CONTINUE
C
C         CALCULATE FUNCTION VALUES IN THE FIRST FEASIBLE POINT
```

```
C
   80 CALL FDF (N,M,X,DF,F)
      CALL FQQ (FHH,N,M,X,DF,F,1,0)
      FMM0=F(1)
      DO 90 I=1,M
      FMM0=AMAX1(FMM0,F(I))
   90 CONTINUE
      XN=XZERO
      DO 100 I=1,N
      XN=XN+X(I)*X(I)
  100 CONTINUE
      XN=SQRT(XN)
      NACT0=0
      NCALL=1
C
C        ITERATIVE LOOP STARTS HERE
C
  110 NACT=NACT0
      IF (NACT.EQ.0) GO TO 130
      DO 120 I=1,NACT
      KSET(I)=KSET0(I)
  120 CONTINUE
C
C        SOLVE THE LINEAR SUBPROBLEM
C
  130 CALL MMLPA (F,DF,CLOC,DC,M,N,N1,IC,LEQ,LI,DX,XXN,XX,NACT,KSET,ASET
     1,U,R,W1,W2,F1,WM,WL,WL1,KSTATF,KSTATC,W,SEPS,ACCUM,FMM,IFALL)
      IF (FMM.GE.FMM0) GO TO 400
C
C        CALCULATE FUNCTION VALUES IN THE NEW POINT
C
      DO 140 I=1,N
      X1(I)=X(I)+XX(I)
  140 CONTINUE
      CALL FDF (N,M,X1,DF1,F1)
      NCALL=NCALL+1
      CALL FQQ (FHH,N,M,X1,DF1,F1,NCALL,1)
      IF (MARK.EQ.0) GO TO 410
      FMM1=F1(1)
      DO 150 I=1,M
      FMM1=AMAX1(FMM1,F1(I))
  150 CONTINUE
C
C        REVISE THE STEP LENGTH
C
      IF ((FMM0-FMM1).GT.0.25*(FMM0-FMM)) GO TO 160
      DX=0.25*XXN
      DIV4=.TRUE.
      GO TO 180
  160 IF (DIV4) GO TO 170
      IF ((FMM0-FMM1).GT.0.75*(FMM0-FMM)) DX=XXN+XXN
  170 DIV4=.FALSE.
C
C        UPDATE THE HESSIAN APPROXIMATION
C
  180 DO 190 J=1,N
      W(J)=XZERO
      W1(J)=XZERO
  190 CONTINUE
      DO 210 I=1,NACT
      K=KSET(I)
      IF (K.LE.L) GO TO 210
      KK=K-L
      T=-ASET(I)
      DO 200 J=1,N
```

```
      W1(J)=W1(J)+T*DF1(KK,J)
      W(J)=W(J)+T*DF(KK,J)
  200 CONTINUE
  210 CONTINUE
      DO 220 I=1,N
      W2(I)=W1(I)-W(I)
  220 CONTINUE
      CALL BFGS (B,N,W2,XX,W,SEPS)
C
C       TEST IF THE NEW POINT IS ACCEPTABLE
C
      IF ((FMM0-FMM1).LE.0.01*(FMM0-FMM)) GO TO 320
C
C       COMPARE THE NEW ACTIVE SET WITH THE PRECEDING
C
      IF (NACT0.NE.NACT) GO TO 250
      DO 240 I=1,NACT
      K=KSET(I)
      DO 230 J=1,NACT
      IF (K.EQ.KSET0(J)) GO TO 240
  230 CONTINUE
      GO TO 250
  240 CONTINUE
      KEQSET=KEQSET+1
      GO TO 260
  250 KEQSET=1
C
C       INTRODUCE THE NEW POINT
C
  260 NSTEP=NSTEP+1
      XN=XZERO
      FMM0=FMM1
      NACT0=NACT
      DO 270 I=1,N
      X(I)=X1(I)
      XN=XN+X(I)**2
      DO 270 J=1,M
  270 DF(J,I)=DF1(J,I)
      XN=SQRT(XN)
      DO 280 I=1,M
      F(I)=F1(I)
  280 CONTINUE
      DO 290 I=1,NACT0
      KSET0(I)=KSET(I)
  290 CONTINUE
      IF (L.EQ.0) GO TO 320
      DO 310 I=1,L
      T=C(I)
      DO 300 J=1,N
      T=T+DC(I,J)*X(J)
  300 CONTINUE
      CLOC(I)=T
  310 CONTINUE
C
C       TEST OF CONVERGENCE CRITERION
C
  320 IF (XXN.LE.EPS*XN) GO TO 410
      IF (XXN.LE.SEPS*XN) GO TO 400
      IF (XXN.LE.XM50) GO TO 410
      IF (NCALL.GE.MAXF) GO TO 420
C
C       TEST FOR SWITCH TO STAGE-2
C
      SHIFT=FMM0.LE.FMMREF.AND.KEQSET.GE.KEQS.AND.NSTEP.GE.N
      IF (.NOT.SHIFT) GO TO 110
```

```
      IF (NACT.EQ.N1) GO TO 380
C
C       TEST FOR POSITIVE DEFINITENESS OF THE HESSIAN APPROX.
C       IN A RELEVANT DIRECTION
C
      DO 340 I=1,NACT
      K=KSET(I)
      IF (K.GT.L) GO TO 340
      T=ASET(I)
      DO 330 J=1,N
      W1(J)=W1(J)+T*DC(K,J)
  330 CONTINUE
  340 CONTINUE
      DO 360 I=1,N
      T=XZERO
      DO 350 J=1,N
      T=T+B(I,J)*W1(J)
  350 CONTINUE
      W(I)=T
  360 CONTINUE
      T=XZERO
      DO 370 I=1,N
      T=T+W(I)*W1(I)
  370 CONTINUE
      IF (T.LE.XZERO) GO TO 110
C
C       SHIFT TO STAGE-2
C
  380 NSHIFT=NSHIFT+1
      FMMREF=FMM0-10.0*SEPS*ABS(FMM0)
      XXNMAX=AMAX1(DX0,DX+DX)
      CALL S2LA1Q (FQQ,FHH,FDF,N,M,L,LEQ,C,CLOC,DC,IC,X,XXNMAX,B,NACT,KS
     1ET,ASET,N1,KSTATF,KSTATC,A,XX,NN,F,DF,X1,F1,DF1,W1,W2,EPS,MAXF,NCA
     2LL,XXN,NSTEP,SEPS,IFALL)
      IF (IFALL.LE.4) GO TO 410
      FMM0=-XP73
      DO 390 I=1,M
      FMM0=AMAX1(FMM0,F(I))
  390 CONTINUE
      DX=AMAX1(DX,0.5*XXN)
      KEQSET=1
      GO TO 110
C
C       RETURN
C
  400 IFALL=2
  410 MAXF=NCALL
      KEQS=NSHIFT
      EPS=XXN
      RETURN
  420 IFALL=3
      GO TO 410
      END
C
C
      SUBROUTINE S2LA1Q (FQQ,FHH,FDF,N,M,L,LEQ,C,CLOC,DC,IC,X,XXNMAX,B,N
     1ACT,KSET,ASET,N1,KSTATF,KSTATC,DZ,ZZ,NN,F,DF,X1,F1,DF1,W,W1,EPS,MA
     2XF,NCALL,XXN,NSTEP,SEPS,IFALL)
C
C     STAGE-2 (QUASI-NEWTON) ALGORITHM FOR LINEARLY CONSTRAINED
C     MINIMAX OPTIMIZATION.
C
      DIMENSION C(IC), CLOC(IC), DC(IC,N), X(N), B(N,N), ASET(N1), DZ(NN
     1,NN), ZZ(NN), F(M), DF(M,N), X1(N), F1(M), DF1(M,N), W(N), W1(N)
      INTEGER KSET(N1),KSTATF(M),KSTATC(IC)
```

```
      EXTERNAL FQQ,FHH,FDF
      COMMON /MML000/ MARK
      DATA XZERO,XONE,XP73,XM50/0.0,1.0,1.E73,1.E-50/
C
C       INITIALIZE
C
      LI=L-LEQ
      LE1=LEQ+1
      IFALL=0
      SSEPS=SQRT(SEPS)
      KK0=KSET(NACT)-L
      NACT1=NACT-1
      NZ=N+NACT1
      NSTEP2=0
      XXN=XZERO
      DO 10 I=1,M
      KSTATF(I)=0
   10 CONTINUE
      IF (L.EQ.0) GO TO 30
      DO 20 I=1,L
      KSTATC(I)=0
   20 CONTINUE
   30 DO 40 I=1,NACT
      K=KSET(I)
      IF (K.LE.L) KSTATC(K)=1
      IF (K.GT.L) KSTATF(K-L)=1
   40 CONTINUE
C
C       ITERATIVE LOOP STARTS HERE
C
C       SET UP THE ITERATION MATRIX AND THE RIGHTHAND SIDE
C
   50 DO 70 I=1,N
      DO 60 J=1,N
      DZ(I,J)=B(I,J)
   60 CONTINUE
      ZZ(I)=-DF(KK0,I)
   70 CONTINUE
      IF (NACT.EQ.1) GO TO 150
      DO 140 J=1,NACT1
      K=KSET(J)
      JN=J+N
      IF (K.GT.L) GO TO 90
      ZZ(JN)=-CLOC(K)
      DO 80 I=1,N
      DZ(I,JN)=DC(K,I)
      DZ(JN,I)=DZ(I,JN)
   80 CONTINUE
      GO TO 110
   90 KK=K-L
      ZZ(JN)=F(KK)-F(KK0)
      DO 100 I=1,N
      DZ(I,JN)=DF(KK0,I)-DF(KK,I)
      DZ(JN,I)=DZ(I,JN)
  100 CONTINUE
  110 DO 120 I=N1,NZ
      DZ(I,JN)=XZERO
  120 CONTINUE
      T=ASET(J)
      DO 130 I=1,N
      ZZ(I)=ZZ(I)-T*DZ(JN,I)
  130 CONTINUE
  140 CONTINUE
  150 RES0=XZERO
      DO 160 I=1,NZ
```

```
      RES0=RES0+ZZ(I)**2
  160 CONTINUE
      RES0=SQRT(RES0)
C
C         CALCULATE THE QUASI-NEWTON STEP
C
      CALL LINSYS (DZ,ZZ,NN,NZ,K,SEPS)
      IF (K.EQ.NZ) GO TO 170
      IFALL=8
      RETURN
C
C         CONTROL STEP LENGTH
C
  170 XXN1=XZERO
      ALFA=XONE
      DO 180 I=1,N
      XXN1=XXN1+ZZ(I)**2
  180 CONTINUE
      XXN1=SQRT(XXN1)
      IF (XXN1.GT.XXNMAX) ALFA=XXNMAX/XXN1
C
C         WILL OTHER CONSTRAINTS OR FUNCTIONS BECOME ACTIVE ?
C
      STEP=XP73
      IF (LI.EQ.0) GO TO 210
      DO 200 I=LE1,L
      IF (KSTATC(I).NE.0) GO TO 200
      T=XZERO
      DO 190 J=1,N
      T=T+ZZ(J)*DC(I,J)
  190 CONTINUE
      IF (T.GE.XZERO) GO TO 200
      T=-CLOC(I)/T
      IF (T.GT.STEP) GO TO 200
      STEP=T
  200 CONTINUE
  210 T0=XZERO
      DO 220 I=1,N
      T0=T0+ZZ(I)*DF(KK0,I)
  220 CONTINUE
      F0=F(KK0)
      DO 240 I=1,M
      IF (KSTATF(I).NE.0) GO TO 240
      T=XZERO
      DO 230 J=1,N
      T=T+ZZ(J)*DF(I,J)
  230 CONTINUE
      T=T0-T
      IF (T.GE.XZERO) GO TO 240
      T=(F(I)-F0)/T
      IF (T.GT.STEP) GO TO 240
      STEP=T
  240 CONTINUE
      IF (STEP.GT.ALFA) GO TO 250
      IFALL=9
      ALFA=STEP
C
C         SCALE THE STEP
C
  250 DO 260 I=1,NZ
      ZZ(I)=ALFA*ZZ(I)
  260 CONTINUE
      XXN1=ABS(ALFA)*XXN1
C
C         CALCULATE FUNCTION VALUES AND RESIDUALS IN THE NEW POINT
```

```
C
      XN1=XZERO
      DO 270 I=1,N
      X1(I)=X(I)+ZZ(I)
      XN1=XN1+X1(I)**2
  270 CONTINUE
      XN1=SQRT(XN1)
      NCALL=NCALL+1
      CALL FDF (N,M,X1,DF1,F1)
      CALL FQQ (FHH,N,M,X1,DF1,F1,NCALL,2)
      IF (MARK.EQ.0) GO TO 520
      DASET0=XZERO
      IF (NACT.EQ.1) GO TO 290
      DO 280 I=N1,NZ
      IF (KSET(I-N).GT.L) DASET0=DASET0-ZZ(I)
  280 CONTINUE
  290 RES=XZERO
      T=ASET(NACT)+DASET0
      DO 300 I=1,N
      W(I)=-T*DF(KK0,I)
      W1(I)=-T*DF1(KK0,I)
  300 CONTINUE
      IF (NACT.EQ.1) GO TO 350
      DO 340 J=1,NACT1
      K=KSET(J)
      JN=J+N
      T=ASET(J)+ZZ(JN)
      IF (K.GT.L) GO TO 320
      S=C(K)
      DO 310 I=1,N
      SS=DC(K,I)
      W(I)=W(I)+T*SS
      W1(I)=W1(I)+T*SS
      S=S+SS*X1(I)
  310 CONTINUE
      RES=RES+S**2
      GO TO 340
  320 KK=K-L
      DO 330 I=1,N
      W(I)=W(I)-T*DF(KK,I)
      W1(I)=W1(I)-T*DF1(KK,I)
  330 CONTINUE
      RES=RES+(F1(KK0)-F1(KK))**2
  340 CONTINUE
  350 DO 360 I=1,N
      RES=RES+W1(I)**2
  360 CONTINUE
      RES=SQRT(RES)
C
C       UPDATE THE HESSIAN APPROXIMATION
C
      DO 370 I=1,N
      W1(I)=W1(I)-W(I)
  370 CONTINUE
      CALL BFGS (B,N,W1,ZZ,W,SEPS)
C
C       TEST IF THE RESIDUAL HAS DECREASED
C
      IF (NSTEP2.EQ.0) GO TO 390
      IF (RES.LE.0.999*RES0) GO TO 390
C
C       IF NO - TEST FOR MACHINE ACCURACY
C
      IF (XXN1.GT.SSEPS*(XXNMAX+XN1).OR.NSTEP2.LT.2) GO TO 380
      IFALL=2
```

```
      RETURN
  380 IFALL=5
      RETURN
C
C        IF YES - INTRODUCE THE NEW POINT
C
  390 NSTEP2=NSTEP2+1
      NSTEP=NSTEP+1
      XN=XZERO
      DO 400 I=1,N
      X(I)=X1(I)
      DO 400 J=1,M
  400 DF(J,I)=DF1(J,I)
      XN=XN1
      XXN=XXN1
      FMAX=-XP73
      DO 410 I=1,M
      T=F1(I)
      FMAX=AMAX1(T,FMAX)
      F(I)=T
  410 CONTINUE
      ASET(NACT)=ASET(NACT)+DASET0
      IF (ASET(NACT).GT.XZERO) IFALL=6
      IF (NACT.EQ.1) GO TO 430
      DO 420 I=1,NACT1
      IN=I+N
      ASET(I)=ASET(I)+ZZ(IN)
      IF (KSET(I).GT.LEQ.AND.ASET(I).GT.XZERO) IFALL=6
  420 CONTINUE
  430 IF (L.EQ.0) GO TO 470
      DO 450 J=1,L
      T=C(J)
      DO 440 I=1,N
      T=T+DC(J,I)*X(I)
  440 CONTINUE
      CLOC(J)=T
  450 CONTINUE
C
C        TEST IF THE ACTIVE SET IS COMPLETE
C
      T=FMAX+RES
      DO 460 I=1,M
      IF (F(I).LE.T) GO TO 460
      IFALL=7
      RETURN
  460 CONTINUE
C
C        TEST CONVERGENCE CRITERION
C
  470 IF (XXN.GT.EPS*XN) GO TO 480
      IF (NACT.LT.N1) IFALL=1
      RETURN
  480 IF (XXN.GT.SEPS*XN) GO TO 490
      IFALL=2
      RETURN
  490 IF (XXN.GT.XM50) GO TO 500
      IFALL=0
      RETURN
  500 IF (NCALL.LT.MAXF) GO TO 510
      IFALL=3
      RETURN
  510 IF (IFALL.GT.4) RETURN
      GO TO 50
  520 IFALL=4
      RETURN
```

```
      END
C
C
      SUBROUTINE FEASI (C,DC,IC,LE,LI,N,X,NACT,KSET,ASET,U,R,DL,RIGHT,CU
     1P,DLDC,W,KSTAT,IFALL,ACCUM,SEPS)
C
C     THE SUBROUTINE FINDS A FEASIBLE POINT FOR A SET OF LINEAR
C     EQUALITY AND INEQUALITY CONSTRAINTS.
C
      DIMENSION C(IC), DC(IC,N), X(N), ASET(N), U(N,N), R(N,N), DL(N), R
     1IGHT(N), CUP(IC), DLDC(IC), W(N)
      INTEGER KSET(N),KSTAT(IC)
      LOGICAL ACCUM,OBJECT
      DATA XZERO,XP73/0.0,1.E73/
C
C        INITIALIZE
C
      EPS=(N+10)*SEPS
      ACCUM=.FALSE.
      NACTIN=NACT
      NACT=0
      LE1=LE+1
      LELI=LE+LI
      DO 10 I=1,N
      X(I)=XZERO
   10 CONTINUE
      IFALL=0
      IF (LELI.EQ.0) RETURN
      DO 20 I=1,LELI
      KSTAT(I)=0
   20 CONTINUE
C
C        MAKE ACTIVE THE EQUALITY CONSTRAINTS PLUS OTHER CONSTRAINTS
C        AS DEFINED IN KSET
C
      IF (LE.EQ.0) GO TO 50
      IF (LE.GT.N) GO TO 410
      DO 40 I=1,LE
      RIGHT(I)=-C(I)
      DO 30 J=1,N
      R(J,I)=DC(I,J)
   30 CONTINUE
      KSET(I)=I
      KSTAT(I)=1
   40 CONTINUE
      CALL ADDCL (U,R,N,NACT,LE,RIGHT,W,ACCUM,.FALSE.,EPS)
      IF (NACT.LT.LE) GO TO 410
   50 IF (NACTIN.LT.1) GO TO 80
      DO 70 K=1,NACTIN
      KK=KSET(K)
      IF (KK.LT.LE1.OR.KK.GT.LELI) GO TO 70
      NACT1=NACT+1
      IF (NACT1.GT.N) GO TO 80
      DO 60 I=1,N
      R(I,NACT1)=DC(KK,I)
   60 CONTINUE
      CALL UTTRNS (U,N,NACT,ACCUM,R(1,NACT1),W)
      CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.FALSE.,EPS)
      IF (NACT.LT.NACT1) GO TO 70
      RIGHT(NACT1)=-C(KK)
      KSET(NACT1)=KK
      KSTAT(KK)=1
   70 CONTINUE
   80 CALL TSOLV (R,N,NACT,RIGHT,X)
      IF (NACT.EQ.N) GO TO 100
```

```
      NACT1=NACT+1
      DO 90 I=NACT1,N
      X(I)=XZERO
   90 CONTINUE
  100 CALL UTRNS (U,N,NACT,ACCUM,X,W)
C
C        UPDATE THE CONSTRAINTS
C
      IF (LI.EQ.0) RETURN
      DO 120 I=LE1,LELI
      T=C(I)
      DO 110 J=1,N
      T=T+DC(I,J)*X(J)
  110 CONTINUE
      CUP(I)=T
  120 CONTINUE
C
C        INITIALIZE INEQUALITY CONSTRAINT LOOP
C
      DO 130 I=LE1,LELI
      IF (CUP(I).LT.XZERO.AND.KSTAT(I).EQ.0) KSTAT(I)=-1
  130 CONTINUE
C
C        ACTIVATE VIOLATED INEQUALITY CONSTRAINTS ONE BY ONE
C        USE THE STRONGEST VIOLATED AS OBJECTIVE CONSTRAINT
C
  140 FMIN=XP73
      DO 150 I=LE1,LELI
      IF (KSTAT(I).NE.-1) GO TO 150
      IF (CUP(I).GE.FMIN) GO TO 150
      FMIN=CUP(I)
      NEW=I
  150 CONTINUE
      IF (FMIN.GE.XZERO) RETURN
      DO 160 I=1,N
      RIGHT(I)=DC(NEW,I)
  160 CONTINUE
      CALL UTTRNS (U,N,NACT,ACCUM,RIGHT,W)
      KSTAT(NEW)=1
C
C        CALCULATE MULTIPLIERS FOR THE NEW ACTIVE CONSTRAINT AND DROP
C        CONSTRAINTS WITH POSITIVE MULTIPLIERS IN ORDER TO ACHIEVE
C        THE DIRECTION OF STEEPEST INCREMENT
C
  170 IF (NACT.EQ.LE) GO TO 220
      CALL RSOLV (R,N,NACT,RIGHT,ASET)
      AMAX=-XP73
      DO 180 I=LE1,NACT
      IF (ASET(I).LT.AMAX) GO TO 180
      K=I
      AMAX=ASET(I)
  180 CONTINUE
      IF (AMAX.LT.XZERO) GO TO 220
      KSTAT(KSET(K))=0
      DO 190 I=LE1,LELI
      IF (KSTAT(I).EQ.-2) KSTAT(I)=0
  190 CONTINUE
      IF (ACCUM) GO TO 200
      ACCUM=.TRUE.
      CALL HACUM (U,N,NACT,W)
  200 CALL DELCL (K,U,R,N,NACT,RIGHT,.TRUE.)
      IF (K.GT.NACT) GO TO 170
      DO 210 I=K,NACT
      KSET(I)=KSET(I+1)
  210 CONTINUE
```

```
      GO TO 170
C
C       CALCULATE THE PROJECTED GRADIENT
C
  220 T=XZERO
      DLN2=XZERO
      IF (NACT.EQ.0) GO TO 240
      DO 230 I=1,NACT
      T=T+RIGHT(I)**2
      DL(I)=XZERO
  230 CONTINUE
  240 NACT1=NACT+1
      IF (NACT.EQ.N) GO TO 260
      DO 250 I=NACT1,N
      DLN2=DLN2+RIGHT(I)**2
      DL(I)=RIGHT(I)
  250 CONTINUE
  260 T=T+DLN2
      IF (T.GT.XZERO.AND.DLN2.GT.EPS*EPS*T) GO TO 280
      S=(N+1)*ABS(C(NEW))
      DO 270 I=1,N
      S=S+ABS(DC(NEW,I)*X(I))*(N+3-I)
  270 CONTINUE
      IF (CUP(NEW).LT.-EPS*S) GO TO 410
      KSTAT(NEW)=0
      GO TO 140
  280 CALL UTRNS (U,N,NACT,ACCUM,DL,W)
C
C       PROJECT GRADIENTS ON THE PROJECTED GRADIENT
C
      DO 300 I=LE1,LELI
      T=XZERO
      DO 290 J=1,N
      T=T+DL(J)*DC(I,J)
  290 CONTINUE
      DLDC(I)=T
  300 CONTINUE
C
C       CALCULATE STEP LENGTH "ANES" TO MAKE THE OBJECTIVE CONSTRAINT
C       EQUAL ZERO, AND CALCULATE THE STEP LENGTH "AMIN" TO THE
C       NEAREST INACTIVE CONSTRAINT UNDER CONSIDERATION
C
      ANES=-CUP(NEW)/DLN2
  310 AMIN=XP73
      DO 320 I=LE1,LELI
      IF (KSTAT(I).NE.0) GO TO 320
      T=DLDC(I)
      IF (T.GE.XZERO) GO TO 320
      T=-CUP(I)/T
      IF (T.GT.AMIN) GO TO 320
      AMIN=T
      K=I
  320 CONTINUE
C
C       WILL THE OBJECTIVE CONSTRAINT GET ACTIVE ?
C       IF NOT, MAKE ACTIVE THE CLOSEST
C
      OBJECT=ANES.LE.AMIN
      ALFA=AMIN1(AMIN,ANES)
      NACT1=NACT+1
      IF (OBJECT) GO TO 350
      DO 330 I=1,N
      R(I,NACT1)=DC(K,I)
  330 CONTINUE
      CALL UTTRNS (U,N,NACT,ACCUM,R(1,NACT1),W)
```

```
      CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.TRUE.,EPS)
      IF (NACT1.EQ.NACT) GO TO 340
      KSTAT(K)=-2
      GO TO 310
  340 KSTAT(K)=1
      KSET(NACT)=K
C
C         TAKE THE STEP
C
  350 IF (ALFA.EQ.XZERO) GO TO 380
      DO 360 I=1,N
      X(I)=X(I)+ALFA*DL(I)
  360 CONTINUE
      DO 370 I=LE1,LELI
      T=CUP(I)+ALFA*DLDC(I)
      IF (KSTAT(I).EQ.-1.AND.T.GE.XZERO) KSTAT(I)=0
      CUP(I)=T
  370 CONTINUE
  380 IF (.NOT.OBJECT) GO TO 170
C
C         ACTIVATE THE OBJECTIVE CONSTRAINT
C
      DO 390 I=1,N
      R(I,NACT1)=RIGHT(I)
  390 CONTINUE
      CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.FALSE.,EPS)
      IF (NACT.EQ.NACT1) GO TO 400
      KSTAT(NEW)=0
      GO TO 140
  400 KSET(NACT)=NEW
      GO TO 140
C
C         NO FEASIBLE POINTS
C
  410 IFALL=3
      RETURN
      END
C
C
      SUBROUTINE MMLPA (F,DF,C,DC,M,N,N1,IC,LE,LI,XNMAX,XN,X,NACT,KSET,A
     1SET,U,R,DL,RIGHT,FUP,DLDF,CUP,DLDC,KSTATF,KSTATC,W,SEPS,ACCUM,FMAX
     2,IFALL)
C
C     THE SUBROUTINE SOLVES A LINEARLY CONSTRAINED LINEAR MINIMAX
C     PROBLEM. THE STARTING POINT MUST BE FEASIBLE.
C
      DIMENSION F(M), DF(M,N), C(IC), DC(IC,N), X(N), ASET(N1), U(N,N),
     1R(N,N), DL(N), RIGHT(N), FUP(M), DLDF(M), CUP(IC), DLDC(IC), W(N)
      INTEGER KSET(N1),KSTATF(M),KSTATC(IC)
      LOGICAL ACCUM
      DATA XZERO,XONE,XP73/0.0,1.0,1.E73/
C
C         INITIALIZE
C
      LE1=LE+1
      LELI=LE+LI
      XNMAX2=XNMAX**2
      XN2=XZERO
      EPS=N*SEPS
      ACCUM=.FALSE.
      IFALL=0
      DO 10 I=1,N
      X(I)=XZERO
   10 CONTINUE
      FMAX=-XP73
```

```
      DO 20 I=1,M
      KSTATF(I)=0
      T=F(I)
      IF (T.LE.FMAX) GO TO 20
      FMAX=T
      KSET0=I
   20 FUP(I)=T
      IF (LELI.EQ.0) GO TO 40
      DO 30 I=1,LELI
      KSTATC(I)=0
      CUP(I)=C(I)
   30 CONTINUE
C
C         ACTIVATE INITIAL ACTIVE SET
C
   40 NACTIN=NACT
      NACT=0
      IF (LE.EQ.0) GO TO 70
      DO 60 I=1,LE
      DO 50 J=1,N
      R(J,I)=DC(I,J)
   50 CONTINUE
      KSET(I)=I
      KSTATC(I)=1
   60 CONTINUE
      CALL ADDCL (U,R,N,NACT,LE,RIGHT,W,ACCUM,.FALSE.,EPS)
      IF (NACT.EQ.LE) GO TO 70
      XN=XZERO
      IFALL=3
      RETURN
   70 IF (NACTIN.LT.LE1) GO TO 100
      DO 90 K=1,NACTIN
      KK=KSET(K)
      IF (KK.LT.LE1.OR.KK.GT.LELI) GO TO 90
      NACT1=NACT+1
      IF (NACT1.GT.N) GO TO 100
      DO 80 I=1,N
      R(I,NACT1)=DC(KK,I)
   80 CONTINUE
      CALL UTTRNS (U,N,NACT,ACCUM,R(1,NACT1),W)
      CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.FALSE.,EPS)
      IF (NACT.LT.NACT1) GO TO 90
      EPS=EPS+SEPS
      KSET(NACT1)=KK
      KSTATC(KK)=1
   90 CONTINUE
C
C         TRANSFORM OBJECTIVE FUNCTION GRADIENT
C
  100 KSTATF(KSET0)=1
      DO 110 J=1,N
      RIGHT(J)=-DF(KSET0,J)
  110 CONTINUE
      KSET0=KSET0+LELI
      CALL UTTRNS (U,N,NACT,ACCUM,RIGHT,W)
C
C         ITERATIVE LOOP
C
C         CALCULATE MULTIPLIERS AND FIND THE LARGEST
C
  120 ASET0=-XONE
      IF (NACT.EQ.0) GO TO 240
      CALL RSOLV (R,N,NACT,RIGHT,ASET)
      IF (NACT.EQ.LE) GO TO 240
      AMAX=-XP73
```

```
      DO 130 I=LE1,NACT
      IF (KSET(I).GT.LELI) ASET0=ASET0-ASET(I)
      IF (ASET(I).LE.AMAX) GO TO 130
      K=I
      AMAX=ASET(I)
  130 CONTINUE
      IF (AMAX.LT.XZERO.AND.ASET0.LT.XZERO) GO TO 240
      IF (AMAX.GT.ASET0) GO TO 180
C
C        CHANGE OBJECTIVE FUNCTION
C
      DO 140 I=LE1,NACT
      IF (KSET(I).LE.LELI) GO TO 140
      K=I
      GO TO 150
  140 CONTINUE
  150 DO 170 I=1,K
      T=R(I,K)
      IF (K.EQ.NACT) GO TO 170
      K1=K+1
      DO 160 J=K1,NACT
      IF (KSET(J).GT.LELI) R(I,J)=R(I,J)-T
  160 CONTINUE
  170 RIGHT(I)=RIGHT(I)+T
      KK=KSET0
      KSET0=KSET(K)
      KSET(K)=KK
C
C        DELETE ACTIVE CONSTRAINT NUMBER K
C
  180 KK=KSET(K)
      IF (KK.GT.LELI) KSTATF(KK-LELI)=0
      IF (KK.LE.LELI) KSTATC(KK)=0
      IF (ACCUM) GO TO 190
      ACCUM=.TRUE.
      CALL HACUM (U,N,NACT,W)
  190 CALL DELCL (K,U,R,N,NACT,RIGHT,.TRUE.)
      EPS=EPS+SEPS
      IF (K.GT.NACT) GO TO 210
      DO 200 I=K,NACT
      KSET(I)=KSET(I+1)
  200 CONTINUE
C
C        DELETE LINEAR DEPENDENCE LABELS
C
  210 DO 220 I=1,M
      IF (KSTATF(I).EQ.-2) KSTATF(I)=0
  220 CONTINUE
      IF (LI.EQ.0) GO TO 120
      DO 230 I=LE1,LELI
      IF (KSTATC(I).EQ.-2) KSTATC(I)=0
  230 CONTINUE
      GO TO 120
C
C        IS THERE AN UNBOUNDED SOLUTION ?
C
  240 IF (NACT.EQ.N) GO TO 490
C
C        CALCULATE THE PROJECTED GRADIENT
C
      K=NACT+1
      T=XZERO
      DLN2=XZERO
      DO 250 I=K,N
      DLN2=DLN2+RIGHT(I)**2
```

```
      DL(I)=RIGHT(I)
  250 CONTINUE
      IF (K.EQ.1) GO TO 270
      DO 260 I=1,NACT
      T=T+RIGHT(I)**2
      DL(I)=XZERO
  260 CONTINUE
  270 T=T+DLN2
      IF (T.GT.XZERO.AND.DLN2.GT.EPS*EPS*T) GO TO 280
      IFALL=2
      GO TO 490
  280 CALL UTRNS (U,N,NACT,ACCUM,DL,W)
C
C        PROJECT GRADIENTS ON THE PROJECTED GRADIENT
C
      DO 300 I=1,M
      T=XZERO
      DO 290 J=1,N
      T=T+DL(J)*DF(I,J)
  290 CONTINUE
      DLDF(I)=T
  300 CONTINUE
      IF (LELI.EQ.0) GO TO 330
      DO 320 I=1,LELI
      T=XZERO
      DO 310 J=1,N
      T=T+DL(J)*DC(I,J)
  310 CONTINUE
      DLDC(I)=T
  320 CONTINUE
C
C        CALCULATE STEP LENGTH
C
  330 SMINC=XP73
      IF (LI.EQ.0) GO TO 350
      DO 340 I=LE1,LELI
      IF (KSTATC(I).NE.0) GO TO 340
      T=DLDC(I)
      IF (T.GE.XZERO) GO TO 340
      T=-CUP(I)/T
      IF (T.GT.SMINC) GO TO 340
      NEWC=I
      SMINC=T
  340 CONTINUE
  350 SMINF=XP73
      K=KSET0-LELI
      T0=DLDF(K)
      F0=FUP(K)
      DO 360 I=1,M
      IF (KSTATF(I).NE.0) GO TO 360
      T=T0-DLDF(I)
      IF (T.GE.XZERO) GO TO 360
      T=(FUP(I)-F0)/T
      IF (T.GT.SMINF) GO TO 360
      SMINF=T
      NEWF=I
  360 CONTINUE
      STEP=AMIN1(SMINF,SMINC)
C
C        IN CASE THE STEP IS TOO LONG REDUCE AND RETURN
C
      S=STEP
      CALL LIMIT (XNMAX2,X,XN2,DL,DLN2,S,N)
      IF (S.EQ.STEP) GO TO 370
      IFALL=1
```

```
      STEP=S
      GO TO 440
C
C        INCLUDE THE NEW FUNCTION/CONSTRAINT
C
  370 NACT1=NACT+1
      KK0=KSET0-LELI
      IF (SMINF.LT.SMINC) GO TO 400
      DO 380 I=1,N
      R(I,NACT1)=DC(NEWC,I)
  380 CONTINUE
      CALL UTTRNS (U,N,NACT,ACCUM,R(1,NACT1),W)
      CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.TRUE.,EPS)
      IF (NACT.EQ.NACT1) GO TO 390
      KSTATC(NEWC)=-2
      GO TO 330
  390 KSTATC(NEWC)=1
      KSET(NACT)=NEWC
      GO TO 430
  400 DO 410 I=1,N
      R(I,NACT1)=DF(KK0,I)-DF(NEWF,I)
  410 CONTINUE
      CALL UTTRNS (U,N,NACT,ACCUM,R(1,NACT1),W)
      CALL ADDCL (U,R,N,NACT,1,RIGHT,W,ACCUM,.TRUE.,EPS)
      IF (NACT.EQ.NACT1) GO TO 420
      KSTATF(NEWF)=-2
      GO TO 330
  420 KSTATF(NEWF)=1
      KSET(NACT)=NEWF+LELI
  430 EPS=EPS+SEPS
      IF (STEP.EQ.XZERO) GO TO 120
C
C        TAKE THE STEP AND UPDATE LINEAR FUNCTIONS
C
  440 FMAX=-XP73
      XN2=XZERO
      DO 450 I=1,N
      X(I)=X(I)+STEP*DL(I)
      XN2=XN2+X(I)**2
  450 CONTINUE
      DO 460 I=1,M
      T=FUP(I)+STEP*DLDF(I)
      IF (T.GT.FMAX) FMAX=T
      FUP(I)=T
  460 CONTINUE
      IF (LELI.EQ.0) GO TO 480
      DO 470 I=1,LELI
      CUP(I)=CUP(I)+STEP*DLDC(I)
  470 CONTINUE
  480 IF (IFALL.EQ.0) GO TO 120
C
C        RETURN
C
  490 XN=SQRT(XN2)
      NACT=NACT+1
      KSET(NACT)=KSET0
      ASET(NACT)=ASET0
      RETURN
      END
C
C
      SUBROUTINE LINSYS (A,B,IDIM,N,NR,EPS)
C
C     THE SUBROUTINE SOLVES A SYSTEM OF LINEAR EQUATIONS
C     USING GAUSSIAN ELIMINATION.
```

```
C
      DIMENSION A(IDIM,IDIM), B(N)
      DATA XONE,XM50/1.0,1.E-50/
      NR=0
C
C       A IS CONSIDERED TO BE OF RANK K-1 IF THE ABSOLUTE VALUE
C       OF THE K-TH PIVOT IS LESS THAN K*EPS.
C
      IF (N-1) 120,10,20
   10 IF (ABS(A(1,1)).LT.XM50) RETURN
      NR=1
      B(1)=B(1)/A(1,1)
      RETURN
C
C       EQUILIBRATION IN THE INFINITY NORM
C
   20 DO 40 I=1,N
      AM=ABS(A(I,1))
      DO 30 J=2,N
      S=ABS(A(I,J))
      IF (AM.LT.S) AM=S
   30 CONTINUE
      IF (AM.LT.XM50) AM=XONE
      B(I)=B(I)/AM
      DO 40 J=1,N
   40 A(I,J)=A(I,J)/AM
C
C       ELIMINATION
C
      N1=N-1
      DO 90 K=1,N1
      NR=K-1
C
C       FIND PIVOTAL ROW
C
      AM=ABS(A(K,K))
      I0=K
      K1=K+1
      DO 50 I=K1,N
      S=ABS(A(I,K))
      IF (S.LE.AM) GO TO 50
      AM=S
      I0=I
   50 CONTINUE
      IF (AM.LT.2*K*EPS) RETURN
      IF (I0.EQ.K) GO TO 70
C
C       INTERCHANGE EQUATIONS K AND I0
C
      DO 60 J=K,N
      S=A(K,J)
      A(K,J)=A(I0,J)
      A(I0,J)=S
   60 CONTINUE
      S=B(K)
      B(K)=B(I0)
      B(I0)=S
C
C       STORE PIVOT IN AM AND ELIMINATE IN ROWS K+1 TO N
C
   70 AM=A(K,K)
      DO 90 I=K1,N
      S=A(I,K)/AM
      DO 80 J=K1,N
      A(I,J)=A(I,J)-S*A(K,J)
```

```
   80 CONTINUE
   90 B(I)=B(I)-S*B(K)
      NR=N1
      IF (ABS(A(N,N)).LT.2*N*EPS) RETURN
C
C       A HAS FULL RANK
C
      NR=N
C
C       BACK SUBSTITUTION
C
      B(N)=B(N)/A(N,N)
      K=N
      DO 110 I=2,N
      K1=K
      K=K-1
      S=B(K)
      DO 100 J=K1,N
      S=S-A(K,J)*B(J)
  100 CONTINUE
      B(K)=S/A(K,K)
  110 CONTINUE
  120 RETURN
      END
C
C
      SUBROUTINE BFGS (B,N,Y,XX,W,SEPS)
C
C     UPDATES A HESSIAN APPROXIMATION USING BFGS-FORMULA.
C
      DIMENSION B(N,N), Y(N), XX(N), W(N)
      DATA XZERO/0.0/
      EPS=(N+10)*SEPS
      DO 20 I=1,N
      T=XZERO
      DO 10 J=1,N
      T=T+B(I,J)*XX(J)
   10 CONTINUE
      W(I)=T
   20 CONTINUE
      YXX=XZERO
      WXX=XZERO
      YN=XZERO
      XXN=XZERO
      WN=XZERO
      DO 30 I=1,N
      YN=YN+Y(I)**2
      XXN=XXN+XX(I)**2
      WN=WN+W(I)**2
      YXX=YXX+Y(I)*XX(I)
      WXX=WXX+W(I)*XX(I)
   30 CONTINUE
      YN=SQRT(YN)
      XXN=SQRT(XXN)
      WN=SQRT(WN)
      IF (YN.EQ.XZERO.OR.WN.EQ.XZERO.OR.XXN.EQ.XZERO) RETURN
      IF (ABS(YXX).LT.EPS*YN*XXN) RETURN
      IF (ABS(WXX).LT.EPS*WN*XXN) RETURN
      DO 40 I=1,N
      B(I,I)=B(I,I)+Y(I)**2/YXX-W(I)**2/WXX
   40 CONTINUE
      IF (N.EQ.1) RETURN
      DO 50 I=2,N
      I1=I-1
      DO 50 J=1,I1
```

```
      B(I,J)=B(I,J)+Y(I)*Y(J)/YXX-W(I)*W(J)/WXX
   50 B(J,I)=B(I,J)
      RETURN
      END
C
C
      SUBROUTINE ADDCL (U,R,N,KCOL,KNEW,RIGHT,W,ACCUM,LRIGHT,EPS)
C
C     UPDATES HOUSEHOLDER FACTORIZATION.
C     THE NEW COLUMNS MUST HAVE BEEN TRANSFORMED AS RIGHTHAND SIDES.
C
      DIMENSION U(N,N), R(N,N), RIGHT(N), W(N)
      LOGICAL ACCUM,LRIGHT
      DATA XZERO/0.0/
      K1=KCOL+1
      K2=KCOL+KNEW
C
C       COLUMN LOOP STARTS HERE
C
      DO 170 K=K1,K2
      S=XZERO
      T=XZERO
      IF (K.EQ.1) GO TO 20
      KK=K-1
      DO 10 I=1,KK
      T=T+R(I,K)**2
   10 CONTINUE
   20 DO 30 I=K,N
      S=S+R(I,K)**2
   30 CONTINUE
      T=T+S
      T=SQRT(T)
      S=SQRT(S)
C
C       RETURN IF THE NEW COLUMN DEPENDS LINEARLY ON THE
C       PRECEDING COLUMNS
C
      IF (T.EQ.XZERO) RETURN
      IF (S.LT.T*EPS) RETURN
C
C       PERFORM HOUSEHOLDER TRANSFORMATION
C
      TT=R(K,K)
      T=ABS(TT)
      ALFA=SQRT(S*(S+T))
      BETA=-SIGN(S,TT)
      R(K,K)=BETA
      W(K)=(TT-BETA)/ALFA
      IF (K.EQ.N) GO TO 80
      KK=K+1
      DO 40 I=KK,N
      W(I)=R(I,K)/ALFA
   40 CONTINUE
C
C       TRANSFORM THE REMAINING COLUMNS
C
      IF (K.EQ.K2) GO TO 80
      DO 70 J=KK,K2
      T=XZERO
      DO 50 I=K,N
      T=T+W(I)*R(I,J)
   50 CONTINUE
      DO 60 I=K,N
      R(I,J)=R(I,J)-T*W(I)
   60 CONTINUE
```

```
   70 CONTINUE
C
C        TRANSFORM THE RIGHTHAND SIDE
C
   80 IF (.NOT.LRIGHT) GO TO 110
      T=XZERO
      DO 90 I=K,N
      T=T+W(I)*RIGHT(I)
   90 CONTINUE
      DO 100 I=K,N
      RIGHT(I)=RIGHT(I)-T*W(I)
  100 CONTINUE
C
C        ACCUMULATE THE TRANSFORMATIONS IN U
C        U MUST HAVE BEEN INITIALIZED
C
  110 IF (ACCUM) GO TO 130
      DO 120 I=K,N
      U(I,K)=W(I)
  120 CONTINUE
      GO TO 170
  130 DO 160 I=1,N
      T=XZERO
      DO 140 J=K,N
      T=T+U(I,J)*W(J)
  140 CONTINUE
      DO 150 J=K,N
      U(I,J)=U(I,J)-T*W(J)
  150 CONTINUE
  160 CONTINUE
  170 KCOL=KCOL+1
      RETURN
      END
C
C
      SUBROUTINE DELCL (K,U,R,N,KCOL,RIGHT,LRIGHT)
C
C     DELETES COLUMN NUMBER K IN THE FACTORIZED MATRIX.
C     K MUST SATISFY       1.LE.K.LE.KCOL
C     U MUST HAVE BEEN ACCUMULATED.
C
      DIMENSION U(N,N), R(N,N), RIGHT(N)
      LOGICAL LRIGHT
C
C        DELETE COLUMN NUMBER K
C
      KCOL=KCOL-1
      IF (K.GT.KCOL) RETURN
      DO 10 J=K,KCOL
      J1=J+1
      DO 10 I=1,J1
   10 R(I,J)=R(I,J1)
C
C        TRANSFORM TO UPPER TRIANGULAR FORM
C        USING STANDARD GIVENS TRANSFORMATIONS
C
      DO 60 KK=K,KCOL
      K1=KK+1
      X=R(KK,KK)
      Y=R(K1,KK)
      A=SQRT(X*X+Y*Y)
      C=X/A
      S=Y/A
      R(KK,KK)=C*X+S*Y
      IF (KK.EQ.KCOL) GO TO 30
```

```
      DO 20 J=K1,KCOL
      X=R(KK,J)
      Y=R(K1,J)
      R(KK,J)=C*X+S*Y
      R(K1,J)=C*Y-S*X
   20 CONTINUE
   30 IF (.NOT.LRIGHT) GO TO 40
      X=RIGHT(KK)
      Y=RIGHT(K1)
      RIGHT(KK)=C*X+S*Y
      RIGHT(K1)=C*Y-S*X
C
C        ACCUMULATE THE TRANSFORMATIONS
C
   40 DO 50 I=1,N
      X=U(I,KK)
      Y=U(I,K1)
      U(I,KK)=C*X+S*Y
      U(I,K1)=C*Y-S*X
   50 CONTINUE
   60 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE UTTRNS (U,N,KCOL,ACCUM,R,W)
C
C     TRANSFORM THE VECTOR R AS A RIGHTHAND SIDE.
C
      DIMENSION U(N,N), R(N), W(N)
      LOGICAL ACCUM
      DATA XZERO/0.0/
C
C        IF THE TRANSFORMATIONS HAVE BEEN ACCUMULATED
C        DO SIMPLE MATRIX-MULTIPLICATION
C        ELSE TRANSFORM RIGHTHAND SIDES
C
      IF (ACCUM) GO TO 40
      IF (KCOL.EQ.0) RETURN
      DO 30 K=1,KCOL
      T=XZERO
      DO 10 I=K,N
      T=T+R(I)*U(I,K)
   10 CONTINUE
      DO 20 I=K,N
      R(I)=R(I)-T*U(I,K)
   20 CONTINUE
   30 CONTINUE
      RETURN
   40 DO 50 I=1,N
      W(I)=R(I)
   50 CONTINUE
      DO 70 K=1,N
      T=XZERO
      DO 60 I=1,N
      T=T+U(I,K)*W(I)
   60 CONTINUE
      R(K)=T
   70 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE UTRNS (U,N,KCOL,ACCUM,R,W)
C
```

```
C     TRANSFORM THE VECTOR R OPPOSITE A RIGHTHAND SIDE.
C
      DIMENSION U(N,N), R(N), W(N)
      LOGICAL ACCUM
      DATA XZERO/0.0/
      K1=KCOL+1
      IF (ACCUM) GO TO 40
      IF (KCOL.EQ.0) RETURN
      DO 30 KK=1,KCOL
      K=K1-KK
      T=XZERO
      DO 10 J=K,N
      T=T+U(J,K)*R(J)
   10 CONTINUE
      DO 20 J=K,N
      R(J)=R(J)-T*U(J,K)
   20 CONTINUE
   30 CONTINUE
      RETURN
   40 DO 50 I=1,N
      W(I)=R(I)
   50 CONTINUE
      DO 70 I=1,N
      T=XZERO
      DO 60 J=1,N
      T=T+U(I,J)*W(J)
   60 CONTINUE
      R(I)=T
   70 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE RSOLV (R,N,KCOL,RIGHT,X)
C
C     PERFORM BACK SUBSTITUTION ON RIGHT.
C
      DIMENSION R(N,N), RIGHT(N), X(N)
C
C       CALCULATE ALFA USING BACK SUBSTITUTION ON R
C
      K=KCOL
      K1=K+1
   10 IF (K.EQ.0) RETURN
      T=RIGHT(K)
      IF (K1.GT.KCOL) GO TO 30
      DO 20 J=K1,KCOL
      T=T-X(J)*R(K,J)
   20 CONTINUE
   30 X(K)=T/R(K,K)
      K1=K
      K=K-1
      GO TO 10
      END
C
C
      SUBROUTINE TSOLV (R,N,KCOL,RIGHT,X)
C
C     PERFORM BACK SUBSTITUTION ON RIGHT USING THE
C     TRANSPOSED TRIANGULAR MATRIX.
C
      DIMENSION R(N,N), RIGHT(N), X(N)
      IF (KCOL.EQ.0) RETURN
      X(1)=RIGHT(1)/R(1,1)
      IF (KCOL.EQ.1) RETURN
```

```
      DO 20 I=2,KCOL
      I1=I-1
      T=RIGHT(I)
      DO 10 J=1,I1
      T=T-X(J)*R(J,I)
   10 CONTINUE
      X(I)=T/R(I,I)
   20 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE HACUM (U,N,KCOL,W)
C
C     ACCUMULATES HOUSEHOLDER VECTORS STORED IN LOWER TRIANGLE
C     OF THE FIRST KCOL COLUMNS OF U IN AN ORTHONORMAL MATRIX U.
C     THE HOUSEHOLDER VECTORS MUST HAVE TWO NORM EQUAL TO TWO.
C     KCOL.GE.1 .
C
      DIMENSION U(N,N), W(N)
      DATA XZERO,XONE/0.0,1.0/
C
C        INITIALIZE USING LAST TRANSFORMATION
C
      K1=KCOL+1
      DO 10 I=KCOL,N
      W(I)=U(I,KCOL)
   10 CONTINUE
      DO 20 I=KCOL,N
      U(I,I)=XONE-W(I)**2
   20 CONTINUE
      IF (KCOL.EQ.N) GO TO 40
      DO 30 I=K1,N
      I1=I-1
      T=W(I)
      DO 30 J=KCOL,I1
      S=-T*W(J)
      U(I,J)=S
   30 U(J,I)=S
   40 IF (KCOL.EQ.1) RETURN
C
C        ACCUMULATE REMAINING TRANSFORMATIONS
C
      DO 100 KK=2,KCOL
      K=K1-KK
      DO 50 I=K,N
      W(I)=U(I,K)
   50 CONTINUE
      T=W(K)
      KP1=K+1
      U(K,K)=XONE-T*T
      DO 60 I=KP1,N
      U(I,K)=-T*W(I)
   60 CONTINUE
      DO 90 L=KP1,N
      S=XZERO
      DO 70 I=KP1,N
      S=S+W(I)*U(I,L)
   70 CONTINUE
      U(K,L)=-T*S
      DO 80 I=KP1,N
      U(I,L)=U(I,L)-S*W(I)
   80 CONTINUE
   90 CONTINUE
  100 CONTINUE
```

```
      RETURN
      END
C
C
      SUBROUTINE LIMIT (XNMAX2,X,XN2,P,PN2,ALFA,N)
C
C     LIMIT THE STEP LENGTH ALFA.
C
      DIMENSION X(N), P(N)
      DATA XZERO/0.0/
      XTP=XZERO
      DO 10 I=1,N
      XTP=XTP+X(I)*P(I)
   10 CONTINUE
      B=XTP/PN2
      T=SQRT(B*B+(XNMAX2-XN2)/PN2)
      AP=T-B
      AM=-T-B
      IF (ALFA.GT.AP) ALFA=AP
      IF (ALFA.LT.AM) ALFA=AM
      RETURN
      END
```

SOC-292

MMLC - A FORTRAN PACKAGE FOR LINEARLY CONSTRAINED MINIMAX OPTIMIZATION

J.W. Bandler and W.M. Zuberek

May 1982, No. of Pages: 78

Revised:

Key Words: Minimax optimization, constrained optimization, nonlinear programming, optimization program, computer-aided design

Abstract: MMLC is a package of subroutines for solving linearly constrained minimax optimization problems. It is an extension and modification of the MMLA1Q package due to Hald. First derivatives of all functions with respect to all variables are assumed to be known. The solution is found by an iteration that uses either linear programming applied in connection with first-order derivatives or a quasi-Newton method applied in connection with first-order and approximate second-order derivatives. The method has been described by Hald and Madsen. The package and documentation are developed for the CDC 170/730 system with the NOS 1.4 operating system and the Fortran 4.8508 compiler.

Description: Contains Fortran listing, user's manual. Source deck or magnetic tape available for $150.00. The listing contains 1972 lines, of which 427 are comments.

Related Work: SOC-218, SOC-280, SOC-281, SOC-291, SOC-294.

Price: $100.00.